How Can We Make Reliable Engineering Computing Explainable

Olga Kosheleva[1] and Vladik Kreinovich[2]
Departments of [1] Teacher Education and [2] Computer Science, University of Texas at El Paso,
olgak@utep.edu, vladik@utep.edu

Abstract. Complex computations often contain difficult-to-detect mistakes. This problem is very acute for AI-based computations – where 5% of the corresponding answers are wrong, but this happens in more traditional computations as well. A natural way to detect such mistakes is to supplement the actual computation results with some easy-to-understand explanations. This is what researchers are trying to do for AI for make its results more reliable, and this is what we propose to do for computations in general. In this paper, we illustrate this idea on the example of reliable engineering computing, where it is important not only to get an estimate, but to also inform the user how accurate is the provided estimate.

Keywords: reliable engineering computing, explainable computing, uncertainty propagation

1. Need for Explainable Computations

1.1. This need is well understood for AI computations

Nowadays, a significant portion of computations – and decisions in general – is performed by tools using Artificial Intelligence (AI) techniques – such as neural networks. While in many cases, there tools are spectacularly effective, in up to 5% of the cases they produce wrong results – this is called *hallucination* (and in the recent past, this percentage was even higher, up to 15%).

Human professionals also make mistakes, but if in doubt, we can ask, e.g., a doctor how he/she came up with the current diagnosis and, based on that, either follow his/her advice or ask for a second opinion – while Large Language Models like ChatGPT – and other AI tools – do not provide such an explanation.

To make AI results more reliable, it is therefore desirable to make them explainable. In AI community, eXplainable AI – XAI – is indeed one of the main research directions; see, e.g., (**?**).

1.2. We need explanable computing beyond AI computations

In non-AI computations, mistakes are not that frequent, but they still occur. To minimize the number of such mistakes, it is therefore desirable to make *all* computations explainable – not only AI-related computations.

Computers are not perfect, we are not perfect when we program them, so when we have results of complex computations, they may be wrong. Many mathematicians remember the first computer-based proof of the four-color theorem (that every planary graph can be colored in 4 colors so that no two neighbors have the same color). This proof was too long to be understandable, and eventually,

a mistake was found. This mistake did not invalidate the correctness of the result, but only until an understandable proof appeared, the mathematical community accepted this theorem as being proven.

One of us (VK) had personal experience with how such back-of-the-envelope computations help avoid mistakes. For several semester, he was teaching Introduction to Computing class for Engineering and Science majors. To make the material more interesting and relevant for students, programming examples in such classes are usually formulated not in abstract computational terms, but in terms simulating a real-life problem. Interestingly, physics students made much fewer mistakes than others – since they had an intuition that allowed them to have back-of-the envelope estimations of the desired result. So, when the computer output differed from these estimates, they knew that something was wrong in their program.

Not only physics majors, we all have this ability to some degree. For example, a recent research (Fadelli, 2026; Zenon et al., 2026) has shown that even when we perform simple arithmetic operations, we usually start – even before we finish understanding the problem – to perform some estimates for the desired result – to make sure that what we compute later is not a mistake. For example, if we are asked to add a two-digit number (e.g., 37) with a one-digit number, then before even we hear the second digit of the first number, we estimate that the sum should start with 3 or 4 – so that if later on our computations will get a number starting with a 5, we will know that we have made a mistake.

1.2.1. *The need to make computations explainable is well understood in physics*

The need to supplement computations with understandable explanations is well-known in physics, where such explanations are known as *back-of-the envelope* computations – since to be truly easy-to-understand, these explanations should fit in the back of the envelope. In some cases, this term is taken literally: there is a well-known story how Einstein's wife was having a tour of a large telescope and naively asked what was its main purpose. When the tour guide explained that this telescope is needed to understand the structure of the Universe, she replied that her husband does it on the back on an envelope.

This may be an anecdote, but Einstein was indeed very good in such simplifying computations (and, by the way, he was not as good in more complex computations: in these, he needed help). It has been sad that any student roaming the streets of Goettingen knew mathematics better than Einstein, but he had intuition that allowed him to solve many physics problems better than anyone else. For example, the most famous mathematician of that time David Hilbert – whom in 1900 the world-wide mathematics community asked to come up with the list of most important challenging problems for the 20 century mathematicians – came up with equations of General Relativity in 1916, only two weeks later than Einstein. But even Hilbert did it two weeks earlier than Einstein, still Einstein would have been praised as one of the main authors of this theory: while all Hilbert did was coming up with a complex systems of partial differential equations, Einstein uses his physics intuition to come up with first approximation solutions. These solutions enabled the researchers to experimentally test and confirm this theory in 1919 (as soon as the First World War was over in late 1918 and large-scale experimental science could be resumed).

1.3. The need for explainable computations is especially important for reliable engineering computing

The need to be explainable is also important for reliable engineering computations, when we not only produce a numerical result, but we also produce a measure of its accuracy – e.g., the upper bound Δ on this result's approximation error (Jaulin et al., 2012; Kubica, 2019; Mayer, 2017; Moore at al., 2009; Rabinovich, 2005) or the standard deviation σ of the approximation error (BIPM, 2028; BIPM, 2028a; BIPM, 2011; Rabinovich, 2005). So, it's important to make computation of Δ and/or σ explainable too.

Explainability is especially important for estimating accuracy of the computation results – since while we often have intuition about the physical phenomena that we are trying to predict and can thus detect computation errors, we rarely have intuition about the computation accuracy. So, without understandable explanations, we may miss computation errors.

1.4. What we do in this paper

In this paper, we provide guidance about how to make reliable engineering computing explainable – or at least more explainable.

1.5. Important comment: there are other reasons why we need explainable computations – reasons that go beyond the topics of this paper

It should be mentioned that in physics and in other disciplines, there are other reason why we need to supplement computations with explanations, a reason that goes beyond what we study in this paper.

1.5.1. *First reason: equations are often approximate*

This reason is that the equations that are used in the computationsm – and the models that underlie these computations – are often only approximate. To a naive mathematician, the equations that a physicist or a chemist provide to him/her may sound like an Ultimate Truth directly from Mount Sinai, but physicists, chemists, and engineers know that these equations and these models are only approximate. So, whatever results we get by using these computations need to taken with a grain of salt: if they do not agree with common sense, there is a good chance that they are caused by the approximate character of the equations. Every non-fundamental equation is approximate:

- Ohm's law is approximate – sometimes we see non-linear dependence of voltage on current,
- laws of chemical kinetics are only approximate – more complex equations are needed when the concentrations become high, etc.

In some cases, it is clear when the results of using approximate models are wrong. For example, in many cases, we observe exponential growth:

- a colony of bacteria – when left with a source of food – grows exponentially,
- population grows exponentially,

– economy grows exponentially during a boom,

– the need for new AI researchers (or any other hot-topic researchers) grows exponentially for some period of time, etc.

These exponential models – and many other approximate models in general – are useful to predict short-term future developments, but long-term, they make no sense. For example, since the need for AI professionals grows faster than the population, after a while, we will have more AI professionals than humans – this makes no sense. In this case, the mistake caused by using approximate models is easy to see, but there are other cases when results are not that crazy but still wrong. In these cases, explainable computations can help detect such erroneous uses of approximate models.

Physicists know that even when we use fundamental physical equations, we need to be careful – since these equations are not perfect. Let us give a mathematically (rather) simple example from (Feynman et al., 2005): what is the overall energy of an electron? It consists of:

– the energy $E = m_0 \cdot c^2$ corresponding to the electron's rest mass m_0, and

– the overall energy of the electric field generated by the electron.

Let us compute this energy. The electron is an indivisible elementary particle, it does not have any separate parts. So, according to special relativity, it must be a pointwise particle – otherwise, if it occupied two different spatial locations, we could be able to consider parts located at these location as separate parts. The electric field of an electron at each spatial point is determined by Coulomb's law $|\vec{E}| = c_1 \cdot q/r^2$, where c_1 is a constant whose value depends on the choice of measuring units, q is the electric charge of the electron, and r is the distance from given point to the electron. It is known that the energy density $\rho(x)$ of the electric field at each spatial location is proportional to the square of the electric field: $\rho(x) = c_2 \cdot |\vec{E}|^2$ for some constant c_2. Thus, we have $\rho(x) = c_3/r^4$, where we denoted $c_3 \stackrel{\text{def}}{=} c_2 \cdot c_1^2$. The overall energy $\mathcal{E}$ of the electric field can be obtained if we integrate the energy density over the whole space:

$$\mathcal{E} = \int \frac{c_3}{r^4}\, dV.$$

Since the integrated function depends only on the distance r, we can integrate over each sphere of this radious r – whose area is $4 \cdot \pi \cdot r^2$ – and then integrate over r:

$$\mathcal{E} = \int_0^\infty \frac{c_2}{r^4} \cdot 4 \cdot \pi \cdot r^2\, dr = c_3 \cdot \int_0^\infty \frac{1}{r^2}\, dr = -\frac{1}{r}\Big|_{r=0}^{r=\infty}.$$

One can see that the result is infinite – which makes no sense from the physical viewpoint: the energy of a tiny electron is clearly bounded and very small.

In this example, it is clear that the result makes no physical sense, but in other cases, when the result is finite, we need to use common sense – i.e., explainable computations – to check that the result makes sense.

1.5.1.1. *Comment* There have been some similar physically meaningless infinite results in classical (pre-quantum) physics. Some of them got resolved with the appearance of quantum ideas – the very

idea of quantum physics appeared when Max Planck tried to explain why the overall energy of a black body radiation is not infinite. However, many meaningless infinities – such as the overall energy of an electron – remain in quantum physics.

The above example does not mean that computing the electron's overall energy remains an open problem – in this and many other cases, physicists have found some ways around – but if we do know that and just try to uncritically use equations, you get nonsense. This is, by the way, why it is still not clear whether quantum computing (see, e.g., (Nilesen and Chuang, 2013)) can lead to an exponential speedup: in contrast to problems like $P \stackrel{?}{=} NP$, where the mathematical formulation is clear and precise but no one knows how to solve it, for the quantum question, we do not even have a precise formulation of the problem – since we do not have a fully consistent precise general mathematical formulation of quantum physics.

1.5.2. *Second reason: equations are sometimes too difficult to solve*
Another reason why we need back-of-the envelope computations is that often, the corresponding equations are too complex to be solved. In such cases, it is desirable to have simpler explainable models that would enable us to provide at least some reasonable estimates for the desored quantities.

2. Analysis of the problem

2.1. Why do we perform computations in the first place

Let us go back to our problem: how to make reliable engineering computing more explainable. To better understand this problem, let us first recall why we need computations in the first place. What do we want in general? We want to understand the world, and we want to make it better. Understanding the world means knowing the values of the all the physical quantities that describe the state of the world, knowing these values in all spatial locations and at all moments of time.

- Some of these quantities we can measure directly – e.g., the current temperature and the current wind speed.
- Many other quantities are difficult – or even impossible – to measure directly.

For example:

- it is difficult to measure the temperature at some difficult-to-reach location, and
- it is not possible to directly measure tomorrow's temperature.

To estimate such not-easy-to-directly-measure quantities, we can use known relation between these quantities and some easier-to-measure ones. For example, to predict tomorrow's temperature, we can:

- measure temperature, wind speed, humidity at different locations, and then
- use known equations to make the desired prediction.

In general, such computations are one of the main uses of computers.

Another use of computations is related to changing the world. We would like to predict what will happen:

- if we perform a certain action – e.g., seed the clouds to cause rain, and/or
- if we come up with a design – e.g., a new design for a meteorological rocket.

Ideally, we should also come up with an optimal action and/or optimal design: action and/or design that optimize the appropriate objective function. For this, we also need computations.

2.2. Why computation results are not absolutely accurate: three main reasons and why we concentrate on only one of the them

We would like to know the exact values of the not-easy-to-directly-measure quantities, we would like to exactly know the parameters corresponding to optimal actions and designs. However, as a result of computations, we only get approximate values. There are three reasons for this:

- First, the models that we use are usually only approximate, they only provide an approximate relation between the physical quantities. We, on the computational side, cannot do much about that – this is what physicists have been doing since the beginning of science.
- Second, our algorithms are often only approximate. Whether we use finite element methods to solve systems of partial differential equations, whether we use a continuous-media approximation to describe interactions between atoms and molecules – we do not get exact results. Coming up with better algorithms is difficult, many researchers are working on this. For each specific computational problem, coming up with better algorithms is an important challenge, and we do not see how explainability – the main topic of this paper – can help.
- Finally, the results are not accurate because the values that we feed into the algorithms are not absolutely accurate: they come from measurements, and measurements are never absolutely accurate. In other words, the results are not absolutely accurate because uncertainty of the inputs propagates through our algorithm. The resulting uncertainty is what we will concentrate on in this paper.

2.3. How to describe uncertainty propagation: reminder

Let us denote the inputs to a data processing algorithm by $x_1, \ldots, x_n$, and the result of applying this algorithm by $f(x_1, \ldots, x_n)$. In the ideal world, we should apply the algorithm to the actual values x_i of the corresponding quantities. However, as we have mentioned, we do not know these exact values. Instead, we know approximate values $\widetilde{x}_i$ – that are:

- either obtained directly by measurements
- or obtained indirectly, by applying some other algorithms to measurement results.

For all the inputs, the approximation error $\Delta x_i \stackrel{\text{def}}{=} \widetilde{x}_i - x_i$ is different from 0. Since we are using approximate values of the inputs, instead of the ideal value $y = f(x_1, \ldots, x_n)$, we get an approximate value $\widetilde{y} = f(\widetilde{x}_1, \ldots, \widetilde{x})$. We are interested in estimating the resulting approximation error

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

By definition of the approximation error Δx_i, we have $x_i = \widetilde{x}_i - \Delta x_i$. Thus, the above formula for Δy taes the following form:

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

Measurement errors are usually reasonably small, e.g., about 10%. For such values, their square is much smaller than the error itself: e.g., the square of 10% is 1% which is indeed much smaller. So, to estimate Δy, we can use the techniques that are often used in physics (see, e.g., (Feynman et al., 2005; Thorne and Blanford, 2021)): we can expand the dependence of Δy on Δx_i in Taylor series and ignore quadratic and higher order terms in this expansion. As a result, we get the following formula:

$$\Delta y = \frac{\partial f}{\partial x_1} \cdot \Delta x_1 + \ldots + \frac{\partial f}{\partial x_n} \cdot \Delta x_n.$$

Depending on what we know about Δx_i, we can come up with conclusions about Δy.

Sometimes, we know the probabilities of different possible values of Δx_i. If the mean value of Δx_i is not 0, then we re-calibrate this measuring instrument – by subtracting this mean error from all the measurement results. For example, if before I step on my home scales, I see 2 lbs, I simply subtract 2 lbs from all the measurement results. So, in this case, we can safely assume that the mean value of Δx_i is 0. In this case, the mean value of Δy is also 0, and for the variance σ^2, we have the formula

$$\sigma^2 = \left(\frac{\partial f}{\partial x_1}\right)^2 \cdot \sigma_1^2 + \ldots + \left(\frac{\partial f}{\partial x_n}\right)^2 \cdot \sigma_n^2,$$

where σ_i^2 is the variance of Δx_i.

Sometimes, we only know the upper bound Δ_i on the absolute value $|\Delta x_i|$ of the measurement error Δx_i: $|\Delta x_i| \le \Delta_i$. This case is known as *interval uncertainty*, since in this case, based on measurement result $\widetilde{x}_i$, the only information that we have about the actual (unknown) value x_i is that this value is located in the interval $[\widetilde{x}_i - \Delta_i, \widetilde{x}_i + \Delta_i]$. Why don't we know the probabilities?

- This can happen because we have a state-of-the-art measuring instrument, for which no more-accurate instrument is known, so we can cannot find the probabilities by comparing the results of the two instruments – as these probability are usually determined.

- This may also happen because comparing the results of these two measuring instrument takes time and money, and if we have many sensors, calibrating each of them will cost too much. If we launch a spaceship to the Moon, then yes, we want to perform as many tests as needed, but if we are predicting weather, it does not make economic sense to determine probability distributions for thousands of sensors.

In such interval cases, all we know about Δy is the upper bound Δ on the absolute value $|\Delta y|$ of Δy:

$$\Delta = \left|\frac{\partial f}{\partial x_1}\right| \cdot \Delta_1 + \ldots + \left|\frac{\partial f}{\partial x_n}\right| \cdot \Delta_n.$$

In both probabilistic and interval cases, we can get partial derivatives, e.g., by using numerical differentiation:

$$\frac{\partial f}{\partial x_1} \approx \frac{f(\widetilde{x}_1, \ldots, \widetilde{x}_{i-1}, \widetilde{x}_i + h_i, \widetilde{x}_{i+1}, \ldots, \widetilde{x}_n) - \widetilde{y}}{h_i}$$

for some small h_i.

2.3.0.1. *Comment* Sometimes, the approximation error Δx_i has two components:

- systematic error component about which we only know the upper bound, and
- the random error component, about which we know the standard deviation.

In this case, we need:

- to separately process systematic error components – to get the systematic error component of the resulting error Δy, and
- to separately process random error components – to get the random error component of the resulting error Δy.

2.4. When n is small, the resulting computations are reasonably explainable

When the number of inputs is small, the above procedure is easy to implement and easy to compute – so in this sense, we have a reasonably explainable procedure.

2.5. But what if n is large?

In many practical situations – e.g., in weather prediction – we deal with thousands of inputs – while the prediction itself takes an hour or so on a high-performance computer. If we simply compute all the partial derivatives, this will take too much time. We can use Monte-Carlo simulations – but this required repeating computations are least 25 times – to get the 20% accuracy of estimating σ or Δ, and this will take too much time.

This is where we need explainable computing. For large n, we usually have a few most important inputs – inputs that provide the largest contribution to the overall accuracy estimation. For example, this may be inputs for which the corresponding partial derivatives are the largest. Without losing generality, let us assume that these inputs are $x_1, \ldots, x_r$ for some small r. Then, we can compute – in an explainable way – the values

$$\left(\sigma^{(0)}\right)^2 = \left(\frac{\partial f}{\partial x_1}\right)^2 \cdot \sigma_1^2 + \ldots + \left(\frac{\partial f}{\partial x_r}\right)^2 \cdot \sigma_r^2$$

and

$$\Delta^{(0)} = \left|\frac{\partial f}{\partial x_1}\right| \cdot \Delta_1 + \ldots + \left|\frac{\partial f}{\partial x_r}\right| \cdot \Delta_r.$$

To find the desired values $\sigma^2 = \left(\sigma^{(0)}\right)^2 + \left(\sigma^{(1)}\right)^2$ and $\Delta = \Delta^{(0)} + \Delta^{(1)}$, we need to be able to compute the contribution of all the remaining terms:

$$\left(\sigma^{(1)}\right)^2 = \left(\frac{\partial f}{\partial x_{r+1}}\right)^2 \cdot \sigma_{r+1}^2 + \ldots + \left(\frac{\partial f}{\partial x_n}\right)^2 \cdot \sigma_n^2$$

and

$$\Delta^{(1)} = \left|\frac{\partial f}{\partial x_{r+1}}\right| \cdot \Delta_{r+1} + \ldots + \left|\frac{\partial f}{\partial x_n}\right| \cdot \Delta_n.$$

In both cases, each term corresponding to $n-r$ remaining inputs may be small, but there are many of them, so their overall contribution cannot be ignored.

3. Our main idea and the resulting formulas

3.1. Idea

The need to deal with a large number of terms is ubiquitous in physics. For example, any body consists of about 10^{23} molecules. We know the equations that describe their dynamics, but there is no way we can deal with that many equations. So what do physicists do? Different molecules move differently – and they are largely independent from each other. From our viewpoint, these motions look random. So, instead of considering individual motions, we can assume that they all follow some probability distribution.

This is what statistical physics does. The overall effect is a joint effect of many small independent random effects. In such case, according to the Central Limit Theorem, for large n, the distribution of the resulting effect is close to Gaussian; see, e.g., (Sheskin, 2011). In general, to describe a normal distribution, it is sufficient to know the mean value m and the standard deviation s. Then, with any desired confidence level, we can conclude that then actual values does not exceed $m + k_0 \cdot s$, where k_0 depends on the confidence level:

- to reach 95% confidence, we can take $k_0 = 2$;
- to reach 99.9% confidence, we can take $k_0 = 3$; and
- to reach confidence $1 - 10^{-8}$, we can take $k_0 = 6$.

Let us apply this idea to our case.

3.2. Ket us apply this idea: details

Both for probabilistic and for interval uncertainty, the formula for the desired value V describing uncertainty is the sum of $n - r$ products of the type $d \cdot a$, where:

– d is a factor related to the partial derivative of the data processing function, and

– a is a factor related to input uncertainty.

We assume:

– that the values d and a corresponding to different inputs are independent, and

– that the values d and a corresponding to the same input are also independent of each other.

In this case, the mean value m of the sum V is equal to the sum of the mean values $E[d \cdot a]$, each of which is the product of the mean values: $E[d \cdot a] = m_d \cdot m_a$, where we denoted $m_d \stackrel{\text{def}}{=} E[d]$ and $m_a \stackrel{\text{def}}{=} E[a]$. Thus, we have $m = (n - r) \cdot m_d \cdot m_a$.

Similarly, the variance s^2 of the sum V is equal to the sum of the variances. The variance of the product $d \cdot a$ can be computed as the difference

$$E[d^2 \cdot a^2] - (E[d \cdot a])^2 = s_d \cdot s_a - m_d^2 \cdot m_a^2,$$

where we denoted $s_d \stackrel{\text{def}}{=} E[d^2]$ and $s_a \stackrel{\text{def}}{=} E[a^2]$. Now, with the corresponding confidence, we can conclude that the actual value of v is bounded by $m + k_0 \cdot s$.

Good news is that to find all these expected values, we do not need to consider all inputs and all possible inputs x_i – it is sufficient to consider a random sample. Another good news is that the characteristics of d can be pre-computed beforehand and can be then used for different accuracies a. So, we arrive at the following formulas.

3.3. Resulting formulas: case of probabilistic uncertainty

First, out of the practically encountered input tuples $(x_1, \ldots, x_n)$, randomly select a few. For each of K selected input tuples, randomly select one of the $n - r$ remaining inputs, and use numerical differentiation to compute the corresponding partial derivative D_k. Based on the resulting values $D_1, \ldots, D_K$, compute two arithmetic averages:

$$m_d = \frac{1}{K} \cdot \sum_{k=1}^{K} D_k^2 \text{ and } s_d = \frac{1}{K} \cdot \sum_{k=1}^{K} D_k^4.$$

Then, randomly select K of the remaining $n - r$ inputs $i_1, \ldots, i_K$, and record the corresponding values $\sigma_{i_k}^2$. Based on these values, compute the following two arithmetic averages:

$$m_a = \frac{1}{K} \cdot \sum_{k=1}^{K} \sigma_{i_k}^2 \text{ and } s_a = \frac{1}{K} \cdot \sum_{k=1}^{K} \sigma_{i_k}^4.$$

Then, we can conclude that:

– on average, the standard deviation σ^2 of Δy is equal to $m = (n - r) \cdot m_d \cdot m_a$, and

– that with confidence level depending on k_0, the actual standard deviation is bounded by the value

$$m + k_0 \cdot \sqrt{(n-r) \cdot (s_d \cdot s_a - m_d^2 \cdot m_a^2)}.$$

3.4. Resulting formulas: case of interval uncertainty

First, out of the practically encountered input tuples $(x_1, \ldots, x_n)$, randomly select a few. For each of K selected input tuples, , randomly select one of the $n-r$ remaining inputs, and use numerical differentiation to compute the corresponding partial derivative D_k. Based on the resulting values $D_1, \ldots, D_K$, compute two arithmetic averages:

$$m_d = \frac{1}{K} \cdot \sum_{k=1}^{K} |D_k| \text{ and } s_d = \frac{1}{K} \cdot \sum_{k=1}^{K} D_k^2.$$

Then, randomly select K of the remaining $n-r$ inputs $i_1, \ldots, i_K$, and record the corresponding values Δ_{i_k}. Based on these values, compute the following two arithmetic averages:

$$m_a = \frac{1}{K} \cdot \sum_{k=1}^{K} \Delta_{i_k} \text{ and } s_a = \frac{1}{K} \cdot \sum_{k=1}^{K} \Delta_{i_k}^2.$$

Then, we can conclude that:

– on average, the upper bound Δ of Δy is equal to $m = (n-r) \cdot m_d \cdot m_a$, and

– that with confidence level depending on k_0, the actual standard deviation is bounded by the value

$$m + k_0 \cdot \sqrt{(n-r) \cdot (s_d \cdot s_a - m_d^2 \cdot m_a^2)}.$$

3.5. How we can improve these estimates

In deriving the above estimates, we considered all the inputs. In practice, often, inputs can be naturally divided into several classes. For example, inputs for weather prediction can be divided into:

– inputs corresponding to temperature,

– inputs corresponding to wind speed, and

– inputs corresponding to humidity.

These sensors are different, so it makes sense, instead of averages overall the inputs, to consider averages over each group of inputs. The formulas will be similar, except that now we need to compute several terms. Here is how the above formulas need to be modified when we have G groups of inputs with N_g elements in each of the groups g.

For the case of probabilistic uncertainty, first, out of the practically encountered input tuples $(x_1, \ldots, x_n)$, randomly select a few. For each of K selected input tuples, for each group g, randomly select an input from this group, and use numerical differentiation to compute the corresponding derivative $D_{g,k}$. Based on the resulting values $D_{g,1}, \ldots, D_{g,k}k$, compute, for each group, two arithmetic averages:

$$m_{g,d} = \frac{1}{K} \cdot \sum_{k=1}^{K} D_{g,k}^2 \text{ and } s_{g,d} = \frac{1}{K} \cdot \sum_{k=1}^{K} D_{g,k}^4.$$

Then, in each group g, randomly select K of its inputs $i_{g,1}, \ldots, i_{g,K}$, and record the corresponding values $\sigma^2_{i_{g,k}}$. Based on these values, compute, for each group g, the following two arithmetic averages:

$$m_{g,a} = \frac{1}{K} \cdot \sum_{k=1}^{K} \sigma^2_{i_{g,k}} \text{ and } s_{g,a} = \frac{1}{K} \cdot \sum_{k=1}^{K} \sigma^4_{i_{g,k}}.$$

Then, we can conclude that:

- on average, the standard deviation σ^2 of Δy is equal to

$$m = \sum_{g=1}^{G} N_g \cdot m_{g,d} \cdot m_{g,a},$$

  and

- that with confidence level depending on k_0, the actual standard deviation is bounded by the value

$$m + k_0 \cdot \sqrt{\sum_{g=1}^{G} N_g \cdot (s_{g,d} \cdot s_{g,a} - m_{g,d}^2 \cdot m_{g,a}^2)}.$$

Similarly, for the case of interval uncertainty, first, out of the practically encountered input tuples $(x_1, \ldots, x_n)$, randomly select a few. For each of K selected input tuples, for each groups g, randomly select an input from this group, and use numerical differentiation to compute the corresponding derivative $D_{g,k}$. Based on the resulting values $D_{g,1}, \ldots, D_{g,k}k$, compute, for each group, two arithmetic averages:

$$m_{g,d} = \frac{1}{K} \cdot \sum_{k=1}^{K} |D_{g,k}| \text{ and } s_{g,d} = \frac{1}{K} \cdot \sum_{k=1}^{K} D_{g,k}^2.$$

Then, in each group g, randomly select K of its inputs $i_{g,1}, \ldots, i_{g,K}$, and record the corresponding values $\sigma^2_{i_{g,k}}$. Based on these values, compute, for each group g, the following two arithmetic averages:

$$m_{g,a} = \frac{1}{K} \cdot \sum_{k=1}^{K} \Delta_{i_{g,k}} \text{ and } s_{g,a} = \frac{1}{K} \cdot \sum_{k=1}^{K} \Delta^2_{i_{g,k}}.$$

Then, we can conclude that:

– on average, the upper bound Δ of Δy is equal to

$$m = \sum_{g=1}^{G} N_g \cdot m_{g,d} \cdot m_{g,a},$$

and

– that with confidence level depending on k_0, the actual upper bound Δ is bounded by the value

$$m + k_0 \cdot \sqrt{\sum_{g=1}^{G} N_g \cdot (s_{g,d} \cdot s_{g,a} - m_{g,d}^2 \cdot m_{g,a}^2)}.$$

Acknowledgements

This work was supported by the AT&T Fellowship in Information Technology, by the Institute for Risk and Reliability, Leibniz Universitaet Hannover, Germany, by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) Focus Program SPP 100+ 2388, Grant Nr. 501624329, by the European Union under the project ROBOPROX (No. CZ.02.01.01/00/22 008/0004590), by the Center of Excellence in Econometrics, Faculty of Economics, Chiang Mai University, Thailand, by the Ho Chi Minh City University of Banking, Vietnam, and by Thang Long University, Hanoi, Vietnam.

References

BIPM, IEC, IFCC, ILAC, ISO, IUPAC, IUPAP, OIML. Evaluation of Measurement Data. Guide to the Expression of Uncertainty in Measurement. *Joint Committee for Guides in Metrology (JCGM)* 100:2008, 2008.

BIPM, IEC, IFCC, ILAC, ISO, IUPAC, IUPAP, OIML. Evaluation of Measurement Data. Supplement 1 to the "Guide to the Expression of Uncertainty in Measurement": Propagation of Distributions Using a Monte Carlo Method. *Joint Committee for Guides in Metrology (JCGM) 101:2008, 2008.*

BIPM, IEC, IFCC, ILAC, ISO, IUPAC, IUPAP, OIML. Evaluation of Measurement Data. Supplement 2 to the "Guide to the Expression of Uncertainty in Measurement": Extension to Any Number of Output Quantities, Joint Committee for Guides in Metrology (JCGM) 102:2011, 2011.

Fadelli, I. Mental Math's Shortcut – Pupil Dilation Suggests People Start Solving Befiore All Numbers Are In. *Physics News*, 2026, https://phys.org/news/2026-04-mentak-math-shortcut-pupil-ddilation.html

Feynman, R., R. Leighton, and M. Sands. *The Feynman Lectures on Physics.* Addison Wesley, Boston, Massachusetts, 2005.

Goodfellow, I., Y. Bengio, and A. Courville, *Deep Learning*, MIT Press, Cambridge, Massachusetts, 2016.

Jaulin, L., M. Kiefer, O. Didrit, and E. Walter. *Applied Interval Analysis, with Examples in Parameter and State Estimation, Robust Control, and Robotics.* Springer, London, 2012.

Kubica, B. J. *Interval Methods for Solving Nonlinear Constraint Satisfaction, Optimization, and Similar Problems: from Inequalities Systems to Game Solutions.* Springer, Cham, Switzerland, 2019.

Mayer, G. *Interval Analysis and Automatic Result Verification.* de Gruyter, Berlin, 2017.

Moore, R. E., R. B. Kearfott, and M. J. Cloud. *Introduction to Interval Analysis*. SIAM, Philadelphia, 2009.
Nielsen, M. A., and I. L. Chuang, *Quantum Computation and Quantum Information*, Cambridge University Press, Cambridge, U.K., 2011.
Rabinovich, S. G. *Measurement Errors and Uncertainty: Theory and Practice*. Springer Verlag, New York, 2005.
Sheskin, D. J. Handbook of Parametric and Nonparametric Statistical Procedures. Chapman and Hall/CRC, Boca Raton, Florida, 2011.
Thorne, K. S., and R. D. Blandford. *Modern Classical Physics: Optics, Fluids, Plasmas, Elasticity, Relativity, and Statistical Physics*. Princeton University Press, Princeton, New Jersey, 2021.
Zénon, A., S. Salbagio, and M. Andres. Pupil Size Variations Reveal Bayesian Inference in Cognitive Arithemtic. *Proceedings of the Royal Society B: Biological Sciences*, 293:10151937, 2026, doi 10.1098/rspb.2025.1937