Chapter 1
Mathematically Rigorous Global Optimization and Fuzzy Optimization
A brief comparison of paradigms, methods, similarities and differences

Ralph Baker Kearfott

Abstract Mathematically rigorous global optimization and fuzzy optimization have different philosophical underpinnings, goals, and applications. However, some of the tools used in implementations are similar or identical. We review, compare and contrast basic ideas and applications behind these areas, referring to some of the work in the very large literature base.

1.1 Introduction

Mathematical optimization in general is a huge field, with numerous experts working in each of the plethora of application areas. There are various ways of categorizing the applications, methods, and techniques. For our view, we identify the following areas in which computer methods for optimization are applied:

1. Computation of parameters in a scientific model, where the underlying physical laws are assumed to be well-known and highly accurate.
2. Computation of an optimal engineering design, where the underlying physical laws are assumed to be exact, but where there may be uncertainties in measurements of known quantities.
3. Estimation of parameters in statistical models, where a fixed population is known, a subset of that population is well-defined, and we want to estimate the proportion of the whole population in the sub-population.
4. Decision processes in the social and managerial sciences.
5. Learning in expert systems.

We have put these application areas roughly in order of decreasing “hardness" of the underlying science and formulas. In addition to particular algorithmic techniques that take advantage of problem structure in specific applications in each of these areas,

University of Louisiana at Lafayette
e-mail: rbk@louisiana.edu, rbk@lusfiber.net

these five areas admit differing underlying guiding paradigms. We give the following three such contrasting philosophies.

For scientific and engineering computing in the hard sciences (fields 1 and 2), a philosophy has been that equations would give the results (outputs) exactly, provided known quantities (inputs) are known exactly. In traditional floating point computation, best guesses for the inputs are made, the floating computations are done, and the result (output) of the floating point computations is taken in lieu of the exact answer. With the advent of modern computers, numerical analysts have recognized that small inaccuracies in the model, errors or uncertainties in the inputs, and errors introduced during the computation may have important effects on the result (output), and thus should be taken into account. Various ways of estimating or bounding the effects of these uncertainties and errors are being investigated, and such handling this kind of uncertainty has been the subject of a number of prestigious conferences. Here, we focus on interval analysis for handling this kind of uncertainty.

Statistical computations (field 3) contrast only slightly with scientific and engineering computations. In statistical computations, there aren't deterministic equations that give a unique output (optimum) given unique inputs (parameters in the objective and constraints), but the underlying laws of probability, as well as assumptions about the entire population and the sub-population, can be stated precisely. Thus, we end up with a statistical model whose optimum has certain statistical properties, assuming the model and population assumptions are correct. In this sense, statistical computations are the same as computations stemming from fields 1 and 2. Uncertainty enters into statistical computations in a similar way.

In recent decades, there has been a push to develop models for some problems in the managerial sciences that are similar to scientific and engineering models or statistical models. Linear models (linear programming) have had phenomenal success in managed environments where inputs and model assumptions are precisely known; such environments include military or government procurement and supply, or optimizing operations in a large corporation.

Equation-based models have also been applied to more fundamentally complicated situations in management and economics, such as stock market prediction and portfolio management, in free-market economies. However, certain simplifying assumptions, such as market efficiency (all participants immediately have full current information) and each player acting in their own best interest (e.g. the assumption that pure altruism and ethics are not factors) have been shown to not be universally valid. Furthermore, nonlinearities due to many players acting according to the same model are not well-understood. Thus, even if current inputs are precisely known and the objective is precisely defined, the "physical law" paradigm for analyzing such situations is debatable.

A different type of managerial or every-day decision process is where an unambiguous quantitative statement of the inputs and goals is not well-defined. An example of such a specification might be design of a customer waiting room at a service center, where the manager doesn't want an unneeded expense for an excessively large room, but also doesn't want the customers to feel too crowded, that is, the room shouldn't be too small. Here, "too small" is subjective, and depends on individual

customers, but there are sizes that most people would agree are “too small" for the anticipated number of customers, and there sizes most people would agree are “more than large enough." This is not the same as well-defined inputs but with uncertainty in their measurements. Also, there is no defined subset of “small rooms" among an existing set of “all rooms" for which a statistical estimate of “smallness" can be made. This is *ambiguity*, rather than *uncertainty* or *randomness*. Nonetheless, people routinely make decisions based on this type of ambiguity. A goal of fuzzy set theory is to model and automate decision making based on this kind of ambiguity.

In what follows, we review interval methods for handling uncertainty, and fuzzy set theory and technology for handling ambiguity. The basic ideas, fundamental equations, scope of applicability, and references for interval analysis appear in Section 1.2, while fuzzy set technology is similarly treated in Section 1.3.

We introduce our notation and basic ideas for branch and bound algorithms in Section 1.4. Ubiquitous in mixed integer programming, branch and bound algorithms are a mainstay in interval arithmetic based global optimization, are almost unavoidable in general deterministic algorithms for non-convex problems, and are also used in a significant number of algorithms based on the fuzzy paradigm.

In Section 1.5, we relate some branch and bound techniques to interval computations, while we do the same for fuzzy technology[1] in Section 1.6. There is a huge amount of published and on-going scholarly work in both interval analysis and fuzzy set theory and applications. Here, rather than try to be comprehensive, we restrict the references to a few well-known introductions and some of the work most familiar to us. We apologize to those we have left out, and would like to hear from them. We also realize that we have not comprehensively mentioned all subtleties and terminologies associated with these two areas; please refer to the references we give as starting points.

We contrast the two areas in a concluding section.

1.2 Interval Analysis: Fundamentals and Philosophy

For interval arithmetic, the underlying assumptions are:

- The inputs are well-defined quantities, but only known to within a certain level of uncertainty, such as when we have measurements with known error bars.
- The output is exactly defined, and
  - either the output can be computed exactly, or
  - an exact output can be computed, but with known error bars in the model equations, provided the arithmetic used in producing the output from the inputs is exact.

[1] Our interval details are more comprehensive, since fuzzy technology is such a large field, and since our primary expertise lies in interval analysis.

– If the computer arithmetic is not exact, the computer can provide mathematically rigorous bounds on the roundoff error resulting from an arithmetic operation[2].

1.2.1 Overview

The idea behind interval arithmetic is to encompass both measurement errors in the input and roundoff errors in the computation, to obtain the range of all possible outputs. Specifically, the logical definition of interval arithmetic is, for for $\odot \in \{+, -, \times, \div\}$,

$$\boldsymbol{x} \odot \boldsymbol{y} = \{x \odot y \mid x \in \boldsymbol{x} \text{ and } y \in \boldsymbol{y}\}, \tag{1.1}$$

for intervals $\boldsymbol{x} = [\underline{x}, \overline{x}]$ and $\boldsymbol{y} = [\underline{y}, \overline{y}]$; interval evaluation of a univariate operation or function is defined as

$$\boldsymbol{f}(\boldsymbol{x}) = \{f(x) \mid x \in \boldsymbol{x}\}. \tag{1.2}$$

Computer implementations are practical because

- it is usually sufficient to relax (1.1) to

$$\boldsymbol{x} \odot \boldsymbol{y} \supseteq \{x \odot y \mid x \in \boldsymbol{x} \text{ and } y \in \boldsymbol{y}\}, \tag{1.3}$$

  and to relax (1.2) to

$$\boldsymbol{f}(\boldsymbol{x}) \supseteq \{f(x) \mid x \in \boldsymbol{x}\}, \tag{1.4}$$

  provided the containments in (1.3) and (1.4) aren't too loose.
- The definition (1.1) corresponds to the operational definitions

$$\left.\begin{aligned}
\boldsymbol{x} + \boldsymbol{y} &= [\underline{x} + \underline{y}, \overline{x} + \overline{y}], \\
\boldsymbol{x} - \boldsymbol{y} &= [\underline{x} - \overline{y}, \overline{x} - \underline{y}], \\
\boldsymbol{x} \times \boldsymbol{y} &= [\min\{\underline{x}\underline{y}, \underline{x}\overline{y}, \overline{x}\underline{y}, \overline{x}\overline{y}\}, \max\{\underline{x}\underline{y}, \underline{x}\overline{y}, \overline{x}\underline{y}, \overline{x}\overline{y}\}] \\
\frac{1}{\boldsymbol{x}} &= [\frac{1}{\overline{x}}, \frac{1}{\underline{x}}] \qquad \text{if } \underline{x} > 0 \text{ or } \overline{x} < 0 \\
\boldsymbol{x} \div \boldsymbol{y} &= \boldsymbol{x} \times \frac{1}{\boldsymbol{y}}
\end{aligned}\right\} \tag{1.5}$$

- On a computer with control of rounding modes, such as those adhering to the IEEE 754 Standard for Floating Point Arithmetic[3], the operational definitions (1.5) may be relaxed in a mathematically rigorous way, to (1.3), by rounding the lower end point of the result down and the upper end point of the result

[2] typically, through control of the rounding mode, as specified in the IEEE 754 standard for floating point arithmetic [19].

[3] Most desktops and laptops with appropriate programming language support, and some supercomputers, adhere to this standard.

up. Similarly, using such directed rounding, known error bounds, and common techniques for evaluating standard functions, (1.4) may be implemented in a mathematically rigorous way with floating point arithmetic.

These basic ideas make interval arithmetic highly attractive. However, there are subtleties that can cause problems for the naive, such as interval dependency [63], the wrapping effect when solving differential equations[4], and the clustering effect in global optimization[5] (see [13, 29, 12] and subsequent work of other authors).

Furthermore, the above definitions leave some ambiguity concerning operational definition and semantic interpretation in cases such as $[10, 100]/[-1, 0]$, $[10, 100]/[-1, 2]$ or $[-1, 1]/[-1, 1]$. Various alternatives have been proposed and extensively discussed; consensus has resulted in IEEE 1788-2015, the IEEE standard for interval arithmetic [18].

1.2.2 Interval logic

Logical computations figure prominently in implementations of branch and bound algorithms, and traditional, interval, and fuzzy logic differ significantly.

Logic is often is based on sets. In traditional such logic, we define "true" (**T**), "false" (**F**), the binary logical operators "and" ($\wedge$) and "or" ($\vee$) , and the unary logical operator"not" ($\neg$):

Definition 1 (often-taught traditional logic based on sets) Let r and s be objects and let $\mathcal{T}$ be the set of true objects for which a property is true. Furthermore, denote the truth value of an object r by $\boldsymbol{r} = \mathfrak{T}(r)$.

- Truth values: $\mathfrak{T}(r) = \mathbf{T}$ if $r \in \mathcal{T}$ and $\mathfrak{T}(r) = \mathbf{F}$ if $r \notin \mathcal{T}$.
- The "and" operator: $\mathfrak{T}(r \wedge s) = \mathbf{T}$ if $r \in \mathcal{T}$ and $s \in \mathcal{T}$, and $\mathfrak{T}(r \wedge s) = \mathbf{F}$ otherwise.
- The "or" operator: $\mathfrak{T}(r \vee s) = \mathbf{T}$ if $r \in \mathcal{T}$ or $s \in \mathcal{T}$, and is $\mathfrak{T}(r \vee s) = \mathbf{F}$ otherwise.
- "not" operator: $\mathfrak{T}(\neg r) = \mathbf{T}$ if and only if $r \notin \mathcal{T}$.

Once logical values $\boldsymbol{r}$ have been assigned to objects r, boolean expressions, analogous to arithmetic expressions, may be formed from the logical values $\boldsymbol{r}$, with operators $\wedge$ and $\vee$ corresponding to multiplication and addition, and operator $\neg$ corresponding to negation; this leads to well-known *truth tables* defining these operations. We see truth tables for traditional logic in Table 1.1; such a truth table would be useful, for example, in searching through individual constraints to determine feasibility of an entire system of constraints. In the context of numerical computation, and branch and bound algorithms for optimization in particular, we may without loss of generality think of the logical statements r and s as numerical values and the set $\mathcal{T}$ as $[0, \infty)$, so we have:

[4] first observed in [39, pp. 90–93] and treated by many since then

[5] The clustering effect is actually in common to most basic branch and bound algorithms, whether or not interval technology is used. However, its cause is closely related to the interval dependency problem.

Table 1.1: Truth tables for standard logic

$\wedge$	**T**	**F**
T	**T**	**F**
F	**F**	**F**

$\vee$	**T**	**F**
T	**T**	**T**
F	**T**	**F**

r	$\neg(r)$
T	**F**
F	**T**

Definition 2 (traditional logic in a simplified numerical context)

- Truth values: $\mathfrak{T}(r) = \mathbf{T}$ if $r \geq 0$. and $\mathfrak{T}(r) = \mathbf{F}$ if $r < 0$.
- The "and" operator: $\mathfrak{T}(r \wedge s) = \mathbf{T}$ if $r \geq 0$ and $s \geq 0$, $\mathfrak{T}(r \wedge s) = \mathbf{F}$ otherwise.
- The "or" operator: $\mathfrak{T}(r \vee s) = \mathbf{T}$ if $r \geq 0$ or $s \geq 0$, and $\mathfrak{T}(r \vee s) = \mathbf{F}$ otherwise.
- The "not" operator: $\mathfrak{T}(\neg r) = \mathbf{T}$ if $r < 0$, and $\mathfrak{T}(\neg r) = \mathbf{F}$ otherwise.

In the context of Definition 2, the values r and s are known only to lie within sets that are intervals, that is, $r \in \boldsymbol{r} = [\underline{r}, \overline{r}]$ and $s \in \boldsymbol{s} = [\underline{s}, \overline{s}]$. In this context, what does it mean for $\boldsymbol{r}$ to be true, that is, for $\boldsymbol{r} \geq 0$? Addressing this question reveals various subtleties associated with defining logic based on intervals, when the underlying philosophy is making mathematically rigorous statements. These subtleties are codified, for example, in [18, Table 10.7]. In short, the following three cases can occur:

1. $\overline{r} < 0$, and it is certain r cannot be non-negative;
2. $\underline{r} \geq 0$, and it is certain r is non-negative;
3. $\underline{r} < 0$, and $\overline{r} \geq 0$, and r may or may not be non-negative.

This leads to the following three-valued logic with values **T**, **F**, and **U** ("unknown").

Definition 3 (interval set-based logic in simplified form)

- Truth values:
  - $\mathfrak{T}(\boldsymbol{r}) = \mathbf{T}$ if $\overline{r} \geq 0$.
  - $\mathfrak{T}(\boldsymbol{r}) = \mathbf{F}$ if $\underline{r} < 0$.
  - $\mathfrak{T}(\boldsymbol{r}) = \mathbf{U}$ otherwise.
- The "and" operator: $\mathfrak{T}(\boldsymbol{r} \wedge \boldsymbol{s}) = \mathfrak{T}(\boldsymbol{r} \cap \boldsymbol{s})$.
- The "or" operator: Let $\boldsymbol{t} = \boldsymbol{r} \cup \boldsymbol{s}$. $\mathfrak{T}(\boldsymbol{r} \vee \boldsymbol{s}) = \mathfrak{T}(\boldsymbol{r} \cup \boldsymbol{s})$.
- The "not" operator:
  - $\mathfrak{T}(\neg \boldsymbol{r}) = \mathbf{T}$ if $\underline{r} < 0$.
  - $\mathfrak{T}(\neg \boldsymbol{r}) = \mathbf{F}$ if $\overline{r} > 0$.
  - $\mathfrak{T}(\neg \boldsymbol{r}) = \mathbf{U}$ if $0 \in \boldsymbol{r}$.

Notably, interval logical expressions differ from traditional logical expressions in the sense that $\mathfrak{T}(\neg \boldsymbol{r}) \neq (\neg \mathfrak{T}(\boldsymbol{r}))$. This is important in implementations of branch and bound algorithms where the reported results are meant to be mathematically rigorous. Truth tables for our simplified interval logic appears in Table 1.2.

Table 1.2: Truth tables for interval logic

$\wedge$	T	F	U
T	T	F	U
F	F	F	F
U	U	F	U

$\vee$	T	F	U
T	T	T	T
F	T	F	U
U	T	U	U

$\neg$	
r	$\neg(r)$
T	U*
F	T
U	U

* This can be made more precise if it is known whether or not $\boldsymbol{r}$ corresponds to $\boldsymbol{r}$ with $0 \notin \boldsymbol{r}$.

1.2.3 Extensions

There is significant literature and practice on improving interval arithmetic with similar alternatives and extensions of it. Among these are:

- affine arithmetic [54, 50] to better bound expressions with almost-linear subexpressions;
- Taylor arithmetic to produce notably tighter bounds in various contexts; see, for example, [35, 11, 9] and especially [8] if available.
- random interval arithmetic, that discards mathematical rigor but has advantages if certainty is not needed [60];
- algebraic completion of the set of intervals[6] [6]
- others, with theoretical, practical, and algorithmic goals.

1.2.4 History and references

The ideas underlying interval arithmetic are sufficiently basic that they have been independently discovered, in various contexts, various times, as evidenced in [64], [61], [55], [14], etc.; based on an entry in an 1896 periodical for beginning high school pedagogy, Siegfried Rump even claims interval arithmetic was common knowledge in Germany before the turn of the twentieth century.

The most complete beginning of interval arithmetic in the twentieth century is generally recognized to be Ramon Moore's dissertation [39], and subsequent monograph [40]. Outlines of most current applications to interval arithmetic occur in Moore's dissertation. Moore published a 1979 revision [41]. For an elementary introduction to interval arithmetic, we recommend our 2009 update [42]. Another common reference text is Alefeld and Herzberger's 1972 book [3] and 1983 translation into English [4].

[6] but with complicated interpretation of results in general settings

1.3 Fuzzy Sets: Fundamentals and Philosophy

Amid a huge amount of literature, software manuals, technical descriptions, and mathematical analysis of algorithms and variants, the foundation and basic philosophy of fuzzy technology continues to be well described by its inventor Lofteh Zadeh's seminal article [65], while we also rely on [33] for additional insight. We suggest the reader new to the subject turn to these for additional reading concerning what follows here.

In contrast to interval analysis, meant to make traditional scientific measurement and computation mathematically rigorous, fuzzy set theory and associated areas are meant to allow computers to process ambiguous natural language and to make decisions in the way humans do in the absence of scientific measurements or models. Fuzzy set theory, fuzzy logic, and applications are characterized less by underlying assumptions than lack thereof[7].

The following example is amenable to processing with fuzzy technology.

Example 1 Take a college departmental tenure committee's use of student evaluation of instruction data. The committee wants to grant tenure only to professors with "good" (or sufficiently popular) ratings. However, it is *subjective* what sufficiently good means. If the students rate the professor on a scale of 1 to 5, some committee members might think a rating of 3 or greater is sufficiently good, while others might think the rating is sufficiently good only if it is greater than or equal to 4. If $\mathcal{S}$ represents the set of sufficiently good ratings, a *membership function* is used to describe $\mathcal{S}$ as a fuzzy set.

Fuzzy sets are defined via membership functions. In some ways, membership functions are like characteristic functions, and in some ways, they are like statistical distributions, with notable differences.

Definition 4 (modified from [65]) Let $\mathbb{X}$ be a domain space (a set of objects, words, numbers, etc.; discrete or continuous), and let $x \in \mathbb{X}$. The *degree of membership* of $x \in \mathbb{X}$ in a set $\mathcal{A} \subseteq \mathbb{X}$ is defined by a *fuzzy membership function*

$$\mu_A : \mathbb{X} \rightarrow [0, 1].$$

The general membership function has few mathematical restrictions: Different authors have placed various additional restrictions on membership functions, to facilitate analysis in various applications.

Definition 5 If $\mathcal{A} \subseteq \mathbb{X}$ is as in Definition 4, the pair $(\mathcal{A}, \mu_A)$ is termed a *fuzzy set* $\mathfrak{A}$.

In problem formulations and computations, particular quantities are assumed to be fuzzy or not:

Definition 6 A quantity x is called *fuzzy value* $\tilde{x}$ if it is assumed to belong to some fuzzy set $\mathfrak{X}$ with degree of membership $\mu_X(x)$. Well-defined quantities, that is, those that are not fuzzy, are called *crisp* quantities.

[7] This is not to say that mathematical implications of particular procedures are not highly analyzed.

The interval arithmetic analog of a fuzzy set is a non-degenerate interval[8], whereas a crisp value corresponds to a point, i.e. a degenerate interval.

Any membership function $\mu(t)$ for Example 1 would reasonably be like a cumulative statistical distribution with support $[1, 5]$, since it would seem reasonable for it to increase from 0 to 1 as t increases from 1 to 5. However, while statistical distributions are *derived* based on assumptions and principles, fuzzy membership functions are typically *designed* according to a subjective view of the application, ease-of-computation, and experience with what gives acceptable results[9].

Membership functions can also be similar to probability density functions:

Example 2 (A classic example) In Example 1, suppose the department chair would like to commend faculty whose evaluations are above average and to take other action whose evaluations are below average. Some might think a rating of 2 would qualify as average, and some might think a rating of 4 would qualify as average, but all might agree a rating of 3 would be average.

In Example 2, a well-designed membership function $\mu(t)$ might have $\mu(3) = 1$, $\mu(1) = 0$, $\mu(5) = 0$, and be increasing on $[1, 3]$ and decreasing on $[3, 5]$. However, unlike a probability density function, in general it would not be necessary to have $\int_1^5 \mu(t)dt \neq 1$.

Example 3 Suppose we wish to describe a quantity that we know varies within the interval $[1, 2]$, and we know that every value in the interval $[1, 2]$ is taken on by that quantity, but we know nothing else about the distribution of values of the quantity within $[1, 2]$ An appropriate membership function for the set of values of the quantity takes on might then be $\mu(t) = \chi_{[1,2]}(t)$, where $\chi_{1.2]}$ is the characteristic function of $[1, 2]$.

Example 3 highlights one relationship between interval computation and fuzzy sets.

To do computations with fuzzy sets and to make decisions, degrees of truth are defined, and the user of fuzzy set technology determines an acceptable degree of truth, defined through *alpha cuts*:

Definition 7 The set

$$\mathcal{S}_\alpha = \{x \mid \mu(x) \geq \alpha\} \quad \text{for some } \alpha \in [0, 1] \tag{1.6}$$

is called an *alpha-cut* for the fuzzy set $\mathfrak{S}$. Here, we will call a particular value α_0 the *degree of truth*[10] of the elements of $\mathcal{S}_{\alpha_0}$.

The degree of truth is like a probability, but, in some contexts, we call degrees of truth *possibilities*.

---

[8] Intervals correspond to fuzzy sets with membership function the characteristic function, i.e. indicator function, of the interval.

[9] However, data, and even statistics, can sometimes be used in the design of membership functions.

[10] The terms "degree of truth" and "degree of belief" are common in the literature concerning handling uncertain knowledge. Also, "belief functions," like membership functions, are defined. See [53], for example. We do not guarantee our definition of "degree of truth" is the same as that of all others.

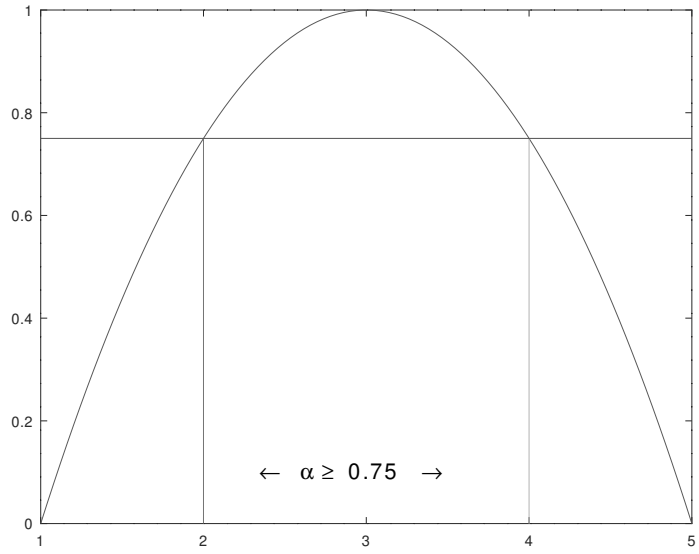

Fig. 1.1: The α-cut $\alpha \geq 0.75$ for Example 4.

For $\mathbb{X}$ an interval subset of $\mathbb{R}$ and under reasonable assumptions on $\mu(t)$, such as μ be unimodal or monotonically increasing, $\mathcal{S}_\alpha$ is an interval for each α; in Example 1, $\mathcal{S}_\alpha$ is an interval subset of $[1,5]$ for each $\alpha \in [0,1]$.

Decisions based on fuzzy computations are made according to what degree of truth α_0 is acceptable.

Example 4 Suppose, in Example 2, the possibilities are modeled by the membership function $\mu(t) = -\frac{1}{4}(t-1)(t-5)$, and the department head decides to accept an evaluation as average if it has a possibility $\alpha \geq 0.75$. Then $\mathcal{S}_{0.75} = [2,4]$; see Figure 1.1.

An important aspect of computations involving fuzzy sets involves functions of variables from a fuzzy set.

Theorem 1 *(well known; see for example [33, Section 3.4] for an explanation) Suppose the interval $\mathcal{X}_\alpha$ is an α-cut for a real quantity x, and suppose a dependency between a quantity y and the quantity $x \in \mathcal{X}$ can be expressed with a computable expression f, i.e. $y = f(x)$, where $\mathcal{X}$ corresponds to a fuzzy set $\mathfrak{X}$. Then, the range $\boldsymbol{f}(\mathcal{X}_\alpha)$ of f over $\mathcal{X}_\alpha$ is the corresponding α-cut for the value $f(x)$.*

Thus, interval arithmetic can be used for bounding α-cuts of functions of variables, and the corresponding membership function for a function f of a fuzzy variable can be approximated by subdividing $[0,1]$ and using interval arithmetic to evaluate f over each sub-interval. See, for example, [33].

Intersection and union of fuzzy sets can be defined in terms of the membership functions, as introduced in Zadeh's seminal work [65]:

Definition 8 (a classical alternative for union and intersection of fuzzy sets) If $\mathcal{S}$ is a fuzzy set with membership function μ_S and $\mathcal{T}$ is a fuzzy set with membership function μ_T, the membership function for the intersection $\mathcal{S} \cap \mathcal{T}$ may be defined as

$$T(\mathcal{S}, \mathcal{T})(x) = \mu_{S \cap T}(x) = \inf \{\mu_S(x), \mu_T(x)\}, \tag{1.7}$$

and the membership function for the union may be defined as

$$S(\mathcal{S}, \mathcal{T})(x) = \mu_{S \cup T}(x) = \sup \{\mu_S(x), \mu_T(x)\}. \tag{1.8}$$

Other, somewhat similar definitions of intersection $T(\mathcal{S}, \mathcal{T})(x)$ and union $S(\mathcal{S}, \mathcal{T})(x)$ of fuzzy sets are also defined and used; the operators $T(\mathcal{S}, \mathcal{T})$ and $S(\mathcal{S}, \mathcal{T})$, obeying certain common properties, are called *T-norms* and *S-norms*. Relations between different T-norms and S-norms have been analyzed. Which particular S- or T-norm is used in an application can depend on subjective or design considerations.

1.3.1 Fuzzy logic

Fuzzy logic is based on fuzzy union and fuzzy intersection. Here, a common definition of the membership function for "not-$\mathcal{S}$" is

$$\mu_{\neg S}(x) = 1 - \mu_S(x). \tag{1.9}$$

If (1.8), (1.7), and (1.9) are used to define truth values in fuzzy logic, then, for each degree of truth α_0 other than $\alpha_0 = 0.5$, the truth tables for the fuzzy logic are the same as the truth tables for traditional logic (Table 1.1), whereas, for $\alpha_0 = 0.5$, $\neg s$ is **T** when s is **T**.

1.3.2 A brief history

Like interval arithmetic, basic concepts in fuzzy set theory have deep roots, going back to Bernoulli and Lambert (see [53, Section 2]). Also, like Ray Moore in his dissertation and subsequent book, Lofti Zadeh clearly defines the philosophy, principles, and underlying mathematics of fuzzy sets in [65]. Since Zadeh's seminal paper, fuzzy information technology has become ubiquitous throughout computer science and technology. There are numerous conferences held world-wide each year on the subject, there are various software packages, there are numerous devices controlled with fuzzy technology, etc. Examples of recent conference proceedings are [27], [7], and [59].

1.4 The Branch and Bound Framework: Some Definitions and Details

Here, we focus on the general non-convex optimization problem, that we pose with this notation:

$$\begin{aligned} &\text{minimize } \varphi(x) \\ &\text{subject to } c_i(x) = 0, \;\; i = 1, \ldots, m_1, \\ &\qquad\qquad g_j(x) \leq 0, \;\; j = 1, \ldots, m_2, \\ &\text{where } \varphi : \mathcal{X} \subseteq \mathbb{R}^n \rightarrow \mathbb{R} \\ &\text{and each } c_i, g_j : \mathcal{X} \rightarrow \mathbb{R}, \text{ with } x = (x_1, \ldots, x_n) \in \mathcal{X}. \end{aligned} \tag{1.10}$$

The subset of $\mathbb{R}^n$ satisfying the constraints is called the *feasible set*. Here, some of the constraints $g_i \leq 0$ may be of the simple form $x_i \leq a_i$ or $\boldsymbol{x}_i \geq b_i$; these *bound constraints* are often handled separately with efficient techniques.

An important general concept in branch and bound algorithms for global optimization is that of *relaxations*:

Definition 9 A *relaxation* of the global optimization problem (1.10) is a related problem, where φ is replaced by some related function $\breve{\varphi}$, each set of constraints is replaced by a related set of constraints, such that, if φ^* is the global optimum to the original problem (1.10) and $\varphi^\dagger$ is the global optimum to the relaxed problem, then $\varphi^\dagger \leq \varphi^*$.

We will say more about relaxations in Section 1.5.

In branch and bound (B&B) methods, an initial domain is adaptively subdivided, and each sub-domain is analyzed. The general structure is outlined in Algorithm 1.

B&B algorithms represent a deterministic framework, that is, a framework within which, should the algorithm complete and use exact arithmetic, the actual global optimum (and, depending on the algorithm, possibly one or possibly all globally optimizing points) will be found. Neumaier calls such algorithms *complete*; see [44] for a perspective on this.

A common choice for the regions $\mathcal{D}$ in Algorithm 1 is rectangular paralellepipeds

$$\boldsymbol{x} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n) = ([\underline{x}_1, \overline{x}_1], [\underline{x}_2, \overline{x}_2], \ldots, [\underline{x}_n, \overline{x}_n]), \tag{1.11}$$

that is, the set

$$\left\{ x = (x_1, x_2, \ldots, x_n) \in \mathbb{R}^n \;\mid\; x_i \in [\underline{x}_i, \overline{x}_i] \text{ for } 1 \leq i \leq n \right\},$$

otherwise known as a *box* or *interval vector*. Subdivision of boxes is easily implemented, typically, but not always, by bisection in a scaled version of the widest coordinate:

$$\boldsymbol{x} \longrightarrow \{\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}\}, \tag{1.12}$$

where

$$\boldsymbol{x}^{(1)} = \left(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{i-1}, [(\underline{x}_i + \overline{x}_i)/2, \overline{x}_i], \boldsymbol{x}_{i+1}, \ldots \boldsymbol{x}_n\right)$$

Data: An initial region $\mathcal{D}^{(0)}$, the objective φ,
the constraints C, a domain stopping tolerance ε_d, and a limit M on the maximum number of regions to be allowed to be processed.
Result: Set the Boolean variable `OK`:
• Set `OK = true`, set the best upper bound $\overline{\varphi}$ for the global optimum, set the list C within which all optimizing points must lie, if the algorithm completed with less than M regions considered.
• Set `OK = false` if the algorithm could not complete.

1 Initialize the list $\mathcal{L}$ of regions to be processed to contain $\mathcal{D}^{(0)}$;
2 Determine an upper bound $\overline{\varphi}$ on the global optimum;
3 $i \leftarrow 1$;
4 **while** $\mathcal{L} \neq \emptyset$ **do**
5 $i \leftarrow i + 1$;
6 **if** $i > M$ **then return** *OK* = `false`;
7 Remove a region $\mathcal{D}$ from $\mathcal{L}$;
8 **Bound:** Determine if $\mathcal{D}$ is not infeasible, and if it is not proven to be infeasible, determine a lower bound $\underline{\varphi}$ on φ over the feasible part of $\mathcal{D}$;
9 **if** $\mathcal{D}$ *is infeasible or* $\underline{\varphi} > \overline{\varphi}$ **then** return to Step 7;
10 **Reduce:** (Possibly) eliminate portions of $\mathcal{D}$ through various efficient techniques, without subdividing.;
11 **Improve upper bound:** Possibly compute a better upper bound $\overline{\varphi}$;
12 **if** *a scaled diameter diam of* $\mathcal{D}$ *satisfies* $diam(\mathcal{D}) < \varepsilon_d$ **then**
13 Store $\mathcal{D}$ in C;
14 Return to Step 7;
15 **else**
16 **Branch:** Split $\mathcal{D}$ into two or more sub-regions whose union is $\mathcal{D}$;
17 Put each of the sub-regions into $\mathcal{L}$;
18 Return to Step 7;
19 **end**
20 **end**
21 **return** *OK* = `true`, $\overline{\varphi}$, *and* C *(possibly empty)*;

Algorithm 1: General Branch and Bound Structure

and

$$\boldsymbol{x}^{(2)} = \left(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{i-1}, [\underline{x}_i, (\underline{x}_i + \overline{x}_i)/2], \boldsymbol{x}_{i+1}, \ldots \boldsymbol{x}_n\right)$$

for the selected coordinate direction i. Besides allowing simple management of the subdivision process, uaing boxes $\boldsymbol{x}$ for the regions $\mathcal{D}$ can provide an alternative way of handling bound constraints $\underline{x} \leq x_i \leq \overline{x}$, provided sufficient care is taken in algorithm design, implementation, and interpretation of results.

Besides boxes for the regions $\mathcal{D}$, *simplexes* have appeared in some of the literature and implementations. An n-simplex is defined geometrically as the convex hull of $n+1$ points $x^{(i)}$, $0 \leq i \leq n$ in $\mathbb{R}^m$, for $m \geq n$; we define the canonical simplex in $\mathbb{R}^n$ as having $x^{(0)} = (0, \ldots, 0) \in \mathbb{R}^n$ and $x^{(i)}$ the i-th coordinate vector e_i for $1 \leq i \leq n$. There are various disadvantages to using simplexes in branch and bound methods, such as lack of simple correspondence to bounds on the variables, complicated geometries resulting from subdivision processes, and the fact that the volume of the

canonical n-simplex is only $\frac{1}{n!}$. However, some have reported success with simplexes in B&B algorithms. For instance, Paulavičius and Žilinskas [66, 46, 45] report successes on general problems; however, their methods use statistical estimates of Lipschitz constants, so are not complete in the sense of Neumaier [44].

Nonetheless, use of simplexes can be advantageous in complete algorithms for several classes of problems. This is because of two alternative characterizations of a simplex in terms of sets of constraints. The set of $n + 1$ inequality constraints

$$\sum_{i=1}^{n} x_i \leq 1, \quad x_i \geq 0 \text{ for } 1 \leq i \leq n \tag{1.13}$$

defines the canonical n-simplex. Alternatively, the set of constraints

$$\sum_{i=0}^{n} x_i = 1, \quad x_i \geq 0 \text{ for } 0 \leq i \leq n \tag{1.14}$$

defines an n simplex in $\mathbb{R}^{n+1}$. Serendipitously, a number of practical problems have such sets of constraints, so their feasible set is a subset of a simplex. An initial investigation with this in mind is [21]. An investigation into B&B algorithms based on simplexes, as well as classes of constraint sets easily converted to simplicial constraint sets (1.13) or (1.14), is currently underway [34].

We will say more about bounding techniques (Step 8 of Algorithm 1) and reducing techniques (Step 10 of Algorithm 1) in in Section 1.5.

1.5 Interval Technology: Some Details

Interval-based methods for global optimization are generally strongly deterministic in the sense they provide mathematically rigorous bounds on the global optimum, global optimizers, or both. They are largely based on branch and bound methods, in which an initial domain is recursively subdivided (branching) to obtain better and better bounds on the range of the objective function. In addition to using interval arithmetic to bound the range of the objective and constraints (if present) over each sub-region, there are various interval-based acceleration processes, such as constraint propagation and interval Newton methods. Books on interval-based global optimization include Hansen and Walster's 1983 book [17] as well as their 2003 update [16], Ratschek and Rokne [47], our monograph [24], [20] (with a special emphasis on applications and constraint propagation), and others.

Implementations of interval-based branch and bound algorithms vary in efficiency, depending on the application and how the problem is formulated. For example, only rigorous and tight bounds on the objective may be desired, bounds on a global optimizer may be desired, or bounds on all global optimizers may be desired, it may be required to prove there are points satisfying all equality constraints, or relaxations of the equality constraints may be permissible, etc.

Interval-based optimization and system solving is having an increasing number of successes in applications, but is not a panacea. This is largely due to interval dependency in the evaluation of expressions[11] and the wrapping effect[12] when integrating systems of differential equations. These problems can often be avoided with astute utilization of problem-specific features.

Here, we outline a few common interval analysis tools for B&B algorithms.

1.5.1 Interval Newton Methods

Interval Newton methods and the related Krawczyk method are a class of methods that are derived from the traditional Newton-Raphson method in several ways.

Definition 10 Suppose $F : D \subseteq \mathbb{R}^n \to \mathbb{R}^n$, suppose $\boldsymbol{x} \in D$ is an interval n-vector, and suppose that $\boldsymbol{F}'(\boldsymbol{x})$ is an interval extension of the Jacobian matrix[13] of F over $\boldsymbol{x}$ (obtained, for example, by evaluating each component with interval arithmetic), and $\check{x} \in \boldsymbol{x}$. Then a *multivariate interval Newton operator* F is any mapping $\boldsymbol{N}(F, \boldsymbol{x}, \check{x})$ from the set of ordered pairs $(\boldsymbol{x}, \check{x})$ of interval n-vectors $\boldsymbol{x}$ and point n-vectors $\check{x}$ to the set of interval n-vectors, such that

$$\tilde{\boldsymbol{x}} \leftarrow \boldsymbol{N}(F, \boldsymbol{x}, \check{x}) = \check{x} + \boldsymbol{v}, \tag{1.15}$$

where $\boldsymbol{v} \in \mathbb{IR}^n$ is any box that bounds the solution set to the linear interval system

$$\boldsymbol{F}'(\boldsymbol{x})v = -F(\check{x}). \tag{1.16}$$

Interval Newton methods can verify non-existence of optimizers in regions $\mathcal{D}$ in Algorithm 1, allowing quick rejection of such regions without extensive subdivision. In contrast to other techniques, such as examination of the feasibility of each constraint individually, interval Newton methods implicitly take advantage of the coupling between constraints and the objective. Furthermore, interval Newton methods can reduce the size of a region $\mathcal{D}$ in an often quadratically convergent iteration process, much more efficiently than if $\mathcal{D}$ would need to be subdivided. They can also sometimes prove existence of critical points of the optimization problem within a specific region $\mathcal{D}$.

This theorem summarizes some of the relevant properties:

Theorem 2 *Suppose $F : D \subseteq \mathbb{R}^n \to \mathbb{R}^n$ represents a system of nonlinear equations $F(x) = 0$, suppose F has continuous first-order partial derivatives, and suppose $\boldsymbol{N}(F, \boldsymbol{x}, \check{x})$ is the image under an interval Newton method of the box $\boldsymbol{x}$. Then any solutions $x^* \in \boldsymbol{x}$ of $F(x) = 0$ must also lie in $\boldsymbol{N}(F, \boldsymbol{x}, \check{x})$. In particular, if $\boldsymbol{N}(F, \boldsymbol{x}, \check{x}) \cap \boldsymbol{x} = \emptyset$, there can be no solutions of F in $\boldsymbol{x}$.*

[11] The results of (1.1) and (1.2) are sharp to within measurement and rounding errors, but are sometimes increasingly un-sharp when operations are combined

[12] see [42], etc.

[13] For details and generalizations, see a text on interval analysis, such as [43], [4], or our work [24].

Typically[14], if $\boldsymbol{N}(F, \boldsymbol{x}, \check{x}) \subseteq \boldsymbol{x}$, then iteration of (1.15) will result in the widths of the components of $\boldsymbol{x}$ converging quadratically to narrow intervals that contain a solution to $F = 0$. Some analysis of this may be found in our work [1, Section 8.4], while more analyses can be found in numerous other works, including [43].

Disadvantages of interval Newton methods include their limited scope of applicability and the large amount of linear algebra computations. In particular, the interval Newton equation (1.15) can be iterated to progressively narrow the bounds on a system of nonlinear equations. Interval Newton methods may not work well if the system of equations is ill-conditioned at a solution, although they do have some applicability in such cases and also when the region $\mathcal{D}$ is larger than a typical region of attraction of Newton's method (for example, see [22] and [31]). In branch and bound algorithms for global optimization, the system of equations can be either a system of equality constraints or the Kuhn-Tucker or Fritz John equations, and those can be ill-conditioned or singular at solutions, for some fairly common situations, as for example, demonstrated in [32].

Regarding linear algebra computations, interval Newton methods are generally effective only if a preconditioner matrix Y is computed, so (1.15) becomes

$$Y\boldsymbol{F}'(\boldsymbol{x})v = -YF(\check{x}), \tag{1.17}$$

and the solution set of the preconditioned system is bounded. A common preconditioner, used in various interval Newton formulations, is for Y to be the inverse of the matrix of midpoints of the interval Jacobian matrix $\boldsymbol{F}'(\boldsymbol{x})$, while other preconditioners involve solving a linear programming problem to obtain each row of the point matrix Y (see [22, 31]). Either method involves more computations than the classical floating point Newton method, and the advantages of using an interval Newton method should we weighed against the smaller cost of simpler and less costly techniques within the steps of a B&B algorithm.

1.5.2 Constraint Propagation

Constraint propagation, an entire sub-field of computer science, is strongly tied to interval arithmetic when continuous variables appear in the problem formulation. Within the context of general non-convex global optimization algorithms, it seldom leads to a precise globally optimal solution, but it can be a relatively inexpensive way of reducing the size of regions $\mathcal{D}$ in Step 10 (the "reduce" step) of Algorithm 1.

At its simplest, constraint propagation involves parsing the expressions defining the objective and constraints into a list of elementary or canonical operations, evaluating the resulting intermediate expressions, then inverting the canonical operations to obtain sharper values on the variables. We give the following very simple example for the benefit of readers new to the subject.

---

[14] under mild conditions on the interval extension of F' and for most ways of computing $\boldsymbol{v}$

Example 5

$$\begin{aligned} &\text{minimize } \varphi(x) = x_1 x_2 \\ &\text{subject to } g(x) = 1 - x_1 x_2 \leq 0 \text{ and } x_i \in [0,1],\ i = 1,2. \end{aligned} \tag{1.18}$$

The variables x_1 and x_2, as well as the intermediate results of the computation and the function and constraint values, can be represented with the following list[15].

$$\begin{aligned} v_1 &= x_1, \\ v_2 &= x_2; \\ v_3 &\leftarrow v_1 v_2, \\ v_4 &\leftarrow 1 - v_3; \\ \varphi &= v_3, \\ g &= v_4 \leq 0. \end{aligned} \tag{1.19}$$

We initialize v_1 to $[0,1]$ and v_2 to $[0,1]$, then do a "forward evaluation" of the list (1.19), obtaining $v_3 \leftarrow [0,1]$ and $v_4 \leftarrow [0,1]$. The condition $v_4 \leq 0$ gives $v_4 \in (-\infty, 0] \cap [0,1] = [0,0]$. We then solve $v_4 = 1 - v_3$ for v_3, giving $v_3 \in [1,1]$. Using this value for v_3 and solving for v_1 in $v_3 = v_1 v_2$ then gives

$$v_1 \leftarrow ([1,1]/[0,1]) \cap [0,1] = [1,\infty) \cap [0,1] = [1,1].$$

We similarly get $v_2 \leftarrow [1,1]$. In this case, a few elementary operations of constraint propagation gives $(x_1, x_2) = (1,1)$ as the only solution.

In addition to using constraints, as in Example 5, upper bounds on the objective from Step 11 of Algorithm 1 can also be used with the back propagation we have just illustrated, to narrow the bounds on possible minimizing solutions.

A notable historical example of interval constraint propagation in general global optimization software is in work of Pascal van Hentenryck ([57, 58]). Van Hentenryck's work is one of many examples, such as [38] or even our own early work [23]. Such interval constraint propagation is found in much current specialized and general global optimization software; a well-known current commercial such package is BARON [51, 52].

The book chapter [10] contains a somewhat recent review of constraint propagation processes. The COPROD series of workshops [62] is held every year, in conjunction either with one of the meetings on interval analysis or on fuzzy technology.

---

[15] Various researchers refer to similar lists as *code lists* (Rall), *directed acyclic graphs* (DAGs, Neumaier), etc. Although, for a given problem, such lists are not unique, they may be generated automatically with computer language compilers or by other means.

1.5.3 Relaxations

If $\mathcal{D}$ is a box, an interval evaluation of φ over $\mathcal{D}$ provides an easy lower bound on φ in the bounding step 8 of a branch and bound process (Algorithm 1). However, such an evaluation can be crude, since it doesn't take account of the constraints at all. For example, many problems, such as minimax and ℓ_1 data fitting, are formulated so the objective consists of a single slack variable, and all of the problem information is in the constraints. Interval Newton methods may not be useful in such situations (see [32], for instance). Furthermore, constraint propagation as explained in Section 1.5.2 doesn't take full advantage of coupling between the constraints, so may not be so useful either for certain problems. This is where relaxations step in.

Relaxations as in Definition 9 do not in general need interval arithmetic, but interval computations are frequently either central to the relaxation or are a part of the relaxation. In particular, we generally replace Problem 1.10, augmented with constraints describing the current region $\mathcal{D}$ under consideration, by a relaxation that is easier to solve than Problem 1.10; the optimum of the relaxation then provides a lower bound[16] on φ over $\mathcal{D}$.

Generally, relaxations are formed by replacing the objective φ by an objective $\tilde{\varphi}$ such that $\tilde{\varphi}(x) \leq \varphi(x)$ for all $x \in \mathcal{D}$, and replacing the feasible set $\mathcal{F}$ (defined as the portion of $\mathcal{D}$ satisfying the constraints) by a set $\tilde{\mathcal{F}}$ such that $\mathcal{F} \subseteq \tilde{\mathcal{F}}$. $\tilde{\mathcal{F}}$ is generally defined by modifying the inequality constraints. For example, $g(x) \leq 0$ can be "relaxed" by replacing g by $\tilde{g}$, where $\tilde{g}(x) \leq g(x)$ for all $x \in \mathcal{D}$; an equality constraint $c(x) = 0$ may be relaxed by first replacing it by two inequality constraints $c(x) \leq 0$ and $-c(x) \leq 0$, then relaxing these.

A classical relaxation for convex univariate φ or g is to replace g (or φ) by a tangent line approximation ℓ: $\ell(x_i) = ax_i + b$ at some point $x^{(0)} = (x_1^{(0)}, \ldots, x_i^{(0)}, \ldots, x_n^{(0)}) \in \mathcal{D}$. This can be done at multiple points $x^{(0)} \in \mathcal{D}$; the more points, the tighter the enclosure $\tilde{\mathcal{F}}$ is to $\mathcal{F}$. Non-convex univariate functions g can be relaxed by finding Lipschitz constants for g over $\mathcal{D}$, and replacing g by the affine lower bound implied by the Lipschitz constant. Mathematically rigorous Lipschitz constants can be computed with interval evaluations of g' over $\mathcal{D}$. However, we cannot tighten the enclosing set $\tilde{\mathcal{F}}$ by adding additional constraints $\ell \leq 0$ corresponding to such non-convex g without subdividing $\mathcal{D}$; if we could, we might be able to use this procedure to prove P = NP.

The parsing procedure illustrated in Section 1.5.2 with Example 5 and formulas (1.19) can be used to decompose the objective and each constraint into constraints depending on only one or two variables. The resulting constraints can then be relaxed with linear functions as we have just described[17]. The resulting relaxation

[16] To obtain a possibly better upper bound on the global optimum, we replace φ in Problem 1.10 by $-\varphi$, then form and solve a relaxation of the resulting problem. However, for mathematical rigor, computing an upper bound on φ is more complicated than computing a lower bound over $\mathcal{D}$, since the region $\mathcal{D}$ may not contain a feasible point.

[17] Two-variable relaxations correspond to multiplication; the literature describes various relaxations to these.

of the overall problem (1.10) is then a sparse linear program whose size depends on the number of operations. Such relaxations are called *McCormick Relaxations*, and, to our knowledge, first appeared in [36, 37]. Various researchers have studied such relaxations; one of our publications on the subject is [30, 26]. In [28], we[18] use the ideas to automatically analyze the difficulty (in terms of amount of non-convexity, in a certain sense) of a B&B method to solve a problem.

Other researchers have improved upon McCormick relaxations for certain problems by defining relaxations for multiple combinations of operations, more than two variables, and by using more general functions (other than linear[19]) in the relaxation. C. Floudas' group and their α-BB algorithm (see [5] and subsequent work) come to mind. In α-BB, the effects of Hessian matrices over domains $\mathcal{D}$ are bounded using interval arithmetic; see [5, 2], etc.

1.5.4 Interval Arithmetic Software

Numerous software packages are available for interval arithmetic and for interval-based global optimization; we refer the reader to the aforementioned general references. One of the most well-known current interval arithmetic packages is Siegfried Rump's Matlab toolbox INTLAB [48, 49]. Our own GlobSol software[25], based on ideas in [24], subsequent developments in constraint propagation, and our own ACM Transactions on Mathematical Software algorithms, have been much cited in comparisons with newer packages.

1.6 Fuzzy Technology: A Few Details

Since Zadeh's seminal work, a large number of papers and reports concerning underlying philosophy and technical details of optimization in a fuzzy context have appeared. One review, containing various references, is [56]; we recommend this review for further reading.

Fuzzy optimization problems can be classified in various ways. One way is according to where the fuzziness occurs:

1. *fuzzy domain, fuzzy range, crisp objective*: The domain $\mathcal{X}$ of the objective φ and the constraints c_i and g_i is a fuzzy set $\mathfrak{X}$, and the range $\mathcal{Y} = \{y \mid y = \varphi(x) \text{ for } x \in \mathcal{X}\}$ is also a fuzzy set $\mathfrak{Y}$). However, the objective φ itself is assumed to be well-defined.
2. *crisp domain, fuzzy range*: The domain $\mathcal{X}$ is a subset of a usual real vector space, but the range Y is a fuzzy set $\mathfrak{Y}$.
3. *fuzzy domain, crisp range*: The domain $\mathcal{X}$ is a fuzzy set $\mathfrak{x}$, but the range Y is a usual real vector space, and φ is well-defined for crisp (real) vectors x.

[18] Another group previously published similar ideas, with a slightly different perspective; see [15].

[19] notably, quadratics, since quadratic programs have been extensively studied

4. *fuzzy function*: Whether or not the domain or range are fuzzy, the φ or the g_i or c_i may be defined in a fuzzy way.

The concept of a fuzzy function (Type 4) corresponds to an interval-valued objective φ, such as a polynomial φ with interval coefficients, in which the values of a function at a point are intervals. Similarly, if the values of some of the c_i or g_i at points are intervals, this can be construed to define a fuzzy domain. However, the concept of a fuzzy range does not otherwise seem to have an analog in interval optimization.

Optimization based on fuzzy sets typically proceeds by somehow reformulating the problem as a non-fuzzy (crisp) optimization problem. For instance, let's consider the case where the objective and constraints are real-valued, as in the global optimization problem (1.10), but when both objective and constraints defining the feasible set are fuzzy. Then, minimizing φ can be interpreted to mean there is $x \in \mathcal{X}$ at which $\varphi(x)$ is sufficiently likely to be judged minimum and at which the degree of membership in the feasible set is acceptably high. To describe this situation, we make the following definitions.

Definition 11 Let $\mathfrak{X} = \{\mathcal{X}, \mu_X\}$ be a fuzzy set, and let f be a function as in Theorem 1. Define the fuzzy set $\mathfrak{X}^{\{f=0\}} = \{\mathcal{X}^{\{f=0\}}, \mu_X^{\{f=0\}}\}$ by

$$\mathcal{X}^{\{f=0\}} = \{x \in \mathcal{X} \mid f(x) = 0\},$$

and define $\mu_X^{\{f=0\}} : \mathcal{X} \rightarrow [0,1]$ to be a membership function[20] of x in $\mathcal{X}^{\{f=0\}}$. Similarly define $\mathfrak{X}^{\{f \leq 0\}}$.

Definition 12 Let $\mathfrak{X}$ be as in Definition 11, and let φ be an objective function as in (1.10). Define $\mu_X^{\{\min \varphi\}} : \mathcal{X} \rightarrow [0,1]$ to be a membership function for the set $\operatorname{argmin}_{x \in \mathcal{X}} \varphi(x)$.

In this context, with this notation, and with the feasible set defined by constraints c_i, $1 \leq i \leq m_1$ and g_j, $1 \leq j \leq m_2$ of (1.10), one crisp optimization problem formulation is:

$$\max_{\mathcal{X}} \min \left\{ \mu_X^{\{\min \varphi\}}(x);\ \ \mu_X^{\{c_i=0\}}(x)\,, 1 \leq i \leq m_1;\ \ \mu_X^{\{g_i \leq 0\}}(x)\,, 1 \leq j \leq m_2 \right\}. \tag{1.20}$$

Solution of the crisp problem (1.20) can proceed, e.g. via a specialized interval-based branch and bound algorithm, by non-rigorous techniques, or by local methods such as descent methods.

For a somewhat simpler example of a reformulation, consider a problem of our Type 3 classification that is unconstrained (no c_i and no g_i), and suppose we have predetermined an acceptable degree of membership α_0. We may then reformulate the problem as

$$\min_{x \in \mathcal{X}} \mu_X^{\{\min \varphi\}}(x) \ \text{ such that } \mu_X^{\{\min \varphi\}}(x) \geq \alpha_0. \tag{1.21}$$

[20] derived from μ_X and f or designed some other way, to take account of perceived goodness of values of φ

Objectives (1.20) and (1.21) are merely examples of formulations of crisp objectives corresponding to fuzzy optimization problems. In contrast to what has been developed in interval optimization, techniques for fuzzy optimization consist more of a guiding way of thinking than prescriptions for deriving algorithms.

A common problem with fuzzy optimization is that crisp reformulations often do not have isolated minimizing points, especially when the range is a fuzzy set (that is, when the values in the range of f are known only to within a degree of membership). For example, suppose we required only that $\mu_X^{\{\min \varphi\}}(x) \geq \alpha_0$ in (1.21), and not that $\mu_X^{\{\min \varphi\}}(x)$ also be minimized? Problem (1.21) then becomes a constraint satisfaction problem that usually would have an open region of solutions.

It is impossible here to cover all of the techniques in the literature for fuzzy technology, especially when dealing with specific applications. A recent proceedings is [27].

1.7 Conclusions

Fuzzy sets and interval analysis have an outwardly similar history in the twentieth century. Techniques and tools for implementing and using interval computations and fuzzy computations are similar, and implementations of fuzzy logic use interval arithmetic. However, the underlying premises are very different, based on the innate difference between measurement uncertainty and ambiguity in human thought and language. Properties of interval arithmetic are largely deduced, while properties of particular fuzzy logic systems are largely designed. While interval arithmetic strives to guarantee the range of all possible outcomes, fuzzy sets and fuzzy logic strive to automate ambiguous human perception and decision processes in a way that the outcome seems reasonable to humans.

We are currently investigating combining fuzzy sets in interval global optimization: fuzzy sets may be a way to control heuristics involved in the branching and fathoming processes.

References

1. Ackleh, A.S., Allen, E.J., Kearfott, R.B., Seshaiyer, P.: Classical and Modern Numerical Analysis: Theory, Methods, and Practice. Taylor and Francis, Boca Raton, Florida (2009)
2. Adjiman, C.S., Dallwig, S., Floudas, C.A., Neumaier, A.: A global optimization method, αBB, for general twice-differentiable constrained NLPs. I. Theoretical advances. Computers and Chemical Engineering **22**(9), 1137–1158 (1998)
3. Alefeld, G., Herzberger, J.: Nullstelleneinschließung mit dem Newton-Verfahren ohne Invertierung von Intervallmatrizen. (German) [Including zeros of nonlinear equations by the Newton method without inverting interval matrices]. Numerische Mathemaik **19**(1), 56–64 (1972)

4. Alefeld, G., Herzberger, J.: Introduction to Interval Computations. Academic Press, New York, USA (1983). Transl. by Jon G. Rokne from the original German 'Einführung in die Intervallrechnung'
5. Androulakis, I.P., Maranas, C.D., Floudas, C.A.: αbb: A global optimization method for general constrained nonconvex problems. J. Global Optimization **7**(4), 337–363 (1995). DOI 10.1007/BF01099647. URL https://doi.org/10.1007/BF01099647
6. Anguelov, R.: The algebraic structure of spaces of intervals: Contribution of Svetoslav Markov to interval analysis and its applications. BIOMATH **2** (2014). DOI 10.11145/j.biomath.2013.09.257
7. Barreto, G.A., Coelho, R. (eds.): Fuzzy Information Processing - 37th Conference of the North American Fuzzy Information Processing Society, NAFIPS 2018, Fortaleza, Brazil, July 4-6, 2018, Proceedings, *Communications in Computer and Information Science*, vol. 831. Springer (2018). DOI 10.1007/978-3-319-95312-0. URL https://doi.org/10.1007/978-3-319-95312-0
8. Berz, M., Makino, K.: Taylor models web page (2020). URL https://bt.pa.msu.edu/index_TaylorModels.htm. Accessed March 30, 2020
9. Berz, M., Makino, K., Kim, Y.K.: Long-term stability of the tevatron by verified global optimization. Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment **558**(1), 1 – 10 (2006). DOI https://doi.org/10.1016/j.nima.2005.11.035. URL http://www.sciencedirect.com/science/article/pii/S0168900205020383. Proceedings of the 8th International Computational Accelerator Physics Conference
10. Bessiere, C.: Chapter 3 - constraint propagation. In: F. Rossi, P. van Beek, T. Walsh (eds.) Handbook of Constraint Programming, *Foundations of Artificial Intelligence*, vol. 2, pp. 29 – 83. Elsevier (2006). DOI https://doi.org/10.1016/S1574-6526(06)80007-6. URL http://www.sciencedirect.com/science/article/pii/S1574652606800076
11. Corliss, G.F., Rall, L.B.: Bounding derivative ranges. In: P.M. Pardalos, C.A. Floudas (eds.) Encyclopedia of Optimization. Kluwer, Dordrecht (1999)
12. Du, K.: Cluster problem in global optimization using interval arithmetic. Ph.D. thesis, University of Southwestern Louisiana (1994)
13. Du, K., Kearfott, R.B.: The cluster problem in global optimization: The univariate case. Computing. Supplementum **9**, 117–127 (1992)
14. Dwyer, P.S.: Matrix inversion with the square root method. Technometrics **6**, 197–213 (1964)
15. Epperly, T.G.W., Pistikopoulos, E.N.: A reduced space branch and bound algorithm for global optimization. J. Global Optim. **11**(3), 287–311 (1997)
16. Hansen, E., Walster, G.W.: Global Optimization Using Interval Analysis. Marcel Dekker, Inc., New York (2003)
17. Hansen, E.R.: Global Optimization Using Interval Analysis. Marcel Dekker, New York, NY, USA (1983)
18. IEEE: 1788-2015 — IEEE Standard for Interval Arithmetic. IEEE Computer Society Press, 1109 Spring Street, Suite 300, Silver Spring, MD 20910, USA (2015). DOI https://doi.org/10.1109/IEEESTD.2015.7140721. URL http://ieeexplore.ieee.org/servlet/opac?punumber=7140719. Approved 11 June 2015 by IEEE-SA Standards Board. http://ieeexplore.ieee.org/servlet/opac?punumber=7140719
19. IEEE Task P754: IEEE 754-2008, Standard for Floating-Point Arithmetic. IEEE, New York, NY, USA (2008). DOI http://dx.doi.org/10.1109/IEEESTD.2008.4610935. URL http://en.wikipedia.org/wiki/IEEE_754-2008;http://ieeexplore.ieee.org/servlet/opac?punumber=4610933
20. Jaulin, L., Keiffer, M., Didrit, O., Walter, E.: Applied Interval Analysis. SIAM, Philadelphia (2001)
21. Karhbet, S., Kearfott, R.B.: Range bounds of functions over simplices, for branch and bound algorithms. Reliable Computing **25**, 53–73 (2017). Special volume containing refereed papers from SCAN 2016, guest editors Vladik Kreinovich and Warwick Tucker
22. Kearfott, R.B.: Preconditioners for the interval Gauss–Seidel method. SIAM Journal on Numerical Analysis **27**(3), 804–822 (1990). DOI http://dx.doi.org/10.1137/0727047

23. Kearfott, R.B.: Decomposition of arithmetic expressions to improve the behavior of interval iteration for nonlinear systems. Computing **47**(2), 169–191 (1991)
24. Kearfott, R.B.: Rigorous Global Search: Continuous Problems. No. 13 in Nonconvex optimization and its applications. Kluwer Academic Publishers, Dordrecht, Netherlands (1996)
25. Kearfott, R.B.: GlobSol User Guide. Optimization Methods and Software **24**(4–5), 687–708 (2009)
26. Kearfott, R.B.: Erratum: Validated linear relaxations and preprocessing: Some experiments. SIAM J. Optim. **21**(1), 415–416 (2011)
27. Kearfott, R.B., Batyrshin, I.Z., Reformat, M., Ceberio, M., Kreinovich, V. (eds.): Fuzzy Techniques: Theory and Applications - Proceedings of the 2019 Joint World Congress of the International Fuzzy Systems Association and the Annual Conference of the North American Fuzzy Information Processing Society IFSA/NAFIPS'2019 (Lafayette, Louisiana, USA, June 18-21, 2019), *Advances in Intelligent Systems and Computing*, vol. 1000. Springer (2019). DOI 10.1007/978-3-030-21920-8. URL https://doi.org/10.1007/978-3-030-21920-8
28. Kearfott, R.B., Castille, J.M., Tyagi, G.: Assessment of a non-adaptive deterministic global optimization algorithm for problems with low-dimensional non-convex subspaces. Optimization Methods and Software **29**(2), 430–441 (2014). DOI 10.1080/10556788.2013.780058. URL https://doi.org/10.1080/10556788.2013.780058
29. Kearfott, R.B., Du, K.: The cluster problem in multivariate global optimization. Journal of Global Optimization **5**, 253–265 (1994)
30. Kearfott, R.B., Hongthong, S.: Validated linear relaxations and preprocessing: Some experiments. SIAM J. Optim. **16**(2), 418–433 (2005). DOI http://dx.doi.org/10.1137/030602186. URL http://epubs.siam.org/sam-bin/dbq/article/60218
31. Kearfott, R.B., Hu, C.Y., Novoa III, M.: A review of preconditioners for the interval Gauss–Seidel method. Interval Computations **1**(1), 59–85 (1991). URL http://interval.louisiana.edu/reliable-computing-journal/1991/interval-computations-1991-1-pp-59-85.pdf
32. Kearfott, R.B., Muniswamy, S., Wang, Y., Li, X., Wang, Q.: On smooth reformulations and direct non-smooth computations in global optimization for minimax problems. Journal of Global Optimization **57**(4), 1091–1111 (2013)
33. Kreinovich, V.: Relations between interval and soft computing. In: C. Hu, R.B. Kearfott, A. de Korvin (eds.) Knowledge processing with interval and soft computing, Advanced information and knowledge processing, pp. 75–97. Springer-Verlag, Berlin, Germany / Heidelberg, Germany / London, UK / etc. (2008). DOI http://dx.doi.org/10.1007/BFb0085718
34. Liu, D.: A Bernstein-polynomial-based branch-and-bound algorithm for polynomial optimization over simplexes. Ph.D. thesis, Department of Mathematics, University of Louisiana, Lafayette, LA 70504-1010 USA (2021). (work in progress)
35. Makino, K., Berz, M.: Efficient control of the dependency problem based on Taylor model methods. Reliable Computing **5**(1), 3–12 (1999)
36. McCormick, G.P.: Converting general nonlinear programming problems to separable nonlinear programming problems. Tech. Rep. T-267, George Washington University, Washington (1972)
37. McCormick, G.P.: Computability of global solutions to factorable nonconvex programs. Math. Prog. **10**(2), 147–175 (1976)
38. Messine, Frederic: Deterministic global optimization using interval constraint propagation techniques. RAIRO-Oper. Res. **38**(4), 277–293 (2004). DOI 10.1051/ro:2004026. URL https://doi.org/10.1051/ro:2004026
39. Moore, R.E.: Interval arithmetic and automatic error analysis in digital computing. Ph.D. dissertation, Department of Mathematics, Stanford University, Stanford, CA, USA (1962). URL http://interval.louisiana.edu/Moores_early_papers/disert.pdf. Also published as Applied Mathematics and Statistics Laboratories Technical Report No. 25.
40. Moore, R.E.: Interval analysis. Prentice-Hall, Upper Saddle River, NJ 07458, USA (1966)
41. Moore, R.E.: Methods and Applications of Interval Analysis. SIAM, Philadelphia (1979)
42. Moore, R.E., Kearfott, R.B., Cloud, M.J.: Introduction to Interval Analysis. SIAM, Philadelphia (2009). URL http://www.loc.gov/catdir/enhancements/fy0906/2008042348-b.

html;http://www.loc.gov/catdir/enhancements/fy0906/2008042348-d.html; http://www.loc.gov/catdir/enhancements/fy0906/2008042348-t.html
43. Neumaier, A.: Interval Methods for Systems of Equations, *Encyclopedia of Mathematics and its Applications*, vol. 37. Cambridge University Press, Cambridge, UK (1990)
44. Neumaier, A.: Complete search in continuous global optimization and constraint satisfaction. In: A. Iserles (ed.) Acta Numerica 2004, pp. 271–369. Cambridge University Press (2004)
45. Paulavčius, R., Žilinskas, J.: Simplicial Global Optimization. Springer Verlag (2014). DOI 10.1007/978-1-4614-9093-7. URL http://dx.doi.org/10.1007/978-1-4614-9093-7
46. Paulavičius, R., Žilinskas, J.: Simplicial global optimization. Journal of Global Optimization **60**(4), 801–802 (2014). URL http://EconPapers.repec.org/RePEc:spr:jglopt:v:60:y:2014:i:4:p:801-802
47. Ratschek, H., Rokne, J.G.: New Computer Methods for Global Optimization. Wiley, New York (1988)
48. Rump, S.M.: INTLAB–INTerval LABoratory. In: T. Csendes (ed.) Developments in Reliable Computing: Papers presented at the International Symposium on Scientific Computing, Computer Arithmetic, and Validated Numerics, SCAN-98, in Szeged, Hungary, *Reliable Computing*, vol. 5(3), pp. 77–104. Kluwer Academic Publishers Group, Norwell, MA, USA, and Dordrecht, The Netherlands (1999). URL: http://www.ti3.tu-harburg.de/rump/intlab/
49. Rump, S.M.: INTLAB – INTerval LABoratory (1999–2020). http://www.ti3.tu-harburg.de/rump/intlab/
50. Rump, S.M., Kashiwagi, M.: Implementation and improvements of affine arithmetic. Nonlinear Theory and Its Applications, IEICE **6**(3), 341–359 (2015). DOI 10.1587/nolta.6.341
51. Sahinidis, N.V.: BARON: A general purpose global optimization software package. Journal of Global Optimization **8**(2), 201–205 (1996)
52. Sahinidis, N.V.: BARON Wikipedia page (2020). URL https://en.wikipedia.org/wiki/BARON. Accessed March 16, 2020
53. Smets, P.: The degree of belief in a fuzzy event. Inf. Sci. **25**, 1–19 (1981)
54. Stolfi, J., de Figueiredo, L.H.: An introduction to affine arithmetic. TEMA (São Carlos) **4**(3), 297–312 (2003). DOI 10.5540/tema.2003.04.03.0297. URL https://tema.sbmac.org.br/tema/article/view/352
55. Sunaga, T.: Theory of interval algebra and its application to numerical analysis. RAAG Memoirs **2**, 29–46 (1958). URL http://www.cs.utep.edu/interval-comp/sunaga.pdf
56. Tang, J.F., Wang, D.W., Fung, R.Y.K., Yung, K.L.: Understanding of fuzzy optimization: Theories and methods. Journal of Systems Science and Complexity **17**(1), 117 (2004). URL http://123.57.41.99/jweb_xtkxyfzx/EN/abstract/article_11437.shtml
57. Van Hentenryck, P.: Constraint Satisfaction in Logic Programming. MIT Press, Cambridge, MA (1989)
58. Van Hentenryck, P., McAllester, D., Kapur, D.: Solving polynomial systems using a branch and prune approach. SIAM Journal on Numerical Analysis **34**(2), 797–827 (1997). DOI 10.1137/S0036142995281504. URL https://doi.org/10.1137/S0036142995281504
59. Vellasco, M., Estevez, P. (eds.): 2018 IEEE International Conference on Fuzzy Systems, FUZZ-IEEE 2018, Rio de Janeiro, Brazil, July 8-13, 2018. IEEE (2018). URL http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=8466242
60. Žilinskas, J., Bogle, D.: A survey of methods for the estimation ranges of functions using interval arithmetic. In: Models and Algorithms for Global Optimization: Essays Dedicated to Antanas Žilinskas on the Occasion of His 60th Birthday, pp. 97–108 (2007). DOI 10.1007/978-0-387-36721-7_6
61. Warmus, M.M.: Approximations and inequalities in the calculus of approximations. classification of approximate numbers. Bull. Acad. Polon. Sci. Ser. Sci. Math. Astronom. Phys. **9**, 241–245 (1961). URL http://www.ippt.gov.pl/~zkulpa/quaphys/warmus.html
62. Wikipedia: COPROD web page (2020). URL http://coprod.constraintsolving.com/. Accessed March 17, 2020
63. Wikipedia: Interval arithmetic Wikipedia page (2020). URL https://en.wikipedia.org/wiki/Interval_arithmetic. Accessed March 30, 2020

64. Young, R.C.: The algebra of many-valued quantities. Math. Ann. **104**, 260–290 (1931)
65. Zadeh, L.A.: Fuzzy sets. Information and Control **8**, 338–353 (1965‘)
66. Žilinskas, J.: Branch and bound with simplicial partitions for global optimization. Mathematical Modelling and Analysis **13**(1), 145–159 (2008). DOI 10.3846/1392-6292.2008.13.145-159. URL http://dx.doi.org/10.3846/1392-6292.2008.13.145-159