7-2021

"Negative" Results -- When the Measured Quantity Is Outside the Sensor's Range -- Can Help Data Processing

Jonatan Contreras
The University of Texas at El Paso, jmcontretas2@utep.edu

Francisco Zapata
The University of Texas at El Paso, fcozpt@outlook.com

Olga Kosheleva
The University of Texas at El Paso, olgak@utep.edu

Vladik Kreinovich
The University of Texas at El Paso, vladik@utep.edu

Martine Ceberio
The University of Texas at El Paso, mceberio@utep.edu

Comments:
Technical Report: UTEP-CS-20-76d

"Negative" Results – When the Measured Quantity Is Outside the Sensor's Range – Can Help Data Processing

Jonatan Contreras, Francisco Zapata, Olga Kosheleva, Vladik Kreinovich, and Martine Ceberio

Abstract In many real-life situations, we know the general form of the dependence $y = f(x, c_1, \ldots, c_m)$ between physical quantities, but the values c_i need to be determined experimentally, based on the results of measuring x and y. In some cases, we do not get any result of measuring y since the actual value is outside the range of the measuring instrument. Usually, such cases are ignored. In this paper, we show that taking these cases into account can help data processing – by improving the accuracy of our estimates of c_i and thus, by improving the accuracy of the resulting predictions of y.

1 A Brief Introduction

In many real-life situations, we know, from experience, that the value of a quantity y is determined by the values of related quantities $x = (x_1, \ldots, x_v)$. For example, we often know that the future value y of a physical quantity – e.g., of the temperature at a given location – is determined by the current values of this and related quantities: temperature, wind speed and direction, humidity, etc. – at this and nearby locations. In many such situations, we know the general type of this dependence, i.e., we know that this dependence takes the form $y = f(x, c_1, \ldots, c_n)$, where the parameters c_i need to be determined from experiments:

- we measure the values of x and y in different situations, and
- we find the value of the parameters c_i which are consistent with the results of these measurements.

Jonatan Contreras, Francisco Zapata, Olga Kosheleva, Vladik Kreinovich, and Martine Ceberio
University of Texas at El Paso, 500 W. University, El Paso, TX 79968, USA
e-mail: jmcontretas2@utep.edu, fcozpt@outlook.com, olgak@utep.edu, vladik@utep.edu, mceberio@utep.edu

Measurements are never 100% accurate, so we can only find the values c_i with some accuracy – and this accuracy also needs to be determined based on the same measurement results. The more measurements we process, the more accurate the results. However, as we process a large number of measurements, we will eventually encounter situations when we could not measure y – because the corresponding value was outside the sensor's range. Usually, in data processing, such situations are simply dismissed – or, at best, used as an additional check of the resulting model. In this paper, we show that such "negative" data can help in all aspects of data processing – not only to select an appropriate model, but also to estimate the model's accuracy.

Preliminary results of this research first appeared in our conference paper [2]. In the current paper, we extend the 1-D case analyzed in [2] into a general multi-D case. We have also added an explicit explanation of where such "negative" data points come from.

2 Formulation of the Problem: A More Detailed Description

What are the main objectives of science and engineering. Crudely speaking, the main objective of science is to *predict* future events. This is the ultimate goal by which we gauge the quality of each new scientific theory: if this theory can make a new prediction, and this prediction is experimentally confirmed, this confirms the theory. This is how General Relativity was confirmed – by measuring the gravitation-related deviation of the light from the straight line. This is how all theories are checked and confirmed.

The main objective of engineering is to come up with designs and/or control strategies that will *improve* our future. To understand which gadgets, which strategies will lead to the desired improvement, we need to predict how exactly the state of the world will change under each such strategy.

For both objectives, we must be able:

- *given* the initial conditions x – the initial state of the world plus, in the engineering case, the description of the changes that we plan to make,
- to *predict* the value of each quantity y that characterizes the future state of the world (or at least the future state of the system in which we are interested).

Comment. In general, the values of the quantity y can be all possible real values – or all the values from some finite (or semi-infinite) interval. Such quantities are known as *continuous.*

In some case, possible values of y are limited to a discrete set. For example, electric charge can only take values which are proportional to the elementary charge. Such quantities are known as *discrete*.

In this paper, we concentrate on continuous quantities. However, our ideas and formulas can be easily extended to the discrete cases as well. For example, for discrete quantities, in the case of probabilistic uncertainty:

- instead of the probability density function (pdf), we can use its discrete analogue – probabilities of different values, and
- instead of the requirement that the integral of the pdf is equal to 1, we have the requirement that the sum of the probabilities is equal to 1.

In most practical situations, we do not know the dependence of y on x. In some cases we know the equations (or even explicit formulas) that relate the available information x and the desired quantity y. In such cases, in principle, we have an algorithm for predicting y. A typical situation of this type is celestial mechanics, where we can predict, e.g., solar eclipses and re-appearance of comets hundreds of years ahead.

In some cases, we may know the dependence, but there is no feasible algorithm for making the corresponding prediction. For example, it is possible to predict – reasonably reliably – in what direction a tornado will turn in the next 15 minutes. However, such computations require several hours of computations on a high-performance computer. This is a good confirmation of the current tornado models, but from the practical viewpoint, this result is so far useless: what is the purpose of these hour-long computations if in 15 minutes we will already know where the tornado moved. The hope is that as computers get faster and faster, we will eventually be able to make the corresponding computations practical.

In many other cases, we do not know how exactly y depends on x. In such cases, we need to determine this dependence based on the known observations $\left(x^{(k)}, y^{(k)}\right)$, in which we measured (or otherwise estimated) both x and y. In many such situations, we know that the dependence of y on x belongs to a known family of functions, i.e., that $y = f(x, c_1, \ldots, c_n)$ for some values of the parameters c_i, but these values need to be determined from experiment.

In other situations, we do not know the family, so we use appropriate machine learning tools to come up with the desired dependence $y = f(x)$. In mathematical terms, the results of machine learning – e.g., the results of deep learning – can also be described as $y = f(c_1, \ldots, c_n)$, where c_i are the parameters that change during training, and the expression $f(x, c_1, \ldots, c_n)$ describes the result of applying, to the input x, the machine learning algorithm with parameters c_i.

Need to take uncertainty into account. Our knowledge of x and y usually comes from measurements, and measurements are never absolutely accurate: the measurement result $\widetilde{y}$ is, in general, different from the actual value y. In some cases, we know the probabilities of measurement errors $\Delta y \stackrel{\text{def}}{=} \widetilde{y} - y$. In such cases, we have a probabilistic information about y; see, e.g., [16].

In other cases, we do not know these probabilities, all we know is the upper bound Δ on the absolute value of the measurement error $|\Delta y| \le \Delta$. In such cases, once we know the measurement result $\widetilde{y}$, the only thing that we can conclude about the actual value y is that this value is located in the interval $[\widetilde{y} - \Delta, \widetilde{y} + \Delta]$.

In this paper, we will consider both cases – of probabilistic and of interval uncertainty.

Important comment about the accuracies of measuring x and y. Usually:

- we set up the values x, and
- measure the corresponding values y.

The values x we set up ourselves, we know exactly the range of the corresponding values, so we can select the most appropriate measuring instrument, and get reasonably accurate values.

In contrast, we only have a crude idea about the future values y; based on this idea, we can guess the range of y, but this guess may be wrong, so the selected measuring instrument may be not the most accurate one for this range. As a result, in general, the accuracy with which we know y is much worse that the accuracy with which we know x.

So, it makes sense – at least in the first reasonable approximation – to ignore the measurement errors corresponding to measuring x – since they are much smaller than the errors of measuring y – and assume that the values x are known exactly.

Positive and negative examples: what are they and why it is important. In most practical situations, we have measurement results describing both x and y. We will call the corresponding examples $\left(x^{(k)}, y^{(j)}\right)$ *positive examples*.

However, sometimes :

- we know the values x, but
- we could not measure y.

Indeed, each measuring instrument has a range of values for which it has been designed:

- a ruler cannot directly measure lengths which are too long;
- scales cannot measure weights which are too heavy – they will simply crush the scales – or too light: they will not be noticed by the scale at all.

In each experiment, we try to select a measuring instrument that will be able to measure the corresponding value – but remember that we are dealing with situations in which we do not know the actual dependence, so we cannot exactly predict the value y. Because of this, sometimes, we have wrong expectations, we select a wrong instrument. Thus, as we have mentioned earlier, the actual signal may not be in the area of the sensor's not-the-best accuracy.

Moreover, the signal can be outside the interval on which we selected and/or trained the measuring instrument: the actual value could be smaller than this interval's lower endpoint or it could be larger that the interval's upper endpoint.

In this case, for the only information about the measured quantity y is that this quantity is *not* located in the (known) interval range of the measuring instrument. We will call such situations *negative* examples.

Usually, negative examples are ignored. Usually, practitioners simply ignore such negative examples – mostly because there are no methods for using them.

At best, such examples are used as an additional conformation of the resulting values c_i: by checking that for each such example, the corresponding value $y^{(k)} = f\left(x^{(k)}, c_1, \ldots, c_n\right)$ is indeed outside the range of the corresponding measuring instrument.

What we do in this paper. Since the measurement results are approximate, the resulting estimates for the parameters c_i are also approximate. From the practical viewpoint, it is important to know this uncertainty – since it affects the accuracy of the resulting future predictions $y = f(x, c_1, \ldots, c_n)$.

In this paper, we show that negative example can help to gauge this uncertainty – and thus, help us make more accurate predictions.

3 Case of Interval Uncertainty

Data processing under interval uncertainty, case of positive examples: a brief reminder. Following our assumption about the accuracies of measuring x and y, we assume that:

- we know the exact values $x^{(k)}$, while
- the values $y^{(k)}$ are only known with interval uncertainty – i.e., for each k, we know the interval $\left[\underline{y}^{(k)}, \overline{y}^{(k)}\right]$ that contains the actual (unknown) value $y^{(k)}$.

In this case, our objective is to find the values $c = (c_1, \ldots, c_n)$ for which the following inequality is satisfied for all k:

$$\underline{y}^{(k)} \le f\left(x^{(k)}, c_1, \ldots, c_n\right) \le \overline{y}^{(k)}, 1 \le k \le K. \tag{1}$$

Data processing under interval uncertainty, case of positive examples: algorithms. For each i, we want to find the range $\left[\underline{c}_i, \overline{c}_i\right]$ of possible values of c_i. To find this range, we need to solve the following two constraint optimization problems:

- to find $\underline{c}_i$, we minimize c_i under the constraints (1); and
- to find $\overline{c}_i$, we maximize c_i under the constraints (1).

In the general non-linear case, this problem is NP-hard (even finding one single combination c that satisfies all the constraints (1) is, in general, NP-hard); see, e.g., [7]. In such cases, constraint solving algorithms (see, e.g., [4]) can lead to approximate ranges: e.g., to enclosures $\left[\underline{c}'_i, \overline{c}'_i\right] \supseteq \left[\underline{c}_i, \overline{c}_i\right]$ for the actual range.

The problem of computing the ranges $\left[\underline{c}_i, \overline{c}_i\right]$ becomes feasible if we consider families that linearly depend on the parameters c_i, i.e., families of the type

$$f(x, c_1, \ldots, c_n) = f_0(x) + c_1 \cdot f_1(x) + \ldots + c_n \cdot f_n(x). \tag{2}$$

In this case, inequalities (1) become linear inequalities in terms of the unknowns c_i:

$$\underline{y}^{(k)} \le f_0\left(x^{(k)}\right) + c_1 \cdot f_1\left(x^{(k)}\right) + \ldots + c_n \cdot f_n\left(x^{(k)}\right) \le \overline{y}^{(k)}, \ \ 1 \le k \le K \tag{3}$$

In this case, e.g., the range $\left[\underline{c}_i, \overline{c}_i\right]$ of possible values of c_i can be obtained by solving the following two *linear programming* problems – i.e., problems of optimizing a linear function under linear constraints:

- to find $\underline{c}_i$, we need to minimize c_i under the linear constraints (3); and
- to find $\overline{c}_i$, we need to maximize c_i under the linear constraints (3).

There are efficient feasible algorithms for solving linear programming problems; see, e.g., [3, 8]. So, the corresponding regression problem can indeed be efficiently solved.

How can we take "negative" intervals into account. As we have mentioned earlier:

- in addition to "positive" intervals – i.e., intervals that contain the y-values $y^{(k)}$, $k = 1, \ldots, K$,
- we can also have "negative" intervals $\left(\underline{y}^{(\ell)}, \overline{y}^{(\ell)}\right)$, $\ell = K+1, \ldots, L$ – i.e., intervals that are known *not* to contain the corresponding values $y^{(\ell)}$.

In this case, in addition to the condition (1), we also have an additional condition that must be satisfied for each ℓ from $K+1$ to L:

$$f\left(x^{(\ell)}, c_1, \ldots, c_n\right) \le \underline{y}^{(\ell)} \text{ or } \overline{y}^{(\ell)} \le f\left(x^{(\ell)}, c_1, \ldots, c_n\right). \tag{4}$$

In this case, we need to find the values $c = (c_1, \ldots, c_n)$ that satisfy *both* constraints (1) and (4).

An example showing that negative intervals can help. Let us first show that the use of negative example can indeed improve the accuracy of the resulting estimates for c_i.

Let us consider a very linear model $y = c_1 \cdot x$, and the simplest case when we have only two observations: for two values $x = -1$ and for $x = 1$, we have the same interval of possible values of y: $y \in [-1, 1]$. One can easily see that in this case, the set of possible values of c_1 is the interval $[-1, 1]$.

In particular, for $x = 2$, the only information that we can conclude based on this estimate for c_1, is that $y \in [-2, 2]$.

Suppose now that, in addition to the above two positive examples, we also have a negative example: namely, we know that for $x = 2$, the value y cannot be in the interval $(-3, 2)$ – the range of the corresponding sensor. In this case, for this x, the set of possible values of y narrows down to a single value $y = 2$. Correspondingly, the set of possible values of c_1 narrows down from the original interval $[-1, 1]$ to a single value $c_1 = 1$.

In this case, the accuracy of the resulting predictions $y = c_1 \cdot x$ drastically increases, from very inaccurate predictions $y \in [-x, x]$ to a very accurate prediction $y = x$.

Towards algorithms: the problem is not easy. Now that we know that negative intervals can drastically improve the prediction accuracy, it is desirable to look for algorithms that will take such intervals into account.

It would have been nice to have a general feasible algorithm. However, unfortunately, one can prove that if we take into account possible negative intervals, then

even for linear constraints, the problem of finding the bounds $\underline{c}_i$ and $\overline{c}_i$ becomes NP-hard.

The proof of this NP-hardness is rather straightforward. Indeed, it is known that the following problem is NP-hard (see, e.g., [7, 15]):

- *given* natural numbers $s_1, \ldots, s_n$ and s,
- *find* a subset of the values s_i that adds up to s.

In other words, we need to find the values $c_i \in \{0, 1\}$ (describing whether to take the i-th value s_i or not) for which $\sum_{i=1}^{n} c_i \cdot s_i = s$.

This problem can be easily reformulated as an interval problem with positive and negative examples. For this purpose, we take a linear model

$$y = c_1 \cdot x_1 + \ldots + c_n \cdot x_n$$

and the following examples:

- a positive example in which $x_i = s_i$ for all i and $y \in [s, s]$; consistency with this positive example means that

$$s = \sum_{i=1}^{n} c_i \cdot s_i;$$

- n additional positive examples; in the i-th example, $x_i = 1, x_j = 0$ for all $j \neq i$, and $y \in [0, 1]$; consistency with each such example means that $c_i \in [0, 1]$; and
- n negative examples; in the i-th example, $x_i = 1, x_j = 0$ for all $j \neq i$, and $y \notin (0, 1)$; consistency with each such example means that $c_i \notin (0, 1)$.

Together with the previous consistency, this means exactly that $c_i \in \{0, 1\}$.

Since we cannot compute the exact range, let us compute the enclosure for the range. NP-hard implies that, unless P = NP (which most computer scientists believe to be impossible), no feasible algorithm is possible that would always compute the exact ranges for c_i – or even check whether the data is consistent with the model.

Since we cannot compute the exact range $[\underline{c}_i, \overline{c}_i]$, a natural idea is to compute an *enclosure* $[\underline{C}_i, \overline{C}_i] \supseteq [\underline{c}_i, \overline{c}_i]$. Let us show how we can do it.

Computing enclosure: first algorithm. Each negative interval $(\underline{y}^{(\ell)}, \overline{y}^{(\ell)})$ means that the actual value of $y^{(\ell)}$ is either in the interval $(-\infty, \underline{y}^{(\ell)}]$ or in the interval $[\overline{y}^{(\ell)}, \infty)$. So, the following is a natural algorithm:

- we can add, to K positive intervals, the first of these two semi-infinite intervals, solve the corresponding linear programming problem, and get ranges $[\underline{C}_i^{(\ell),-}, \overline{C}_i^{(\ell),-}]$ for the coefficients c_i;

- we can also add, to K positive intervals, the second of these two semi-infinite intervals, solve the corresponding linear programming problem, and get ranges $\left[\underline{C}_i^{(\ell),+}, \overline{C}_i^{(\ell),+}\right]$ for the coefficients c_i.

We know that the actual value $y^{(\ell)}$ is either in the first *or* in the second of the semi-infinite intervals. Therefore, the actual range $\left[\underline{c}_i, \overline{c}_i\right]$ of possible values of each c_i is contained in the *union* of the two intervals:

$$\left[\underline{C}_i^{(\ell)}, \overline{C}_i^{(\ell)}\right] = \left[\underline{C}_i^{(\ell),-}, \overline{C}_i^{(\ell),-}\right] \bigcup \left[\underline{C}_i^{(\ell),+}, \overline{C}_i^{(\ell),+}\right]. \tag{5}$$

In order words, we get the enclosure $\left[\underline{C}_i^{(\ell)}, \overline{C}_i^{(\ell)}\right]$, where:

$$\underline{C}_i^{(\ell)} = \min\left(\underline{c}_i^{(\ell),-}, \underline{c}_i^{(\ell),+}\right) \text{ and } \overline{C}_i^{(\ell)} = \max\left(\overline{c}_i^{(\ell),-}, \overline{c}_i^{(\ell),+}\right). \tag{6}$$

The actual value c_i belongs to *all* these intervals, so we can conclude that it belongs to the intersection $\left[\underline{c}_i, \overline{c}_i\right]$ of all these intervals:

$$\left[\underline{C}_i, \overline{C}_i\right] = \bigcap_{\ell=K+1}^{L} \left[\underline{C}_i^{(\ell)}, \overline{C}_i^{(\ell)}\right]. \tag{7}$$

For this new enclosure, we have:

$$\underline{C}_i = \max_\ell \underline{C}_i^{(\ell)} \text{ and } \overline{C}_i = \min_\ell \overline{C}_i^{(\ell)}. \tag{8}$$

If this intersection is empty, this means that the model is inconsistent with observations.

Computing enclosure: second algorithm. In the above algorithm, at each step, we only take into account *one* negative example. Instead, we can take into account *two* negative examples. Then, for each pair (ℓ, ℓ') of negative examples, we have four possible cases:

- we can have the case $a = --$ when $y^\ell \in \left(-\infty, \underline{y}^{(\ell)}\right]$ and $y^{\ell'} \in \left(-\infty, \underline{y}^{(\ell')}\right]$;
- we can have the case $a = -+$ when $y^\ell \in \left(-\infty, \underline{y}^{(\ell)}\right]$ and $y^{\ell'} \in \left[\overline{y}^{(\ell')}, \infty\right)$;
- we can have the case $a = +-$ when $y^\ell \in \left[\overline{y}^{(\ell)}, \infty\right)$ and $y^{\ell'} \in \left(-\infty, \underline{y}^{(\ell')}\right]$; and
- we can have the case $a = ++$ when $y^\ell \in \left[\overline{y}^{(\ell)}, \infty\right)$ and $y^{\ell'} \in \left[\overline{y}^{(\ell')}, \infty\right)$.

For each of these four cases $a = --, -+, +-, ++$, we can add the corresponding two semi-infinite intervals to K positive intervals, and find the enclosures $\left[\underline{C}_i^{(\ell,\ell'),a}, \overline{C}_i^{(\ell,\ell'),a}\right]$ for the actual range $\left[\underline{c}_i, \overline{c}_i\right]$ of each coefficient c_i. Then, we can conclude that the actual value of c_i belongs to the union of these four intervals:

$$\left[\underline{C}_i^{(\ell,\ell')}, \overline{C}_i^{(\ell,\ell')}\right] = \bigcup_a \left[\underline{C}_i^{(\ell,\ell'),a}, \overline{C}_i^{(\ell,\ell'),a}\right], \tag{9}$$

i.e., we take

$$\underline{C}_i^{(\ell,\ell')} = \min_a \underline{C}_i^{(\ell,\ell'),a} \text{ and } \overline{C}_i^{(\ell,\ell')} = \max_a \overline{C}_i^{(\ell,\ell'),a}. \tag{10}$$

The actual value c_i belongs to *all* these enclosures. So, we can conclude that the actual value c_i belongs to the intersection $\left[\underline{C}_i, \overline{C}_i\right]$ of all these enclosures:

$$\left[\underline{C}_i, \overline{C}_i\right] = \bigcap_{K+1 \le \ell, \ell' \le L} \left[\underline{C}_i^{(\ell,\ell')}, \overline{C}_i^{(\ell,\ell')}\right], \tag{11}$$

i.e., we take

$$\underline{C}_i = \max_{\ell,\ell'} \underline{C}_i^{(\ell,\ell')} \text{ and } \overline{C}_i = \min_{\ell,\ell'} \overline{C}_i^{(\ell,\ell')}. \tag{12}$$

By applying this algorithm:

- on the one hand, we get, in general, a better range – with smaller excess width;
- however now, instead of considering $O(L-K)$ cases as in the first algorithms, we need to consider $O\left((L-K)^2\right)$ cases.

Possible other algorithms. We can get even more accurate estimates for the range if we consider all possible triples, 4-tuples, etc., of negative intervals. However, in this case, we will need to consider $O\left((L-K)^3\right)$, $O\left((L-K)^4\right)$, etc. cases – i.e., we get a much longer computation time.

4 What If We Are Interested in Several Quantities

Negative information that we analyzed so far: a reminder. In the previous text, we considered negative examples corresponding to the case when a value of a quantity y cannot be detected by a sensor tuned for values from some interval $(\underline{y}, \overline{y})$.

In this case, we can conclude that the actual value y is outside this interval.

There are other types of negative information. A similar – but somewhat more complicated – situation occurs if we train, e.g., a camera to a certain area or a microphone to a certain area of spatial directions. In such cases, we are not simply limiting the range of possible values of a single quantity y. Instead, we simultaneously limit the value of two or more quantities $y_1, \ldots, y_m$ to corresponding intervals:

$$y_1 \in \left(\underline{y}_1, \overline{y}_1\right), \ldots, y_m \in \left(\underline{y}_m, \overline{y}_m\right). \tag{13}$$

In this case, in contrast to the previously analyzed case, if we do not detect anything, we cannot conclude that, e.g., the value y_1 is necessarily not in the corresponding

interval $\left[\underline{y}_1, \overline{y}_1\right]$ – this value may well be within this interval, but one of the other quantities is outside its interval. All we know is that the tuple $y = (y_1, \ldots, y_m)$ is *not* located inside the corresponding box

$$y \notin \left(\underline{y}_1, \overline{y}_1\right) \times \ldots \times \left(\underline{y}_m, \overline{y}_m\right). \tag{14}$$

Because of this inter-relation between different variables, to deal with such situations, we can no longer concentrate on one of the quantities y_i – there are no restrictions on each value y_i per se – we need to simultaneously consider *all* related quantities $y_1, \ldots, y_m$.

Let us describe the resulting problem in precise terms: general case. We know that several quantities $y_1, \ldots, y_m$ depend on the quantities $x = (x_1, \ldots, x_v)$. We assume that this dependence is described by functions from a certain family of functions, characterized by parameters $c_1, \ldots, c_n$:

$$y_1 = f_1(x, c_1, \ldots, c_n), \ldots, y_m = f_m(x, c_1, \ldots, c_n). \tag{15}$$

We have several measurements in which the vector $y = (y_1, \ldots, y_m)$ was inside the corresponding box, i.e., when we had, for all k from 1 to K:

$$\begin{gathered} \underline{y}_1^{(k)} \le f_1\left(x^{(k)}, c_1, \ldots, c_m\right) \le \overline{y}_1^{(k)}, \\ \ldots \\ \underline{y}_m^{(k)} \le f_m\left(x^{(k)}, c_1, \ldots, c_m\right) \le \overline{y}_m^{(k)}. \end{gathered} \tag{16}$$

We also have negative examples, for which, for each ℓ from $K+1$ to L, the following condition must be satisfied:

$$\begin{gathered} f_1\left(x^{(\ell)}, c_1, \ldots, c_n\right) \le \underline{y}_1^{(\ell)}, \text{ or } \overline{y}_1^{(\ell)} \le f_1\left(x^{(\ell)}, c_1, \ldots, c_n\right), \text{ or} \\ \ldots, \text{ or} \\ f_m\left(x^{(\ell)}, c_1, \ldots, c_n\right) \le \underline{y}_m^{(\ell)}, \text{ or } \overline{y}_m^{(\ell)} \le f_m\left(x^{(\ell)}, c_1, \ldots, c_n\right). \end{gathered} \tag{17}$$

The question is to find the values $c = (c_1, \ldots, c_n)$ that satisfy all the constraints (16) and (17).

In particular, for each of the parameters c_i, we need to find the range $\left[\underline{c}_i, \overline{c}_i\right]$ of possible values of this parameter:

- Each value $\underline{c}_i$ can be obtained by minimizing c_i under the constraints (16) and (17).
- Similarly, each value $\overline{c}_i$ can be obtained by maximizing c_i under the constraints (16) and (17).

Important case when the dependence on parameters is linear. As we have mentioned, in the general case, the corresponding problems are NP-hard even when we do

not have negative examples. In this no-negative-examples case, however, the problem becomes feasible if we consider the common situations in which the dependence on the parameters c_i is linear, i.e., in which

$$\begin{aligned} y_1 &= f_{1,0}(x) + c_1 \cdot f_{1,1}(x) + \ldots + c_n \cdot f_{1,n}(x), \\ &\ldots \\ y_m &= f_{m,0}(x) + c_1 \cdot f_{m,1}(x) + \ldots + c_n \cdot f_{m,n}(x). \end{aligned} \tag{18}$$

In this case, the condition (16) corresponding to each measurement k takes the form:

$$\begin{aligned} \underline{y}_1^{(k)} &\le f_{1,0}\left(x^{(k)}\right) + c_1 \cdot f_{1,1}\left(x^{(k)}\right) + \ldots + c_n \cdot f_{1,n}\left(x^{(k)}\right) \le \overline{y}_1^{(k)}, \\ &\ldots \\ \underline{y}_m^{(k)} &\le f_{m,0}\left(x^{(k)}\right) + c_1 \cdot f_{m,1}\left(x^{(k)}\right) + \ldots + c_n \cdot f_{m,n}\left(x^{(k)}\right) \le \overline{y}_m^{(k)}. \end{aligned} \tag{19}$$

Similarly, the condition (17) corresponding to each measurement ℓ takes the form

$$\begin{aligned} &f_{1,0}\left(x^{(\ell)}\right) + c_1 \cdot f_{1,1}\left(x^{(\ell)}\right) + \ldots + c_n \cdot f_{1,n}\left(x^{(\ell)}\right) \le \underline{y}_1^{(\ell)}, \text{ or} \\ &\overline{y}_1^{(\ell)} \le f_{1,0}\left(x^{(\ell)}\right) + c_1 \cdot f_{1,1}\left(x^{(\ell)}\right) + \ldots + c_n \cdot f_{1,n}\left(x^{(\ell)}\right), \text{ or} \\ &\ldots, \text{ or} \\ &f_{m,0}\left(x^{(\ell)}\right) + c_1 \cdot f_{m,1}\left(x^{(\ell)}\right) + \ldots + c_n \cdot f_{n,n}\left(x^{(\ell)}\right) \le \underline{y}_m{}^{(\ell)}, \text{ or} \\ &\overline{y}_m{}^{(\ell)} \le f_{m,0}\left(x^{(\ell)}\right) + c_1 \cdot f_{m,1}\left(x^{(\ell)}\right) + \ldots + c_n \cdot f_{m,n}\left(x^{(\ell)}\right). \end{aligned} \tag{20}$$

For each ℓ, we know that one of $2m$ possible inequalities (20) is satisfied.

How can we solve this problem? As we have mentioned, in the presence of negative examples, even for this case – when the dependence on the parameters is linear – the exact computations of the bounds $\underline{c}_i$ and $\overline{c}_i$ is an NP-hard problem already for $m = 1$.

However, we can use the same ideas as in the previous section and come up with a feasible algorithm for computing an *enclosure* for the desired range $[\underline{c}_i, \overline{c}_i]$, i.e., for computing the interval that *contains* the desired range. To be more precise, similarly to the above case $m = 1$, we have a family of feasible algorithms that can bring us closer and closer to the desired range.

First algorithm. The first algorithm in this sequence is when on each step, we take into account only one negative example ℓ. For each ℓ:

- we add one of the $2m$ inequalities (20) to the system (19); thus, we get $2m$ problems of minimizing c_i and $2m$ problems of maximizing c_i; for each of $2m$ pairs of linear programming problems, we thus find an interval of possible values of c_i;

- since *one* of these inequalities is satisfied, we can conclude that the desired range is contained in the *union* of the resulting $2m$ intervals; we can compute this union by computing the smallest of $2m$ lower endpoints and the largest of $2m$ upper endpoints.

For each ℓ, we know that the actual range is contained in the corresponding union – thus, it is contained in the *intersection* of these unions. To compute such an intersection:

- we compute the largest of the lower endpoints corresponding to different ℓ, and
- we compute the smallest of the upper endpoints corresponding to different ℓ.

More accurate – but more time-consuming – algorithms. To get a more accurate estimate of the desired range $[\underline{c}_i, \overline{c}_i]$, instead of taking only *one* negative example ℓ into account in each linear programming problem, we take *two* such negative examples ℓ and ℓ'i into account. We have $2m$ possible inequalities for ℓ and we have $2m$ possible inequalities for ℓ', so we have $(2m)^2$ pairs of possible inequalities.

For each pair (ℓ, ℓ'):

- we add one of $2m$ inequalities (20) corresponding to ℓ and one of $2m$ inequalities corresponding to ℓ' to the system (19); thus, we get $(2m)^2$ problems of minimizing c_i and $(2m)^2$ problems of maximizing c_i; for each of $(2m)^2$ pairs of linear programming problems, we thus find an interval of possible values of c_i;
- since *one* of these pairs of inequalities is satisfied, we can conclude that the desired range is contained in the *union* of the resulting $(2m)^2$ intervals; we can compute this union by computing the smallest of $(2m)^2$ lower endpoints and the largest of $(2m)^2$ upper endpoints.

For each pair (ℓ, ℓ'), we know that the actual range is contained in the corresponding union – thus, it is contained in the *intersection* of these unions. To compute such an intersection:

- we compute the largest of the lower endpoints corresponding to different pairs (ℓ, ℓ'), and
- we compute the smallest of the upper endpoints corresponding to different pairs (ℓ, ℓ').

Instead of pairs, we can consider triples, quadruples, etc. Every time we consider tuples with one more element, the computation time increases – but we get more accurate enclosures.

5 Case of Probabilistic Uncertainty

Probabilistic uncertainty means that for each measurement k, we know the probabilities of different possible values of the measurement error

$$\Delta y^{(k)} = y^{(k)} - y = y^{(k)} - f\left(x^{(k)}, c_1, \ldots, c_n\right),$$

i.e., we know, e.g., the probability density function $\rho_k\left(y^{(k)} - y\right)$ describing these probabilities.

In this case, the probability that a model $y = f(x, c_1, \ldots, c_n)$ is consistent with the k-th observation is proportional to $\rho_k\left(y^{(k)} - f\left(x^{(k)}, c_1, \ldots, c_n\right)\right)$. It is usually assumed that different measurements are independent. Thus, the probability that a model is consistent with all K observations is equal to the product of the corresponding probabilities

$$\prod_{k=1}^{K} \rho_k\left(y^{(k)} - f\left(x^{(k)}, c_1, \ldots, c_n\right)\right). \tag{21}$$

A natural idea is to select the values $c_1, \ldots, c_n$ for which this probability is the largest possible. This is known as the Maximum Likelihood method.

If we want to find the range of possible values of c, then we must look for all the values c for which the expression (21) is larger than or equal to a certain threshold value ρ_0 – where this threshold value can be determined, e.g., from the condition that the probability to be outside this region is equal to some pre-selected value $\varepsilon > 0$.

What if we have negative examples? In this case, instead of considering all possible combinations c, we need to consider only combinations which are consistent with all $L - K$ negative examples, i.e., combinations that satisfy the property (4) for all $\ell = K + 1, \ldots, L$.

6 Case of Fuzzy Uncertainty

What is fuzzy uncertainty: a brief reminder. In some cases, we do not know the probabilities of different values of the measurement errors, but we have expert estimates of which values are possible. These expert estimates are usually formulated by using imprecise ("fuzzy") words from natural language. For example, in the situation when the guaranteed upper bound 0.1 on the absolute value of the measurement error, an expert may say that this absolute value is not much larger than 0.05. To formalize such imprecise ("fuzzy") knowledge, Lotfi Zadeh invented special techniques – that he called *fuzzy*; see, e.g., [1, 5, 6, 10, 12, 13, 14, 17].

Degree of certainty and a membership function. In fuzzy techniques:

- for each imprecise expert statement about a quantity,
- we ask an expert to estimate, on a scale from 0 to 1, his/her degree of confidence that the expert's statement holds for this value (e.g., that 0.06 is not much larger than 0.05).

A function that assigns this degree to each possible value is called a *membership function*.

"And"- and "or"-operations. The degrees of confidence a, b, ... in individual statements $A, B, \ldots$ enable us also to estimate degrees of confidence in composite statements such as $A \,\&\, B$, $A \vee B$, etc.

The algorithms $f_{\&}(a, b)$ and $f_{\vee}(a, b)$ for such estimates are called *"and"*- and *"or"-operations*, or, for historical reasons, *t-norms* and *t-conorms*.

The most widely used "and"-operations are $\min(a, b)$ and $a \cdot b$.

Data processing under fuzzy uncertainty: a brief reminder. In line with our general assumption, let us assume that:

- we know the values $x^{(k)}$ exactly, and
- we know the corresponding y-valued $y^{(k)}$ with fuzzy uncertainty – i.e., that for each example k and for each possible value y of this quantity, we know our degree of confidence $\mu_k\left(y^{(k)} - y\right)$ that the corresponding value $y^{(k)} - y$ of the measurement error is possible.

In this case, the degree to which a model $y = f(x, c_1, \ldots, c_n)$ is consistent with the k-th observation is equal to $\mu_k\left(y^{(k)} - f\left(x^{(k)}, c_1, \ldots, c_n\right)\right)$. By applying an appropriate "and"-operation, we can conclude that the degree to which a model is consistent with all K observations is equal to

$$f_{\&}\left(\mu_1\left(y^{(1)} - f\left(x^{(1)}, c\right)\right), \ldots, \mu_K\left(y^{(K)} - f\left(x^{(K)}, c\right)\right)\right). \tag{22}$$

A natural idea is to select the values $c = (c_1, \ldots, c_n)$ for which this degree is the largest possible – and if we want to find the range of possible values of c, to select the range of all the values for which the degree (22) is larger than or equal to a certain threshold μ_0.

What if we have negative examples? In this case, instead of considering all possible combinations c, we need to consider only combinations which are consistent with all $L - K$ negative examples, i.e., combinations that satisfy the property (4) for all $\ell = K + 1, \ldots, L$.

7 Conclusions

What we did. In this paper, we provided a theoretical foundation for using negative examples in data processing, and we showed, on simplified toy examples, that the resulting algorithms indeed lead to more accurate models.

What still needs to be done. Now that the theoretical foundation has been formulated, we hope that the resulting algorithms and ideas will be applied to real-life problems.

Acknowledgments

This work was supported in part by the National Science Foundation grants 1623190 (A Model of Change for Preparing a New Generation for Professional Practice in Computer Science) and HRD-1242122 (Cyber-ShARE Center of Excellence).

The authors are thankful to Boris Kovalerchuk for his encouragement, and to the anonymous referees for valuable suggestions.

References

1. R. Belohlavek, J. W. Dauben, and G. J. Klir, *Fuzzy Logic and Mathematics: A Historical Perspective*, Oxford University Press, New York, 2017.
2. J. Contreras, F. Zapata, O. Kosheleva, V. Kreinovich, and M. Ceberio, "Let us use negative examples in regression-type problems too", *Proceedings of the 24th International Conference on Information Visualisation IV'2020*, Vienna and Melbourne, September 7–11, 2020, pp. 296–300.
3. T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, MIT Press, Cambridge, MA, 2009.
4. L. Jaulin, M. Kiefer, O. Dicrit, and E. Walter, *Applied Interval Analysis*, Springer, London, 2001.
5. G. Klir and B. Yuan, *Fuzzy Sets and Fuzzy Logic*, Prentice Hall, Upper Saddle River, New Jersey, 1995.
6. B. Kovalerchuk, "Relationship between probability and possibility theiories", In: V. Kreinovich (ed.), *Uncertainty Modeling*, Springer Verlag, Cham, Switzerland, 2017, pp. 97–122.
7. V. Kreinovich, A. Lakeyev, J. Rohn, and P. Kahl, *Computational Complexity and Feasibility of Data Processing and Interval Computations*, Kluwer, Dordrecht, 1998.
8. D. G. Luenberger and Y. Ye, *Linear and Nonlinear Programming*, Springer, Cham, Switzerland, 2016.
9. G. Mayer, *Interval Analysis and Automatic Result Verification*, de Gruyter, Berlin, 2017.
10. J. M. Mendel, *Uncertain Rule-Based Fuzzy Systems: Introduction and New Directions*, Springer, Cham, Switzerland, 2017.
11. R. E. Moore, R. B. Kearfott, and M. J. Cloud, *Introduction to Interval Analysis*, SIAM, Philadelphia, 2009.
12. H. T. Nguyen and V. Kreinovich, "Nested intervals and sets: concepts, relations to fuzzy sets, and applications", In: R. B. Kearfott and V. Kreinovich (eds.), *Applications of Interval Computations*, Kluwer, Dordrecht, 1996, pp. 245–290.
13. H. T. Nguyen, C. Walker, and E. A. Walker, *A First Course in Fuzzy Logic*, Chapman and Hall/CRC, Boca Raton, Florida, 2019.
14. V. Novák, I. Perfilieva, and J. Močkoř, *Mathematical Principles of Fuzzy Logic*, Kluwer, Boston, Dordrecht, 1999.
15. C. Papadimitriou, *Computational Complexity*, Addison-Wesley, Reading, Massachusetts, 1994.
16. S. G. Rabinovich, *Measurement Errors and Uncertainties: Theory and Practice*, Springer, New York, 2005.
17. L. A. Zadeh, "Fuzzy sets", *Information and Control*, 1965, Vol. 8, pp. 338–353.