No-Free-Lunch Result for Interval and Fuzzy Computing: When Bounds Are Unusually Good, Their Computation Is Unusually Slow

Martine Ceberio and Vladik Kreinovich
University of Texas, El Paso, Texas 79968, USA
emails mceberio@utep.edu, vladik@utep.edu

1. Abstract

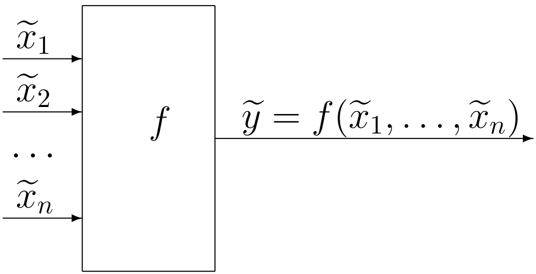
- On several examples from interval and fuzzy computations and from related areas, we show that:
  - when the results of data processing are unusually good,
  - their computation is unusually complex.
- This makes us think that there should be an analog of Heisenberg's uncertainty principle:
  - when we an unusually beneficial situation in terms of results,
  - it is not as perfect in terms of computations leading to these results.
- In short, nothing is perfect.

2. Need for Data Processing

- In science and engineering:
  - we want to *understand* how the world works,
  - we want to *predict* the results of the world processes, and
  - we want to *design* a way to control and change these processes so that the results will be most beneficial.
- Usually, we know the equations that describe how these systems change in time.
- For example, if we want to predict the trajectory of the spaceship, we need to find its current location and velocity.
- Then, we can use Newton's equations to find the future locations of the spaceship.
- Such computations (*data processing*) are the main reason why computers were invented in the first place.

3. Need to Take Input Uncertainty into Account

- In all the data processing tasks:
  - we start with the current and past values $x_1, \ldots, x_n$ of some quantities, and
  - we use a known algorithm $f(x_1, \ldots, x_n)$ to produce the desired result $y = f(x_1, \ldots, x_n)$.
- The values x_i come from measurements which are never absolutely accurate: the results $\widetilde{x}_i$ are $\neq x_i$:



- How do approximation errors $\Delta x_i \stackrel{\text{def}}{=} \widetilde{x}_i - x_i$ affect the resulting error $\Delta y \stackrel{\text{def}}{=} \widetilde{y} - y$?

4. From Probabilistic to Interval Uncertainty

- Manufacturers of the measuring instruments provide bounds Δ_i on the measurement errors: $|\Delta x_i| \le \Delta_i$.
- Often, in addition to these bounds, we also know the *probabilities* of different possible values $\Delta x_i \in [-\Delta_i, \Delta_i]$.
- These probabilities can be found by comparing with a "standard" (much more accurate) instrument.
- However, there are two important situations when we do not know these probabilities:
  - cutting-edge measurements, when there is no more accurate instrument, and
  - cutting-cost manufacturing, when we want to save on this calibration.
- In such cases, after the measurement, all we know about the actual value x_i is that $x_i \in [\widetilde{x}_i - \Delta_i, \widetilde{x}_i + \Delta x_i]$.

5. Interval Computations

- Different values x_i from the corresponding intervals lead, in general, to different values of $y = f(x_1, \ldots, x_n)$.
- We thus need to find the range of possible values of y:

  $\mathbf{y} = [\underline{y}, \overline{y}] = \{f(x_1, \ldots, x_n) : x_1 \in [\underline{x}_1, \overline{x}_1], \ldots, [\underline{x}_n, \overline{x}_n]\}.$
- Computing this range under interval uncertainty is called *interval computations*.
- In general, the problem of computing the exact range $\mathbf{y}$ is NP-hard even for quadratic functions $f(x_1, \ldots, x_n)$.
- This problem is NP-hard even for the sample variance:

$$f(x_1, \ldots, x_n) = \frac{1}{n} \cdot \sum_{i=1}^{n} x_i^2 - \left(\frac{1}{n} \cdot \sum_{i=1}^{n} x_i\right)^2.$$

- Crudely speaking, NP-hard means that no feasible algorithm always computes the exact range.

6. Case of Small Measurement Errors

- In many practical situations, the measurement errors are relatively small.
- We can then safely ignore terms which are quadratic or higher order in terms of these errors:

$$\Delta y = \widetilde{y} - y = f(\widetilde{x}_1, \ldots, \widetilde{x}_n) - f(x_1, \ldots, x_n) =$$

$$f(\widetilde{x}_1, \ldots, \widetilde{x}_n) - f(\widetilde{x}_1 - \Delta x_1, \ldots, \widetilde{x}_n - \Delta x_n) \approx$$

$$\sum_{i=1}^{n} c_i \cdot \Delta_i, \text{ where } c_i = \frac{\partial f}{\partial x_i}.$$

- The largest possible value of Δy is attained when each term $c_i \cdot \Delta x_i$ is the largest for $\Delta x_i \in [-\Delta_i, \Delta_i]$:
  - when $c_i \geq 0$, the function is increasing, so maximum is when $\Delta x_i = \Delta_i$, and equals $c_i \cdot \Delta_i$;
  - when $c_i \leq 0$, the function is decreasing, so maximum is when $\Delta x_i = -\Delta_i$, and equals $c_i \cdot (-\Delta_i)$.

7. Case of Small Measurement Errors (cont-d)

- In both cases $c_i \geq 0$ and $c_i \leq 0$, the largest possible value of i-th term is $|c_i| \cdot \Delta_i$.
- So, the largest value of the sum is $\Delta = \sum\limits_{i=1}^{n} |c_i| \cdot \Delta_i$.
- Derivatives c_i can be computed by numerical differentiation:

$$c_i = \frac{f(\ldots, x_{i-1}, x_i + h, x_{i+1}, \ldots) - f(\ldots, x_{i-1}, x_i, x_{i+1}, \ldots)}{h}.$$

- Thus:
  - if the function $f(x_1, \ldots, x_n)$ is feasible (= computable in polynomial time T_f),
  - then we can compute Δ in time $(n+1) \cdot T_f$ which is also polynomial in n (= feasible).

8. Cases When the Resulting Error Is Unusually Small

- In general, the resulting approximation error Δ is a linear function of the error bounds $\Delta_1, \ldots, \Delta_n$.
- In other words, the resulting approximation error is of the same order as the original bounds Δ_i.
- In this general case, the above technique provide a good estimate for Δ, an estimate
  - with an absolute accuracy of order Δ_i^2 and thus,
  - with a relative accuracy of order Δ_i.
- There are unusually good cases, when all the derivatives $c_i = \dfrac{\partial f}{\partial x_i}$ are equal to 0 at the point $(\widetilde{x}_1, \ldots, \widetilde{x}_n)$.
- In this case, the resulting approximation error is of order $\Delta_i^2 \ll \Delta_i$ – unusually small.

9. Cases When the Resulting Error Is Unusually Small (cont-d)

- When linear terms are 0s, to estimate Δ, we must consider next (quadratic) terms in Taylor expansion:

$$\Delta y = -\frac{1}{2} \cdot \sum_{i=1}^{n} \sum_{j=1}^{n} \frac{\partial^2 f}{\partial x_i \partial x_j} \cdot \Delta x_i \cdot \Delta x_j + \ldots$$

- In such situations, the resulting approximation error is unusually small – proportional to Δ_i^2 instead of Δ_i.
- However, estimating Δ means solving an interval computations problem for a quadratic function $f(x_1, \ldots, x_n)$.
- We have mentioned that this problem is NP-hard; thus:
  - when bounds are unusually small,
  - their computation is an unusually difficult task.

10. Discussion

- The above observation us think that there should be an analog of Heisenberg's uncertainty principle:
  - when we an unusually beneficial situation in terms of results,
  - it is not as perfect in terms of computations leading to these results.
- In short, nothing is perfect.
- Other examples – given below – seem to confirm this conclusion.

11. Need for Fuzzy Computations

- In some cases:
  - in addition to (and/or instead of) measurement results x_i,
  - we have expert estimates for the corresponding quantities.
- These estimates are usually formulated by using words from natural language, like "about 10".
- A natural way to describe such expert estimates is to use fuzzy techniques, i.e., to assign,
  - to each possible value x_i,
  - a degree $\mu_i(x_i)$ to which the expert is confident that this value is possible.
- The corresponding function $\mu_i(x_i)$ is called a *membership function.*

12. Second Case Study: Fuzzy Computations

- *Given:* a memb. function $\mu_i(x_i)$ for each input x_i.
- *Needed:* the fuzzy number Y (membership function) that describes $y = f(x_1, \ldots, x_n)$.
- *Idea:* Zadeh's extension principle:

  $$\mu(y) = \sup\{\min(\mu_1(x_1), \ldots, \mu_n(x_n)) : f(x_1, \ldots, x_n) = y\}.$$

- *Reduction to interval computations:* for α-cuts $\mathcal{X}(\alpha) \stackrel{\text{def}}{=} \{x : \mu(x) \geq \alpha\}$, we have

  $$\mathcal{Y}(\alpha) = \{f(x_1, \ldots, x_n) : x_1 \in \mathcal{X}_1(\alpha), \ldots, x_n \in \mathcal{X}_n(\alpha)\}.$$

- So, fuzzy data processing means repeating interval computations for $\alpha = 0, 0.1, \ldots, 0.9, 1.0$.
- Thus, for fuzzy computations too:
  - when the resulting bounds are unusually good,
  - their computation is unusually difficult.

13. Third Case Study: When Computing Variance under Interval Uncertainty Is NP-Hard

- The variance $V = \frac{1}{n} \cdot \sum\limits_{i=1}^{n} (x_i - E)^2$ describes the average deviation from the mean $E = \frac{1}{n} \cdot \sum\limits_{i=1}^{n} x_i$.
- Variance is the smallest when all values are equal to the mean E: $x_1 = \ldots = x_n = E$.
- So, under interval uncertainty, $\underline{V}$ is the smallest if all intervals have a common point.
- Interestingly, NP-hardness of computing $[\underline{V}, \overline{V}]$ is proven exactly on such an example; thus:
  - when we have an unusually beneficial situation in terms of results,
  - it is not as perfect in terms of computation time.

14. Fourth Case Study: Kolmogorov Complexity

- In many application areas, we need to compress data (e.g., an image).
- The original data can be, in general, described as a string x of symbols.
- What does it mean to compress a sequence?
  - instead of storing the original sequence,
  - we store a compressed data string *and* a program describing how to un-compress the data.
- This pair can be viewed as a single program p which, when run, generates the original string x.
- Thus, the quality of a compression can be described as the length of the shortest program p that generates x.
- This shortest length is called *Kolmogorov complexity* $K(x)$ of the string x: $K(x) \stackrel{\text{def}}{=} \min\{\text{len}(p) : p \text{ generates } x\}$.

15. Kolmogorov Complexity (cont-d)

- The smaller the Kolmogorov complexity $K(x)$, the more we can compress the original sequence x.
- It turns out that, for most strings, the Kolmogorov complexity $K(x)$ is approximately equal to their length.
- These strings are what physicists would call *random*.
- For such strings, $K(x)$ can thus be efficiently computed (at least approximately) as length(x).
- However, there are strings which are not random, strings which can be drastically compressed.
- It turns out that no algorithm for computing $K(x)$ for such strings (even approximately) is possible; so:
  - when situations are unusually good,
  - computations are unusually complex.

16. Acknowledgments

- This work was supported in part:
  - by the National Science Foundation grants HRD-0734825 and DUE-0926721 and
  - by Grant 1 T36 GM078000-01 from the National Institutes of Health.
- The authors thank Didier Dubois for valuable discussions, and anonymous referees for their help.