On-Line Algorithms for Computing Mean and Variance of Interval Data, and Their Use in Intelligent Systems

Vladik Kreinovich
Computer Science Dept.
University of Texas
El Paso, TX 79968, USA
vladik@cs.utep.edu

Hung T. Nguyen
Dept. of Mathem. Sciences
New Mexico State Univ.
Las Cruces, NM 88003, USA
hunguyen@nmsu.edu

Berlin Wu
Dept. of Mathematical Sciences
National Chengchi University
Taipei, Taiwan
berlin@math.nccu.edu.tw

Abstract

When we have only interval ranges $[\underline{x}_i, \overline{x}_i]$ of sample values $x_1, \ldots, x_n$, what is the interval $[\underline{V}, \overline{V}]$ of possible values for the variance V of these values? There are quadratic time algorithms for computing the exact lower bound $\underline{V}$ on the variance of interval data, and for computing $\overline{V}$ under reasonable easily verifiable conditions. The problem is that in real life, we often make additional measurements. In traditional statistics, if we have a new measurement result, we can modify the value of variance in constant time. In contrast, previously known algorithms for processing interval data required that, once a new data point is added, we start from the very beginning. In this paper, we describe new algorithms for statistical processing of interval data, algorithms in which adding a data point requires only $O(n)$ computational steps.

1 Introduction: Data Processing in Intelligent Systems – From Probabilities to Intervals

Let's start with a big picture. Before we describe a specific problem that we solve in this paper, let us first describe how, in our view, this problem fits into a big picture of information processing in intelligent systems. Readers who are familiar with this big picture and/or who are only interested in our technical results can skip this subsection.

One of the main specific features of information processing in intelligent systems is that in such systems, we often have very limited knowledge. As a result, processing of *imprecise* information is necessary in intelligent systems.

A typical example is the processing of linguistic information, i.e., information represented by experts in terms of words from a natural language. This information can be modeled, e.g., by fuzzy sets (see, e.g., [16, 25]). For such modeling, when an expert states that a value is, say, small but not very small, we describe this expert information in terms of an appropriate fuzzy set.

A particular case of such a statement is when an expert states that the actual value is between, say, 0.1 and 0.3. After such a statement, the only information about the actual (unknown) value of the desired quantity is that it belongs to the *interval* $[0.1, 0.3]$ – and each interval (and, more generally, each set) can be viewed as a particular example of a more general concept of a fuzzy set.

Since the knowledge about each quantity is represented in such a form, it is necessary to be able to develop *inference procedures* for such observations. Mathematical analysis of this problem is therefore crucial for designing intelligent systems. In this paper, we analyze an important particular case of this set-valued data. Specifically, in this paper, we investigate the computational aspects of processing interval-valued data. Let us now describe our problem and its motivation in more detail.

Why data processing? In intelligent systems, there are at least two sources of information about physical quantities: measurements and expert estimates.

In many real-life situations, we are interested in the value of a physical quantity y that is difficult or impossible to measure directly and difficult for experts to estimate. Examples of such quantities are the distance to a star and the amount of oil in a given well.

Since we cannot measure or estimate the value y of the desired physical quantity directly, a natural idea is to measure or estimate y *indirectly*. Specifically, we find some easier-to-measure or easier-to-estimate quantities $x_1, \ldots, x_n$ which are related to y by a known relation $y = f(x_1, \ldots, x_n)$. For example, to find the resistance R, we measure or estimate current I and voltage V, and then use the known relation $R = V/I$ to estimate resistance as $\widetilde{R} = \widetilde{V}/\widetilde{I}$. This relation may be a simple functional transformation, or a complex algorithm (e.g., for the amount of oil, a numerical solution to an inverse problem). It is worth mentioning that In the vast majority of these cases, the function $f(x_1, \ldots, x_n)$ that describes the dependence between physical quantities is continuous. In such cases, to estimate y, we first measure or estimate the values of the quantities $x_1, \ldots, x_n$, and then we use the results $\widetilde{x}_1, \ldots, \widetilde{x}_n$ of these measurements or estimates to to compute an estimate $\widetilde{y}$ for y as $\widetilde{y} = f(\widetilde{x}_1, \ldots, \widetilde{x}_n)$.

Comment. In this paper, for simplicity, we consider the case when the relation between x_i and y is known exactly; in practical situations, we often only know an approximate relation between x_i and y.

Why interval computations? From probabilities to intervals. Neither measurements nor estimates are 100% accurate, so in reality, the actual value x_i of quantity i can differ from the result $\widetilde{x}_i$ obtained by measurement or by estimation. Because of these *measurement (estimation) errors* $\Delta x_i \stackrel{\text{def}}{=} \widetilde{x}_i - x_i$, the result $\widetilde{y} = f(\widetilde{x}_1, \ldots, \widetilde{x}_n)$ of data processing is, in general, different from the actual value $y = f(x_1, \ldots, x_n)$ of the desired quantity y [29]. It is desirable to describe the error $\Delta y \stackrel{\text{def}}{=} \widetilde{y} - y$ of the result of data processing. To do that, we must have some information about the errors of direct measurements and/or estimates.

What do we know about the errors Δx_i related to expert estimation? Often, an expert can provide *bounds* $\underline{x}_i$ and $\overline{x}_i$ for the estimated quantity x_i. Then, the actual (unknown) value of x_i belongs to the interval $\mathbf{x}_i = [\underline{x}_i, \overline{x}_i]$. Often, these bounds come in the form of an unsigned error estimate Δ_i on the expert's estimation accuracy: for example, an expert may say that the actual fish population in a lake is 50,000 ± 20,000. In this case, $\widetilde{x}_i =$50,000, $\Delta_i =$20,000, so $\underline{x}_i = \widetilde{x}_i - \Delta_i$ and $\overline{x}_i = \widetilde{x}_i + \Delta_i$.

Comment. For readers who may be interested in how the above description is related to fuzzy sets, here is an explanation. Often, in addition to (or instead of) the bounds, an expert can provide bounds that contain x_i with a certain degree of confidence (not necessarily represented by a probability). Often, we know several such bounding intervals corresponding to different degrees of confidence. Such a nested family of intervals is also called a *fuzzy set*, because it turns out to be equivalent to a more traditional definition of fuzzy set [6, 16, 23, 24, 25] (if a traditional fuzzy set is given, then different intervals from the nested family can be viewed as α-cuts corresponding to different levels of uncertainty α).

What do we know about the errors Δx_i of direct measurements? First, the manufacturer of the measuring instrument must supply us with an upper bound Δ_i on the measurement error. If no such upper bound is supplied, this means that no accuracy is guaranteed, and the corresponding "measuring instrument" is practically useless. In this case, once we perform a measurement and get a measurement result $\widetilde{x}_i$, we know that the actual (unknown) value x_i of the measured quantity belongs to the interval $\mathbf{x}_i = [\underline{x}_i, \overline{x}_i]$, where $\underline{x}_i = \widetilde{x}_i - \Delta_i$ and $\overline{x}_i = \widetilde{x}_i + \Delta_i$.

In many practical situations, we not only know the interval $[-\Delta_i, \Delta_i]$ of possible values of the measurement or estimation error; we also know the probability of different values Δx_i within this interval. This knowledge underlies the traditional engineering approach to estimating the error of indirect measurement, in which we assume that we know the probability distributions for measurement errors Δx_i.

In practice, we can determine the desired probabilities of different values of Δx_i by comparing the results of measuring with this instrument (or results of expert estimation) with the results of measuring the same quantity by a standard (much more accurate) measuring instrument. Since the standard measuring instrument is much more accurate than the one used, the difference between these two measurement results is practically equal to the measurement error; thus, the empirical distribution of this difference is close to the desired probability distribution for measurement error. There are two cases, however, when this determination is not done:

- First is the case of cutting-edge measurements, e.g., measurements in fundamental science. When the Hubble telescope detects the light from a distant galaxy, there is no "standard" (much more accurate) telescope floating nearby that we can use to calibrate the Hubble: the Hubble telescope is the best we have.

- Second is the case of many commercial measuring instruments. In this case, in principle, every sensor can be thoroughly calibrated, but sensor calibration is so costly – usually costing ten times more than the sensor itself – that manufacturers rarely do it.

In both cases, we have no information about the probabilities of the Δx_i; the only information we have is the upper bound on the measurement or estimation error. Therefore, after we performed a measurement and got a measurement result $\widetilde{x}_i$, the only information that we have about the actual value x_i of the measured quantity is that it belongs to the interval $\mathbf{x}_i = [\widetilde{x}_i - \Delta_i, \widetilde{x}_i + \Delta_i]$. In such situations, the only information that we have about the actual value of $y = f(x_1, \ldots, x_n)$ is that y belongs to the range $\mathbf{y} = [\underline{y}, \overline{y}]$ of the function f over the box $\mathbf{x}_1 \times \ldots \times \mathbf{x}_n$:

$$\mathbf{y} = [\underline{y}, \overline{y}] = \{f(x_1, \ldots, x_n) \,|\, x_1 \in \mathbf{x}_1, \ldots, x_n \in \mathbf{x}_n\}.$$

For continuous functions $f(x_1, \ldots, x_n)$, this range is an interval. The process of computing this interval range based on the input intervals $\mathbf{x}_i$ is called *interval computation*; see, e.g., [13, 14, 15, 22].

Comment. When, instead of a single interval, we have several intervals corresponding to different levels of confidence, we must perform interval computations on each level [6, 16, 23, 24, 25].

Interval computations techniques: brief reminder. 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 $f(a, b)$, if we know the intervals $\mathbf{a}$ and $\mathbf{b}$ for a and b, we can compute the exact range $f(\mathbf{a}, \mathbf{b})$. The corresponding formulas form the so-called *interval arithmetic*. For example,

$$[\underline{a}, \overline{a}] + [\underline{b}, \overline{b}] = [\underline{a} + \underline{b}, \overline{a} + \overline{b}]; \quad [\underline{a}, \overline{a}] - [\underline{b}, \overline{b}] = [\underline{a} - \overline{b}, \overline{a} - \underline{b}];$$

$$[\underline{a}, \overline{a}] \cdot [\underline{b}, \overline{b}] = [\min(\underline{a} \cdot \underline{b}, \underline{a} \cdot \overline{b}, \overline{a} \cdot \underline{b}, \overline{a} \cdot \overline{b}), \max(\underline{a} \cdot \underline{b}, \underline{a} \cdot \overline{b}, \overline{a} \cdot \underline{b}, \overline{a} \cdot \overline{b})].$$

In straightforward interval computations, we repeat the computations forming the expression for f (or, more generally, a program for computing f) step-by-step, replacing each operation on real numbers by the corresponding operation on intervals. It is known that, as a result, we get an enclosure $\mathbf{Y} \supseteq \mathbf{y}$ for the desired range.

In some cases, this enclosure is exact. In more complex cases (see examples below), the enclosure has excess width.

There exist more sophisticated techniques for producing a narrower enclosure, e.g., a centered form method. However, for each of these techniques, there are cases when we get an excess width. The reason for such an excess width is that, as shown in [18, 32], the problem of computing the exact range is known to be NP-hard even for polynomial functions $f(x_1, \ldots, x_n)$ (actually, even for quadratic functions f).

In this paper, we analyze a specific class of interval computations problems – when the algorithm $f(x_1, \ldots, x_n)$ is one of the traditional statistical data processing algorithms such as computing the mean or a variance of the population sample $x_1, \ldots, x_n$.

From the statistical viewpoint, this problem is a particular case of robust statistics. Interval uncertainty means that we do not know the exact probability distribution for measurement or estimation error; instead, we only know that this distribution belongs to a known *collection* of distribution – namely, to the collection of all probability distributions that are non-zero only in the given interval. Situations when we only know a collection of distributions are described by *robust statistics* (see, e.g., [12]), and our problem of estimating population variance is in line with the problems traditionally solved by robust statistics: many known algorithms in the area of robust statistics also return a guaranteed robust estimate for the population mean and population variance, which holds for a collection of distributions.

One may expect that these problems have already been solved in robust statistics. However, while robust statistics does have a lot of useful and interesting results about the guaranteed bounds on the mean for many classes of distributions, the problem of how to actually compute guaranteed bounds on the population variance has not (as we have been able to determine) yet been solved.

Comment. In this paper, we solve a very specific problem related to a combination of interval and probabilistic uncertainty. For a more general context and for other practical problems related to such a combination, see, e.g., [2, 3, 4, 5, 7, 10, 20, 21, 23, 30, 31, 33, 34] and references therein.

2 Error Estimation for Traditional Statistical Data Processing Algorithms under Interval Uncertainty: Known Results

Formulation of the problem. When we have n results $x_1, \ldots, x_n$ of repeated measurement or repeated expert estimation of the same quantity (at different points, or at different moments of time, or by different experts), traditional statistical approach usually starts with computing their population mean $E = (x_1 + \ldots + x_n)/n$ and their (population) variance

$$V = \frac{(x_1 - E)^2 + \ldots + (x_n - E)^2}{n} \tag{1}$$

(or, equivalently, the population standard deviation $\sigma = \sqrt{V}$); see, e.g., [29].

In this paper, we consider situations when we do not know the exact values of the quantities $x_1, \ldots, x_n$, we only know the intervals $\mathbf{x}_1, \ldots, \mathbf{x}_n$ of possible values of x_i. In such situations, for different possible values $x_i \in \mathbf{x}_i$, we get different values of E and V. The question is: what are the intervals $\mathbf{E}$ and $\mathbf{V}$ of possible values of E and V?

The practical importance of this question was emphasized, e.g., in [26, 27] on the example of processing geophysical data.

Bounds on E. For E, the straightforward interval computations leads to the exact range:

$$\mathbf{E} = \frac{\mathbf{x}_1 + \ldots + \mathbf{x}_n}{n}, \text{ i.e., } \underline{E} = \frac{\underline{x}_1 + \ldots + \underline{x}_n}{n}, \text{ and } \overline{E} = \frac{\overline{x}_1 + \ldots + \overline{x}_n}{n}.$$

For variance, the problem is difficult. For V, straightforward interval computations lead to excess width. For example, for $n = 2$, we have $E = (x_1 + x_2)/2$, hence $x_1 - E = (x_1 - x_2)/2$; similarly, $x_2 - E = (x_2 - x)1)/2$. Therefore, $V = ((x_1 - E)^2 + (x_2 - E)^2)/2 = (x_1 - x_2)^2/4$. Thus, when $\mathbf{x}_1 = \mathbf{x}_2 = [0, 1]$, the actual range of the variance is equal to $\mathbf{V} = [0, 0.25]$. On the other hand, $\mathbf{E} = [0, 1]$, hence

$$\frac{(\mathbf{x}_1 - \mathbf{E})^2 + (\mathbf{x}_2 - \mathbf{E})^2}{2} = [0, 1] \supset [0, 0.25].$$

Even more sophisticated methods of interval computations also sometimes lead to an excess width.

Reason: in the formula for the mean E, each variable only occurs once, and it is known that for such formulas, straightforward interval computations lead to the exact range (see, e.g., [11]). In the expression for variance, each variable x_i occurs several times: explicitly, in $(x_i - E)^2$, and implicitly, in the expression for E. In such cases, often, dependence between intermediate computation results leads to excess width in the results of straightforward interval computations. Not surprisingly, we do get excess width when applying straightforward interval computations to formula (1).

For variance, it is known that computing $\overline{V}$ is NP-hard [8]. The very fact that computing the range of a quadratic function is NP-hard was first proven by Vavasis [32] (see also [18]). [8] shows that this difficulty happens even for very simple quadratic functions frequently used in data processing.

A natural question is: does the difficulty come from the requirement that the range be computed exactly? In practice, it is often sufficient to compute, in a reasonable amount of time, a usefully accurate estimate $\widetilde{\overline{V}}$ for $\overline{V}$, i.e., an estimate $\widetilde{\overline{V}}$ which is accurate with a given accuracy $\varepsilon > 0$: $\left|\widetilde{\overline{V}} - \overline{V}\right| \le \varepsilon$. Alas, it can be shown (see, e.g., [8]), that for any ε, such computations are also NP-hard.

It is worth mentioning that $\overline{V}$ can be computed exactly in exponential time $O(2^n)$: it is sufficient to try all 2^n possible combinations of values $\underline{x}_i$ and $\overline{x}_i$ [8].

Feasible algorithm for computing $\underline{V}$. For computing $\underline{V}$, there exists a feasible algorithm [8]: specifically, our algorithm is *quadratic-time*, i.e., it requires $O(n^2)$ computational steps (arithmetic operations or comparisons) for n interval data points $\mathbf{x}_i = [\underline{x}_i, \overline{x}_i]$.

This algorithm $\underline{\mathcal{A}}$ is as follows:

- First, we sort all $2n$ values $\underline{x}_i$, $\overline{x}_i$ into a sequence $x_{(1)} \le x_{(2)} \le \ldots \le x_{(2n)}$.

- Second, we compute $\underline{E}$ and $\overline{E}$ and select all "zones" $[x_{(k)}, x_{(k+1)}]$ that intersect with $[\underline{E}, \overline{E}]$.

- For each of the selected zones $[x_{(k)}, x_{(k+1)}]$, we compute the ratio $r_k = S_k/N_k$, where

$$S_k \stackrel{\text{def}}{=} \sum_{i:\underline{x}_i \ge x_{(k+1)}} \underline{x}_i + \sum_{j:\overline{x}_j \le x_{(k)}} \overline{x}_j, \qquad (2)$$

  and N_k is the total number of such is and js . If $r_k \in [x_{(k)}, x_{(k+1)}]$, then we compute $V_k = W_k/n$, where

$$W_k \stackrel{\text{def}}{=} \sum_{i:\underline{x}_i \ge x_{(k+1)}} (\underline{x}_i - r_k)^2 + \sum_{j:\overline{x}_j \le x_{(k)}} (\overline{x}_j - r_k)^2. \qquad (3)$$

  If $N_k = 0$, we take $V_k \stackrel{\text{def}}{=} 0$.

- Finally, we return the smallest of the values V_k as $\underline{V}$.

Comment. A reader may be somewhat puzzled by the fact that there is a feasible algorithm for computing the smallest possible value $\underline{V}$ of the variance V, while the problem of computing its largest possible value $\overline{V}$ is NP-hard.

Such a difference could not be possible, e.g., for the mean E. Indeed, for the mean, $E(-x_1, \ldots, -x_n) = -E(x_1, \ldots, x_n)$. Since replacing E with $-E$ reverses the order, the smallest possible value of $E(-x_1, \ldots, -x_n)$ corresponds to the largest possible value of $E(x_1, \ldots, x_n)$: $\underline{E}(-\mathbf{x}_1, \ldots, -\mathbf{x}_n) = -\overline{E}(\mathbf{x}_1, \ldots, \mathbf{x}_n)$. Thus, if we have a fast algorithm for computing $\underline{E}$, then we could apply it to the intervals $-\mathbf{x}_1, \ldots, -\mathbf{x}_n$ and therefore, compute $\overline{E}$ as $\overline{E}(\mathbf{x}_1, \ldots, \mathbf{x}_n) = -\underline{E}(-\mathbf{x}_1, \ldots, -\mathbf{x}_n)$. For the variances, however, this idea will not work because $V(-x_1, \ldots, -x_n) = V(x_1, \ldots, x_n)$ hence $V(-\mathbf{x}_1, \ldots, -\mathbf{x}_n) = V(\mathbf{x}_1, \ldots, \mathbf{x}_n)$.

Another explanation is that the function $V(x_1, \ldots, x_n)$ is a convex function. It is known that computing a minimum of a convex function is rather easy, but computing its maximum can be time-consuming.

Feasible algorithm for computing $\overline{V}$. NP-hardness of computing $\overline{V}$ means, crudely speaking, that there are no general ways for solving all particular cases of this problem (i.e., computing $\overline{V}$) in reasonable time.

However, there are algorithms for computing $\overline{V}$ for certain common situations. For example, there exists an efficient ($O(n^2)$) algorithm [8] that computes $\overline{V}$ for the case when all the interval midpoints ("measured or estimated values") $\widetilde{x}_i = (\underline{x}_i + \overline{x}_i)/2$ are definitely different from each other, in the sense that the "narrowed" intervals $[\widetilde{x}_i - \Delta_i/n, \widetilde{x}_i + \Delta_i/n]$ – where $\Delta_i = (\underline{x}_i - \overline{x}_i)/2$ is the interval's half-width – do not intersect with each other. This situation is common because the actual values $x_1, \ldots, x_n$ are usually different, so if we measure them with a sufficient accuracy, we get non-intersecting intervals $[\widetilde{x}_i - \Delta_i/n, \widetilde{x}_i + \Delta_i/n]$.

This algorithm $\overline{\mathcal{A}}$ is as follows:

- First, we sort all $2n$ endpoints of the narrowed intervals $\widetilde{x}_i - \Delta_i/n$ and $\widetilde{x}_i + \Delta_i/n$ into a sequence $x_{(1)} \le x_{(2)} \le \ldots \le x_{(2n)}$. This enables us to divide the real line into $2n + 1$ segments ("zones") $[x_{(k)}, x_{(k+1)}]$, where we denoted $x_{(0)} \stackrel{\text{def}}{=} -\infty$ and $x_{(2n+1)} \stackrel{\text{def}}{=} +\infty$.
- Second, we compute $\underline{E}$ and $\overline{E}$ and pick all "zones" $[x_{(k)}, x_{(k+1)}]$ that intersect with $[\underline{E}, \overline{E}]$.
- For each of remaining zones $[x_{(k)}, x_{(k+1)}]$, for each i from 1 to n, we pick the following value of x_i:
  - if $x_{(k+1)} < \widetilde{x}_i - \Delta_i/n$, then we pick $x_i = \overline{x}_i$;
  - if $x_{(k)} > \widetilde{x}_i + \Delta_i/n$, then we pick $x_i = \underline{x}_i$;
  - for all other i, we consider both possible values $x_i = \overline{x}_i$ and $x_i = \underline{x}_i$.

  As a result, we get one or several sequences of x_i. For each of these sequences, we check whether the mean E of the selected values $x_1, \ldots, x_n$ is indeed within this zone, and if it is, compute the variance by using the formula (2).
- Finally, we return the largest of the computed variances as $\overline{V}$.

This algorithm also works when, for some fixed K, any collection of more than K "narrowed" intervals does not have a common point.

3 On-Line Statistical Analysis: Problem and Results

Formulation of the problem. In practice, measurements and expert estimates can arrive one after another. It is desirable to start processing them as they come, without waiting for all of them to arrive. This is also important because often, as a result of the statistical analysis of the existing measurement results and/or expert estimates, we conclude that we do not have enough measurements and estimates; hence, we make additional measurements or expert estimates. For traditional statistical methods, this can be easily accomplished: once we know the mean E of n values $x_1, \ldots, x_n$ and the corresponding variance V, and a new measurement result (or a new expert estimate) x_{n+1} arrives, we can compute the new values E' and V' as follows:

$$E' = \frac{n \cdot E + x_{n+1}}{n+1}; \quad M = V + E^2; \quad M' = \frac{n \cdot M + x_{n+1}^2}{n+1}; \quad V' = M' - (E')^2,$$

where $M \stackrel{\rm def}{=} \dfrac{x_1^2 + \ldots + x_n^2}{n}$ is a (population) second moment.

Comment. For readers who are not very familiar with formulas from mathematical statistics, here is a simple derivation of the formula $M = E + V^2$: since $(x_i - E)^2 = x_i^2 - 2x_i \cdot E + E^2$, the average V of the squares $(x_i - E)^2$ can be represented as follows:

$$V = \frac{1}{n} \cdot \sum_{i=1}^{n} (x_i - E)^2 = \frac{1}{n} \cdot \sum_{i=1}^{n} x_i^2 - \left(\frac{1}{n} \cdot \sum_{i=1}^{n} x_i \right) \cdot E + E^2 = M - 2E^2 + E^2 = M - E^2.$$

In other words, if we have a new measurement result or a new expert estimate, we can modify the value of the variance in constant time – i.e., by using the number of computational steps that does not grow with n. The above formulas enable us to easily update the statistical characteristics once the new measurement results and/or expert estimates are available.

Similar algorithms can be described for computing $\underline{E}$ and $\overline{E}$:

$$\underline{E}' = \frac{n \cdot \underline{E} + \underline{x}_{n+1}}{n+1}; \quad \overline{E}' = \frac{n \cdot \overline{E} + x_{n+1}}{n+1}. \tag{4}$$

However, the above algorithms for computing $\underline{V}$ and $\overline{V}$ start with sorting the values $\underline{x}_i$ and $\overline{x}_i$. Thus, we cannot even start these algorithms unless we already know all the (interval) values $\mathbf{x}_1, \ldots, \mathbf{x}_n$ before we start computations.

So, if we have a new measurement result or a new expert estimate, and we want to recompute the bounds on V, we must start from scratch and again apply $O(n^2)$ computational steps. Thus, if we add measurement results/expert estimates one by one, we need $O(1^2 + 2^2 + \ldots + n^2) = O(n^3)$ computational steps.

A natural question is: if we simply add a new (interval) value $\mathbf{x}_{n+1}$, can we use the previous computations to re-compute $\mathbf{V}$ faster? In this paper, we show that such a speed-up is indeed possible. Specifically, we will show that it is possible to modify the algorithms in such a way that each algorithm requires only $O(n)$ steps after a new data point x_{n+1} is added. In these new algorithms, to process n measurement results and expert estimates one after another, we need $O(1 + 2 + \ldots + n) = O(n^2)$ computational steps – same as before, but now we do not have to wait until all the measurement results and expert estimates are available.

New algorithm for computing $\underline{V}$: main idea. This new algorithm is a modification of the above described algorithm $\underline{\mathcal{A}}$. Let us first describe the main three differences between the new algorithm and the previous one.

The first difference is that, in contrast to $\underline{\mathcal{A}}$, we will compute the values S_k, N_k, r_k, and V_k for *all* zones $[x_{(k)}, x_{k+1})]$, not just for the zones that intersect with $[\underline{E}, \overline{E}]$ and/or for which r_k belongs to the zone. (Of course, when we compute $\underline{V}$, we compute only the smallest of the values V_k corresponding to the zones that intersect with $\mathbf{E}$ and for which r_k belongs to the zone.)

Second, instead of computing V_k by using formula (3), we use the following equivalent formula:

$$W_k = M_k - 2S_k \cdot r_k + N_k \cdot r_k^2, \tag{5}$$

where

$$M_k \stackrel{\rm def}{=} \sum_{i:\underline{x}_i \ge x_{(k+1)}} \underline{x}_i^2 + \sum_{j:\overline{x}_j \le x_{(k)}} \overline{x}_j^2. \tag{6}$$

Comment. The formula (5) is similar to the above-mentioned known relation $V = M - E^2$ between the variance V, the second moment M, and the mean E, and its proof is similar to the proof of that relation. Indeed, since $(x-r_k)^2 = x^2-2x\cdot r_k+r_k^2$, we can represent each of the two sums in the formula (3) as three sums, corresponding to the sums of $\underline{x}_i^2$ and $\overline{x}_i^2$, the sum of $\underline{x}_i$ and $\overline{x}_i$, and the sum of N_k identical terms r_k^2. The first sum leads to M_k, the second sum – by definition of S_k – leads to $-2S_k \cdot r_k$, and the third sum results in $N_k \cdot r_k^2$.

The third difference is that at the end of this algorithm, we keep not only the final value $\underline{V}$, but we also keep all the intermediate computational results: the sequence $x_{(i)}$, the values $\underline{E}$ and $\overline{E}$, and the values S_k, N_k, r_k, M_k, and V_k.

New algorithm for computing $\underline{V}$: description and computational complexity. Let us now describe how this new algorithm works. Suppose that we have already finished applying the algorithm to n intervals $\mathbf{x}_1, \ldots, \mathbf{x}_n$, and a new interval $\mathbf{x}_{n+1} = [\underline{x}_{n+1}, \overline{x}_{n+1}]$ arrives.

First, we recompute the values $\underline{E}$ and $\overline{E}$ by applying the formulas (4). This requires a constant number of computational steps.

Then, we find the place for the new bounds $\underline{x}_{n+1}$ and $\overline{x}_{n+1}$ in the sorted sequence $x_{(1)} \le x_{(2)} \le \ldots \le x_{(2n)}$. Since the sequence $x_{(i)}$ is sorted, finding a place for each of the bounds within this sequence can be done by bisection (binary search), i.e., in $O(\log(n))$ steps (see, e.g., [1]).

Each of the added bounds is either within one of the previous zone – in which case this zone splits into two new smaller zones, or it is before or after all the previous zones – in which case a single new zone is added. In both cases, adding one bound adds at most two new zones, so adding two bounds means that we have at most 4 new zones.

To proceed, we must update the values S_k, N_k, r_k, M_k, and V_k corresponding to the old zones, and compute the values S_k, N_k, r_k, M_k, and V_k corresponding to the new zones.

For each old zone $[x_{(k)}, x_{(k+1)}]$, the value of S_k will only change if either $\underline{x}_{n+1} \ge x_{(k+1)}$ or $\overline{x}_{n+1} \ge x_{(k)}$. In the first case, we add $\underline{x}_{n+1}$ to S_k; in the second case, we add $\overline{x}_{n+1}$ to S_k. In both cases, we add 1 to N_k.

Similarly, the value of M_k will only change if we either $\underline{x}_{n+1} \ge x_{(k+1)}$ or $\overline{x}_{n+1} \ge x_{(k)}$. In the first case, we add $\underline{x}_{n+1}^2$ to M_k; in the second case, we add $\overline{x}_{n+1}^2$ to S_k.

For each old zone k, once the values of S_k, N_k, and M_k are updated, we can compute r_k and V_k in $O(1)$ steps, i.e., by using the number of computational steps that remains bounded by a constant when n increases and thus, does not increase with n.

Thus, for each old zone, we need $O(1)$ computational steps for the update.

For each new zone, explicit computation of S_k and M_k requires that we go over all n intervals, i.e., it requires linear time $O(n)$.

Thus, the update of all intermediate values requires a constant time $O(1)$ for each of $O(n)$ old zones and a linear time $O(n)$ for each of $O(1)$ new zones. Therefore, the total number of computational steps needed for an update is equal to $O(1) \cdot O(n) + O(n) \cdot O(1) = O(n)$. In other words, we need linear time to update.

Finally, we compute $\underline{V}$ as the smallest of $\le n$ values V_k; this also requires linear time. We have therefore proven that our algorithm indeed requires linear time to update the lower bound $\underline{V}$ on the variance V.

New algorithm for computing $\underline{V}$: numerical example. Let us illustrate the above algorithm on the example when we process the following 3 intervals: $\mathbf{x}_1 = [2.1, 2.6]$, $\mathbf{x}_2 = [2.0, 2.1]$, and $\mathbf{x}_3 = [2.2, 2.9]$. We start with the interval $\mathbf{x}_1 = [2.1, 2.6]$. We only have a single interval, so we only have two bounds: 2.1 and 2.6. These bounds are endpoints of the same interval, so they are already sorted, hence $x_{(1)} = 2.1$ and $x_{(2)} = 2.6$. This is a degenerate case. In this case, we have only one zone $[x_{(1)}, x_{(2)}] = [2.1, 2.6]$. For this zone, $S_1 = 0$, $N_1 = 0$, $r_1 = S_1/N_1$ is undefined, $M_1 = 0$, and $V_1 = 0$.

Then, we add the second interval $\mathbf{x}_2 = [2.0, 2.1]$. To get the ordering of all 4 bounds, we must find the place for the two new bounds, $\underline{x}_2 = 2.0$ and $\overline{x}_2 = 2.1$, in the sorted sequence $x_{(1)} = 2.1 < x_{(2)} = 2.6$. We find the place for each of these bounds by bisection, so we get $2.0 < 2.1 = 2.1 < 2.6$. No new bounds split the old zone $[2.1, 2.6]$, so this zone remains, In addition to this old zone, we also have a new zone $[2.0, 2.1]$.

In accordance with the algorithm, let us start with re-computing the values $S_k, \ldots$ corresponding to the old zone. The new interval $\mathbf{x}_2$ is completely to the left of the old zone, so its upper bound 2.1 is added to S_1 and 1 to N_k. As a result, for this zone, we get $S = 0+2.1 = 2.1$ and $N = 0+1 = 1$. Hence, for this zone, $r = S/N = 2.1$.

Similarly, the value M changes by adding 2.1^2, so the new value of M is $0+2.1^2=4.41$. Finally, we compute

$$V=\frac{M-2S\cdot r+N\cdot r^2}{n}=\frac{4.41-2\cdot 2.1\cdot 2.1+1\cdot 2.1^2}{2}=0.$$

For the new zone, we explicitly compute S and M. In our case, $S=2.1$, $N=1$, $r=S/N=2.1$, $M=2.1^2=4.41$, and

$$V=\frac{4.41-2\cdot 2.1\cdot 2.1+1\cdot 2.1^2}{2}=0.$$

Let us now add the third interval $\mathbf{x}_3=[2.2,2.9]$. First, we find the place for the new bounds 2.2 and 2.9 in the sorted sequence $2.0<2.1<2.6$. As a result, we get an enlarged sorted sequence $2.0<2.1<2.2<2.6<2.9$. The zone $[2.0,2.1]$ stays, the zone $[2.1,2.6]$ is now split into two new zones: $[2.1,2.2]$ and $[2.2,2.6]$, and a new zone $[2.6,2.9]$ has appeared.

For the old zone $[2.0,2.1]$, since $\underline{x}_3=2.2$ is larger than the upper bound of this zone, we recalculate S by adding the value 2.2 corresponding to the new interval $\mathbf{x}_3$, i.e., replace the old value $S=2.1$ with $S=2.1+2.2=4.3$. Correspondingly, we replace the old value $N=1$ with the new value $N=1+1=2$. Hence, $r=S/N=2.15$. Similarly, since $\underline{x}_3\geq 2.1$, the value M is changed from the old value 4.41 to the new value $4.41+2.2^2=7.25$. Hence,

$$V=\frac{7.25-2\cdot 4.3\cdot 2.15+2\cdot 2.15^2}{3}=0.875.$$

For the new zone $[2.1,2.2]$, straightforward computations describe S as $S=2.1+2.2=4.3$ and $N=2$, hence $r=S/N=2.15$. Here, $M=2.1^2+2.2^2=7.25$, hence, similarly to the previous zone, we have $V=0.875$.

For the new zone $[2.2,2.6]$, we have $S=2.1$ and $N=1$, hence $r=S/N=2.1$. Here, $M=2.1^2=4.41$, hence $V=(4.41-2\cdot 2.1\cdot 2.1+1\cdot 4.41)/3=0$.

Finally, for the new zone $[2.6,2.9]$, we have $S=2.1+2.6=4.7$ and $N=2$, hence $r=S/N=2.35$. Here, $M=2.1^2+2.6^2=11.17$, hence

$$V=\frac{11.17-2\cdot 4.7\cdot 2.35+2\cdot 2.35^2}{3}=0.541666\ldots$$

If these three intervals are all we have, then to get the actual value of $\underline{V}$, we consider only those zones for which r is within this zone. Out of our 4 zones, only one zone has this property: $[2.1,2.2]$. For this zone, $V=0.875$, so this is the desired lower endpoint $\underline{V}$.

New algorithm for computing $\overline{V}$: main idea. Let us now describe how we can modify the above algorithm $\overline{\mathcal{A}}$ so that it will require linear time to update for the case when, for some fixed K, any collection of more than K "narrowed" intervals does not have a common point.

Let us first describe the main difference between this modification and the original algorithm.

The first difference is that, in contrast to $\overline{\mathcal{A}}$, we will perform the computations for *all* zones $[x_{(k)},x_{k+1)}]$, not just for the zones that intersect with $[\underline{E},\overline{E}]$. (Of course, when we compute $\overline{V}$, we compute only the largest of the values V corresponding to the zones that intersect with $\mathbf{E}$.)

Second, at the end of this algorithm, we keep not only the final value $\overline{V}$, but we also keep all the intermediate computational results: the sequence $x_{(i)}$, and, for each zone, all selected sequences $x_1,\ldots,x_n$ and the values E and V corresponding to these sequences.

New algorithm for computing $\overline{V}$: description and computational complexity. Let us now describe how this new algorithm works. Suppose that we have already finished applying the algorithm to n intervals $\mathbf{x}_1,\ldots,\mathbf{x}_n$, and a new interval $\mathbf{x}_{n+1}=[\underline{x}_{n+1},\overline{x}_{n+1}]$ arrives.

First, we recompute the values $\underline{E}$ and $\overline{E}$ by applying formulas (4). This requires a constant number of computational steps.

Then, we find the place for the new bounds $\underline{x}_{n+1}$ and $\overline{x}_{n+1}$ in the sorted sequence $x_{(1)}\leq x_{(2)}\leq\ldots\leq x_{(2n)}$. Since the sequence $x_{(i)}$ is sorted, finding a place for each of the bounds within this sequence can be done by bisection (binary search), i.e., in $O(\log(n))$ steps (see, e.g., [1]).

Similarly to the previous modified algorithm, each of the added bounds is either within a previous zone – in which case this zone splits into two new smaller zones, or it is before or after all the previous zones – in which case a single new zone is added. In both cases, adding one bound adds at most two new zones, so adding two bounds means that we have at most 4 new zones.

To proceed, we must update the sequences and the corresponding values E and V corresponding to the old zones, and compute the values corresponding to the new zones.

For each old zone, and for each corresponding sequence, we must update this sequence by adding the corresponding value of x_{n+1}, and then re-computing E and V. Since no more than K narrowed intervals can have a common point, for each zone, there are no more than 2^K corresponding sequences. When K is fixed, this means that we have a constant number $O(1)$ of such sequences. For each sequence, updating E and V can be done (as we have already mentioned) in $O(1)$ steps, i.e., in the number of steps that does not depend on n.

For each new zone, we need to find all the sequences and compute the corresponding values E and V. Finding all the sequences requires $\leq 2^K \cdot n = O(n)$ steps, and computing E and V for each of these sequences also requires linear time.

Thus, the update of all intermediate values requires a constant time $O(1)$ for each of $O(n)$ old zones and a linear time $O(n)$ for each of $O(1)$ new zones. Therefore, the total number of computational steps needed for an update is equal to $O(1) \cdot O(n) + O(n) \cdot O(1) = O(n)$. In other words, we do need linear time to update.

Finally, we compute $\overline{V}$ as the largest of $\leq n$ values V; this also requires linear time. We have therefore proven that our algorithm indeed requires linear time to update the lower bound $\overline{V}$ on the variance V.

Acknowledgments

This work was supported in part by NASA grant NCC5-209, by the AFOSR grant F49620-00-1-0365, by NSF grants EAR-0112968, EAR-0225670, and EIA-0321328, by IEEE/ACM SC2003 Minority Serving Institutions Participation Grant, by the Army Research Laboratories grant DATM-05-02-C-0046, by a research grant from Sandia National Laboratories as part of the Department of Energy Accelerated Strategic Computing Initiative (ASCI), by Applied Biomathematics, and by the Small Business Innovation Research grant 9R44CA81741 from the National Cancer Institute (NCI), a component of NIH.

The authors are thankful to all the participants of InTech'03 conference for valuable discussions.

The anomynous referees' detailed suggestions were so thorough and so helpful that the three referees should be viewed as true co-authors of this paper; we are very thankful to them.

References

[1] Cormen Th. H., Leiserson C. E., Rivest R. L., and Stein C. (2001), *Introduction to Algorithms*, MIT Press, Cambridge, MA.

[2] Berleant D. (1993), "Automatically verified arithmetic with both intervals and probability density functions", *Interval Computations*, No. 2, 48–70.

[3] Berleant D. (1996), "Automatically verified arithmetic on probability distributions and intervals", In: R. B. Kearfott and V. Kreinovich (eds.) *Applications of Interval Computations*, Kluwer, Dordrecht.

[4] Berleant D., and Goodman-Strauss C. (1998), "Bounding the results of arithmetic operations on random variables of unknown dependency using intervals", *Reliable Computing, vol. 4*, No. 2, 147–165.

[5] Berleant D., Xie L., and Zhang J. (2003), "Statool: A Tool for Distribution Envelope Determination (DEnv), an Interval-Based Algorithm for Arithmetic on Random Variables", *Reliable Computing, vol. 9*, No. 2, 91–108.

[6] Bojadziev G. and Bojadziev M. (1995), *Fuzzy Sets, Fuzzy Logic, Applications*, World Scientific, Singapore.

[7] Ferson S. (2002), *RAMAS Risk Calc 4.0: Risk Assessment with Uncertain Numbers*, CRC Press, Boca Raton, Florida.

[8] Ferson S., Ginzburg L., Kreinovich V., Longpré L., and Aviles M. (2002), "Computing Variance for Interval Data is NP-Hard", *ACM SIGACT News, vol. 33*, 108–118.

[9] Ferson S., Ginzburg L., Kreinovich V., and Lopez J. (2002), "Absolute Bounds on the Mean of Sum, Product, etc.: A Probabilistic Extension of Interval Arithmetic", *Extended Abstracts of the 2002 SIAM Workshop on Validated Computing*, Toronto, Canada, May 23–25, 70–72.

[10] Ferson S., Myers D., and Berleant D. (2001), *Distribution-free risk analysis: I. Range, mean, and variance*, Applied Biomathematics, Technical Report, 2001.

[11] Hansen E. (1997), “Sharpness in interval computations”, *Reliable Computing, vol. 3*, 7–29.

[12] Huber P. J. (1981), *Robust statistics*, Wiley, New York.

[13] Jaulin L., Kieffer M., Didrit O., and Walter E. (2001), *Applied Interval Analysis*, Springer-Verlag, Berlin.

[14] Kearfott R. B. (1996), *Rigorous Global Search: Continuous Problems*, Kluwer, Dordrecht.

[15] Kearfott R. B. and Kreinovich V., eds. (1996), *Applications of Interval Computations*, Kluwer, Dordrecht.

[16] Klir G., and Yuan B. (1995), *Fuzzy Sets and Fuzzy Logic: Theory and Applications*, Prentice Hall, Upper Saddle River, New Jersey.

[17] Kreinovich, V. “Probabilities, Intervals, What Next? Optimization Problems Related to Extension of Interval Computations to Situations with Partial Information about Probabilities”, *Journal of Global Optimization* (to appear).

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

[19] Kreinovich V., Nguyen H. T., Ferson S., and Ginzburg L. (2002), “From Computation with Guaranteed Intervals to Computation with Confidence Intervals”, *Proc. 21st Int’l Conf. of North American Fuzzy Information Processing Society NAFIPS’2002*, New Orleans, Louisiana, 418–422.

[20] Kuznetsov V. P. (1991), *Interval Statistical Models*, Radio i Svyaz, Moscow (in Russian).

[21] Lodwick W. A., and Jamison K. D. (2003), “Estimating and Validating the Cumulative Distribution of a Function of Random Variables: Toward the Development of Distribution Arithmetic”, *Reliable Computing, vol. 9*, No. 2, 127–141.

[22] Moore R. E. (1979), *Methods and Applications of Interval Analysis*, SIAM, Philadelphia.

[23] Moore R. E., and Lodwick W. A. (2003), “Interval Analysis and Fuzzy Set Theory”, *Fuzzy Sets and Systems, vol. 135*, No. 1, 5–9.

[24] Nguyen H. T., and Kreinovich V. (1996), “Nested Intervals and Sets: Concepts, Relations to Fuzzy Sets, and Applications”, In: Kearfott R. B. et al., *Applications of Interval Computations*, Kluwer, Dordrecht, 245–290.

[25] Nguyen H. T., and Walker E. A. (1999), *First Course in Fuzzy Logic*, CRC Press, Boca Raton, Florida.

[26] Nivlet P., Fournier F., and Royer J. (2001), “A new methodology to account for uncertainties in 4-D seismic interpretation”, *Proceedings of the 71st Annual International Meeting of the Society of Exploratory Geophysics SEG’2001*, San Antonio, Texas, September 9–14, 1644–1647.

[27] Nivlet P., Fournier F., and Royer J. (2001), “Propagating interval uncertainties in supervised pattern recognition for reservoir characterization”, *Proceedings of the 2001 Society of Petroleum Engineers Annual Conference SPE’2001*, New Orleans, Louisiana, September 30–October 3, paper SPE-71327.

[28] Osegueda R., Kreinovich V., Potluri L., and Aló R. (2002), “Non-Destructive Testing of Aerospace Structures: Granularity and Data Mining Approach”, *Proceedings of FUZZ-IEEE’2002*, Honolulu, Hawaii, May 12–17, *Vol. 1*, 685–689.

[29] Rabinovich S. (1993), *Measurement Errors: Theory and Practice*, American Institute of Physics, New York.

[30] Regan H., Ferson S., and Berleant D., “Equivalence of five methods for bounding uncertainty”, *Journal of Approximate Reasoning* (to appear).

[31] Rowe N. C. (1988), “Absolute bounds on the mean and standard deviation of transformed data for constant-sign-derivative transformations”, *SIAM Journal of Scientific Statistical Computing, vol. 9*, 1098–1113.

[32] Vavasis S. A. (1991), *Nonlinear Optimization: Complexity Issues*, Oxford University Press, N.Y.

[33] Walley P. (1991), *Statistical Reasoning with Imprecise Probabilities*, Chapman and Hall, N.Y.

[34] Williamson R. and Downs T. (1990), “Probabilistic arithmetic I: numerical methods for calculating convolutions and dependency bounds”, *International Journal of Approximate Reasoning, vol. 4*, 89–158.