

# Are There Efficient Necessary and Sufficient Conditions for Straightforward Interval Computations To Be Exact?

Vladik Kreinovich<sup>1</sup>, Luc Longpré<sup>1</sup>, and James J. Buckley<sup>2</sup>

<sup>1</sup>Department of Computer Science, U. of Texas at El Paso  
El Paso, TX 79968, USA, {vladik,longpre}@cs.utep.edu

<sup>2</sup>Mathematics Department, U. of Alabama at Birmingham  
Birmingham, AL 35294-1170, USA, buckley@math.uab.edu

## Abstract

We prove that no efficient necessary and sufficient conditions are possible for checking whether straightforward interval computations lead to the exact result.

**One of the main problems of interval computations.** One of the main problems of interval computations is to find a range of a given function on given intervals. To be more precise, the problem is: given  $n$  input intervals  $\mathbf{x}_1, \dots, \mathbf{x}_n$  and an algorithm  $f(x_1, \dots, x_n)$  that transforms  $n$  real numbers  $x_1, \dots, x_n$  into a real number  $y = f(x_1, \dots, x_n)$ , find the range

$$\mathbf{y} = f(\mathbf{x}_1, \dots, \mathbf{x}_n) = \{f(x_1, \dots, x_n) \mid x_1 \in \mathbf{x}_1, \dots, x_n \in \mathbf{x}_n\}.$$

Usually, the endpoints of the intervals  $\mathbf{x}_i$  come from measurements, and measurement usually produces rational numbers, so we can assume that the intervals  $\mathbf{x}_i$  have rational endpoints. If we cannot compute the exact range, we can at least try to find an enclosure  $\mathbf{Y} \supseteq \mathbf{y}$  for the range.

**Straightforward interval computations: its advantages and drawbacks.** Historically the first method for computing the enclosure for the range is the method which is sometimes called “straightforward” interval computations. This method is based on the fact that inside the computer, every algorithm consists of elementary operations (arithmetic operations, min, max, etc.). For each elementary operation  $f(x, y)$ , if we know the intervals  $\mathbf{x}$  and  $\mathbf{y}$  for  $x$  and  $y$ , we can compute the exact range  $f(\mathbf{x}, \mathbf{y})$ . The corresponding formulas form the so-called *interval arithmetic*. In straightforward interval computations, we repeat the computations forming the program  $f$  step-by-step, replacing each operation

with real numbers by the corresponding operation of interval arithmetic. It is known that, as a result, we get an enclosure for the desired range.

In some important cases, the enclosure obtained by using straightforward interval computations is actually the exact range. There are several sufficient conditions for straightforward interval computations to be exact: e.g., it is exact when  $f(x_1, \dots, x_n)$  is an explicit expression in which each variable occurs only once; another condition is given in [4].

However, there are known cases when the resulting enclosure is much larger than the actual range. For example, for the expression  $f(x_1, x_2) = x_1 + x_1 \cdot x_2$ , straightforward interval computations are exact when  $\underline{x}_2 \geq 0$  and not exact when, e.g.,  $\mathbf{x}_1 = [\underline{x}_1, \bar{x}_1]$  is a non-degenerate interval and  $\mathbf{x}_2 = [-1, -1]$ . Indeed, in the second case,  $f(x_1, x_2) = 0$ , so we have a 1-point range  $[0, 0]$ , but straightforward interval computations result in  $[\underline{x}_1 - \bar{x}_1, \bar{x}_1 - \underline{x}_1]$ .

**More sophisticated methods and the first methodological question.**

Several methods have been proposed to reduce the overestimation: centered form, bisection, monotonicity check, etc. Some methods – like Hansen’s generalized interval arithmetic [3] – decrease the overestimation by taking into account dependence between interval variables; in these methods, the range of  $x_1 + x_1 \cdot (-1)$  is correctly computed as  $[0, 0]$ .

Each new method improves the enclosures, often reducing the enclosure to the exact range, but for each known method, there are cases when this method still overestimates.

In such situations, when many methods have been proposed and none of them is perfect, a natural question is: *Is a perfect method* – that would always return the exact range in reasonable time – *possible at all?* This methodological question is important for algorithm designers:

- If a perfect method is possible, then it is reasonable to spend some time looking for it.
- On the other hand, if such a method is not possible at all, then looking for a perfect method would be a waste of time – like looking for a solution-in-radicals of a general fifth order algebraic equation or for a ruler-and-compass angle trisection.

If no general perfect method is possible, then, instead of wasting time looking for such a method, we should look either for *classes* of functions and/or domains for which it is possible to compute the exact range, or for algorithms that still overestimate, but produce *better* estimates than the existing ones.

**A (known) answer to the first methodological question.** For interval computations, this important methodological question was answered in 1981, when Gaganov proved [1, 2] that the problem of computing the range is NP-hard (see, e.g., [5] and references therein).

Crudely speaking, NP-hard means that there are no general ways for solving this problem (i.e., computing the exact range) in reasonable time. (As an aside,

it is possible to compute the range exactly in time that increases exponentially with  $n$  [5].) Of course, every NP-hard problem has easier-to-solve subclasses, and the problem of range estimation is no exception: as we have mentioned, there are several important classes of functions for which we can compute the exact range in reasonable time. However, the NP-hardness result means that when we design a general range estimation algorithm, we can, in general, only compute *enclosures* for the desired range.

Maybe the difficulty from the requirement that the range be computed exactly? In practice, it is often sufficient to compute, in a reasonable amount of time, usefully accurate bounds for  $\mathbf{y}$ , i.e., bounds which are accurate within a given accuracy  $\varepsilon > 0$ . Alas, for any  $\varepsilon$ , such computations are also NP-hard.

This NP-hardness is not so bad from a practical viewpoint as it may sound. For example, in [5], we analyzed “in what sense” the computation is NP-hard: with respect to the dimension  $n$  or to the number of operations?

- We showed that the problem remains NP-hard if we only consider quadratic functions of the arbitrary number of variables – so it is, in this sense, “NP-hard with respect to the dimension  $n$ ”.
- However, if we fix the dimension, then it is already possible to have a polynomial-time (feasible) algorithm for exact range estimation – i.e., the problem is *not* “NP-hard with respect to the number of operations”.

Since in many practical problems, dimensionality  $n$  is reasonably small, we have a lot of practical problems for which a feasible algorithm is possible.

**Second methodological question.** When we use an algorithm – e.g., straightforward interval computations – to estimate the range, we know that the result *may* be an overestimation. But is it?

As we have mentioned, there are many important sufficient conditions under which straightforward interval computations produce an exact range. New better sufficient conditions are being discovered. However, none of the known conditions is necessary: for each of these conditions, there are cases not covered by this condition in which the results are nevertheless exact.

Again, we have a natural question: are perfect (i.e., efficient, necessary and sufficient) conditions possible at all? If they are possible, then it is reasonable to spend some time looking for them. If such conditions are not possible, then looking for such perfect conditions would be a useless waste of time.

**Our answer to this question.** Let us consider algorithms  $f(x_1, \dots, x_n)$  that consist only of the operations  $+$ ,  $-$ ,  $\cdot$ ,  $\min$ , and  $\max$ .

**Theorem.** *The problem of checking whether for a given algorithm  $f(x_1, \dots, x_n)$  and given intervals  $\mathbf{x}_1, \dots, \mathbf{x}_n$ , straightforward interval computations are exact, is NP-hard.*

**Proof.** In the proof of Theorem 3.1 from [5], we have shown that a known NP-hard problem – checking satisfiability of a propositional formula  $F$  – can

be reduced to checking whether for an appropriate quadratic polynomial  $f$ , the lower endpoint  $\underline{y}$  of the range  $\mathbf{y}$  is  $\underline{y} = 0$  (if  $F$  is satisfiable) or  $\underline{y} \geq 0.09$  (if  $F$  is not satisfiable). It is easy to check that for all the corresponding polynomials  $f$ ,  $\bar{y} \geq 0.25$ . Thus, for a function  $f_0(x_1, \dots, x_n) = \max(0.04, \min(f(x_1, \dots, x_n), 0.25))$ , the actual range  $\mathbf{y}_0$  is  $[0.04, 0.25]$  if  $F$  is satisfiable and  $[\underline{y}, 0.25]$  otherwise. If we could check whether straightforward interval computations are exact, i.e., whether  $\mathbf{Y}_0 = \mathbf{y}_0$ , we could thus check whether  $F$  is satisfiable:

- If  $\underline{Y}_0 > 0.04$ , this means that  $\underline{y}_0 \geq \underline{Y}_0 > 0$  hence  $F$  is not satisfiable.
- If  $\underline{Y}_0 = 0.04$  and the result of straightforward interval computations is exact, then  $\underline{y}_0 = 0.04$  hence  $F$  is satisfiable.
- If  $\underline{Y}_0 = 0.04$  and the result of straightforward interval computations is not exact, then  $\underline{y}_0 > 0.04$ , and so the propositional formula  $F$  is not satisfiable.

This reduction to a known NP-hard problem proves that our problem is NP-hard as well. The theorem is proven.

*Comment.* A similar result holds if we allow division as well.

**Practical conclusions.** In other words, no feasible necessary and sufficient conditions are possible for checking whether the estimate obtained by using straightforward computations is exact. As a result, instead of trying to find such conditions, we should fully concentrate on identifying classes of functions (or functions and box values) for which straightforward computations lead to the exact range. For example, it is known that for the Gauss elimination algorithm, under certain conditions, we get the exact intervals for the solution; it is also known that completing the square of any quadratic function of one variable can be used to compute its exact range. Finding more cases like that is worth the effort.

*Mathematical comment.* This result easily implies that the exact computation of the range is NP-hard, but we know of no easy way to deduce our new result from the NP-hardness of interval computations. In this sense, our new result is stronger than the known result that computing the range is NP-hard.

**Related open problems.** As we have mentioned, in practice, it is usually sufficient to compute the range within a given accuracy  $\varepsilon$ . How difficult is it to check whether for a given algorithm  $f(x_1, \dots, x_n)$  and given intervals  $\mathbf{x}_1, \dots, \mathbf{x}_n$ , straightforward interval computations are accurate within the given accuracy? We think that this problem is NP-hard, but we could not prove it.

What if we consider other methods – such as centered form? Again, we could not prove it either way.

**Acknowledgments.** This work was supported in part by NASA grants NCC5-209 and NCC 2-1232, by NSF grants CDA-9522207, ERA-0112968 and 9710940 Mexico/Conacyt, and by AFOSR grant F49620-00-1-0365. The authors are thankful to Eldon Hansen, Weldon A. Lodwick, Bill Walster, and to the anonymous referees for fruitful discussions.

## References

- [1] A. A. Gaganov, *Computational complexity of the range of the polynomial in several variables*, Leningrad University, Math. Department, M.S. Thesis, 1981 (in Russian).
- [2] A. A. Gaganov, “Computational complexity of the range of the polynomial in several variables”, *Cybernetics*, 1985, pp. 418–421.
- [3] E. R. Hansen, “A generalized interval arithmetic”, In: K. Nickel (ed.), *Interval mathematics*, Springer Lecture Notes in Computer Science, 1975, Vol. 29, pp. 7–18.
- [4] E. Hansen, “Sharpness in interval computations”, *Reliable Computing*, 1997, Vol. 3, pp. 7–29.
- [5] V. Kreinovich, A. Lakeyev, J. Rohn, and P. Kahl, *Computational complexity and feasibility of data processing and interval computations*, Kluwer, Dordrecht, 1997.