Proceedings of the ASME 2015 International Mechanical Engineering Congress & Exposition
IMECE 2015
November 13-19, 2015, Houston, Texas

IMECE 2015-50339

HOW TO TAKE INTO ACCOUNT MODEL INACCURACY WHEN ESTIMATING THE UNCERTAINTY OF THE RESULT OF DATA PROCESSING

Vladik Kreinovich*, Olga Kosheleva,
Andrzej Pownuk, and Rodrigo Romero
Cyber-ShARE Center
University of Texas at El Paso
El Paso, Texas 79968
Emails: vladik@utep.edu, olgak@utep.edu
ampownuk@utep.edu, raromero2@utep.edu

ABSTRACT

In engineering design, it is important to guarantee that the values of certain quantities such as stress level, noise level, vibration level, etc., stay below a certain threshold in all possible situations, i.e., for all possible combinations of the corresponding internal and external parameters. Usually, the number of possible combinations is so large that it is not possible to physically test the system for all these combinations. Instead, we form a computer model of the system, and test this model. In this testing, we need to take into account that the computer models are usually approximate. In this paper, we show that the existing techniques for taking model uncertainty into account overestimate the uncertainty of the results. We also show how we can get more accurate estimates.

INTRODUCTION

Bounds on unwanted processes: an important part of engineering specifications. An engineering system is designed to perform certain tasks. In the process of performing these tasks, the system also generates some undesirable side effects: it can generate noise, vibration, heat, stress, etc.

We cannot completely eliminate these undesired effects, but specifications for an engineering system usually require that the size q of each of these effects does not exceed a certain predefined threshold (bound) t. It is therefore important to check that this specification is always satisfied, i.e., that $q \le t$ in all possible situations.

How can we check that specifications are satisfied for all possible situations: simulations are needed. To fully describe each situation, we need to know the values of all the parameters $p_1, \ldots, p_n$ that characterize this situation.

These may be external parameters such as wind speed, load, etc., for a bridge. This may be internal parameters such as the exact value of the Young module for a material used in the design.

For each of these parameters, we know the interval of possible values $[\underline{p}_i, \overline{p}_i]$. For many parameters p_i, this interval is described by setting a nominal value $\widetilde{p}_i$ and the bound Δ_i on possible deviations from this nominal value. In such a setting, the interval of possible values has the form

$$[\underline{p}_i, \overline{p}_i] = [\widetilde{p}_i - \Delta_i, \widetilde{p}_i + \Delta_i]. \quad (1)$$

In other cases, the bounds $\underline{p}_i$ and $\overline{p}_i$ are given directly. However, we can always describe the resulting interval in the form (1) if we take the midpoint of this interval as $\widetilde{p}_i$ and its half-width as Δ_i:

$$\widetilde{x}_i \stackrel{\text{def}}{=} \frac{\underline{p}_i + \overline{p}_i}{2}; \ \ \Delta_i \stackrel{\text{def}}{=} \frac{\overline{p}_i - \underline{p}_i}{2}. \quad (2)$$

*Address all correspondence to this author.

Thus, without losing generality, we can always assume that the set of possible values of each parameter p_i is given by the expression (1).

We would like to make sure that the quantity q satisfies the desired inequality $q \le t$ for *all* possible combinations of values $p_i \in [\underline{p}_i, \overline{p}_i]$. Usually, there are many such parameters, and thus, there are many possible combinations – even if we limit ourselves to extreme cases, when each parameter p_i is equal to either $\underline{p}_i$ or to $\overline{p}_i$, we will still get 2^n possible combinations. It is therefore not feasible to physically check how the system behaves under all such combination. Instead, we need to rely on computer simulations.

Formulation of the problem. There are known techniques for using computer simulation to check that the system satisfies the given specifications for all possible combinations of these parameters. These techniques, however, have been originally designed for the case when we have an exact model of the system.

In principle, we can also use these techniques in more realistic situations, when the corresponding model is only approximate. However, as we show in this paper, the use of these techniques leads to overestimation of the corresponding uncertainty. We also show that a proper modification of these techniques leads to a drastic decrease of this overestimation and thus, to more accurate estimations.

HOW TO CHECK SPECIFICATIONS WHEN WE HAVE AN EXACT MODEL OF A SYSTEM: REMINDER

Case of an exact model: description. To run the corresponding computer simulations, we need to have a computer model that, given the values of the parameters $p_1, \ldots, p_n$, estimates the corresponding value of the parameter q. Let us first consider situations when this computer model is exact, i.e., when this model enables us to compute the exact value q:

$$q = q(p_1, \ldots, p_n). \tag{3}$$

In most engineering situations, deviations from nominal values are small. Usually, possible deviations $\Delta p_i \stackrel{\text{def}}{=} p_i - \widetilde{p}_i$ from nominal values are reasonably small; see, e.g., [10]. In this paper, we will restrict ourselves to such situations.

In such situations, we can plug in the values $p_i = \widetilde{p}_i + \Delta p_i$ into the formula (3), expand the resulting expression in Taylor series in terms of small values Δp_i, and ignore terms which are quadratic (or of higher order) in terms of Δp_i:

$$q(p_1, \ldots, p_n) = \widetilde{q} + \sum_{i=1}^{n} c_i \cdot \Delta p_i, \tag{4}$$

where we denote

$$\widetilde{q} \stackrel{\text{def}}{=} q(\widetilde{x}_1, \ldots, \widetilde{x}_n) \text{ and } c_i \stackrel{\text{def}}{=} \frac{\partial q}{\partial p_i}. \tag{5}$$

How to use the linearized model to check that specifications are satisfied: analysis of the problem. To make sure that we always have $q \le t$, we need to guarantee that the largest possible value $\overline{q}$ of the function q does not exceed t.

How can we compute this upper bound $\overline{q}$? The maximum of the sum (4) is attained when each of n terms $c_i \cdot \Delta p_i$ attains the largest possible value. Each of these terms is a linear function of $\Delta p_i \in [-\Delta_i, \Delta_i]$, so the desired maximum has the form (see, e.g., see, e.g., [5, 10]):

$$\overline{q} = \widetilde{q} + \sum_{i=1}^{n} |c_i| \cdot \Delta_i. \tag{6}$$

How to estimate the derivatives c_i? Sometimes, we have an explicit formula for computing $q(p_1, \ldots, p_n)$. In this case, by explicitly differentiating the corresponding expression, we can get formulas for computing the derivatives c_i.

In most real-life situations, however, there is no explicit formula. To find the value $q(p_1, \ldots, p_n)$ corresponding to the parameter values $p_1, \ldots, p_n$ – e.g., to find the corresponding stress – we need to solve a system of partial differential equations. In such situations, the dependence $q(p_1, \ldots, p_n)$ is given in terms of a complex algorithm (and not an explicit formula), and thus, computing the derivative is not as straightforward.

Since we do not have an analytical expression for the derivative c_i, we need to use *numerical differentiation* to estimate c_i. In the linear approximation,

$$q(\widetilde{p}_1, \ldots, \widetilde{p}_{i-1}, \widetilde{p}_i + h_i, \widetilde{p}_{i+1}, \ldots, \widetilde{p}_n) = \widetilde{q} + c_i \cdot h_i, \tag{7}$$

so

$$c_i = \frac{q(\widetilde{p}_1, \ldots, \widetilde{p}_{i-1}, \widetilde{p}_i + h_i, \widetilde{p}_{i+1}, \ldots, \widetilde{p}_n) - \widetilde{q}}{h_i}. \tag{8}$$

In particular, substituting this expression for c_i, with $h_i = \Delta_i$, into the formula (4), we get

$$\overline{q} = \widetilde{q} + \sum_{i=1}^{n} |q_i - \widetilde{q}|, \tag{9}$$

where we denoted

$$q_i \stackrel{\text{def}}{=} q(\widetilde{p}_1, \ldots, \widetilde{p}_{i-1}, \widetilde{p}_i + \Delta_i, \widetilde{p}_{i+1}, \ldots, \widetilde{p}_n). \tag{10}$$

Thus, we arrive at the following technique (see, e.g., [5]).

How to use the linearized model to check that specifications are satisfied: resulting technique. We know:

- an algorithm $q(p_1, \ldots, p_n)$ that, given the values of the parameters $p_1, \ldots, p_n$, computes the value of the quantity q;
- a threshold t that needs to be satisfied;
- for each parameter p_i, we know its nominal value $\widetilde{p}_i$ and the largest possible deviation Δ_i from this nominal value.

Based on this information, we need to check whether $q(p_1, \ldots, p_n) \le t$ for all possible combinations of values p_i from the corresponding intervals $[\widetilde{p}_i - \Delta_i, \widetilde{p}_i + \Delta_i]$.

We can perform this checking as follows:

1) first, we apply the algorithm q to compute the value $\widetilde{q} = q(\widetilde{p}_1, \ldots, \widetilde{p}_n)$;
2) then, for each i from 1 to n, we apply the algorithm q to compute the value $q_i = q(\widetilde{p}_1, \ldots, \widetilde{p}_{i-1}, \widetilde{p}_i + \Delta_i, \widetilde{p}_{i+1}, \ldots, \widetilde{p}_n)$;
3) after that, we compute $\overline{q} = \widetilde{q} + \sum_{i=1}^{n} |q_i - \widetilde{q}|$;
4) finally, we check whether $\overline{q} \le t$.

If $\overline{q} \le t$, this means that the desired specifications are always satisfied. If $\overline{q} > t$, this means that for some combinations of possible values p_i, the specifications are not satisfied.

Possibility of a further speed-up. The formula (9) requires $n+1$ calls to the program that computes q for given values of parameters. In many practical situations, the program q takes a reasonably long time to compute, and the number of parameters is large. In such situations, the corresponding computations require a very long time.

A possibility to speed up the corresponding computations comes from the properties of the Cauchy distribution, i.e., a distribution with a probability density function

$$\rho(x) = \frac{1}{\pi \cdot \Delta} \cdot \frac{1}{1 + \left(\frac{x}{\Delta}\right)^2}. \tag{11}$$

The possibility to use Cauchy distributions comes from the fact that they have the following property: if η_i are independent variables which are Cauchy distributed with parameters Δ_i, then for each tuple of real numbers $c_1, \ldots, c_n$, the linear combination $\sum_{i=1}^{n} c_i \cdot \eta_i$ is also Cauchy distributed, with parameter $\Delta = \sum_{i=1}^{n} |c_i| \cdot \Delta_i$.

Thus, we can find Δ as follows [7]:

1) first, for $k = 1, \ldots, N$, we simulate random variables $\eta_i^{(k)}$ which are Cauchy-distributed with parameters Δ_i;
2) for each k, we then estimate $\Delta y^{(k)} = \sum_{i=1}^{n} c_i \cdot \eta_i^{(k)}$ as $\Delta y^{(k)} = y^{(k)} - \widetilde{y}$, where

$$y^{(k)} = q(\widetilde{p}_1 + \eta_1^{(k)}, \ldots, \widetilde{p}_n + \eta_n^{(k)}); \tag{12}$$

3) based on the population of N values $\Delta y^{(1)}, \ldots, \Delta y^{(N)}$ which is Cauchy-distributed with parameter Δ, we find this parameter;
4) finally, we follow the formula (6) and compute $\overline{q} = \widetilde{q} + \Delta$.

(see [7] for technical details).

In this Monte-Carlo-type technique, we need $N+1$ calls to the program that computes q. The accuracy of the resulting estimate depends only on the sample size N and *not* on the number of inputs n. Thus, for a fixed desired accuracy, when n is sufficiently large, this method requires much fewer calls to q and is, thus, much faster. For example, if we want to estimate Δ with relative accuracy 20%, then we need $N = 100$ simulations, so for $n \gg 200$, this method is much faster that a straightforward application of the formula (9).

For many practical problems, we can achieve an even faster speed-up. In both methods described in this section, we apply the original algorithm $q(p_1, \ldots, p_n)$ several times: first, to the tuple of nominal values $(\widetilde{p}_1, \ldots, \widetilde{p}_n)$ and then, to several other tuples $(\widetilde{p}_1 + \eta_1, \ldots, \widetilde{p}_n + \eta_n)$. For example, in the linearized method (9), we apply the algorithm q to tuples $(\widetilde{p}_1, \ldots, \widetilde{p}_{i-1}, \widetilde{p}_i + \Delta_i, \widetilde{p}_{i+1}, \ldots, \widetilde{p}_n)$ corresponding to $i = 1, \ldots, n$.

In many practical cases, once we have computed the value $\widetilde{q} = q(\widetilde{p}_1, \ldots, \widetilde{p}_n)$, we can then compute the values

$$q(\widetilde{p}_1 + \eta_1, \ldots, \widetilde{p}_n + \eta_n) \tag{13}$$

faster than by directly applying the algorithm q to the corresponding tuple. This happens, for example, if the algorithm for computing $q(p_1, \ldots, p_n)$ solves a system of nonlinear equations $F_j(q_1, \ldots, q_k, p_1, \ldots, p_n) = 0$, $1 \le j \le k$, and then returns $q = q_1$.

In this case, once we know the values $\widetilde{q}_j$ for which $F_j(\widetilde{q}_1, \ldots, \widetilde{q}_k, \widetilde{p}_1, \ldots, \widetilde{p}_n) = 0$, we can find the values $q_j = \widetilde{q}_j + \Delta q_j$ for which

$$F_j(\widetilde{q}_1 + \Delta q_1, \ldots, \widetilde{q}_k + \Delta q_k, \widetilde{p}_1 + \eta_1, \ldots, \widetilde{p}_n + \eta_n) = 0 \tag{14}$$

by linearizing this system into an easy-to-solve system of linear equations

$$\sum_{j'=1}^{k} a_{jj'} \cdot \Delta q_{j'} + \sum_{i=1}^{n} b_{ji} \cdot \eta_i = 0, \tag{15}$$

where $a_{jj'} \stackrel{\text{def}}{=} \dfrac{\partial F_j}{\partial q_{j'}}$ and $b_{ji} \stackrel{\text{def}}{=} \dfrac{\partial F_j}{\partial p_i}$.

A similar simplification is possible when the value q corresponding to given values $p_1, \ldots, p_n$ comes from solving a system of nonlinear differential equations

$$\frac{dx_j}{dt} = f_j(x_1, \ldots, x_k, p_1, \ldots, p_n). \tag{16}$$

In this case, once we find the solution $\widetilde{x}_j(t)$ to the system of differential equations

$$\frac{d\widetilde{x}_j}{dt} = f_j(\widetilde{x}_1, \ldots, \widetilde{x}_k, \widetilde{p}_1, \ldots, \widetilde{p}_n) \tag{17}$$

corresponding to the nominal values, we do not need to explicitly solve the modified system of differential equations

$$\frac{dx_j}{dt} = f_j(x_1, \ldots, x_k, \widetilde{p}_1 + \eta_1, \ldots, \widetilde{p}_n + \eta_n) \tag{18}$$

to find the corresponding solution. Instead, we can take into account that the differences η_i are small; thus, the resulting differences $\Delta x_j(t) \stackrel{\text{def}}{=} x_j(t) - \widetilde{x}_j(t)$ are small. So, we can linearize the resulting differential equations

$$\frac{\Delta x_j(t)}{dt} = f_j(\widetilde{x}_1 + \Delta x_1, \ldots, \widetilde{x}_k + \Delta x_k, \widetilde{p}_1 + \eta_1, \ldots, \widetilde{p}_n + \eta_n) -$$

$$f_j(\widetilde{x}_1, \ldots, \widetilde{x}_k, \widetilde{p}_1, \ldots, \widetilde{p}_n) \tag{19}$$

into easier-to-solve *linear* equations

$$\frac{d\Delta x_j}{dt} = \sum_{j'=1}^{k} a_{jj'} \cdot \Delta x_{j'} + \sum_{i=1}^{n} b_{ji} \cdot \eta_i, \tag{20}$$

where $a_{jj'} \stackrel{\text{def}}{=} \dfrac{\partial f_j}{\partial x_{j'}}$ and $b_{ji} \stackrel{\text{def}}{=} \dfrac{\partial f_j}{\partial p_i}$.

This idea – known as *local sensitivity analysis* – is successfully used in many practical applications; see, e.g., [2, 11].

Comment. As we have mentioned earlier, in this paper, we only consider situations when the deviations Δp_i from the nominal values are small. In some practical situations, some of these deviations are not small. In such situations, we can no longer use linearization, we need to use global optimization techniques of *global sensitivity*; see, e.g., [2, 11].

WHAT IF WE TAKE INTO ACCOUNT MODEL INACCURACY

Models are rarely exact. Engineering systems are usually complex. As a result, it is rarely possible to find explicit expressions for q as a function of the parameters $p_1, \ldots, p_n$. Usually, we have some approximate computations. For example, if q is obtained by solving a system of partial differential equations, we use, e.g., the Finite Element method to find the approximate solution and thus, the approximate value of the quantity q.

How model inaccuracy is usually described. In most practical situations, at best, we know the upper bound ε on the accuracy of the computational model. In such cases, for each tuple of parameters $p_1, \ldots, p_n$, once we apply the computational model and get the value $Q(p_1, \ldots, p_n)$, the actual (unknown) value $q(p_1, \ldots, p_n)$ of the quantity q satisfies the inequality

$$|Q(p_1, \ldots, p_n) - q(p_1, \ldots, p_n)| \le \varepsilon. \tag{21}$$

How this model inaccuracy affects the above checking algorithms: analysis of the problem. Let us start with the formula (9). This formula assumes that we know the exact values of $\widetilde{q} = q(\widetilde{p}_1, \ldots, \widetilde{p}_n)$ and q_i (as defined by the formula (10)). Instead, we know the values

$$\widetilde{Q} \stackrel{\text{def}}{=} Q(\widetilde{p}_1, \ldots, \widetilde{p}_n) \tag{22}$$

and

$$Q_i \stackrel{\text{def}}{=} Q(\widetilde{p}_1, \ldots, \widetilde{p}_{i-1}, \widetilde{p}_i + \Delta_i, \widetilde{p}_{i+1}, \ldots, \widetilde{p}_n) \tag{23}$$

which are ε-close to the values $\widetilde{q}$ and q_i. We can apply the formula (9) to these approximate values, and get

$$\overline{Q} = \widetilde{Q} + \sum_{i=1}^{n} |Q_i - \widetilde{Q}|. \tag{24}$$

Here, $|\widetilde{Q} - \widetilde{q}| \le \varepsilon$ and $|Q_i - q_i| \le \varepsilon$, hence $|(Q_i - \widetilde{Q}) - (q_i - \widetilde{q})| \le 2\varepsilon$ and $||Q_i - \widetilde{Q}| - |q_i - \widetilde{q}|| \le 2\varepsilon$. By adding up all these inequalities, we conclude that

$$|\overline{q} - \overline{Q}| \le (2n+1) \cdot \varepsilon. \tag{25}$$

Thus, the only information that we have about the desired upper bound $\overline{q}$ is that $\overline{q} \le B$, where

$$B \stackrel{\text{def}}{=} \overline{Q} + (2n+1) \cdot \varepsilon. \tag{26}$$

Hence, we arrive at the following method.

How this model inaccuracy affects the above checking algorithms: resulting method. We know:

- an algorithm $Q(p_1,\ldots,p_n)$ that, given the values of the parameters $p_1,\ldots,p_n$, computes the value of the quantity q with a known accuracy ε;
- a threshold t that needs to be satisfied;
- for each parameter p_i, we know its nominal value $\widetilde{p}_i$ and the largest possible deviation Δ_i from this nominal value.

Based on this information, we need to check whether $q(p_1,\ldots,p_n)\le t$ for all possible combinations of values p_i from the corresponding intervals $[\widetilde{p}_i-\Delta_i,\widetilde{p}_i+\Delta_i]$.

We can perform this checking as follows:

1) first, we apply the algorithm Q to compute the value

$$\widetilde{Q}=Q(\widetilde{p}_1,\ldots,\widetilde{p}_n); \tag{27}$$

2) then, for each i from 1 to n, we apply the algorithm Q to compute the value

$$Q_i=Q(\widetilde{p}_1,\ldots,\widetilde{p}_{i-1},\widetilde{p}_i+\Delta_i,\widetilde{p}_{i+1},\ldots,\widetilde{p}_n); \tag{28}$$

3) after that, we compute

$$B=\widetilde{Q}+\sum_{i=1}^{n}|Q_i-\widetilde{Q}|+(2n+1)\cdot\varepsilon; \tag{29}$$

4) finally, we check whether $B\le t$.

If $B\le t$, this means that the desired specifications are always satisfied. If $B>t$, this means that we cannot guarantee that the specifications are always satisfied.

Comment 1. Please note that, in contrast to the case of the exact model, if $B>t$, this does not necessarily mean that the specifications are not satisfied: maybe they are satisfied, but we cannot check that since we only know approximate values of q.

Comment 2. Similar bounds can be found for the estimates based on the Cauchy distribution.

Comment 3. The above estimate B is not the best that we can get, but it has been proven that computing the best estimate would require un-realistic exponential time [4, 8] – i.e., time which grows as 2^s with the size s of the input; thus, when we only consider feasible algorithms, overestimation is inevitable.

Comment 4. Similar to the methods described in the previous section, instead of directly applying the algorithm Q to the modified tuples, we can, wherever appropriate, to use the above-mentioned local sensitivity analysis technique.

Problem. When n is large, then, even for reasonably small inaccuracy ε, the value $(2n+1)\cdot\varepsilon$ is large.

In this paper, we show how we can get better estimates for the difference between the desired bound $\widetilde{q}$ and the computed bound $\overline{Q}$.

HOW TO GET BETTER ESTIMATES

Main idea. As we have mentioned earlier, usually, we know the partial differential equations that describe the engineering system. Model inaccuracy comes from the fact that we do not have an analytical solution to this system of equations, so we have to use numerical (approximate) methods.

Usual numerical methods for solving systems of partial differential equations involve discretization of space – e.g., the use of Finite Element Methods.

Strictly speaking, the resulting inaccuracy is deterministic. However, in most cases, for all practical purposes, this inaccuracy can be viewed as random: when we select a different combination of parameters, we get an unrelated value of discretization-based inaccuracy.

In other words, we can view the differences

$$Q(p_1,\ldots,p_n)-q(p_1,\ldots,p_n) \tag{30}$$

corresponding to different tuples $(p_1,\ldots,p_n)$ as independent random variables. In particular, this means that the differences $\widetilde{Q}-\widetilde{q}$ and Q_i-q_i are independent random variables.

Technical details. What is a probability distribution for these random variables?

All we know about each of these variables is that its values are located somewhere in the interval $[-\varepsilon,\varepsilon]$. We do not have any reason to assume that some values from this interval are more probable than others, so it is reasonable to assume that all the values are equally probable, i.e., that we have a uniform distribution on this interval.

For this uniform distribution, the mean is 0, and the standard deviation is $\sigma=\dfrac{\varepsilon}{\sqrt{3}}$.

Auxiliary idea: how to get a better estimate for $\widetilde{q}$. In our main algorithm, we apply the computational model Q to $n+1$ different tuples. What we suggest it to apply it to one more tuple (making it $n+2$ tuples), namely, computing an approximation

$$M\stackrel{\text{def}}{=}Q(\widetilde{p}_1-\Delta_1,\ldots,\widetilde{p}_n-\Delta_n) \tag{31}$$

to the value

$$m\stackrel{\text{def}}{=}q(\widetilde{p}_1-\Delta_1,\ldots,\widetilde{p}_n-\Delta_n). \tag{32}$$

In the linearized case (4), one can easily check that

$$\widetilde{q}+\sum_{i=1}^{n} q_i + m = (n+2)\cdot\widetilde{q}, \tag{33}$$

i.e.,

$$\widetilde{q} = \frac{1}{n+2}\cdot\left(\widetilde{q}+\sum_{i=1}^{n} q_i + m\right). \tag{34}$$

Thus, we can use the following formula to come up with a new estimate $\widetilde{Q}_{\rm new}$ for $\widetilde{q}$:

$$\widetilde{Q}_{\rm new} = \frac{1}{n+2}\cdot\left(\widetilde{Q}+\sum_{i=1}^{n} Q_i + m\right). \tag{35}$$

For the differences $\Delta\overline{q}_{\rm new} \stackrel{\rm def}{=} \overline{Q}_{\rm new} - \overline{q}$, $\Delta\overline{q} \stackrel{\rm def}{=} \overline{Q} - \overline{q}$, $\Delta\widetilde{q} \stackrel{\rm def}{=} \widetilde{Q} - \widetilde{q}$, $\Delta q_i \stackrel{\rm def}{=} Q_i - q_i$, and $\Delta m \stackrel{\rm def}{=} M - m$, we have the following formula:

$$\Delta\widetilde{q}_{\rm new} = \frac{1}{n+2}\cdot\left(\Delta\widetilde{q}+\sum_{i=1}^{n} \Delta q_i + \Delta m\right). \tag{36}$$

The left-hand side is the arithmetic average of $n+2$ independent identically distributed random variables, with mean 0 and variance $\sigma^2 = \dfrac{\varepsilon^2}{3}$. Hence (see, e.g., [12]), the mean of this average $\Delta\widetilde{q}_{\rm new}$ is the average of the means, i.e., 0, and the variance is equal to $\sigma^2 = \dfrac{\varepsilon^2}{3\cdot(n+2)} \ll \dfrac{\varepsilon^2}{3} = \sigma^2[\Delta\widetilde{q}]$.

Thus, this average $\widetilde{Q}_{\rm new}$ is a more accurate estimation of the quantity $\widetilde{q}$ than $\widetilde{Q}$.

Let us use this better estimate for $\widetilde{q}$ when estimating the upper bound $\overline{q}$. Since the average $\widetilde{Q}_{\rm new}$ is a more accurate estimation of the quantity $\widetilde{q}$ than $\widetilde{Q}$, let us use this average instead of $\widetilde{Q}$ when estimating $\overline{Q}$. In other words, instead of the estimate (24), let us use a new estimate

$$\overline{Q}_{\rm new} = \widetilde{Q}_{\rm new} + \sum_{i=1}^{n} |Q_i - \widetilde{Q}|. \tag{37}$$

Let us estimate the accuracy of this new approximation.

The formula (9) can be described in the following equivalent form:

$$\overline{q} = \widetilde{q} + \sum_{i=1}^{n} s_i\cdot(q_i - \widetilde{q}) = \left(1 - \sum_{i=1}^{n} s_i\right)\cdot\widetilde{q} + \sum_{i=1}^{n} s_i\cdot q_i, \tag{38}$$

where $s_i \in \{-1, 1\}$ are the signs of the differences $q_i - \widetilde{q}$.

Similarly, we get

$$\overline{Q}_{\rm new} = \left(1 - \sum_{i=1}^{n} s_i\right)\cdot\widetilde{Q}_{\rm new} + \sum_{i=1}^{n} s_i\cdot Q_i. \tag{39}$$

Thus, for the difference $\Delta\overline{q} \stackrel{\rm def}{=} \overline{Q}_{\rm new} - \overline{q}$, we have

$$\Delta\overline{q}_{\rm new} = \left(1 - \sum_{i=1}^{n} s_i\right)\cdot\Delta\widetilde{q}_{\rm new} + \sum_{i=1}^{n} s_i\cdot\Delta q_i. \tag{40}$$

Here, the differences $\Delta\widetilde{q}_{\rm new}$ and Δq_i are independent random variables. According to the Central Limit Theorem (see, e.g., [12]), for large n, the distribution of a linear combination of many independent random variables is close to Gaussian. The mean of the resulting distribution is the linear combination of the means, thus equal to 0.

The variance of a linear combination $\sum\limits_i k_i\cdot\eta_i$ of independent random variables η_i with variances σ_i^2 is equal to $\sum\limits_i k_i^2\cdot\sigma_i^2$. Thus, in our case, the variance σ^2 of the difference $\Delta\overline{q}$ is equal to

$$\sigma^2 = \left(1 - \sum_{i=1}^{n} s_i\right)^2\cdot\frac{\varepsilon^2}{3\cdot(n+2)} + \sum_{i=1}^{n}\frac{\varepsilon^2}{3}. \tag{41}$$

Here, since $|s_i| \le 1$, we have $\left|1 - \sum\limits_{i=1}^{n} s_i\right| \le n+1$, so (41) implies that

$$\sigma^2 \le \frac{\varepsilon^2}{3}\cdot\left(\frac{(n+1)^2}{n+2} + n\right). \tag{42}$$

Here, $\dfrac{(n+1)^2}{n+2} \le \dfrac{(n+1)^2}{n+1} = n+1$, hence

$$\sigma^2 \le \frac{\varepsilon^2}{3}\cdot(2n+1). \tag{43}$$

For a normal distribution, with almost complete certainty, all the values are concentrated within k_0 standard deviations away from the mean: within 2σ with confidence 0.95, within 3σ with degree of confidence 0.999, within 6σ with degree of confidence $1 - 10^{-8}$. Thus, we can safely conclude that

$$\overline{q} \le \overline{Q}_{\rm new} + k_0\cdot\sigma \le \overline{Q}_{\rm new} + k_0\cdot\frac{\varepsilon}{\sqrt{3}}\cdot\sqrt{2n+1}. \tag{44}$$

Here, inaccuracy grows as $\sqrt{2n+1}$, which is much better than in the traditional approach, where it grows proportionally to $2n+1$ – and we achieve this drastic reduction of the overestimation, basically by using one more run of the program Q in addition to the previously used $n+1$ runs.

So, we arrive at the following method.

Resulting method. We know:

- an algorithm $Q(p_1,\ldots,p_n)$ that, given the values of the parameters $p_1,\ldots,p_n$, computes the value of the quantity q with a known accuracy ε;
- a threshold t that needs to be satisfied;
- for each parameter p_i, we know its nominal value $\widetilde{p}_i$ and the largest possible deviation Δ_i from this nominal value.

Based on this information, we need to check whether $q(p_1,\ldots,p_n)\le t$ for all possible combinations of values p_i from the corresponding intervals $[\widetilde{p}_i-\Delta_i,\widetilde{p}_i+\Delta_i]$.

We can perform this checking as follows:

1) first, we apply the algorithm Q to compute the value

$$\widetilde{Q}=Q(\widetilde{p}_1,\ldots,\widetilde{p}_n); \tag{45}$$

2) then, for each i from 1 to n, we apply the algorithm Q to compute the value

$$Q_i=Q(\widetilde{p}_1,\ldots,\widetilde{p}_{i-1},\widetilde{p}_i+\Delta_i,\widetilde{p}_{i+1},\ldots,\widetilde{p}_n); \tag{46}$$

3) then, we compute

$$M=Q(\widetilde{p}_1-\Delta_1,\ldots,\widetilde{p}_n-\Delta_n); \tag{47}$$

4) we compute

$$\widetilde{Q}_{\text{new}}=\frac{1}{n+2}\cdot\left(\widetilde{Q}+\sum_{i=1}^{n}Q_i+M\right); \tag{48}$$

5) we compute

$$b=\widetilde{Q}_{\text{new}}+\sum_{i=1}^{n}\left|Q_i-\widetilde{Q}_{\text{new}}\right|+k_0\cdot\sqrt{2n+1}\cdot\frac{\varepsilon}{\sqrt{3}}, \tag{49}$$

where k_0 depends on the level of confidence that we can achieve;

6) finally, we check whether $b\le t$.

If $b\le t$, this means that the desired specifications are always satisfied. If $b>t$, this means that we cannot guarantee that the specifications are always satisfied.

Comment. For the Cauchy method, similarly, after computing $\widetilde{Q}=Q(\widetilde{p}_1,\ldots,\widetilde{p}_n)$ and $Y^{(k)}=Q(\widetilde{p}_1+\eta_1^{(k)},\ldots,\widetilde{p}_n+\eta_n^{(k)})$, we can compute the improved estimate $\widetilde{Q}_{\text{new}}$ for $\widetilde{q}$ as

$$\widetilde{Q}_{\text{new}}=\frac{1}{N+1}\cdot\left(\widetilde{Q}+\sum_{k=1}^{N}Y^{(k)}\right), \tag{50}$$

and estimate the parameter Δ based on the more accurate differences $\Delta y_{\text{new}}^{(k)}=Y^{(k)}-\widetilde{Q}_{\text{new}}$.

Experimental testing. We tested our approach on the example of the seismic inverse problem in geophysics, where we need to reconstruct the velocity of sound at different spatial locations and at different depths based on the times that it takes for a seismic signal to get from the set-up explosion to different seismic stations. In this reconstruction, we used (a somewhat improved version of) the finite element technique that was originated by John Hole [3]; the resulting techniques are described in [1, 6, 9].

According to Hole's algorithm, we divide the 3-D volume of interest (in which we want to find the corresponding velocities) into a rectangular 3-D grid of N small cubic cells. We assume that the velocity is constant within each cube; the value of velocity in the j-th cube is denoted by v_j. Each observation j means that we know the time t_j that it took the seismic wave to go from the site of the corresponding explosion to the location of the observING sensor.

This algorithm is iterative. We start with the first-approximation model of the Earth, namely, with geology-motivated approximate values $v_i^{(1)}$. At each iteration k, we start with the values $v_i^{(k)}$ and produce the next approximation $v_i^{(k+1)}$ as follows.

First, based on the latest approximation $v_i^{(k)}$, we simulate how the seismic waves propagate from the explosion site to the sensor locations. In the cube that contains the explosion site, the seismic signal propagates in all directions. When the signal's trajectory approaches the border between the two cubes i and i', the direction of the seismic wave changes in accordance with the Snell's law $\dfrac{\sin(\theta_i)}{\sin(\theta_{i'})}=\dfrac{v_i^{(k)}}{v_{i'}^{(k)}}$, where θ_i is the angle between the seismic wave's trajectory in the i-th cube and the vector orthogonal to the plane separating the two cubes. Snell's law enables us to find the trajectory's direction in the next cube i'. Once the way reaches the location of the sensor, we can estimate the travel time as $t_j^{(k)}=\sum\limits_i\dfrac{\ell_{ji}}{v_i^{(k)}}$, where the sum is taken over all the cubes through which this trajectory passes, and ℓ_{ji} is the length of the part of the trajectory that lies in the i-th cube.

Each predicted value $t_j^{(k)}$ is, in general, different from the observed value t_j. To compensate for this difference, the velocity model $v_i^{(k)}$ is corrected: namely, the inverse value $s_i^{(k)} \stackrel{\text{def}}{=} \dfrac{1}{v_i^{(k)}}$ is replaced with an updated value

$$s_i^{(k+1)} = s_i^{(k)} + \frac{1}{n_i} \cdot \sum_j \frac{t_j - t_j^{(k)}}{L_j}, \tag{51}$$

where the sum is taken over all trajectories that pass through the i-th cube, n_i is the overall number of such trajectories, and $L_j = \sum\limits_i \ell_{ji}$ is the overall length of the j-th trajectory.

Iterations stop when the process converges; for example, it is reasonable to stop the process when the velocity models obtained on two consecutive iterations becomes close:

$$\sum_i (v_i^{(k+1)} - v_i^{(k)})^2 \le \varepsilon \tag{52}$$

for some small value $\varepsilon > 0$. The quality of the resulting solution can be gauged by how well the predicted travel times $t_i^{(k)}$ match the observations; usually, by the root mean square (rms) approximation error $\sqrt{\dfrac{1}{N} \cdot \sum\limits_i (t_i^{(k)} - t_i)^2}$.

In this problem, there are two sources of uncertainty. The first is the uncertainty with which we can measure each travel time t_j. The travel time is the difference between the time when the signal arrives at the sensor location and the time of the artificially set explosion. The explosion time is known with a very high accuracy, but the arrival time is not. In the ideal situation, when the only seismic signal comes from the our explosion, we could exactly pinpoint the arrival time as the time when the sensor starts detecting a signal. In real life, there is always a background noise, so we can only determine the arrival time with some accuracy.

The second source of uncertainty comes from the fact that our discrete model is only an approximate description of the continuous real Earth.

In [1, 6, 9], we used the formula (9) and the Cauchy-based techniques to estimate how the measurement uncertainty affects the results of data processing. To test the method described in this section, we used the above formulas to compute the new value $\widetilde{Q}_{\text{new}}$. These new values indeed lead to a better fit with data than the original values $\widetilde{Q}$: in our 16 experiments, we only in one case, the rms approximation error decreased, on average, by 15%. It should also be mentioned that only in one case the rms approximation error increased (and not much, only by 7%); in all other 15 cases, the rms approximation error decreased.

FUTURE WORK: CAN WE FURTHER IMPROVE THE ACCURACY?

How to improve the accuracy: a straightforward approach. As we have mentioned, the inaccuracy $Q \ne q$ is caused by the fact that we are using a Finite Element method with a finite size elements. In the traditional Finite Element method, when we assume that the values of each quantity within each element are constant, this inaccuracy comes from the fact that we ignore the difference between the values of the corresponding parameters within each element. For elements of linear size h, this inaccuracy Δx is proportional to $x' \cdot h$, where x' is the spatial derivative of x. In other words, the inaccuracy is proportional to the linear size h.

A straightforward way to improve the accuracy is to decrease h. For example, if we reduce h to $\dfrac{h}{2}$, then we decrease the resulting model inaccuracy ε to $\dfrac{\varepsilon}{2}$.

This decrease requires more computations. The number of computations is, crudely speaking, proportional to the number of nodes. Since the elements fill the original area, and each element has volume h^3, the number of such elements is proportional to h^{-3}.

So, if we go from the original value h to the smaller value h', then we increase the number of computations by a factor of

$$K \stackrel{\text{def}}{=} \frac{h^3}{(h')^3}. \tag{53}$$

This leads to decreasing the inaccuracy by a factor of $\dfrac{h}{h'}$, which is equal to $\sqrt[3]{K}$.

For example, in this straightforward approach, if we want to decrease the accuracy in half $\left(\sqrt[3]{K} = 2\right)$, we will have to increase the number of computation steps by a factor of $K = 8$.

An alternative approach: description. An alternative approach is as follows. We select K small vectors $\left(\Delta_1^{(k)}, \ldots, \Delta_n^{(k)}\right)$, $1 \le k \le K$, which add up to 0. For example, we can arbitrarily select the first $K-1$ vectors and take $\Delta p_i^{(K)} = -\sum\limits_{k=1}^{K-1} \Delta_i^{(k)}$.

Then, every time we need to estimate the value $q(p_1, \ldots, p_n)$, instead of computing $Q(p_1, \ldots, p_n)$, we compute the average

$$Q_K(p_1, \ldots, p_n) = \frac{1}{K} \cdot \sum_{k=1}^{K} Q\left(p_1 + \Delta_1^{(k)}, \ldots, p_n + \Delta_n^{(k)}\right). \tag{54}$$

Why this approach decreases inaccuracy. We know that $Q(p_1 + \Delta p_1, \ldots, p_n + \Delta p_n) = q(p_1 + \Delta p_1, \ldots, p_n + \Delta p_n) + \delta q$,

where, in the small vicinity of the original tuple $(p_1, \ldots, p_n)$, the expression $q(p_1 + \Delta p_1, \ldots, p_n + \Delta p_n)$ is linear, and the differences δq are independent random variables with zero mean.

Thus, we have

$$Q_K(p_1, \ldots, p_n) = \frac{1}{K} \cdot \sum_{k=1}^{K} q\left(p_1 + \Delta_1^{(k)}, \ldots, p_n + \Delta_n^{(k)}\right) + \frac{1}{K} \cdot \sum_{k=1}^{K} \Delta q^{(k)}. \quad (55)$$

Due to linearity and the fact that $\sum_{k=1}^{K} \Delta_i^{(k)} = 0$, the first average in (55) is equal to $q(p_1, \ldots, p_n)$. The second average is the average of K independent identically distributed random variables, and we have already recalled that this averaging decreases the inaccuracy by a factor of $\sqrt{K}$.

Thus, in this alternative approach, we increase the amount of computations by a factor of K, and as a result, we decrease the inaccuracy by a factor of $\sqrt{K}$.

The new approach is better than the straightforward one. In general, $\sqrt{K} > \sqrt[3]{K}$. Thus, with the same increase in computation time, the new method provides a better improvement in accuracy than the straightforward approach.

Comment. The above computations only refer to the traditional Finite Element approach, when we approximate each quantity within each element by a *constant*. In many real-life situations, it is useful to approximate each quantity within each element not by a constant, but rather by a *polynomial* of a given order (see, e.g., [13]): by a linear function, by a quadratic function, etc. In this case, for each element size h, we have smaller approximation error but larger amount of computations. It is desirable to extend the above analysis to such techniques as well.

ACKNOWLEDGMENT

This work was supported in part by the National Science Foundation grants HRD-0734825 and HRD-1242122 (Cyber-ShARE Center of Excellence) and DUE-0926721. The authors are greatly thankful to the anonymous referees for valuable suggestions.

REFERENCES

[1] M. G. Averill, *Lithospheric Investigation of the Southern Rio Grande Rift*, University of Texas at El Paso, Department of Geological Sciences, PhD Dissertation, 2007.

[2] D. G. Cacuci, *Sensitivity and Uncertainty Analysis: Theory*, Chapman & Hall/CRC, Boca Raton, Florida, 2007.

[3] J. A. Hole, "Nonlinear high-resolution three-dimensional seismic travel time tomography, *Journal of Geophysical Research*, 192, Vol. 97, No. B5, pp. 6553–6562.

[4] V. Kreinovich, "Error estimation for indirect measurements is exponentially hard", *Neural, Parallel, and Scientific Computations*, 1994, Vol. 2, No. 2, pp. 225–234.

[5] V. Kreinovich, "Interval Computations and Interval-Related Statistical Techniques: Tools for Estimating Uncertainty of the Results of Data Processing and Indirect Measurements", In: F. Pavese and A. B. Forbes (eds.), *Data Modeling for Metrology and Testing in Measurement Science*, Birkhauser-Springer, Boston, 2009, pp. 117–145.

[6] V. Kreinovich, J. Beck, C. Ferregut, A. Sanchez, G. R. Keller, M. Averill, and S. A. Starks, "Monte-Carlo-type techniques for processing interval uncertainty, and their potential engineering applications", *Reliable Computing*, 2007, Vol. 13, No. 1, pp. 25–69.

[7] V. Kreinovich and S. Ferson, "A New Cauchy-Based Black-Box Technique for Uncertainty in Risk Analysis", *Reliability Engineering and Systems Safety*, 2004, Vol. 85, No. 1–3, pp. 267–279.

[8] V. Kreinovich, A. Lakeyev, J. Rohn, and P. Kahl, *Computational Complexity and Feasibility of Data processing and Interval Computations*, Kluwer, Dordrecht, 1997.

[9] P. Pinheiro da Silva, A. Velasco, M. Ceberio, C. Servin, M. G. Averill, N. Del Rio, L. Longpré, and V. Kreinovich, "Propagation and provenance of probabilistic and interval uncertainty in cyberinfrastructure-related data processing and data fusion", In: R. L. Muhanna and R. L. Mullen (eds.), *Proceedings of the International Workshop on Reliable Engineering Computing REC'08*, Savannah, Georgia, February 20–22, 2008, pp. 199–234.

[10] S. Rabinovich, *Measurement Errors and Uncertainties: Theory and Practice*, Springer Verlag, New York, 2005.

[11] A. Saltelli, K. Chan, and E. M. Scott, *Sensitivity Analysis*, Wiley, Chichester, UK, 2009.

[12] D. J. Sheskin, *Handbook of Parametric and Nonparametric Statistical Procedures*, Chapman & Hall/CRC, Boca Raton, Florida, 2011.

[13] P. Solin, K. Segeth, and I. Dolezel, *Higher-Order Finite Element Methods*, Chapman & Hall/CRC, Boca Raton, Florida, 2003.