On the Approximation of Interval Functions

Klaus Meer*

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

Abstract. Many problems in interval arithmetic in a natural way lead to a quantifier elimination problem over the reals. By studying closer the precise form of the latter we show that in some situations it is possible to obtain a refined complexity analysis of the problem. This is done by structural considerations of the special form of the quantifiers and its implications for the analysis in a real number model of computation. Both can then be used to obtain as well new results in the Turing model. We exemplify our approach by dealing with different versions of the approximation problem for interval functions.

1 Introduction

When studying a problem in interval arithmetics one of the basic assumptions is that the input data is not known accurately. Instead of dealing with say an input sequence $(x_1, \dots, x_n)$ of rationals describing our precise data, we consider a sequence of intervals $[\underline{x}_1, \overline{x}_1], \dots, [\underline{x}_n, \overline{x}_n]$ such that the actual data points x_i are only known to belong to $[\underline{x}_i, \overline{x}_i], 1 \le i \le n$.

Example 1. (see [11]) Consider the problem: Given a matrix $A \in \mathbb{Q}^{m \times n}, b \in \mathbb{Q}^m$, is there a vector $x \in \mathbb{R}^n$ such that $Ax = b$? As a problem in interval arithmetic we should start with an *interval matrix* $\mathfrak{A}$, i.e. a set of intervals $[\underline{a}_{ij}, \overline{a}_{ij}] \subseteq \mathbb{R}$ for each position (i, j) and an *interval vector* $\mathfrak{b}$ of intervals $[\underline{b}_i, \overline{b}_i]$ as components. The only knowledge about the original data (A, b) then is the information $A \in \mathfrak{A}$, i.e. $\forall i, j\ a_{ij} \in [\underline{a}_{ij}, \overline{a}_{ij}]$ and similarly for $b \in \mathfrak{b}$. A typical question then could be: Is there any choice $A \in \mathfrak{A}, b \in \mathfrak{b}$ such that there is an $x \in \mathbb{R}^n$ with $Ax = b$?

The above interval linear systems problem is known to be NP-hard [8]. From an informal point of view it is easy to get an intuition why this is the case. The interval framework "automatically" introduces a linear number (in the problem dimension) of quantifiers ranging over the *real* numbers; they are used to describe the interval information available about the input data:

$$\exists\, A \in \mathfrak{A}\ \exists\, b \in \mathfrak{b}\ \exists\, x \in \mathbb{R}^n\ Ax = b\ .$$

* Partially supported by the EU Network of Excellence PASCAL Pattern Analysis, Statistical Modelling and Computational Learning and by the Danish Natural Science Research Council SNF. This publication only reflects the author's views.

The first two blocks of existential quantifiers are not present in the exact setting and increase the problem's complexity. They naturally lead to the consideration of *algebraic* models of computation like the BSS model introduced by Blum, Shub, and Smale, [2].

In real-life computers, operations with real numbers are hardware supported. Almost universally, floating-point arithmetic is used for scientific computation. Therefore, if we have a certain number of bit operations as part of a single operation with real numbers, then these operations will be performed much faster than if they were unrelated operations with bits. From this viewpoint, in addition to the standard Turing complexity, it is desirable to investigate the complexity in terms of how many operations with real numbers are necessary. This cost measure is studied in the real number BSS model of computation (resp. in algebraic complexity theory). The approach is substantiated by the fact that (see [17]) "most practioneers of scientific computing do not experience much difference between the real number model and floating point arithmetic." Of course, there are situations in which a more careful analysis has to be performed. However, for numerically stable algorithms the real number model seems to reflect pretty well the floating-point costs. Moreover, as already mentioned, in the framework of interval arithmetic the real number model comes in quite naturally already through the formulation of problems.

Note as well that in the BSS model the problem of deciding general existential formulas in the first-order theory $\mathrm{FO}_{\mathbb{R}}$ over the reals is $\mathrm{NP}_{\mathbb{R}}$-complete, where $\mathrm{NP}_{\mathbb{R}}$ denotes a real analogue of NP. For more details see [2].

Example 2. This example deals with the best linear approximation of quadratic interval functions (BLA for short, see [10,8]). A problem instance is given by a box $B = [\underline{b}_1, \overline{b}_1] \times \ldots \times [\underline{b}_n, \overline{b}_n] \subseteq \mathbb{R}^n$, a bound $M \in \mathbb{R}^n$ and a quadratic interval function $f : B \mapsto \{\text{intervals in } \mathbb{R}\}$. The latter means that there are two quadratic polynomials $\underline{f}, \overline{f} : B \mapsto \mathbb{R}$ such that $\forall\, y \in B\ \underline{f}(y) \le \overline{f}(y)$ and the only information we have about f is that $\forall\, y \in B\ f(y) \in [\underline{f}(y), \overline{f}(y)]$. The task is to approximate f as best as possible with respect to the maximum norm on B by two linear functions $\underline{X}, \overline{X} : B \to \mathbb{R}$, i.e. to compute

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

under the constraints

$$\forall\, y \in B : \quad \begin{array}{l} \overline{X}(y) \ge \overline{f}(y) \\ \underline{X}(y) \le \underline{f}(y) \end{array}$$

As decision problem we ask whether the minimal value is at most M. This problem is a linear semi-infinite optimization problem that was studied in [10]. There, it was shown that the problem (when restricted to rational input data) is NP-hard. However, no upper bound is known, i.e. the problem is not known to belong to a certain level of the polynomial hierarchy. It is neither known what the complexity for a fixed input dimension n is (Koshelev et al. [10] showed polynomial solvability for $n = 1$.)

For our approach it is most interesting to see that the BLA problem logically can be expressed as a $\Sigma_2^{\mathbb{R}}$ formula in the first order theory over the reals. The first block of existential quantifiers asks for the linear functions $\underline{X}, \overline{X}$, the second block of universal quantifiers checks the constraints. The entire formula thus has the shape

$$\exists\, \underline{X}, \overline{X} \in \mathbb{R}^n \; \forall\, y \in B \; \Phi(\underline{X}, \overline{X}, y) \; ,$$

where Φ is a quantifier free $\mathrm{FO}_{\mathbb{R}}$ formula. However, note that this does not automatically imply the problem to belong to Σ_2 in the classical polynomial hierarchy when inputs are restricted to be rational numbers.

In this extended abstract we show how a further analysis of the logical structure of the above formula results in refined and even new complexity results concerning the BLA problem, both in the Turing and the BSS model. For full details on this part see [13]. We then also discuss more generally the limits of our approach.

2 Framework and Techniques

We shall use Example 2 to clarify our general ideas. Given the real quantifiers naturally involved in many such interval problems we start from real number complexity theory as described in [2]. In this framework we perform an analysis that allows us as well to obtain new and to refine known results for the problem's complexity in the Turing model. In particular, we show:

- many of the NP-hard interval problems do likely not capture the full complexity of the analogue class $\mathrm{NP}_{\mathbb{R}}$;
- for expressing many such problems the full power of real quantifier elimination is not necessary;
- a new upper bound for the BLA problem in the Turing model: BLA $\in \Sigma_2$;
- a new fixed parameter result: BLA is polynomial time solvable in the Turing model for fixed variable dimension $n \in \mathbb{N}$.

The proofs of the above results are divided into two main parts. In the first we analyse consequences the logical description of a problem has with respect to its real number complexity. In the second we check the particular problem we are interested in with respect to the special logical form it can be expressed in. Whereas the first part is using structural complexity theory in the BSS model, the second part is depending very much on the problem under consideration. In case of the BLA problem, the Reduction Ansatz from semi-infinite optimization together with arguments from [12] are crucial.

2.1 Digital Quantifiers

Similar to the class NP in the BSS model the real analogue $\mathrm{NP}_{\mathbb{R}}$ can be characterized as all real number decision problems A such that there is a (real) decision problem $B \in \mathrm{P}_{\mathbb{R}}$ together with a polynomial p such that

$$A = \{x \in \mathbb{R}^n | \exists\, y \in \mathbb{R}^m \; m \le p(n) \text{ and } (x, y) \in B \text{ for some } n, m\} \; .$$

The class $\mathrm{DNP}_{\mathbb{R}}$ is the subclass of problems for which in the above characterization it is sufficient to let the existential quantifiers range over some $\{0,1\}^m$, only, see [3].

Thus, a problem is in $\mathrm{DNP}_{\mathbb{R}}$ if the verification procedure can be performed using a proof from a finite search space instead of one from an uncountable set $\mathbb{R}^m$. Clearly, it is $\mathrm{P}_{\mathbb{R}} \subseteq \mathrm{DNP}_{\mathbb{R}} \subseteq \mathrm{NP}_{\mathbb{R}}$. Similarly for the real version $\Sigma_2^{\mathbb{R}}$ of Σ_2 (and for all the other levels of the real polynomial time hierarchy) we can define the digital subclass $\mathrm{D}\Sigma_2^{\mathbb{R}}$. Though we cannot expect to be able to prove $\mathrm{DNP}_{\mathbb{R}} \neq \mathrm{NP}_{\mathbb{R}}$ or $\mathrm{D}\Sigma_2^{\mathbb{R}} \neq \Sigma_2^{\mathbb{R}}$ it is possible to substantiate these conjectures by the theorem below. Before stating it let us shortly explain two notions used in the theorem.

Weak reductions in the BSS model were introduced by Koiran [7] as a way to penalize computations that produce high degree polynomials by repeated squaring. For example, under the weak cost measure performing n times a repeated squaring of an input $x \in \mathbb{R}$ has exponential cost since the polynomial generated when x is seen as a variable has degree 2^n. For a real number complexity class $\mathcal{C}$ we denote by the superscript $\mathcal{C}^w$ the corresponding real number complexity classes when implicit complexity statements refer to the weak cost measure. For example, $\mathrm{DNP}_{\mathbb{R}}^w$ is the class of problems verifiable in weak polynomial time by guessing a sequence in $\{0,1\}^\infty$. Similarly, $\mathrm{D}\Sigma_2^{\mathbb{R},w}$ is the weak version of $\mathrm{D}\Sigma_2^{\mathbb{R}}$.

The family of (particular) resultant polynomials RES_n is known from algebraic geometry, see f.e. [5] and [16]:

Definition 1. *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 n homogeneous degree two polynomials. It is the unique (up to sign) irreducible polynomial with integer coefficients that generates the variety of (coefficient vectors of) systems that have a non-trivial complex zero. The set of these systems is denoted by $\mathcal{H}$, the solvable ones by $\mathcal{H}_0$.*

Resultants are conjectured not to be efficiently computable, see [14] for some hardness results and [5] for the general theory of resultants.

Theorem 1. *a) No problem in* $\mathrm{DNP}_{\mathbb{R}}^w$ *is* $\mathrm{NP}_{\mathbb{R}}$*-complete under weak polynomial time reductions. No problem in $D\Sigma_2^{\mathbb{R},w}$ is* $\mathrm{NP}_{\mathbb{R}}$*-hard under weak polynomial time reductions.*
b) Suppose there is no *(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* no *problem in* $\mathrm{DNP}_{\mathbb{R}}$ *is* $\mathrm{NP}_{\mathbb{R}}$*-complete and* <u>*no*</u> *problem in $D\Sigma_2^{\mathbb{R}}$ is* $\mathrm{NP}_{\mathbb{R}}$*-hard under polynomial time reductions in the BSS model.*
c) Part b) also holds with respect to Turing instead of many-one reductions. It is as well true for computational (instead of decision) problems that can be solved in polynomial (real) time using oracle calls to a $DNP_{\mathbb{R}}$-oracle.

Proof. This is done by generalizing related results in [4] and [12]. Part a) follows almost straightforwardly from those results. For part b) assume the hardness of the problems under consideration. Now study the following real number decision problem: Given a system f of n homogeneous real polynomials of degree 2, is there a common non-trivial (complex) zero. This problem clearly belongs to $\mathrm{NP}_{\mathbb{R}}$ and thus, under the assumption that the theorem is false, can be polynomially reduced to a $\mathrm{D}\Sigma_2^{\mathbb{R}}$ problem (respectively one in $\mathrm{DNP}_{\mathbb{R}}$). This potential reduction is now combined with the Dubois-Risler real Nullstellensatz. It is then shown that the Nullstellensatz implies the claimed efficient computation of a non-zero multiple of RES_n to be part of the reduction algorithm. Part c) follows directly from the proof of part b): Also from a potential Turing reduction the desired algorithm for computing the resultant polynomials (modulo the cited conditions) can be designed. □

Since the computation of RES_n is conjectured to be difficult, the above theorem informally can be phrased as: A problem in DNP$_{\mathbb{R}}$ or in D$\Sigma_2^{\mathbb{R}}$ is NP$_{\mathbb{R}}$-hard only if the potentially difficult problem of computing RES_n is efficiently solvable (modulo the above conditions). We thus take a proof that a $\Sigma_2^{\mathbb{R}}$-problem actually belongs to D$\Sigma_2^{\mathbb{R}}$ as indication for it not to be NP$_{\mathbb{R}}$-hard. Concerning weak classes and reductions the statements are absolute; note that for the weak versions of the real number complexity classes P$_{\mathbb{R}}$ and NP$_{\mathbb{R}}$ their inequality was proven in [4].

2.2 BLA Lies in D$\Sigma_2^{\mathbb{R}}$

For a concrete problem in $\Sigma_2^{\mathbb{R}}$ it might of course be unclear whether it belongs to D$\Sigma_2^{\mathbb{R}}$ and how to prove it. In case of the BLA problem this can be achieved by combining techniques from semi-infinite optimization and structural results from classical quadratic programming. The proof will not only establish BLA $\in$ D$\Sigma_2^{\mathbb{R}}$ but allows as well to conclude new results for BLA in the Turing model.

Theorem 2. *a) The* BLA *problem with real input data lies in*

$$D\Sigma_2^{\mathbb{R}} \; (actually \; in \; D\Sigma_2^{\mathbb{R},w}).$$

b) The BLA *problem with rational input data lies in* Σ_2.
c) For fixed variable number $n \geq 1$ *both in the BSS and in the Turing model* BLA *is solvable in polynomial time.*

Proof. The main work of the proof is to show part a). The overall idea is to find a solution $(\underline{X}, \overline{X})$ and verify its correctness as linear approximation as well as verify whether this approximation realizes the demanded bound M of the maximum norm of $\overline{X} - \underline{X}$ on B. This program is performed along the following steps

1) Assuming $(\underline{X}, \overline{X})$ to be known we guess certain discrete information which then is used to compute at least two points $y^{(i)}$ and $y^{(j)}$ satisfying necessary optimality conditions resulting from the Reduction Ansatz of semi-infinite optimization, see [6]. This reduces the infinitely many $y \in B$ to finitely many. However, these finitely many points still are real, i.e. this step does not yet remove the real quantifiers.
2) From step 1) we deduce conditions that have to be fulfilled by an optimal solution $(\underline{X}, \overline{X})$. These conditions lead to a linear programming problem. The solution of the latter, using additional work, can be shown to be computable by a digital $\Sigma_1^{\mathbb{R}}$ algorithm.
3) Finally, the candidate obtained in 2) is checked for optimality. This mainly requires to check the constraints, which result in two quadratic programs in y representing the lower level of the initial semi-infinite problem. Using the results of [12] this problem belongs to class *co*-DNP$_{\mathbb{R}}$ $= D\Pi_{\mathbb{R}}^1$. Together, we obtain a D$\Sigma_2^{\mathbb{R}}$ algorithm.

We remark that the above analysis actually implies all computations to be executable in the corresponding weak classes as well.
b) The precise analysis of the proof of part a) shows that in case of rational input data the numbers of 0's and 1's quantified in the verification procedure are only chosen to select certain positions in the initial matrices and vectors giving the problem coefficients. They do neither depend on intermediately generated rationals of larger bit-size nor on newly introduced real constants. It follows membership in Σ_2 for rational inputs.
c) Similarly, the proof of a) shows 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 bound variables. Finally, a property in complexity class P has to be checked. $\square$

Part a) above allows to conclude that BLA is unlikely to be $\text{NP}_{\mathbb{R}}$-hard under full and is not $\text{NP}_{\mathbb{R}}$-hard under weak polynomial reductions. Part b) gives a new upper bound for the rational version extending the NP-hardness result in [10]. Finally, part c) answers an open question of [8].

3 How Far Does the Approach Lead?

A huge number of NP-hard problems in interval arithmetics is studied in [8]. One might ask in how far the above methods will work at least for some of those problems as well. It should be clear that there are a lot of interval problems not only NP-hard in the Turing model, but also in the real number framework. If we start from an algebraic problem that already in its exact form is $\text{NP}_{\mathbb{R}}$-hard (when real inputs are considered), then its interval version maintains this property as well.

Let us clarify this by some examples. The problems of deciding the existence of a common real root for a system of quadratic (real) polynomials or existence of a real root of a single (real) degree 4 polynomial are well known to be $\text{NP}_{\mathbb{R}}$-complete [2]. Thus, as soon as one of these problems is involved in the exact formulation the interval version is at least as hard and the above techniques do not apply. An extremely important question is where the boundary is. For the Best Linear Approximation problem of Example 2 the decisive properties are

- the Reduction Ansatz and
- the structural properties of (non-convex) quadratic programming allowing for a discrete coding of optimal points.

It is unclear whether, in particular, the techniques related to the second property can be extended to higher degree polynomials as objective functions. The above result is based on the fact that quadratic programs (i.e. a quadratic polynomial as objective function together with linear constraints) attain their infimum if they are bounded from below. This results in the possibility to reduce the problem to linear programs by using necessary first order optimality conditions. And the Linear Programming decision problem belongs to class $\text{DNP}_{\mathbb{R}}$.

If we consider degree 4 polynomials as objective functions everything is different. The related approximation problem *is* $\text{NP}_{\mathbb{R}}$-hard.

Before we prove this result we state the

Theorem 3. *Given $n \in \mathbb{N}$, a compact box $B \subset \mathbb{R}^n$ and a degree 4 polynomial $f \in \mathbb{R}[x_1, \dots, x_n]$ that satisfies $f(x) \geq 0$ for all $x \in B$, it is NP$_\mathbb{R}$-complete to decide whether f has a real zero in B. We denote this problem by BOX-4-FEAS.* □

Due to space limitations we postpone the proof of this result to a future paper [9]. It is basically a straightforward though tedious application of the known quantifier elimination algorithms and their complexity bounds as given, for example, in [1].

Theorem 4. *Consider the problem of finding the best linear approximation of an interval function of degree 4 (cf. Example 2): Given a (compact) box $B \subset \mathbb{R}^n$, a quartic interval function $f : B \mapsto \{$intervals in $\mathbb{R}\}$, i.e. there are two degree 4 polynomials $\underline{f}, \overline{f} : B \mapsto \mathbb{R}$ such that $\forall\, y \in B\ \underline{f}(y) \ \leq\ f(y)\ \leq\ \overline{f}(y)$. Find two linear functions $\underline{X}, \overline{X} : B \to \mathbb{R}$ that solve the constrained optimization problem*

$$\min_{\underline{X},\overline{X}} \max_{y \in B} \overline{X}(y) - \underline{X}(y)$$

under the constraints

$$\forall\, y \in B : \qquad \begin{array}{l} \overline{X}(y) \ \geq\ \overline{f}(y) \\ \underline{X}(y) \ \leq\ \underline{f}(y). \end{array}$$

Then this problem is NP$_\mathbb{R}$-hard.

Similarly, for the decision version it holds: Given, in addition to f, n and B, a bound $M \in \mathbb{R}$, does the best linear approximation $(\overline{X}, \underline{X})$ satisfy

$$\max_{x \in B} |\overline{X}(x) - \underline{X}(x)| \ \leq\ M\ ?$$

is NP$_\mathbb{R}$-complete.

Proof. We reduce the BOX-4-FEAS problem to the decision problem under consideration. Given n, f and B as input for the former it is easy to compute an upper bound $M \in \mathbb{R}$ such that $f(x) \leq M$ for all $x \in B$. Just plug in the largest possible value B for all x_i and sum up the absolute values of all monomials occuring in f. This bound clearly is computable in polynomial time.

As input for the approximation problem we now choose

$$\overline{f}(x) := M\ \forall\, x \in B\ ,\ \underline{f}(x) := f(x)\ \forall\, x \in B\ .$$

Since $\overline{f}$ is a constant function it follows that the optimal choice for $\overline{X}$ is as well the constant function M. As continuous function $f = \underline{f}$ attains its infimum on the compact box B, say in a point $x^* : f(x^*) = \min\limits_{x \in B} f(x)$. By assumption, $f(x) \geq 0$ for all choices of x. Let $\underline{X}$ be an optimal choice (there might be several) for the approximation problem. It is clear that

$$\max_{x \in B} \overline{X}(x) - \underline{X}(x) \ =\ \max_{x \in B} M - \underline{X}(x) \ \leq\ M,$$

since $\underline{X}(x) := 0$ is a suitable candidate for computing $\min\limits_{\underline{X}} \max\limits_{x \in B} M - \underline{X}(x)$. In the latter inequality we actually get equality if and only if f has a real zero in B. Otherwise, the choice $\underline{X}(x) := f(x^*) > 0$ gives a better (the optimal!) result. □

The theorem shows that if we share the general believe that $\text{DNP}_{\mathbb{R}} \neq \text{NP}_{\mathbb{R}}$, then a limit for our techniques is reached with the linear approximation problem for interval functions of degree 4.

Open Problem: What's about the real number complexity of the best linear approximation problem for (polynomial) interval functions of degree 3? In particular: Does the decision version belong to $\text{DNP}_{\mathbb{R}}$?

The above arguments are closely related to the range problem of interval functions. This issue will be discussed in more detail in [9].

For many other problems related to *linear* interval systems (for a collection of such problem see Chapters 11 and 12 in [8]) it seems that our results can be applied. They yield some more optimistic upper bounds than what could be feared from solving them with general quantifier elimination methods. It might be fruitful to analyze further such problems under this point of view.

We give one such result here. It deals with the computation of a solution interval of an interval linear system. As in Example 1 let $\mathfrak{A} := [\underline{A}, \overline{A}]$ be an interval matrix and $\mathfrak{b} := [\underline{b}, \overline{b}]$ an interval vector in $\mathbb{R}^m$.

Definition 2. *(see [8]) a) A* possible solution *of the interval linear system $\mathfrak{A} \cdot x = \mathfrak{b}$ is a vector $y \in \mathbb{R}^n$ such that there exist $A \in \mathfrak{A}, b \in \mathfrak{b}$ for which $A \cdot y = b$.*
*b) By the j-*th solution interval $\mathbf{y}_j = [\underline{y}_j, \overline{y}_j]$ *of the system we mean a (possibly infinite) interval such that for all possible solutions y of the system the j-th component y_j of y belongs to $[\underline{y}_j, \overline{y}_j]$. Moreover, the values $\underline{y}_j, \overline{y}_j$ do occur as components of certain possible solutions (i.e. they are the minimal and maximal such components).*

We consider the problem of computing the solution intervals of an interval linear system. It was shown by Rohn [15] that

Theorem 5. *(Rohn) The problem of computing the $\mathbf{y}_j$'s (or even computing what is called a* possibly finite enclosure*) is NP-hard in the Turing model.*

Here, we show that in the real number framework Theorem 2, part c) applies, i.e. the problem likely does not share the full difficulty of class $\text{NP}_{\mathbb{R}}$.

Theorem 6. *The problem to compute the solution intervals of an interval linear system can be solved in $FP_{\mathbb{R}}^{DNP_{\mathbb{R}}}$, i.e. by a polynomial time real number oracle algorithm that has access to an oracle for problems in $DNP_{\mathbb{R}}$. Thus, Theorem 2, part c) applies.*

Proof. Let $\mathfrak{A}, \mathfrak{b}$ be the data of an interval linear system. We use the following result from [11] to characterize the set $\mathcal{R}$ of possible solutions of $\mathfrak{A} \cdot x = \mathfrak{b}$:

$$\mathcal{R} = \{y^{(1)} - y^{(2)} | y^{(1)} \geq 0,\ y^{(2)} \geq 0,\ {y^{(1)}}^T \cdot y^{(2)} = 0,$$
$$\underline{A} \cdot y^{(1)} - \overline{A} \cdot y^{(2)} \leq \overline{b},\ \overline{A} \cdot y^{(1)} - \underline{A} \cdot y^{(2)} \geq \underline{b}\}.$$

Clearly, for $1 \le j \le n$ the endpoints $\underline{y}_j, \overline{y}_j$ of the solution interval $\mathbf{y}_j$ are given as solutions of the following optimization problems: $\underline{y}_j$ is the solution of

$$\begin{aligned}
&\min_{y^{(1)},y^{(2)} \in \mathbb{R}^n} \; y_j^{(1)} - y_j^{(2)} \text{ subject to the constraints}\\
&y^{(1)} \ge 0,\; y^{(2)} \ge 0,\; {y^{(1)}}^T \cdot y^{(2)} = 0,\\
&\underline{A} \cdot y^{(1)} - \overline{A} \cdot y^{(2)} \le \overline{b},\; \overline{A} \cdot y^{(1)} - \underline{A} \cdot y^{(2)} \ge \underline{b}\,.
\end{aligned}$$

The solution of the corresponding maximization problem is $\overline{y}_j$.

Given the previous results we now can easily design an $\mathrm{FP}_{\mathbb{R}}$ algorithm for computing the solutions of these optimization problems. First, we use the $\mathrm{DNP}_{\mathbb{R}}$ oracle to remove the complementary slackness condition ${y^{(1)}}^T \cdot y^{(2)} = 0$ by figuring out which components of the two vectors are zero in an optimal solution. This clearly is a $\mathrm{DNP}_{\mathbb{R}}$ problem since we only have to code the component indices. We plug in the corresponding zero values into the problem. This turns it into a Linear Programming problem. As mentioned before, a solution of a Linear Programming problem (with real data) can be computed using a $\mathrm{DNP}_{\mathbb{R}}$ oracle, thus finishing the proof. □

Let us finally mention that, once more using the characterizations given in [11], similar results hold as well for the linear interval systems problem from Example 1 and some related variants, see [13].

4 Conclusions

We have seen that some problems in interval arithmetic in a very natural manner give rise to study the problem in the BSS model. A concise analysis concerning the logical expressibility of a problem then allows to obtain structural complexity results not only in the BSS model but also in the Turing model. The latter can refine investigations that do not go further than stating NP-hardness of a problem as soon as real quantifiers come into play.

Acknowledgement

I want to thank G.W. Weber for helpful discussions on semi-infinite programming. Thanks are due to V. Kreinovich and two anonymous referees for useful comments concerning the preparation of this paper.

References

1. S. Basu, R. Pollack, M.F. Roy. On the combinatorial and algebraic complexity of quantifier elimination. *Journal of the ACM*, 43(6), 1002–1045, 1996.
2. L. Blum, F. Cucker, M. Shub, S. Smale. Complexity and Real Computation. 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. S.Å. Gustafson, K.O. Kortanek. Semi-infinte programming and applications. *Mathematical Programming: The State of the Art*, A. Bachem, M. Grötschel, B. Korte, eds., Springer, 132–157, 1983.
7. P. Koiran. A weak version of the Blum-Shub-Smale model. 34^{th} *Annual IEEE Symposium on Foundations of Computer Science*, 486–495, 1993.
8. V. Kreinovich, A.V. Lakeyev, J. Rohn, P. Kahl. Computational Complexity and Feasibility of Data Processing and Interval Computations. Kluwer, 1997.
9. V. Kreinovich, K. Meer. Complexity results for the range problem in interval arithmetic. In preparation.
10. M. Koshelev, L. Longpré, P. Taillibert. Optimal Enclusure of Quadratic Interval Functions. *Reliable Computing*, 4, 351–360, 1998.
11. 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. *Russian Acad. Sci. Dokl. Math.*, 47(3), 518–523, 1993.
12. K. Meer. On the complexity of quadratic programming in real number models of computation. *Theoretical Computer Science*, 133, 85–94, 1994.
13. K. Meer. On a refined analysis of some problems in interval arithmetic using real number complexity theory. *Reliable Computing*, 10(3), 209–225, 2004.
14. D.A. Plaisted. New NP-hard and NP-complete polynomial and integer divisibility problems. *Theoretical Computer Science*, 31, 125–138, 1984.
15. J. Rohn. Enclosing solutions of linear interval equations is NP-hard. *Computing*, 53, 365–368, 1094.
16. B. Sturmfels. Introduction to resultants. *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
17. H. Woźniakowski: Why does information-based complexity use the real number model? Theoretical Computer Science 219, 451 – 465, 1999.