Sharp polynomial upper bound on the variance

Marco de Angelis

University of Strathclyde, Glasgow, G1 1XJ, United Kingdom
marco.de-angelis@strath.ac.uk

Abstract. This short paper presents a new idea leading to a very sharp upper bound on the variance of interval-valued data. The computation of this sharp bound can be carried out in polynomial time in the worst case. For a whole class of interval data this bound is exact as it can be shown that it coincides with the maximum. The algorithm derives from posing the optimisation problem in probabilistic terms, i.e. thinking beyond the deterministic interpretation of an interval. Interval-valued variance can be seen from an imprecise probability's perspective. There are two alternative and non-competing connotations of an interval: a set of real values, and a credal set of all possible probability measures in the given interval. These two connotations do not precipitate any quantitative discrepancies for interval arithmetic. In fact, interval arithmetic provides a means to compute with these imprecise probabilistic objects. This work demonstrates the computational advantage that originates from looking at intervals from an imprecise probabilistic angle, and it may serve as a testimony towards filling the computational void that has for too long discouraged practitioners from computing with interval statistics.

Keywords: Imprecise statistics, interval variance, interval computation

1 Background

Before this work, computing the variance of a data set whose entries are intervals was NP-hard in the general case, e.g. for nested interval data sets [1]. Because the variance is a quadratic form, and because finding the maximum of general quadratic forms has exponential complexity, it is believed that finding the maximum of the variance is exponential too. However if one looks at the problem from a probabilistic standpoint, new exciting polynomial approximations become possible. After all, the variance is not just any quadratic form. The computation of the variance lower and upper bounds for interval-valued data has been extensively studied by Ferson and Kreinovich and documented in [2], who discovered for the lower bound a *quadratic* algorithm in the worst case. In the same study, it is shown that computing the upper bound has exponential complexity in the worst case. Due to such exponential computational barrier, Ferson et al. [2] have resorted to pre-processing strategies for classifying interval-valued data (e.g. non-overlapping, thin, scattered, nested, etc.) to isolate the cases where such an upper bound can be computed in linear or quadratic time [3]. Nested intervals belong to the hardest class: they are known

to have a zero lower bound but an upper bound only computable in exponential time. Other cases displaying concurrent overlaps and nesting can also fall in this class. Kreinovich's algorithms implicitly assume that the underlying data are real-valued points, about which there is uncertainty in the form of intervals. Such intervals will be denominated *real-valued* or *point-valued* to emphasise that there is a single point value somewhere in each interval. In this paper, we show that if we go beyond point-valued intervals new more efficient algorithms can be discovered.

2 Problem statement

The *biased*, or *uncorrected*, or simply *population* variance of a collection of n real-valued numbers x_i is

$$\begin{aligned} V &= \frac{1}{n}\sum (x_i - \mu)^2 \\ &= \frac{1}{n}\sum x_i^2 - \left(\frac{1}{n}\sum x_i\right)^2 . \end{aligned} \tag{1}$$

where $\mu = \sum x_i/n$ is the arithmetic mean. If the data x_i are imprecisely characterised as intervals this formula cannot conveniently be used because of the dependence problem of repeated variables in interval computation.

2.1 Intervals

An interval is denoted $[x] = [\underline{x}, \overline{x}] = \{x \in \mathbb{R} : \underline{x} \le x \le \overline{x}\} \in \mathbb{IR}$, where $\underline{x}$ is the lower bound and $\overline{x}$ is the upper bound and $\mathbb{IR}$ is the space of intervals. The interval arithmetic operations between two intervals $[\underline{x}, \overline{x}]$ and $[\underline{y}, \overline{y}]$ for computing (1) are

$$\begin{aligned} &[\underline{x}, \overline{x}] + [\underline{y}, \overline{y}] = [\underline{x} + \underline{y}, \overline{x} + \overline{y}], \\ &[\underline{x}, \overline{x}] - [\underline{y}, \overline{y}] = [\underline{x} - \overline{y}, \overline{x} - \underline{y}], \\ &[\underline{x}, \overline{x}]^2 = [\max\{0,\ \operatorname{sgn}(\underline{x}\overline{x}) \min\{\underline{x}^2, \overline{x}^2\}\},\ \max\{\underline{x}^2, \overline{x}^2\}]. \end{aligned}$$

While these operations are guaranteed to enclose the true answer in the respective intervals, their use in expressions with repeated variables yields inflated intervals due to the dependence problem [4]. The midpoint and width of an interval are

$$\begin{aligned} m &= \operatorname{mid}\ [\underline{x}, \overline{x}] = (\underline{x} + \overline{x})/2, \\ w &= \operatorname{wid}\ [\underline{x}, \overline{x}] = \overline{x} - \underline{x}. \end{aligned}$$

An n-interval $[\mathrm{x}] \in \mathbb{IR}^n$ is a vector whose entries are intervals

$$[\mathrm{x}] = \left([\underline{x}_1, \overline{x}_1], ..., [\underline{x}_n, \overline{x}_n]\right) = \left([x_1], ..., [x_n]\right).$$

A real-valued function $f : \mathbb{R}^n \to \mathbb{R}^m$ whose expression f is evaluated with interval operations satisfies the fundamental theorem of interval analysis. Such an interval extension $[\mathsf{f}] : \mathbb{IR}^n \to \mathbb{IR}^m$ is said to be inclusion monotonic $[\mathsf{f}]([\mathrm{x}]) \subseteq [\mathsf{f}]([\mathrm{y}])$, whenever $[\mathrm{x}] \subseteq [\mathrm{y}]$.

2.2 Quadratic form

In standard compact matrix form the interval extension of $V = f(\mathrm{x})$ is

$$\mathsf{f}([\mathrm{x}]) = [\mathrm{x}]^T \; Q \; [\mathrm{x}],$$

where $Q = I - E/n$, I is the $n \times n$ identity and E is $n \times n$ matrix of ones. The maximum for V follows from the optimization problem

$$\begin{aligned} \max_{\mathrm{x}} \quad & \mathrm{x}^T \; Q \; \mathrm{x} \\ \text{s.t.} \quad & \underline{\mathrm{x}} \leq \mathrm{x} \leq \overline{\mathrm{x}}. \end{aligned} \tag{2}$$

The quadratic form $\mathrm{x}^T \; Q \; \mathrm{x}$ is convex, thus all its local maxima are located at the corners of [x]. This form makes it easier to see that the maximisation problem (2) is equivalent to an integer programming problem in the space $\{0,1\}^n$. The global maximum can be found examining all 2^n corners of the search space. **Proposition**. The maximum of (2) is attained at one (or possibly more) of the 2^n corners, combination of endpoints, of the search box $[x_1] \times \cdots \times [x_n]$. The proof is given in [2], §Proof of Theorem 3.3, page 20.

3 Intervals as imprecise probabilistic objects

3.1 Variance of a dispersive mixture distribution

Although an interval is often considered to be a set of real numbers, from the perspective of imprecise probability, an interval can be seen as a credal set, i.e., a set containing all probability distributions with supports strictly inside the interval. For a single interval $[\underline{x}, \overline{x}]$, the best possible estimate of the mean is just $[\underline{x}, \overline{x}]$, while its variance is $[0, (\overline{x} - \underline{x})^2/4]$. In contrast, the variance of a point-valued interval is simply zero. In an imprecise setting, an interval is the set of all possible self bounded probability distributions. The collection of intervals [x] can be seen as the credal set of all mixtures of bounded probability distributions. The variance of an equal-weight mixture of n distributions with means μ_i and variances σ_i^2 is

$$\begin{aligned} V_{\text{mixture}} &= \frac{1}{n} \sum \left(\sigma_i^2 + \mu_i^2 \right) - \left(\frac{1}{n} \sum \mu_i \right)^2 \\ &= \frac{1}{n} \sum \sigma_i^2 + \frac{1}{n} \sum \mu_i^2 - \left(\frac{1}{n} \sum \mu_i \right)^2 . \end{aligned} \tag{3}$$

The second line in (3) makes it immediate to see that for point-valued intervals, i.e. when there is certainty about the value within the interval, $\sigma_i = 0$, $\mu_i = x_i$, the expression (3) becomes equivalent to (1). Therefore, a global maximum for the mixture is always an upper bound on the point-valued variance. This can be shown by noticing that the first term of the second line of (3) is always positive, so the variance for the mixture is always larger than the corresponding

point-valued variance. Given the integer-programming problem that follows from §2.2 proposition, and as far as the maximum of (3) is concerned, the search is restricted to the space of discrete probability distributions whose mass is placed on the *endpoints* of the mixture.

This makes intuitive sense because the best-possible upper bound for the variance of an interval $(\overline{x} - \underline{x})^2/4$ is given by a 2-discrete distribution with equal masses at the endpoints. These 2-discrete distributions with endpoint mass make up a so called *dispersive mixture* distribution. The mean and variance of each distribution in the dispersive mixture are

$$\begin{aligned} \mu_i &= p_i \ \underline{x}_i + (1 - p_i) \ \overline{x}_i, \\ \sigma_i^2 &= p_i \ (\underline{x}_i - \mu_i)^2 + (1 - p_i) \ (\overline{x}_i - \mu_i)^2, \end{aligned} \tag{4}$$

where $i = 1, ..., n$, $p_i \in [0, 1]$ is the probability mass placed on the left endpoint, and $1 - p_i$ the probability mass on the right endpoint. Combining and rearranging the equations in (4) we have

$$\sigma_i^2 + \mu_i^2 \equiv p_i \ \underline{x}_i^2 + (1 - p_i) \ \overline{x}_i^2, \tag{5}$$

substituting (5) in (3), the variance of the dispersive mixture is

$$V_{\text{dispersive}} = \frac{1}{n} \ \sum \left(p_i \ \underline{x}_i^2 + (1 - p_i) \ \overline{x}_i^2 \right) - \left(\frac{1}{n} \sum p_i \ \underline{x}_i + (1 - p_i) \ \overline{x}_i \right)^2. \tag{6}$$

3.2 Maximum of the dispersive mixture variance

Let $v : [0, 1]^n \to \mathbb{R}$ be the function that maps a vector of masses $\mathrm{p} = (p_1, ..., p_n)$ to the variance (6) of the dispersive mixture $V_{\text{dispersive}} = v(\mathrm{p})$. The maximum of such a function follows from solving the optimisation problem

$$\begin{aligned} \max_{\mathrm{p}} \quad & v(\mathrm{p}) \\ \text{s.t.} \quad & \mathrm{p} \in [0, 1]^n. \end{aligned} \tag{7}$$

This optimisation is different from (2) not just because the domain is different, but also because $v(\mathrm{p})$ is a different quadratic form compared to $f(\mathrm{x})$. Note that the interval extension $v([\mathrm{p}])$ where $[\mathrm{p}] = [0, 1]^n$ is also affected by the dependence problem of interval arithmetic because p is repeated in (6). Let us look at the properties of this quadratic form. The first partial derivatives are

$$\begin{aligned} \frac{\partial v(\mathrm{p})}{\partial p_k} &= \frac{\partial}{\partial p_k} \left(\frac{1}{n} \sum_i \left(p_i \ \underline{x}_i^2 + (1 - p_i) \ \overline{x}_i^2 \right) - \left(\frac{1}{n} \sum_i p_i \ \underline{x}_i + (1 - p_i) \ \overline{x}_i \right)^2 \right) \\ &= \frac{2}{n} \left(\underline{x}_k - \overline{x}_k \right) \left(\frac{\underline{x}_k + \overline{x}_k}{2} + \frac{1}{n} \sum_i p_i \left(\overline{x}_i - \underline{x}_i \right) - \frac{1}{n} \sum_i \overline{x}_i \right). \end{aligned} \tag{8}$$

The interval width and midpoint for the k-th component are $w_k = \overline{x}_k - \underline{x}_k$ and $m_k = \left(\underline{x}_k + \overline{x}_k\right)/2$, while the mean upper bound is $\overline{\mu} = \frac{1}{n}\sum \overline{x}_i$. Substituting these quantities in (8), the k-th component of the gradient is

$$\frac{\partial v(\mathrm{p})}{\partial p_k} = -\frac{2}{n} w_k \left(m_k + \frac{1}{n} \sum_i p_i w_i - \overline{\mu} \right). \tag{9}$$

The sign of the first derivative is positive (negative) if the sign of the parenthesis is negative (positive), since $w_k > 0$ for all intervals of the real line

$$\begin{aligned} &\frac{\partial v(\mathrm{p})}{\partial p_k} > 0, \text{ iff} \\ &\quad m_k < \overline{\mu} - \frac{1}{n} \sum p_i w_i. \end{aligned}$$

The second derivative is

$$\begin{aligned} \frac{\partial^2 v(\mathrm{p})}{\partial p_j p_k} &= \frac{\partial}{\partial p_j} \left(-\frac{2}{n} w_k \left(m_k + \frac{1}{n} \sum_i p_i w_i - \overline{\mu} \right) \right) \\ &= -\frac{2}{n} w_j \left(\frac{1}{n} w_k \right) = -\frac{2\ w_j w_k}{n^2}. \end{aligned} \tag{10}$$

Since the second derivative (10) is negative for all values of $\mathrm{p} \in [0,1]^n$, we conclude that $v(\mathrm{p})$ is a concave quadratic form with respect to the mass vector p, thus it admits only a single global maximum in [p]. The concavity and positiveness of the quadratic form ensures that the global maximum is unique.

4 Algorithm

In this section we sketch the algorithm that solves the optimisation problem (7) in polynomial time. Two consecutive tasks are undertaken: 1) a *threshold task*, and 2) a *gradient task*, which is only invoked for a particular outcome of 1).

4.1 Threshold task

This task labels each problem either to a low or a high complexity class. If the problem has low complexity the algorithm need not make any additional steps and can terminate returning the global maximum. For low complexity problems, not only the algorithm is fastest, i.e. as fast as computing a dot product of size n, but also returns an exact upper bound on the point values variance. For high complexity problems the gradient task must be invoked. The expression of the first derivative (9) suggests the existence of a "centre of rotation" around which the masses must be spread out for (6) to be maximised. Intuitively, given the discriminating role of the midpoints m_i and the linear dependence on the widths w_i, a strategic centre is the mean midpoint weighted by the widths

$$\tilde{m} = \frac{\sum m_i w_i}{\sum w_i}. \tag{11}$$

The threshold $t(\mathrm{p}) = \overline{\mu} - \frac{1}{n}\sum p_i w_i$ is introduced for the sign of the first derivative

$$\operatorname{sgn}\frac{\partial v(\mathrm{p})}{\partial p_i} = \begin{cases} +1 & m_i < t(\mathrm{p}) \\ & \\ -1 & t(\mathrm{p}) < m_i \end{cases}. \tag{12}$$

An initial vector of masses $\mathrm{p}^{(0)}$ can be derived simply by looking at which midpoints lie before or after the centre $\tilde{m}$

$$p_i^{(0)} = \begin{cases} 1 & m_i < \tilde{m} \\ & \\ 0 & \tilde{m} < m_i \end{cases}. \tag{13}$$

The algorithm for this threshold task is articulated in the following steps

1. Evaluate $t(\mathrm{p}^{(0)}) = \frac{1}{n}\sum p_i^{(0)} w_i$,
2. Compare $t(\mathrm{p}^{(0)})$ against $\tilde{m}$ and terminate if any of the following is true
   - $t(\mathrm{p}^{(0)}) = \tilde{m}$, or
   - $t(\mathrm{p}^{(0)}) < \tilde{m}$ and $m_i \notin [t(\mathrm{p}^{(0)}),\ \tilde{m}],\ i = \{1, ..., n\}$, or
   - $t(\mathrm{p}^{(0)}) > \tilde{m}$ and $m_i \notin [\tilde{m},\ t(\mathrm{p}^{(0)})],\ i = \{1, ..., n\}$.
3. Otherwise gradient iterations are needed.

If step 2. is successful, $\mathrm{p}^{(0)}$ is the maximal dispersive distribution and $v(\mathrm{p})$ is an exact upper bound for the point values variance. The complexity of this task is dominated by the dot product $t(\mathrm{p})$, which can optimistically be deemed $O(n)$.

4.2 Gradient task

This task is only invoked if $\mathcal{H} \neq \emptyset$, where $\mathcal{H} = \{h : m_h \in [\underline{d}, \overline{d}]\ \wedge\ m_h = t(p_h)\}$,

$$[\underline{d}, \overline{d}] = \begin{cases} [t(\mathrm{p}^{(0)}),\ \tilde{m}] & t(\mathrm{p}^{(0)}) < \tilde{m} \\ & \\ [\tilde{m},\ t(\mathrm{p}^{(0)})] & \tilde{m} < t(\mathrm{p}^{(0)}) \end{cases}. \tag{14}$$

Let $\mathrm{p}^* = (p_1^*, ..., p_n^*)$ be the maximal mass vector, solution of the optimisation (7). Then the following statements are true

- The masses $\mathrm{p}^*_{\mathcal{H}}$ are interior solutions, i.e. $p_h^* \in (0, 1),\ \forall h \in \mathcal{H}$.
- The midpoints $\mathrm{m}_{\mathcal{H}}$ must be all equal.

Collections of symmetric intervals whose midpoints are all equal have the largest cardinality $\#\mathcal{H} = n$ and its solution is $p_h^* = 0.5$ for all $h \in \mathcal{H}$. Collections of intervals whose midpoints are all mutually different have the smallest cardinality $\#\mathcal{H} = 1$. Symmetric or partially symmetric intervals are not uncommon in the applications. The gradient task must only be invoked for $h \in \mathcal{H}$, because only the masses $p_h \in (0, 1)$ need updating. The algorithm articulates as follows

1. Assign $p_i = p_i^{(0)}$.
2. Find the subset of indices $\mathcal{H} \subseteq \{1, ..., n\}$.
3. $\forall i \notin \mathcal{H}$, set $p_i = 1$, for $m_i < m_h$ and $p_i = 0$, for $m_h < m_i$.
4. Seek an interior solution for p_h using gradient ascent.

The algorithm terminates when $|t(\mathrm{p}) - m_h| \leq \varepsilon$ for any one $h \in \mathcal{H}$ or for $|v(p^*) - v(p^{(K)})| < \epsilon$, where ε and ϵ are small positive real numbers, and K is the last iteration. Step 3. implements the gradient step

$$\mathrm{p}^{(k+1)} = \mathrm{p}^{(k)} + \gamma \ \nabla v(\mathrm{p}^{(k)}). \tag{15}$$

The learning rate γ is designed to never make $\mathrm{p}^{(k+1)}$ overshoot outside $[0, 1]$. This is done by normalising the gradient, and considering only the remaining portion of the space, $1 - p_h$ if $\nabla v(p_h) > 0$ or p_h if $\nabla v(p_h) < 0$. This intuition can be implemented in a single line using the indicator function

$$I\mathrm{p} = \begin{cases} 1 & \nabla v(\mathrm{p}) \geq 0 \\ & \\ 0 & \nabla v(\mathrm{p}) < 0 \end{cases}. \tag{16}$$

With this engineered learning rate, that is no longer a parameter and need not be an input of the algorithm, the gradient step becomes

$$\mathrm{p}^{(k+1)} = \mathrm{p}^{(k)} + \frac{\nabla v(\mathrm{p}^{(k)})}{||\nabla v(\mathrm{p}^{(k)})||_2} \left((1 - \mathrm{p}^{(k)}) \ I\mathrm{p}^{(k)} \ + \ \mathrm{p}^{(k)} \ (1 - I\mathrm{p}^{(k)}) \right). \tag{17}$$

These iterations are guaranteed to converge because of the concavity of the objective function and the engineered learning rate.

Complexity. Typically the gradient descent for convex functions converges after $O(1/\epsilon)$ iterations in the worst case, where $\epsilon = |v(\mathrm{p}^*) - v(\mathrm{p}^{(K)})|$ Each iteration must only evaluate the gradient $\nabla v(\mathrm{p})$ a dot product. So, a sufficiently small $\epsilon = 10^{-5}$ leads to 10^5 evaluation of dot product. Under stronger regularity conditions, e.g. Lipschitz gradient, strong convexity, convergence can be even faster $O(\log(1/\epsilon))$. Overall the complexity of the algorithm is $O(mn)$, where $m = 1/\epsilon$ or $m = \log(1/\epsilon)$ depends on the quality of the gradient ascent approximation ϵ.

4.3 Examples

For the sake of demonstrating the efficiency of the algorithm, we present three examples: D1) a collection of $n = 9$ intervals with some nesting and overlaps, D2) a collection of $n = 9$ fully nested intervals, D3) a collection of $n = 20$ fully nested intervals with some coinciding midpoints. Table 1 shows that the algorithm's upper bound is very close to the exact one. Table 2 shows the number of iterations as the size of the data increases, which confirms that the algorithm is $O(mn)$. The slight increase in K with n is due to the fact that more gradient iterations are needed to find $\mathcal{H}$. The example data sets are:
D1=[[3.5,6.4],[6.9,8.8],[6.1,8.4],[2.8,6.7],[3.5,9.7],[6.5,9.9],[0.15,3.8], [4.5,4.9],[7.1,7.9]].
D2=[[1.0,9.0],[1.125,8.25],[1.25,7.5],[1.375,6.75],[1.5,6.0],[0.,10.],[1.625,5.25],[1.75,4.5],[1.875,3.75],[2.0,3.0]].
D3=[[-.75,10.75],[-.5,10.5],[-.25,10.25],[-.05,10.05],[.05,9.95],[.25,9.75],[.5,9.5],[.75,9.25],[1.,9.],
[1.125,8.25],[1.25,7.5],[1.375,6.75],[1.5,6],[1.625,5.25],[1.75,4.5],[1.875,3.75],[2,3],[2.25,2.75],[2.495,2.505]].

Table 1. Interval arithmetic (IA), exact, and proposed sharp upper bound. K is the number of iterations needed by the gradient task to converge to $\epsilon = \varepsilon = 10^{-9}$, and $\#\mathcal{H}$ is the cardinality of the interior solution set.

	IA $O(n)$	Exact $O(2^n)$	Algorithm, $O(mn)$, $\varepsilon = \epsilon = 10^{-9}$			
	$\bar{f}([x])$	$f(x^*)$	$v(p^{(0)})$	$v(p^*)$	K	$\#\mathcal{H}$
D_1, $n = 9$	21.744	10.9744	10.9744	10.9744	0	0
D_2, $n = 9$	28.427	9.7257	9.7257	9.7609	0	1
D_3, $n = 20$	57.997	17.5888	16.9293	17.5890	21	10

Table 2. Number of iterations K with increasing n for $\epsilon = \varepsilon = 10^{-9}$.

	$n = 9$	$n = 63$	$n = 207$	$n = 369$	$n = 711$	$n = 1053$
D_1	0	0	0	0	0	0
D_2	0	467	854	1144	1591	1938
	$n = 20$	$n = 140$	$n = 460$	$n = 850$	$n = 1580$	$n = 2340$
D_3	21	77	149	202	285	349

5 Conclusions

This short paper has introduced a new idea to unlock very efficient interval computation for the variance. The algorithm presented in this work can compute in polynomial time a sharp approximation of the upper bound on the variance of interval-valued data especially for cases previously only deemed NP-hard. This work concludes an important chapter on computing with interval statistics that has been open now for about two decades, and it creates exciting computational prospects for imprecise statistics and its applications. May this idea ignite future imprecise probability research and many more of these algorithms appear in the literature to address other challenging descriptive statistics.

Acknowledgements. Thanks to Scott Ferson and the anonymous reviewer.

References

1. Ferson, S., Ginzburg, L., Kreinovich, V., Longpré, L., Aviles, M.: Computing variance for interval data is NP-hard. In: ACM SIGACT News, pp. 108–118. (2002).
2. Ferson, S., Ginzburg, L., Kreinovich, V., Longpré, L., Aviles, M.: Exact Bounds on Sample Variance of Interval Data. In: Extended Abstracts of 2002 SIAM Workshop on Validated Computing, Toronto, Canada, May 23-25, pp. 67-69. (2002).
3. Ferson, S., Kreinovich, V., Hajagos, J., Oberkampf, W., Ginzburg, L.: Experimental uncertainty estimation and statistics for data having interval uncertainty. In: SAND2007–0939, pp. 910198. (2007).
4. Moore, R.: Methods and applications of interval analysis. SIAM. (1979).