Alcom-FT Technical Report Series
ALCOMFT-TR-03-34

On the complexity of some problems in interval arithmetic

K. Meer*

Department of Mathematics and Computer Science
Syddansk Universitet, Campusvej 55, 5230 Odense M, Denmark
e–mail: meer@imada.sdu.dk

Abstract. We study some problems in interval arithmetic treated in *Kreinovich et al.* [12]. First, we consider the best linear approximation of a quadratic interval function. Whereas this problem (as decision problem) is known to be NP-hard in the Turing model, we analyze its complexity in the real number model and the analoguous class $NP_{\mathbb{R}}$. Our results substantiate that most likely it does not any longer capture the difficulty of $NP_{\mathbb{R}}$ in such a real number setting. More precisely, we give upper complexity bounds for the approximation problem for interval functions by locating it in $D\Sigma^2_{\mathbb{R}}$ (a real analogue of Σ^2). This result allows several conclusions:
- the problem is not (any more) $NP_{\mathbb{R}}$-hard under so called weak polynomial time reductions and likely not to be $NP_{\mathbb{R}}$-hard under (full) polynomial time reductions;
- for fixed dimension the problem is polynomial time solvable; this extends the results in *Koshelev et al.* [13] and answers a question left open in [12].

We also study several versions of interval linear systems and show similar results as for the approximation problem.

Our methods combine structural complexity theory with issues from semi-infinite optimization theory.

1 Introduction

Problems in interval arithmetic model situations in which input data only is known within a certain accuracy. Starting from an exact description with input values $a_i, i \in I$ (say $a_i \in \mathbb{R}$ or $\in \mathbb{Q}, I$ an index set), a corresponding formalization in terms of interval arithmetic would only supply the information that the a_i's belong to some given intervals $[\underline{a}_i, \overline{a}_i] \subset \mathbb{R}$. This framework provides a way to formalize and study problems related to the presence of uncertainties. The latter both includes data errors occuring during data measurements and rounding errors during the performance of computer algorithms. Interval arithmetic thus can be seen as an approach to validate numerical calculations.

* partially supported by the Future and Emerging Technologies programme of the EU under contract number IST-1999-14186 (ALCOM-FT) and by the Danish Natural Science Research Council SNF.

The computational complexity of solving a problem in the interval setting might be significantly larger than solving the corresponding problem with accurate input data. In fact, many such results are known in interval arithmetic. As an example, consider the solvability problem for a linear equation system $A \cdot x = b$. If A and b are given precisely Gaussian elimination efficiently yields computation of a solution (or proves its non-existence). This holds both for the bit measure and the algebraic cost measure, see below. However, if we only know the entries in A and b to belong to given intervals, then the complexity changes dramatically; deciding the question whether concrete choices for A and b exist within the given (rational) interval bounds such that the resulting system for these choices is solvable is NP-complete, see [14, 12].
From a logical point of view this increase in the complexity (provided $P \neq NP$) is due to the presence of additional quantifiers in the interval problem description. For example, in the above linear system problem the interval approach has additional existential quantifiers ranging over the intervals of the given accuracy bounds and asking for the existence of "right" coefficients followed by the linear equation problem. Since the new quantifiers are ranging over real intervals, they in principle introduce a quantified formula in the first-order theory of the reals (considered as a real closed field; this of course only holds for algebraic problems). Even though this theory is decidable the complexity of currently known algorithms is tremendous. It is well known that already the existential theory over $\mathbb{R}$ is an $NP_{\mathbb{R}}$-complete problem; here, $NP_{\mathbb{R}}$ denotes the real analogue of NP in the framework of real Turing (or Blum-Shub-Smale, shortly: BSS) machines, see [1]. Therefore, it is natural to consider interval problems in that framework.
In this paper we want to analyze whether for interval problems known to be hard in the Turing setting the shift from the Turing to the real number model implies $NP_{\mathbb{R}}$-completeness or $NP_{\mathbb{R}}$-hardness of the problem in the latter as well. We shall substantiate that in the real number model a finer complexity analysis can be done. More precisely, for some problems the interval formulation will likely not lead to $NP_{\mathbb{R}}$-hardness, even though the restriction to rational problems and the Turing model implies NP-hardness (or completeness, respectively). This will be due to the fact that even though formally the new quantifiers range over the reals, in certain situations they can be replaced by Boolean quantifiers, i.e. such ranging over $\{0, 1\}$, only.
We study the following problems treated in [12] for clarifying our approach: Best approximation of quadratic interval functions by linear ones as well as several different versions of interval linear systems. The first problem is known to be NP-hard in the Turing model, [13]; note, however, that membership in the polynomial hierarchy is not established in [13]. The interval linear system problems are known to be NP-complete.
Let us start with some basic definitions from interval arithmetic, see [12]:

Definition 1 *(a) Let $B := [\underline{b}_1, \overline{b}_1] \times \ldots [\underline{b}_n, \overline{b}_n]$ be a box in $\mathbb{R}^n, \underline{b}_i < \overline{b}_i$ for $1 \le i \le n$. An* interval function *f on B is a mapping which assigns to each point $y \in B$ an interval $f(y) := [\underline{f}(y), \overline{f}(y)] \subseteq \mathbb{R}$. If both functions $\underline{f}$ and $\overline{f}$ are linear or quadratic functions, i.e., if they are polynomials of degree 1 or 2, resp., we call f a* linear *respectively a* quadratic *interval function.*
(b) Given a box B as above, a linear interval function $X := [\underline{X}, \overline{X}]$ and a quadratic interval function $f := [\underline{f}, \overline{f}]$ on B, we say that X approximates *f on B iff*

$$[\underline{f}(y), \overline{f}(y)] \subseteq [\underline{X}(y), \overline{X}(y)] \text{ for all } y \in B.$$

The approximation problem we study can be defined as a decision or a computational problem. We will mainly deal with the former.

Definition 2 *(a) The problem $BLAQIF$ (*best linear approximation of a quadratic interval function*) is defined as follows: given $n \in \mathbb{N}$, a box $B \subseteq \mathbb{R}^n$, a quadratic interval function $f(y) := [\underline{f}(y), \overline{f}(y)]$ on B and a bound $M \in \mathbb{R}$, is there a linear approximation $X = [\underline{X}, \overline{X}]$ of f on B such that*

$$\max_{y \in B} \overline{X}(y) - \underline{X}(y) \le M \ ?$$

(b) As a computational problem $BLAQIF$ asks for computing $\min\limits_{X} \max\limits_{y \in B} \overline{X}(y) - \underline{X}(y)$ under the constraints that $X = (\underline{X}, \overline{X})$ is an approximation of f.

Our main results now can be summarized as follows:
Main results: (i) In the real number model $BLAQIF$ is not $NP_{\mathbb{R}}$-complete under weak polynomial time reductions and likely (to be precised!) neither $NP_{\mathbb{R}}$-complete under (full) polynomial time reductions.
(ii) In the Turing model $BLAQIF$ can be located in Σ_2. For fixed input dimension n both the decision and the computational version can be solved in polynomial time.
Part (ii) complements the results in [13] by providing an upper complexity bound in the Turing setting. It also answers a question posed in [12].
The main results will be proved in Section 3, and Section 4 provides similar results for interval linear systems.

2 Basic notations; structural properties

We first recall the definition of some complexity classes important for our results. Then, we analyze the consequence for problems belonging to one of these classes w.r.t. completeness or hardness properties in the real number model.

2.1 Complexity classes

Though there are different equivalent definitions for the classes we need, for our purposes those based on alternating quantifiers are most appropriate. For more details see [18] for the Turing model and [1] for the real number model, where also Koiran's [11] weak model is treated.

Definition 3 *Let $k \in \mathbb{N}$ be fixed.*
a) A decision problem A over the finite alphabet $\{0,1\}$ belongs to class Σ^k iff there are a problem $B \in P$ and polynomials $p_1, \ldots, p_k$ such that

$$
\begin{aligned}
x \in A \Longleftrightarrow\ & Q_1 y_1 \in \{0,1\}^{p_1(|x|)} Q_2 y_2 \in \{0,1\}^{p_2(|x|)} \ldots Q_k y_k \in \{0,1\}^{p_k(|x|)} \\
& (x, y_1, \ldots, y_k) \in B \ ,
\end{aligned}
$$

where the variable blocks y_i range over $\{0,1\}^{p_i(|x|)}$ and the quantifiers $Q_i \in \{\exists, \forall\}$ alternate, starting with $Q_1 = \exists$ (and $|x|$ describes the bit size of x). Similarly, Π^k is obtained by the same condition, now starting with $Q_1 = \forall$.
b) A decision problem A over $\mathbb{R}^\infty := \bigoplus_{i=1}^{\infty} \mathbb{R}^i$ belongs to class $\Sigma_{\mathbb{R}}^k$ in the real number model iff there are a problem $B \in P_{\mathbb{R}}$ and polynomials $p_1, \ldots, p_k$ such that

$$
\begin{aligned}
x \in A \Longleftrightarrow\ & Q_1 y_1 \in \mathbb{R}^{p_1(|x|_{\mathbb{R}})} Q_2 y_2 \in \mathbb{R}^{p_2(|x|_{\mathbb{R}})} \ldots Q_k y_k \in \mathbb{R}^{p_k(|x|_{\mathbb{R}})} \\
& (x, y_1, \ldots, y_k) \in B \ ,
\end{aligned}
$$

where the variable blocks y_i range over $\mathbb{R}^{p_i(|x|_{\mathbb{R}})}$ and the quantifiers $Q_i \in \{\exists, \forall\}$ alternate, starting with $Q_1 = \exists$ (and $|x|_{\mathbb{R}}$ describes the algebraic size of x). Similarly, $\Pi_{\mathbb{R}}^k$ is obtained by the same condition, now starting with $Q_1 = \forall$.
c) If we in b) restrict the quantifiers to be Boolean ones, i.e. if the variable blocks range over $\{0,1\}^*$ instead of $\mathbb{R}^\infty$ we obtain the digital *classes $D\Sigma_{\mathbb{R}}^k$ and $D\Pi_{\mathbb{R}}^k$, respectively.*

Clearly, $\Sigma^1 = NP, \Sigma_{\mathbb{R}}^1 = NP_{\mathbb{R}}$ and $D\Sigma_{\mathbb{R}}^1 = DNP_{\mathbb{R}}$, where the latter is the class *digital* $NP_{\mathbb{R}}$ of problems in $NP_{\mathbb{R}}$ that require a discrete search space for verification, only. Here, we are most interested in $D\Sigma_{\mathbb{R}}^2$ and $DNP_{\mathbb{R}}$.

2.2 The real number complexity of problems in $D\Sigma_{\mathbb{R}}^2$

This section is on structural complexity of problems in $D\Sigma_{\mathbb{R}}^2$. The main goal is to argue that problems in $D\Sigma_{\mathbb{R}}^2$ likely do not bear the full complexity of $\Sigma_{\mathbb{R}}^2$, and even not of $NP_{\mathbb{R}}$-hard problems. This later on substantiates that for several problems in interval arithmetic that are known to be NP-hard in the Turing model, in the real number model a finer analysis can be done. We turn this into more precise statements as follows. We give an absolute statement with respect to so called *weak reductions* (introduced by Koiran [11] for a *weak* version of the BSS model): No problem in $D\Sigma^2_{\mathbb{R}}$ is $NP_{\mathbb{R}}$-hard under weak reductions. Then, we give an analogous statement for (general) polynomial time reductions under a widely believed hypothesis concerning computations of resultant polynomials: No problem in $D\Sigma^2_{\mathbb{R}}$ is $NP_{\mathbb{R}}$-hard under polynomial time reductions unless there is a (non-uniform) polynomial time algorithm computing a multiple of the resultant polynomial on a Zariski-dense subset. Though some definitions are necessary to precisely state these results, the proofs are almost straightforward extensions of similar statements for $DNP_{\mathbb{R}}$ given in [4] and [15].

Definition 4 *(Weak running time, [11]) Let M be a real machine with (real) machine constants $c := (c_1, \ldots, c_s)$ and having a running time bounded by a function t of the (algebraic) input size. Any intermediate result computed by M on x is a rational function of the form $\frac{p(x,c_1,\ldots,c_s)}{q(x,c_1,\ldots,c_s)}$, where p and q are polynomials with integer coefficients over x and c. The* weak costs *of computing this intermediate result are given as the maximum among the degrees of p, q and the bit sizes of any of its coefficient. Other operations of M (like branches and copying) have weak costs 1.*
The weak running time *of M on input $x \in \mathbb{R}^\infty$ is the sum of the weak costs of all intermediate results and branch-nodes along the computational path of M on x.*

Note that using the weak cost measure we still allow real number algorithms, but operation sequences like repeated squaring now get more expensive than in the BSS model, see [11].

The next definition introduces (a particular subcase of) the well known resultant polynomials. Consider the problem of deciding whether a given system $f := (f_1, \ldots, f_n) \in \mathbb{R}[x_1, \ldots, x_n]^n$ of n homogeneous polynomial systems of degree 2 in n variables has a zero $x \in \mathbb{C} \setminus \{0\}$, i.e. $f_i(x) = 0\ \forall i$. We denote by $\mathcal{H}$ the set of all such systems and by $\mathcal{H}_0$ those being solvable in $\mathbb{C} \setminus \{0\}$.
The implicitly stated claims in the following definition are well known, see f.e. [19, 5].

Definition 5 *Let $n \in \mathbb{N}, N := \frac{1}{2} \cdot n^2 \cdot (n+1)$. The* resultant polynomial $RES_n : \mathbb{R}^N \to \mathbb{R}$ *is a polynomial which as its indeterminates takes the coefficient vectors of homogeneous systems in $\mathcal{H}$. It is the unique (up to sign) irreducible polynomial with integer coefficients that generates the variety of (coefficient vectors of) solvable instances $\mathcal{H}_0$ of problems in $\mathcal{H}$, i.e.*

$$RES_n(f) = 0 \iff f \in \mathcal{H} \text{ has a zero } x \in \mathbb{C}^n \setminus \{0\}.$$

In this notation, $RES_n(f)$ is interpreted as evaluating RES_n on the coefficient vector of f in $\mathbb{R}^N$.

It is generally believed that no efficient algorithms for computing RES_n exist. This is, for example, substantiated by the close relation of this problem to other potentially hard computational problems like the computation of mixed volumes. For more on this see [19] and the literature cited in there. Hardness results for certain resultant computations can be found in [16]; relations between computation of resultants and the real $P_{\mathbb{R}}$ versus $NP_{\mathbb{R}}$ question are studied in [17].

Theorem 6. *a) No problem in $DNP_{\mathbb{R}}$ is $NP_{\mathbb{R}}$-complete under weak polynomial time reductions. No problem in $D\Sigma^2_{\mathbb{R}}$ is $NP_{\mathbb{R}}$-hard under weak polynomial time reductions.*
b) Suppose there is <u>no</u> (non-uniform) polynomial time algorithm which for each $n \in \mathbb{N}$ computes a non-zero multiple of RES_n on a Zariski-dense subset of $\mathcal{H}_0$. Then <u>no</u> problem in $DNP_{\mathbb{R}}$ is $NP_{\mathbb{R}}$-complete and <u>no</u> problem in $D\Sigma^2_{\mathbb{R}}$ is $NP_{\mathbb{R}}$-hard under polynomial time reductions in the BSS model.

Proof. The statements for $DNP_{\mathbb{R}}$ are proven in [4] and [15], resp.
ad a) Suppose $A \in D\Sigma^2_{\mathbb{R}}$ to be $NP_{\mathbb{R}}$-hard under weak reductions. In [4] the class *weak-$PAR_{\mathbb{R}}$* of problems solvable in weak parallel polynomial time is defined [1] and $NP_{\mathbb{R}} \not\subseteq$ *weak-$PAR_{\mathbb{R}}$* is shown. Our claim follows by showing $D\Sigma^2_{\mathbb{R}} \subseteq$ *weak-$PAR_{\mathbb{R}}$* because in that case a weak reduction from an $NP_{\mathbb{R}}$ problem to $A \in D\Sigma^2_{\mathbb{R}}$ would yield the contradiction $NP_{\mathbb{R}} \subseteq$ *weak-$PAR_{\mathbb{R}}$*. But $D\Sigma^2_{\mathbb{R}} \subseteq$ *weak-$PAR_{\mathbb{R}}$* follows from the same kind of argument used in [4]. First, note that $D\Sigma^2_{\mathbb{R}}$ equals its weak version *weak-$D\Sigma^2_{\mathbb{R}}$*, i.e. the class we obtain if in the definition of $D\Sigma^2_{\mathbb{R}}$ the algorithm has to run in weak polynomial time. This can be seen in precisely the same way as the analogous statement $NP_{\mathbb{R}} =$ *weak-$NP_{\mathbb{R}}$* in [4], using the completeness result for the digital classes in [3]. Secondly, by standard arguments the digital polynomial hierarchy over $\mathbb{R}$ is contained in *weak-$PAR_{\mathbb{R}}$*, i.e. in particular

$$\textit{weak-}D\Sigma^2_{\mathbb{R}} \;=\; D\Sigma^2_{\mathbb{R}} \;\subseteq \textit{weak-}PAR_{\mathbb{R}}\ .$$

ad b) Now suppose $A \in D\Sigma^2_{\mathbb{R}}$ to be $NP_{\mathbb{R}}$-hard under polynomial time reductions. Then, the $NP_{\mathbb{R}}$-problem $\mathcal{H}$ can be polynomially time reduced to A and thus belongs to $D\Sigma^2_{\mathbb{R}}$. We extend the arguments from [15] from $DNP_{\mathbb{R}}$ to $D\Sigma^2_{\mathbb{R}}$. Following Definition 3 suppose

$$f \in \mathcal{H}_0 \iff \exists\ y \in \{0,1\}^{p(|f|_{\mathbb{R}})}\ \ \forall\ z \in \{0,1\}^{q(|f|_{\mathbb{R}})}\ (f,y,z) \in B$$

for fixed polynomials p, q and a decision problem $B \in P_{\mathbb{R}}$. The arguments in [15] guarantee that there is a choice $y^* \in \{0,1\}^{p(|f|_{\mathbb{R}})}$ and a Zariski-dense

[1] $L \in$ *weak-$PAR_{\mathbb{R}}$* means that $x \in L$? can be decided using a machine that works with exponentially many processors each of which runs in weak polynomial time. The class was denoted by PAR_w in [4].

subset $\mathcal{Z}$ of $\mathcal{H}_0$ such that

$$f \in \mathcal{Z} \iff \forall \;\; z \in \{0,1\}^{q(|f|_{\mathbb{R}})} \; (f, y^*, z) \in B \; .$$

Moreover, on the complementary set $\mathbb{R}^N \setminus \mathcal{H}_0$ we have

$$f \in \mathbb{R}^N \setminus \mathcal{H}_0 \iff \exists z \in \{0,1\}^{q(|f|_{\mathbb{R}})} \; (f, y^*, z) \notin B \; .$$

Applying the arguments in [15] once more, the existence of a point $z^* \in \{0,1\}^{q(|f|_{\mathbb{R}})}$ can be concluded such that

$$f \in \mathcal{Z} \iff (f, y^*, z^*) \in B \text{ and}$$

$$\text{there is an } \tilde{f} \in \mathbb{R}^N \setminus \mathcal{H}_0 \text{ such that } (\tilde{f}, y^*, z^*) \notin B \; .$$

Now the condition on the right-hand side together with $B \in P_{\mathbb{R}}$ implies the existence of a polynomial time algorithm deciding $\mathcal{H}_0$ in $\mathcal{H}$ at least on $\mathcal{Z} \cup \{\tilde{f}\}$. We are again in the scope of [15] to apply the Dubois-Risler real Nullstellensatz, see [2], and
conclude that such an algorithm has to compute a non-zero multiple of RES_n on a Zariski-dense set in $\mathcal{H}_0$. Note that the importance of $\tilde{f}$ above is to rule out the trivial multiple 0. □

Remark 1. i) The above theorem can be generalized to the full digital polynomial hierarchy over the reals.
ii) Absolute results similar to a) exist for other reducibility notions, f.e. first-order reductions, see [6].
iii) Note that in b) we cannot expect an absolute statement of non-completeness like in a) unless $P_{\mathbb{R}} \neq NP_{\mathbb{R}}$ is proven (for the weak model *weak*-$P_{\mathbb{R}} \neq$ *weak*-$NP_{\mathbb{R}}$ is shown in [4]).

In the next sections the above theorem is used to substantiate the conjecture that interval problems which either belong to $DNP_{\mathbb{R}}$ or $D\Sigma^2_{\mathbb{R}}$ do not share the full difficulty of complete problems in $NP_{\mathbb{R}}$. Thus, in the real number model their complexities seem to be not the hardest possible among all (algebraic) interval problems in the corresponding real complexity class.

Remark 2. Of course, there are many interval problems which also in the real number model are $NP_{\mathbb{R}}$-hard or $NP_{\mathbb{R}}$-complete. If we start from an algebraic problem which already in its exact form has one of these properties, its interval version has it as well. An example is the solvability of interval quadratic systems.

3 Approximation of Interval Functions

As was already indicated the $BLAQIF$ problem can be considered both in the real number and the Turing model. In the former, the input size consists of $2n$ reals for the box B, the real coefficient vectors of the quadratic polynomials $\underline{f}$ and $\overline{f}$ and the real bound M (of size 1). Thus, the algebraic problem size is polynomially bounded in n. In the Turing model we restrict inputs to be rationals and add the bit-sizes to the algebraic size measure.

$BLAQIF$ is closely related to a semi-infinite optimization problem. Towards this end, suppose for a while that we have found an optimal linear approximation

$$\overline{X}(y) := \overline{x}_0 + \overline{x}_1 y_1 + \ldots + \overline{x}_n y_n, \overline{X}(y) - \overline{f}(y) \geq 0 \; \forall \; y \in B$$

$$\underline{X}(y) := \underline{x}_0 + \underline{x}_1 y_1 + \ldots + \underline{x}_n y_n, \underline{f}(y) - \underline{X}(y) \geq 0 \; \forall \; y \in B \; .$$

As shown in [12] it is easy to calculate $\max\limits_{y \in B} \overline{X}(y) - \underline{X}(y)$ once $\overline{X}, \underline{X}$ are known. The components y_i^* of the optimal y^* are determined by the signs of $\overline{x}_i - \underline{x}_i$ according to

$$y_i^* \; := \; \begin{cases} \overline{b}_i \text{ if } \overline{x}_i \geq \underline{x}_i \\ \underline{b}_i \text{ if } \overline{x}_i < \underline{x}_i \end{cases}$$

Knowing these signs we obtain a linear semi-infinite programming problem. For example, if we suppose $\overline{x}_i \geq \underline{x}_i \; \forall \; 1 \leq i \leq n$ the problem turns into

$$(LSI) \begin{cases} \min \overline{x}_0 + \sum\limits_{i=1}^{n} \overline{x}_i \cdot \overline{b}_i - \underline{x}_0 - \sum\limits_{i=1}^{n} \underline{x}_i \cdot \overline{b}_i & \\ & \\ \text{s.t.} \qquad \overline{x}_0 + \overline{x}^T \cdot y - \overline{f}(y) \geq 0 & \forall \; y \in B \\ \qquad\qquad \underline{f}(y) - \underline{x}_0 - \underline{x}^T \cdot y \geq 0 & \forall \; y \in B \\ \qquad\qquad\qquad \overline{x}_i \; \geq \; \underline{x}_i & \forall \; 1 \leq i \leq n \; , \end{cases}$$

where $\overline{x} := (\overline{x}_1, \ldots, \overline{x}_n)$, and similarly for $\underline{x}$.

This problem is linear on the upper variable level (i.e. the $2n + 2$ many x-variables) and quadratic for the lower variable level (i.e. y). It is semi-infinite because there are infinitely many side-constraints for $\overline{X}, \underline{X}$, parametrized through y.

Note that in general we do not know in advance which sign-conditions for the components of an optimal solution $\overline{X}, \underline{X}$ hold. Later on, we shall guess the right conditions as the first part of our $D\Sigma_{\mathbb{R}}^2$ algorithm and start from the resulting (LSI) problem.

General assumption: For sake of simplicity, in the following we assume w.l.o.g. $\overline{x}_i \geq \underline{x}_i \; \forall \; 1 \leq i \leq n$ and deal with the above (LSI).

It is easy to see that the decision version of $BLAQIF$ belongs to $\Sigma_{\mathbb{R}}^2$.

Lemma 1. $BLAQIF \in \Sigma^2_{\mathbb{R}}$.

Proof. A $\Sigma^2_{\mathbb{R}}$-machine guesses an optimal linear approximation $\overline{X}, \underline{X}$. Then, it checks the sign-conditions for the components and constructs the corresponding (LSI). It next computes the objective value and compares it to the bound M. The universal quantifier involved in $\Sigma^2_{\mathbb{R}}$ finally is used in order to check validity of the side constraints for all $y \in B$. □

The result, however, is not strong enough for what we want. It neither proves a similar statement in the Turing model (since we do not know how to bound the bit-sizes of the guessed reals) nor does it give any information that $BLAQIF$ in a real setting is likely to be an easier problem than complete ones for class $\Sigma^2_{\mathbb{R}}$ (and even for class $NP_{\mathbb{R}}$) are. In order to see how general quantifier elimination procedures over $\mathbb{R}$ can be avoided when solving the real $BLAQIF$ problem we have to deeper study semi-infinite optimization problems.

3.1 Optimality conditions for (LSI)

A fundamental idea for studying (LSI) is to reduce the infinitely many constraints to finitely many in order to apply common optimization criteria. For a detailed description of such methods and the regularity conditions under which they can applied the reader may consult one of the many publications on this topic, f.e. [7, 9, 8, 10]. In order to give the reader the possibility to check our statements in the original literature, we restate our problem using the terminology of [7].

Due to the fact that some of the constraints (namely the sign constraints on $\overline{x}_i - \underline{x}_i$) do not depend on y we enlarge the set B to $S := B \times T$, where $T := \{-1, 0, 1, \ldots, n\}$ is an index addressing the different constraints. Define functions $u := (u_1, \ldots, u_{2n+2}) : S \to \mathbb{R}^{2n+2}$ and $u_{2n+3} \to \mathbb{R}$ as follows:

$$u(y,-1) = \begin{pmatrix} 1 \\ y \\ 0 \end{pmatrix} \begin{matrix} \} \in \mathbb{R} \\ \} \in \mathbb{R}^n \\ \} \in \mathbb{R}^{n+1} \end{matrix} \quad ; u_{2n+3}(y,-1) := \overline{f}(y)$$

$$u(y,0) = \begin{pmatrix} 0 \\ -1 \\ -y \end{pmatrix} \begin{matrix} \} \in \mathbb{R}^{n+1} \\ \} \in \mathbb{R} \\ \} \in \mathbb{R}^n \end{matrix} \quad ; u_{2n+3}(y,0) := -\underline{f}(y)$$

$$u(y,k) = \begin{pmatrix} 0 \\ 1 \\ 0 \\ -1 \\ 0 \end{pmatrix} \begin{matrix} \} \in \mathbb{R}^k \\ \} \in \mathbb{R} \\ \} \in \mathbb{R}^n \\ \} \in \mathbb{R} \\ \} \in \mathbb{R}^{n-k} \end{matrix} \quad ; u_{2n+3}(y,k) := 0 \text{ for } 1 \le k \le n$$

Thus, problem (LSI) can be rewritten as

$$
(LSI)_2 \quad \begin{cases} \min \ (\underline{X}, \overline{X})^T \cdot \begin{pmatrix} 1 \\ b \\ -1 \\ -b \end{pmatrix} \\ \text{s.t.} \ \ (\underline{X}, \overline{X})^T \cdot u(y,t) \geq u_{2n+3} \ \ \forall \ (y,t) \in S \end{cases}
$$

Note that the cone $\{\begin{pmatrix} u(y,t) \\ u_{2n+3}(y,t) \end{pmatrix} |(y,t) \in B \times T\}$ is compact in $\mathbb{R}^{2n+3}$ since B is a box; furthermore, because both $\overline{f}$ and $\underline{f}$ are bounded on B there exists a $(\underline{X}, \overline{X}) \in \mathbb{R}^{2n+2}$ such that $(\underline{X}, \overline{X})^T \cdot u(y,t) > u_{2n+3} \ \forall \ (y,t) \in B \times T$. These two conditions are sufficient to obtain the following optimality criteria (see [7] p. 141f).

Theorem 7. *A feasible point $(\underline{X}, \overline{X})$ is optimal for $(LSI)_2$ iff the following conditions are satisfied: there exist two sets $\{y^{(i)}, i \in I\}$ and $\{y^{(j)}, j \in J\}$, each of at most n points in B, together with Lagrange parameters $\lambda_i, i \in I, \nu_j, j \in J$ and $\mu_k, 1 \leq k \leq n$ such that*

i) $\sum\limits_{i \in I} \begin{pmatrix} 1 \\ y^{(i)} \\ 0 \\ \underline{0} \end{pmatrix} \cdot \lambda_i + \sum\limits_{j \in J} \begin{pmatrix} 0 \\ \underline{0} \\ -1 \\ -y^{(j)} \end{pmatrix} \cdot \nu_j + \begin{pmatrix} 0 \\ \underline{\mu} \\ 0 \\ -\underline{\mu} \end{pmatrix} = \begin{pmatrix} 1 \\ \overline{b} \\ -1 \\ -\overline{b} \end{pmatrix}$

where $\mu := (\mu_1, \ldots, \mu_n)$ and $\underline{0} \in \mathbb{R}^n$;

ii) $\lambda_i \geq 0 \ \forall \ i \in I, \nu_j \geq 0 \ \forall \ j \in J, \mu_k \geq 0 \ \forall 1 \leq k \leq n$;

iii) either $\lambda_i = 0$ or the point $y^{(i)}$ is optimal for the problem

$$\min_{y \in B} \ \overline{x}_0 + \overline{x}^T \cdot y^{(i)} - \overline{f}(y),$$

and the optimal value is 0;

iv) either $\nu_j = 0$ or the point $y^{(j)}$ is optimal for the problem

$$\min_{y \in B} \ \underline{f}(y) - \underline{x}_0 - \underline{x}^T \cdot y^{(j)},$$

and the optimal value is 0;

v) $\mu_k \cdot (\overline{x}_k - \underline{x}_k) = 0 \ \ \forall \ 1 \leq k \leq n.$

This theorem is most important for obtaining our results by the following reasons. First, it states that at least one point $y^{(i)}$ and one point $y^{(j)}$ satisfying conditions iii) and iv), respectively, exist; this follows from $\sum\limits_{i \in I} \lambda_i = 1 = \sum\limits_{j \in J} \nu_j$. Therefore, we can search for it and are sure to find it if we guarantee the search to be exhaustive. Secondly, as global optima for the corresponding subproblems $y^{(i)}$ and $y^{(j)}$ satisfy the following optimality conditions on the lower level of the semi-infinite problem.

Corollary 1. *Using the setting of Theorem 7 let $y^{(i)}$ be a point satisfying condition iii), where $\lambda_i > 0$. Let*

$$AC(y^{(i)}) := \{k | y_k^{(i)} = \overline{b}_k \text{ or } y_k^{(i)} = \underline{b}_k\}$$

be the set of active components of $y^{(i)}$ in B. Then there exist Lagrange parameters $\eta_j \geq 0, j \in AC(y^{(i)})$ such that

$$\overline{x} - D_y \overline{f}(y^{(i)}) = \sum_{j \in AC(y^{(i)})} \eta_j \cdot (\pm e_j) \ ,$$

where e_j is the j-th unit vector and the sign is $+1$ iff $y_j^{(i)} = \overline{b_j}$ and -1 iff $y_j^{(i)} = \underline{b_j}$.

Proof. This is Lagrange's theorem; note that the linear independence condition is satisfied for the constraints $\underline{b}_k \leq y_k^{(i)} \leq \overline{b}_k \ \forall\, 1 \leq k \leq n$. □

3.2 Linear approximation of quadratic functions is in $D\Sigma_{\mathbb{R}}^2$

The previous results on the relations of $BLAQIF$ and (LSI) are used in this subsection in order to prove membership in $D\Sigma_{\mathbb{R}}^2$ resp. in Σ^2 as follows. The overall goal is to find a solution $(\underline{X}, \overline{X})$ and check that it realizes the demanded bound M. It has to be shown how that can be realized using binary (digital) quantifiers. Towards this end

1) we guess the right set of signs for $\overline{x}_k - \underline{x}_k, 1 \leq k \leq n$ and produce the corresponding (LSI); w.l.o.g. we again assume all these signs to be 0 or 1.
2) Assuming $(\underline{X}, \overline{X})$ to be known we guess certain discrete information which then is used to compute at least one point $y^{(i)}, i \in I$ and one point $y^{(j)}, j \in J$ satisfying Theorem 7. This is done using Corollary 1 and the ideas developed in [15].
3) From the corollary and the information obtained in 2) we deduce conditions that have to be fulfilled by an optimal solution $(\underline{X}, \overline{X})$. These conditions lead to a linear programming problem. By means of a $DNP_{\mathbb{R}}$ algorithm we obtain a candidate $(\underline{X}, \overline{X})$ for the optimum.
4) Finally, the candidate obtained in 3) is checked for optimality. This mainly requires to check the constraints, which now are quadratic programs in y. Using the results of [15] this problem belongs to class $co\text{-}DNP_{\mathbb{R}} = D\Pi_{\mathbb{R}}^1$. Together, we obtain a $D\Sigma_{\mathbb{R}}^2$ algorithm.

Now towards the details.

Theorem 8. *Let $y^{(i)} \in S$ and $y^{(j)} \in S$ be two points in the statement of Theorem 7 such that the corresponding Lagrange parameters λ_i and ν_j are positive. Suppose that we do neither know an optimal $(\underline{X}, \overline{X})$ nor $y^{(i)}, y^{(j)}$. Then having the correct information about the signs for $\overline{x}_i - \underline{x}_i$ of an optimal solution and about the active components of $y^{(i)}$ and $y^{(j)}$ in S (i.e. those components that correspond either to $\overline{b}_k$ or to $\underline{b}_k$) we can compute an optimal solution $(\underline{X}, \overline{X})$ of (LSI) as (any) solution of a specific linear programming problem. Moreover, the latter linear programming problem can be constructed deterministically in polynomial time if the active components are known.*

Corollary 2. *There is a $D\Sigma^1_{\mathbb{R}}$ algorithm which computes a set $\mathcal{X}$ of vectors in which an optimal solution of (LSI) can be found, i.e. a non-deterministic algorithm that guesses a vector in $\{0,1\}^*$ of polynomial length in n and produces a candidate $(\underline{X}, \overline{X})$ for each guess such that at least one of the candidates produced is an optimal solution.*

Proof. The active components of $y^{(i)}$ and $y^{(j)}$ obviously can be coded by a bit-vector. Using Theorem 8 we construct in polynomial time the linear programming problem. It was shown in [15] that $LP \in DNP_{\mathbb{R}}$ and the algorithm presented there actually uses a guessed binary vector in order to compute an optimal solution of the problem. Theorem 8 (see also its proof below) guarantees such a solution to solve (LSI). □

Proof. (of Theorem 8) As it can be seen from the proof below it will be sufficient to argue for one of the points $y^{(i)}, y^{(j)}$ only, so let us consider $y^{(i)}$. W.l.o.g. suppose the first s many components to be active and to satisfy $y^{(i)}_k = \underline{b}_k$ for $1 \le k \le s$. This actually is the most difficult case because the values of the active components $y^{(i)}_k = \underline{b}_k$ do not correspond to the assumed inequalities $\overline{x}_k - \underline{x}_k \ge 0$ which in the objective function result in the terms $(\overline{x}_k - \underline{x}_k) \cdot \overline{b}_k$ (instead of $(\overline{x}_k - \underline{x}_k) \cdot \underline{b}_k$ which would correspond to $y^{(i)}_k = \underline{b}_k$). However, the difference only results in an additional LP-problem which is of no concern in our analysis.
We plug the active components into the constraint

$$\overline{x}_0 + \overline{x}^T \cdot y^{(i)} - \overline{f}(y^{(i)}) \ge 0$$

and obtain a quadratic minimization problem in the remaining components $y_{s+1}, \ldots, y_n$:

$$(*) \quad \begin{cases} \min \overline{x}_0 + \sum\limits_{k=1}^{s} \overline{x}_k \cdot \underline{b}_k + \sum\limits_{k=s+1}^{n} \overline{x} \cdot y_k - \overline{f}(\underline{b}_1, \ldots, \underline{b}_s, y_{s+1}, \ldots, y_n) \\ \\ \text{such that } \underline{b}_k < y_k < \overline{b}_k \ , \ s+1 \le k \le n \end{cases}$$

If the guess was correct we know that an interior solution for $\tilde{y} := y_{s+1}, \ldots, y_n$ exists. Now define

$$\overline{f}(\underline{b}_1, \ldots, \underline{b}_s, y_{s+1}, \ldots, y_n) := \frac{1}{2}\tilde{y}^T \cdot \overline{D} \cdot \tilde{y} + \overline{h}^T \cdot \tilde{y} + \overline{e} \ ,$$

where $\overline{D} \in \mathbb{R}^{(n-s)\times(n-s)}, \overline{h} \in \mathbb{R}^{n-s}, \overline{e} \in \mathbb{R}$.
Then Corollary 1 together with a straightforward calculation gives:

i) an optimal (interior) solution $\tilde{y}$ lies in the kernel of $\overline{D}$;
ii) an optimal (interior) solution $\tilde{y}$ satisfies

$$\begin{pmatrix} \overline{x}_{s+1} \\ \vdots \\ \overline{x}_n \end{pmatrix} = \overline{D} \cdot \tilde{y} + \overline{h} \ .$$

Thus, i) implies $(\overline{x}_{s+1}, \ldots, \overline{x}_n)^T = \overline{h}$ and we can compute these components of the (LSI) solution directly;
iii) the optimal value of $(*)$ is 0; together with ii) this results in

$$\overline{x}_0 + \sum_{k=1}^{s} \overline{x}_k \cdot \underline{b}_k = \overline{e} \ .$$

In a completely analogue fashion we obtain a similar condition for the part $\underline{X}$ of a solution when studying an optimal $y^{(j)}$ for $\min \underline{f}(y) - \underline{x}^T \cdot y - \underline{x}_0$. If w.l.o.g. the last $n-\ell$ components of a solution $y^{(j)}$ are active we get $(\underline{x}_1, \ldots, \underline{x}_\ell)^T = \underline{h}$ as well as

$$\underline{x}_0 + \sum_{k=1}^{s} \underline{x}_k \cdot \underline{b}_k = \underline{e}$$

for appropriate values $\underline{h}, \underline{e}$ that can easily be computed knowing the active components of $y^{(j)}$.
Putting all the information together we have the following situation: knowing the active components of $y^{(i)}, y^{(j)}$ we can directly compute in polynomial time from the problem input those components of an optimal solution $(\underline{X}, \overline{X})$ that correspond to non-active $y_k^{(i)}, y_k^{(j)}$. The remaining ones can be obtained as optimal solution of the linear program

$$\begin{aligned} \min\ & \overline{x}_0 + \sum_{k=1}^{s} \overline{x}_k \cdot \overline{b}_k + \sum_{k=s+1}^{n} \overline{h}_k \cdot \overline{b}_k - \underline{x}_0 + \sum_{k=1}^{\ell} \underline{h}_k \cdot \underline{b}_k - \sum_{k=\ell+1}^{n} \underline{x}_k \cdot \overline{b}_k \\ \text{s.t. } & \overline{x}_0 + \sum_{k=1}^{s} \overline{x}_k \cdot \underline{b}_k = \overline{e} \\ & \underline{x}_0 + \sum_{k=\ell+1}^{n} \underline{x}_k \cdot \underline{b}_k = \underline{e} \end{aligned}$$

Following [15] such a solution can be computed non-deterministically in polynomial time using a binary vector as guess. □

The theorem can be used to prove the main result of this section:

Theorem 9. *The BLAQIF decision problem belongs to $D\Sigma^2_{\mathbb{R}}$ in the real number model and to Σ^2 in the Turing model.*

Proof. It is clear that there exists a best linear approximation for each instance. The first sequence of binary existential quantifiers is used to find the correct (LSI) version, i.e. to guess the correct signs for $\overline{x}_k - \underline{x}_k$ in an optimal solution. We use the guess to construct the right objective function for the problem (as described before). According to Theorem 7 there exist two points $y^{(i)}, y^{(j)}$ as described in Theorem 8. Moreover, we can guess a binary vector of polynomial length in the algebraic input size, perform the algorithm described in the proof of Theorem 8 and compute a candidate $(\underline{X}, \overline{X})$ for an optimal solution of the (LSI) instance. The proof also guarantees that if we would handle in a deterministic (but inefficient) algorithm all possible guesses at least one would give an optimal solution, see Corollary 2.
In the remaining part of the $D\Sigma^2_{\mathbb{R}}$ algorithm we have to verify that the computed candidate $(\underline{X}, \overline{X})$ indeed is feasible and gives a bound $\leq M$ for the objective function. Whereas the latter is done by a simple evaluation the former requires the computation of an optimal point for two quadratic programming problems with linear constraints. These problems have the lower level variables y as unknowns and are obtained by plugging $\overline{X}$ and $\underline{X}$ into the lower level equations. We have seen this problem to be in *co*-$DNP_{\mathbb{R}}$. Note that if we want to get a globally minimal point we have to compare it with all other candidates. Thus, this part corresponds to checking validity of a formula containing a sequence of $O(n)$ many universal binary quantifiers. This implies $BLAQIF \in D\Sigma^2_{\mathbb{R}}$.
In the Turing model the above structure of binary quantifiers still describes the Σ^2 procedure. The only point to check is that the intermediate computations can be performed in polynomial time w.r.t. the bit-measure. This is true for the arguments relying in [15] as well as for the proof of Theorem 8: No additional constants are introduced in these algorithms and the construction of intermediate matrices and LP-subproblems is done by only rearranging some of the input data. □

Corollary 3. *$BLAQIF$ is not $NP_{\mathbb{R}}$-hard under weak polynomial time reductions; it is not $NP_{\mathbb{R}}$-hard under polynomial time reductions unless a non-zero multiple of RES_n can be computed non-uniformly in polynomial time on a Zariski-dense subset of $\mathcal{H}_0$.* □

Theorem 9 also answers a question posed in [12], chapter 19, concerning the complexity of the rational $BLAQIF$ problem if the dimension n is fixed. Our result extends the one in [13].

Theorem 10. *Let $n \in \mathbb{N}$ be fixed. The computational version of the $BLAQIF$ problem for rational inputs and fixed dimension n is solvable in polynomial time in the Turing model.*

Proof. The proofs of Theorems 8 and 9 do not only show $BLAQIF \in \Sigma^2$ (for rational inputs), but also establish that the number of quantified variables in the corresponding Σ^2 formula is a (linear) function of n. Thus, if n is fixed we can eliminate the quantifiers in polynomial time by evaluating all possible 0-1 assignments for the binded variables. Finally, a property in P has to be checked. An optimal solution can be computed from one of the now constantly many candidates, see Corollary 2. □

4 Interval linear systems

In this final section we want to apply the previous ideas to a few other problems around interval linear systems, see [12]. Due to already known results (see [14] and [12]) it will be easier to locate these problems in the class $DNP_{\mathbb{R}}$. We consider the following problems, see [14]:

Definition 11 *Consider as input an $(m \times n)$ interval matrix $\mathfrak{A}$ and an m-dimensional interval vector $\mathfrak{b}$. The entries $\mathfrak{a}_{ij}$ and $\mathfrak{b}_j$ of $\mathfrak{A}$ and $\mathfrak{b}$ are intervals in $\mathbb{R}$ representing the sets from which possible coefficients and right-hand sides, respectively, can be chosen. For a matrix $A \in \mathbb{R}^{m\times n}$ we write $A \in \mathfrak{A}$ iff the components a_{ij} belong to the interval $\mathfrak{a}_{ij}, \forall i, j$, and similarly for a vector $b \in \mathfrak{b}$. We define the following problems:*

- $\mathcal{P}_1$: *does there exist a point $x \in \mathbb{R}^n$, a matrix $A \in \mathfrak{A}$ and a point $b \in \mathfrak{b}$ such that $A \cdot x = b$?*
- $\mathcal{P}_2$: *does there exist a vector $x \in \mathbb{R}^n$ such that for all $A \in \mathfrak{A}$ we have $A \cdot x \in \mathfrak{b}$?*
- $\mathcal{P}_3$: *does there exist a vector $x \in \mathbb{R}^n$ such that for all $b \in \mathfrak{b}$ there is an $A \in \mathfrak{A}$ such that $A \cdot x = b$?*
- $\mathcal{P}_4$: *is the intersection of the solution sets for $\mathcal{P}_2$ and $\mathcal{P}_3$ non-void?*

We denote by $\mathcal{R}_i$ the solution set of $\mathcal{P}_i$, i.e. those $x \in \mathbb{R}^n$ that satisfy the corresponding requirements.

These problems were studied in [14] and the results obtained there will be needed to prove the following theorem.

Theorem 12. *In the real number model all problems $\mathcal{P}_i, 1 \le i \le 4$ belong to the class $DNP_{\mathbb{R}}$. Thus, Theorem 6 applies to them.*

Proof. Following [14] one can represent the solution set of $\mathcal{P}_1$ as

$$\mathcal{R}_1 = \{x - y | x \ge 0,\ y \ge 0,\ x^T \cdot y = 0,\ \underline{A} \cdot A - \overline{A} \cdot y \le \underline{b},\ \overline{A} \cdot x - \underline{A} \cdot y \ge \underline{b}\},$$

where $\mathfrak{A} =: [\underline{A}, \overline{A}]$ and $\mathfrak{b} =: [\underline{b}, \overline{b}]$. A $DNP_{\mathbb{R}}$-verification first uses complementary slackness and guesses two maximal index sets $S_1, S_2 \subseteq \{1, \ldots, n\}$ such that there is a solution x^*, y^* and $x_i^* = 0\ \forall i \in S_1, y_j^* = 0\ \forall j \in S_2$. Maximality of S_1, S_2 here means that the remaining components of x^*, y^* have to be positive.

Plugging into the description of $\mathcal{R}_1$ the zero components we get a linear programming problem

$$\tilde{A} \cdot z \le \tilde{b}\ ,\ \tilde{\tilde{A}} \cdot z \ge \tilde{\tilde{b}}\ ,\ z > 0$$

in variables z (which represent the non-zero variables among x^*, y^*). Now a solution of a linear programming problem according to [15] can be computed using a $DNP_{\mathbb{R}}$-algorithm. We perform this non-deterministic algorithm, extend the solution we obtain by adding zero components to get two points x^*, y^* and check whether (x^*, y^*) is a solution. If the problem is solvable we are sure to find a solution that way.

For the sets $\mathcal{R}_2, \mathcal{R}_3, \mathcal{R}_4$ similar descriptions exist according to [14] and we can prove membership in $DNP_{\mathbb{R}}$ along the same lines. □

The previous theorem together with Theorem 6 shows that none of the problems $\mathcal{P}_i, 1 \le i \le 4$ is $NP_{\mathbb{R}}$- complete under weak reductions and they are unlikely to be $NP_{\mathbb{R}}$-complete under full reductions. This is basically interesting for the problems $\mathcal{P}_1, \mathcal{P}_3, \mathcal{P}_4$, only, because they are NP-complete in the Turing setting. Problem $\mathcal{P}_2$ is a linear programming problem (due to absence of the non-linear complementary slackness condition) and thus in P over the rationals. The question whether Linear Programming is in $P_{\mathbb{R}}$ in the real number model remains currently (to the author's knowledge) one of the main open problems in linear optimization.

5 Concluding remarks

We have studied some problems in interval arithmetic treated in [12] like approximation of interval functions and different forms of interval linear systems. Whereas these problems are NP-hard in the Turing model we studied their complexity in a real number setting. Due to an intrinsic discrete structure it turned out that these problems are likely to loose their hardness resp. completeness properties for the real analogue $NP_{\mathbb{R}}$ of NP. This implies that the full power of quantifier elimination methods over real closed fields is not necessary to solve these problems. Our results on approximation of quadratic interval functions as well have an application to the Turing model: The location of the problem $BLAQIF$ in class Σ^2 for rational inputs gives a polynomial time algorithm for the computational version in fixed dimensions in the Turing model, thus answering a question posed in [12].

6 Acknowledgement

I want to thank G.W. Weber for helpful discussions on semi-infinite programming. Thanks are as well due to V. Kreinovich for providing me with a preliminary version of [12] some years ago.

References

1. L. Blum, F. Cucker, M. Shub, S. Smale: Complexity and Real Computation. Springer, 1998.
2. J. Bochnak, M. Coste, M.F. Roy: Real Algebraic Geometry. Springer 1998.
3. F. Cucker, M. Matamala: On digital nondeterminism. Mathematical Systems Theory, 29, 635 – 647, 1996.
4. F. Cucker, M. Shub, S. Smale: Complexity separations in Koiran's weak model. Theoretical Computer Science, 133, 3 – 14, 1994.
5. I.M. Gelfand, M.M. Kapranov, A.V. Zelevinsky: Discriminants, resultants, and multidimensional determimants Birkhäuser , 1994.
6. E. Grädel, K. Meer: Descriptive Complexity Theory over the Real Numbers. In: Lectures in Applied Mathematics, J. Renegar, M. Shub, S. Smale (eds.), 32, 381–403, 1996.
7. S.Å. Gustafson, K.O. Kortanek: Semi-infinte programming and applications. In: Mathematical Programming: The State of the Art, A. Bachem, M. Grötschel, B. Korte, eds., Springer, 132 – 157, 1983.
8. R. Hettich, H.Th. Jongen: Semi-infinite Programming: conditions of optimality and applications. In: Optimization Techniques, J. Stoer , ed., Part 2, Lecture Notes in Control and Inform. Sci., 7, Springer, 1 – 11, 1978.
9. R. Hettich, K.O. Kortanek: Semi-infinite Programming.SIAM Review, vol.35, no.3, 380 – 429, 1993.
10. H.Th. Jongen, F. Twilt, G.W. Weber: Semi-infinite optimization: structure and stability of the feasible set. Journal Opt. Theory Applications 72, 529 – 552, 1992.
11. P. Koiran: A weak version of the Blum-Shub-Smale model. In 34^{th} Annual IEEE Symposium on Foundations of Computer Science, 486 – 495, 1993.
12. V. Kreinovich, A.V. Lakeyev, J. Rohn, P. Kahl: Computational Complexity and Feasibility of Data Processing and Interval Computations. Kluwer, 1997.
13. M. Koshelev, L. Longpré, P. Taillibert: Optimal Enclusure of Quadratic Interval Functions. Reliable Computing 4, 351 – 360, 1998.
14. A.V. Lakeyev, S.I. Noskov: A description of the set of solutions of a linear equation with interval defined operator and right-hand side. Russain acad. Sci. Dokl. Math., Vol. 47, No. 3, 518 – 523, 1993.
15. K. Meer: On the complexity of quadratic programming in real number models of computation. Theoretical Computer Science 133, 85 – 94, 1994.

16. D.A. Plaisted: New NP-hard and NP-complete polynomial and integer divisibility problems. Theoretical Computer Science 31, 125 – 138, 1984.
17. M. Shub: Some remarks on Bezout's theorem and complexity theory. In: M. Hirsch, J. Marsden, M. Shub (eds.), Form topology to computation: Proc. of the Smalefest, Springer, 443 – 455, 1993.
18. L. Stockmeyer: The polynomial-time hierarchy. Theoretical Computer Science, 3, 1 – 22, 1977.
19. B. Sturmfels: Introduction to resultants. In: Application of Computational Algebraic Geometry, D.A. Cox B. Sturmfels (eds.), Proc. of Symposia in Applied Mathematics, Vol. 53, American Mathematical Society, 25 – 39, 1998