Outlier Detection Under Interval Uncertainty: Algorithmic Solvability and Computational Complexity

Vladik Kreinovich, Luc Longpré,

and Praveen Patangay

Computer Science Department

University of Texas at El Paso

El Paso, TX 79968, USA

{vladik, longpre, praveen}@cs.utep.edu

Scott Ferson and Lev Ginzburg

Applied Biomathematics

100 North Country Road

Setauket, NY 11733, USA

{scott, lev}@ramas.com

Detecting Outliers Is Important

- In many application areas, it is important to detect *outliers*, i.e., unusual, abnormal values.
- In *medicine*, unusual values may indicate disease.
- In *geophysics*, abnormal values may indicate a mineral deposit (or an erroneous measurement result).
- In *structural integrity* testing, abnormal values may indicate faults in a structure.

Traditional Engineering Approach to Outlier Detection

- First, we collect measurement results $x_1, \ldots, x_n$ corresponding to normal situations.
- Then, we compute the sample average

$$E \stackrel{\text{def}}{=} \frac{x_1 + \ldots + x_n}{n}$$

and the (sample) standard deviation $\sigma = \sqrt{V}$, where

$$V \stackrel{\text{def}}{=} \frac{(x_1 - E)^2 + \ldots + (x_n - E)^2}{n};$$

- A new measurement result x is classified as an outlier if $x \notin [L, U]$, where

$$L \stackrel{\text{def}}{=} E - k_0 \cdot \sigma, \quad U \stackrel{\text{def}}{=} E + k_0 \cdot \sigma,$$

and $k_0 > 1$ is pre-selected.

- *Comment:* most frequently, $k_0 = 2$, 3, or 6.

Outlier Detection Under Interval Uncertainty: A Problem

- In some practical situations, we only have intervals $\mathbf{x}_i = [\underline{x}_i, \overline{x}_i]$.
- *Example:* value $\widetilde{x}_i$ measured by an instrument with measurement error $\leq \Delta_i$; then $x_i \in [\widetilde{x}_i - \Delta_i, \widetilde{x}_i + \Delta_i]$.
- For different values $x_i \in \mathbf{x}_i$, we get different k_0-sigma intervals $[L, U]$.
- A *possible* outlier is a value outside *some* k_0-sigma interval.
- *Example:* structural integrity – not to miss a fault.
- A *guaranteed* outlier is a value outside *all* k_0-sigma intervals.
- *Example:* before a surgery, we want to make sure that there is a micro-calcification.

Outlier Detection Reformulated in Terms of Ranges

- Let $[\underline{L}, \overline{L}]$ and $[\underline{U}, \overline{U}]$ be ranges of L and U.
- A value x is *not* a possible outlier if $x \in \cap[L, U]$, i.e., if $x \in [\overline{L}, \underline{U}]$.
- Thus, a value x *is* a possible outlier if $x \notin [\overline{L}, \underline{U}]$.
- A value x is *not* a guaranteed outlier if $x \in \cup[L, U]$, i.e., if $x \in [\underline{L}, \overline{U}]$.
- Thus, a value x *is* a guaranteed outlier if $x \notin [\underline{L}, \overline{U}]$.
- In real life, we often have an interval $\mathbf{x}$ for x. Then:
  - x is a possible outlier if $\mathbf{x} \not\subseteq [\overline{L}, \underline{U}]$;
  - x is a guaranteed outlier if $\mathbf{x} \cap [\underline{L}, \overline{U}] = \emptyset$.
- *Conclusion:* to detect outliers, we must know the ranges of L and U.

What Was Known Before and Why It Is Not Enough

- *We need:* to detect outliers, we must compute the ranges of $L = E - k_0 \cdot \sigma$ and $U = E + k_0 \cdot \sigma$.
- *We know:* previously, we have shown how to compute the ranges $\mathbf{E}$ and $[\underline{\sigma}, \overline{\sigma}]$ for E and σ.
- *Possibility:* use interval computations to conclude that $L \in \mathbf{E} - k_0 \cdot [\underline{\sigma}, \overline{\sigma}]$ and $L \in \mathbf{E} + k_0 \cdot [\underline{\sigma}, \overline{\sigma}]$.
- *Problem:* the resulting intervals for L and U are *wider* than the actual ranges.
- *Reason:* E and σ use the same inputs $x_1, \ldots, x_n$ and are hence not independent from each other.
- *Practical consequence:* we miss some outliers.
- *Desirable:* compute *exact* ranges for L and U.
- *What we will do:* exactly this.

Detecting Possible Outliers: Idea

- To detect possible outliers, we need $\overline{L}$ and $\underline{U}$.
- The minimum $\underline{U}$ of a smooth function U on an interval $[\underline{x}_i, \overline{x}_i]$ is attained:
  - either inside, when $\dfrac{\partial U}{\partial x_i} = 0$ – i.e., when
    $$x_i = \mu \stackrel{\text{def}}{=} E - \alpha \cdot \sigma \text{ (where } \alpha \stackrel{\text{def}}{=} 1/k_0);$$
  - or at $x_i = \underline{x}_i$, when $\dfrac{\partial U}{\partial x_i} \geq 0$ – i.e., when $\mu \leq \underline{x}_i$;
  - or at $x_i = \overline{x}_i$, when $\dfrac{\partial U}{\partial x_i} \leq 0$ – i.e., when $\overline{x}_i \leq \mu$.
- Thus, once we know how μ is located w.r.t. all the intervals $\mathbf{x}_i$, we can find the optimal values of x_i.
- *Comment.* the value μ can be obtained from the condition $E - \alpha \cdot \sigma = \mu$.
- Hence, to find $\min U$, we analyze how the endpoints $\underline{x}_i$ and $\overline{x}_i$ divide the real line, consider all the resulting sub-intervals, and take the smallest U.

Computing $\underline{U}$: Algorithm

- First, sort all $2n$ values $\underline{x}_i$, $\overline{x}_i$ into a sequence $x_{(1)} \le x_{(2)} \le \ldots \le x_{(2n)}$; take $x_{(0)} \stackrel{\text{def}}{=} -\infty$, $x_{(2n+1)} \stackrel{\text{def}}{=} +\infty$.
- For each zone $[x_{(k)}, x_{(k+1)}]$, we compute the values

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

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

  and $n_k =$ the total number of such i's and j's.
- Solve equation $A - B \cdot \mu + C \cdot \mu^2 = 0$, where

$$A \stackrel{\text{def}}{=} e_k^2 \cdot (1 + \alpha^2) - \alpha^2 \cdot m_k \cdot n,$$

$$B \stackrel{\text{def}}{=} 2e_k \cdot \left((1 + \alpha^2) \cdot n_k - \alpha^2 \cdot n\right); \quad C \stackrel{\text{def}}{=} B \cdot \frac{n_k}{2e_k};$$

  select $\mu \in$ zone for which $\mu \cdot n_k \le e_k$.
- $E_k \stackrel{\text{def}}{=} \dfrac{e_k}{n} + \dfrac{n - n_k}{n} \cdot \mu$, $M_k \stackrel{\text{def}}{=} \dfrac{m_k}{n} + \dfrac{n - n_k}{n} \cdot \mu^2$, $U_k \stackrel{\text{def}}{=} E_k + k_0 \cdot \sqrt{M_k - (E_k)^2}$.
- $\underline{U}$ is the smallest of these values U_k.

Computing $\overline{L}$: Algorithm

- First, sort all $2n$ values $\underline{x}_i$, $\overline{x}_i$ into a sequence $x_{(1)} \le x_{(2)} \le \ldots \le x_{(2n)}$; take $x_{(0)} \stackrel{\text{def}}{=} -\infty$, $x_{(2n+1)} \stackrel{\text{def}}{=} +\infty$.
- For each zone $[x_{(k)}, x_{(k+1)}]$, we compute the values

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

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

and $n_k =$ the total number of such i's and j's.

- Solve equation $A - B \cdot \mu + C \cdot \mu^2 = 0$, where

$$A \stackrel{\text{def}}{=} e_k^2 \cdot (1 + \alpha^2) - \alpha^2 \cdot m_k \cdot n,$$

$$B \stackrel{\text{def}}{=} 2e_k \cdot \left((1 + \alpha^2) \cdot n_k - \alpha^2 \cdot n\right); \quad C \stackrel{\text{def}}{=} B \cdot \frac{n_k}{2e_k};$$

select $\mu \in$ zone for which $\mu \cdot n_k \ge e_k$.

- $E_k \stackrel{\text{def}}{=} \dfrac{e_k}{n} + \dfrac{n - n_k}{n} \cdot \mu$, $M_k \stackrel{\text{def}}{=} \dfrac{m_k}{n} + \dfrac{n - n_k}{n} \cdot \mu^2$, $L_k \stackrel{\text{def}}{=} E_k - k_0 \cdot \sqrt{M_k - (E_k)^2}$.
- $\overline{L}$ is the largest of these values L_k.

Computational Complexity of Outlier Detection

- *Detecting possible outliers:* The above algorithm $\underline{\mathcal{A}}_U$ always computes $\underline{U}$ in quadratic time.
- *Detecting possible outliers:* The above algorithm $\overline{\mathcal{A}}_L$ always computes $\overline{L}$ in quadratic time.
- *Detecting guaranteed outliers:* For every $k_0 > 1$, computing the upper endpoint $\overline{U}$ of the interval $[\underline{U}, \overline{U}]$ of possible values of $U = E + k_0 \cdot \sigma$ is NP-hard.
- *Detecting guaranteed outliers:* For every $k_0 > 1$, computing the lower endpoint $\underline{L}$ of the interval $[\underline{L}, \overline{L}]$ of possible values of $L = E - k_0 \cdot \sigma$ is NP-hard.
- *Comment.* For interval data, the NP-hardness of computing the upper bound for σ was known before.

How Can We Actually Detect Guaranteed Outliers?

- *1st result:* if $1 + (1/k_0)^2 < n$, then $\max U$ and $\min L$ are attained at endpoints of $\mathbf{x}_i$.
- *Example:* $k_0 > 1$ and $n \geq 2$.
- *Resulting algorithm:* test all 2^n combinations of values $\underline{x}_i$ and $\overline{x}_i$.
- *Important case:* often, measured values $\widetilde{x}_i$ are definitely different from each other, in the sense that the "narrowed" intervals

$$\left[\widetilde{x}_i - \frac{1+\alpha^2}{n} \cdot \Delta_i, \widetilde{x}_i + \frac{1+\alpha^2}{n} \cdot \Delta_i\right]$$

do not intersect with each other.

- *Slightly more general case:* for some C, no more than C "narrowed" intervals can have a common point.

Computing $\overline{U}$

- Sort all endpoints of the narrowed intervals into a sequence $x_{(1)} \le x_{(2)} \le \ldots \le x_{(2n)}$, with $x_{(0)} \stackrel{\text{def}}{=} -\infty$, $x_{(2n+1)} \stackrel{\text{def}}{=} +\infty$.
- For each zone $[x_{(i)}, x_{(i+1)}]$, for each j, pick x_j:
  - if $x_{(i+1)} < \widetilde{x}_j - \dfrac{1+\alpha^2}{n} \cdot \Delta_j$, pick $x_j = \overline{x}_j$;
  - if $x_{(i+1)} > \widetilde{x}_j + \dfrac{1+\alpha^2}{n} \cdot \Delta_j$, pick $x_j = \underline{x}_j$;
  - for all other j, consider both $x_j = \overline{x}_j$ and $x_j = \underline{x}_j$.
- We get $\le 2^C$ sequences of x_j for each zone.
- For each sequence x_j, check whether $E - \alpha \cdot \sigma$ is within the zone.
- If $E - \alpha \cdot \sigma \in$ zone, compute $U \stackrel{\text{def}}{=} E + k_0 \cdot \sigma$.
- Finally, we return the largest of the computed values U as $\overline{U}$.

Computing $\underline{L}$

- Sort all endpoints of the narrowed intervals into a sequence $x_{(1)} \le x_{(2)} \le \ldots \le x_{(2n)}$, with $x_{(0)} \stackrel{\text{def}}{=} -\infty$, $x_{(2n+1)} \stackrel{\text{def}}{=} +\infty$.
- For each zone $[x_{(i)}, x_{(i+1)}]$, for each j, pick x_j:
  - if $x_{(i+1)} < \widetilde{x}_j - \dfrac{1+\alpha^2}{n} \cdot \Delta_j$, pick $x_j = \overline{x}_j$;
  - if $x_{(i+1)} > \widetilde{x}_j + \dfrac{1+\alpha^2}{n} \cdot \Delta_j$, pick $x_j = \underline{x}_j$;
  - for all other j, consider both $x_j = \overline{x}_j$ and $x_j = \underline{x}_j$.
- We get $\le 2^C$ sequences of x_j for each zone.
- For each sequence x_j, check whether $E + \alpha \cdot \sigma$ is within the zone.
- If $E + \alpha \cdot \sigma \in$ zone, compute $L \stackrel{\text{def}}{=} E - k_0 \cdot \sigma$.
- Finally, we return the smallest of the computed values L as $\underline{L}$.

Computational Complexity

- *1st result:* for the case when $\leq C$ narrowed intervals can have a common point, the above algorithm $\overline{\mathcal{A}}_U$ always computes $\overline{U}$ in quadratic time.
- *2nd result:* for the case when $\leq C$ narrowed intervals can have a common point, the above algorithm $\underline{\mathcal{A}}_L$ always computes $\underline{L}$ in quadratic time.
- *Comment:* the corresponding computation times are quadratic in n but grow exponentially with C.
- *Corollary:* when C grows, this algorithm requires more and more computation time.
- *Comment:* in the examples on which we prove NP-hardness, $n/2$ out of n narrowed intervals have a common point.

Conclusions

- In many applications, it's important to detect outliers.
- Traditional idea: $x \notin [E - k_0 \cdot \sigma, E + k_0 \cdot \sigma]$.
- We often have only interval ranges $\mathbf{x}_i = [\underline{x}_i, \overline{x}_i]$.
- For different values $x_i \in \mathbf{x}_i$, we get different k_0-sigma intervals $[L, U]$.
- x a *guaranteed* outlier if outside *all* k_0-sigma intervals.
- x a *possible* outlier if outside *some* k_0-sigma interval.
- To detect guaranteed and possible outliers, we must thus be able to compute the *ranges* $\mathbf{L} = [\underline{L}, \overline{L}]$ and $\mathbf{U} = [\underline{U}, \overline{U}]$.
- We show that computing these ranges is, in general, NP-hard.
- We also provide efficient algorithms that compute these ranges under reasonable conditions.

Acknowledgments

This work was supported in part:

- by NASA grant NCC5-209 and NCC2-1232,
- by Air Force Office of Scientific Research grant F49620-00-1-0365,
- by NSF grants EAR-0112968 and EAR-0225670,
- by a grant from the Army Research Lab,
- by IEEE/ACM SC2001 and SC2002 Minority Serving Institutions Participation Grants,
- by grant 9R44CA81741 to Applied Biomathematics from the National Cancer Institute (NCI), a component of the National Institutes of Health (NIH), and
- by a research grant from Sandia National Laboratories as part of the Department of Energy Accelerated Strategic Computing Initiative (ASCI).