Polyarc bounded complex interval arithmetic

Gábor Geréb[1*] and András Sándor[2,3]

[1*]Department of Informatics, University of Oslo.
[2]Central European University.
[3]HUN-REN Alfréd Rényi Institute of Mathematics.

*Corresponding author(s). E-mail(s): gaborge@uio.no;
Contributing authors: sandora@renyi.hu;

Abstract

Complex interval arithmetic is a powerful tool for the analysis of computational errors. The naturally arising rectangular, polar, and circular (together called primitive) interval types are not closed under simple arithmetic operations, and their use yields overly relaxed bounds. The later introduced polygonal type, on the other hand, allows for arbitrarily precise representation of the above operations for a higher computational cost. We propose the polyarcular interval type as an effective extension of the previous types. The polyarcular interval can represent all primitive intervals and most of their arithmetic combinations precisely and has an approximation capability competing with that of the polygonal interval. In particular, in antenna tolerance analysis it can achieve perfect accuracy for lower computational cost then the polygonal type, which we show in a relevant case study. In this paper, we present a rigorous analysis of the arithmetic properties of all five interval types, involving a new algebro-geometric method of boundary analysis.

Keywords: Interval analysis, Interval arithmetic, Tolerance analysis, Computational geometry, Algebraic geometry, Geometric algebra

MSC Classification: 14Q30 , 51M15 , 51N20 , 53A04 , 53Z30 , 65GXX , 65E05 , 08Axx

Author contributions: Conceptualization and methodology: Gábor Geréb; Formal analysis and investigation: Gábor Geréb , András Sándor; Visualization: Gábor Geréb; Writing – original draft preparation: Gábor Geréb , András Sándor; Writing – review and editing: Gábor Geréb , András Sándor;

1 Introduction

Interval analysis is an effective mathematical technique that allows for numerical analysis of problems involving sets (Moore et al, 2009). It has long been used to put bounds on computational errors and has recently been applied in robotics and robust control. It also proved to be useful for finding all the solutions of nonlinear equations and inequalities (Jaulin et al, 2001). Interval arithmetic studies the properties of the numerical representation and arithmetic operations of intervals, and therefore it is an essential part of interval analysis.

Pioneered by Moore (1963), the interval arithmetic for analyzing real-valued computational errors was soon extended to complex numbers in the form of Cartesian and polar products by Boche (1965), and as circular regions by Gargantini and Henrici (1971). Hansen (1975) studied the linear algebra of complex intervals and introduced a generalized interval arithmetic. The field received increased attention in the 1990s: Ohta et al (1990) introduced the polygon interval arithmetic, a journal titled Interval Computations (later renamed to Reliable Computing) was established (Kreinovich and Lauter, 2023; Kearfott et al, 1995), the Matlab/Octave software package called INTLAB was published by Rump (1999), and the BLAS FORTRAN package received an interval extension (Dongarra, 1995). Books on interval analysis and arithmetic were published by Petkovic and Petkovic (1998), Jaulin et al (2001), Moore et al (2009) and Dawood (2011).

Complex interval arithmetic benefited greatly from its interconnections with geometric algebra, which is widely used in the fields of computer aided design, image processing, mathematical morphology, geometrical optics, and dynamical stability analysis. The application of the Minkowski algebra to complex intervals opened up the possibility of the representation and arithmetic combination of intervals bounded by arbitrary explicit and implicit curves in the complex plane (Farouki et al, 2001, 2005). Efficient algorithms for calculating the Minkowski sum of polygons, borrowed from computational geometry, have been successfully applied in the tolerance analysis of antenna arrays (Ohta et al, 1990; Ohta, 2000; de Berg et al, 2008; Anselmi et al, 2015; Tenuti et al, 2017) The polygonal representation produced much more accurate results than the original representations, given that the vertex count was high enough (Fig. 1). However, a high vertex count came with a higher computational cost.

In the tolerance analysis of sensor arrays, polar intervals defined by independent amplitude and phase intervals are typically the primary operands of the evaluation. Motivated by the success of the polygonal representation, and its shortcomings in representing polar and circular intervals and their arithmetic combinations, we set out to find a more suitable interval type. Replacing vertices by circular arcs came as a natural extension, and lead to a new interval type, the polyarc bounded (polyarcular) interval. By providing perfect representation for a much wider set of intervals in return for a moderate increase in complexity, the new interval type suited our application well.

Polyarc is an explicit curve type, defined by an ordered set of circular arcs and consists of the defining arcs and implicit edges between them (Fig. 2). It allows the exact representation of the boundary of the rectangular, polar, circular, and polygonal intervals. It is closed under the addition, negative, reciprocal, union, and intersection operations, and in some cases under multiplication, too. As a by-product of the development process, we reviewed the literature on the existing complex interval types, and collected the relevant theorems from geometric algebra to produce a summary of the arithmetic and computational properties of complex interval types (Fig. 3).

In this paper, we provide a review of the rectangular, circular, polar, and polygonal interval types and present the previously unpublished polyarcular type. We present a rigorous mathematical analysis of their arithmetic properties based on Minkowski algebraic theorems, and compare their computational properties. To demonstrate the advantages of the new type, we show an interval analytical case study where it outperforms the existing types.

We expect this article to be of interest to both theoretical and applied researchers; therefore, we collected results with more theoretical relevance in Sections 2 and 3, and those with more application relevance in Sections 4 and 5.

In Section 2 we define the metric space of complex intervals and the subspaces of its finite data representations. We show that primitive intervals can be approximated by polygonal intervals and perfectly described by polyarcular intervals.

In Section 3 we consider basic arithmetic and set operations applied to complex intervals. We consider their boundaries relying on Minkowski algebra and Matheron (1975), as well as on geometric algebra studies of Farouki et al (2000, 2001). We show that by Gauss map matching we can identify the operand boundary segments that are relevant for the evaluation of the result boundary. We also present three methods for the analysis of the result boundaries of unary and binary operations on complex intervals, one of which has never been used in this context so far to the best of our knowledge. Then we apply these methods to straight edges and circular arcs, which constitute the boundary segments of all the complex interval types considered in this paper, and show that the arithmetic properties of the newly proposed polyarcular interval is better than those of the primitive and polygonal intervals.

In Section 4 we consider the data representation and computation of complex intervals. We introduce the tightness measure and define the type casting operation. We show how type casting and arithmetic operations can cause a loss of tightness. Finally, we show two utility processes: the trimming for extracting the simple boundary when an operation results a self-intersecting boundary, and the backtracking for identifying the subsets of the operands of an operation that maps to a point in the result interval.

In Section 5, we present a case study of antenna tolerance analysis, where the polyarcular interval type outperform the other four types.

Typesetting	Meaning	Example
Calligraphic $(\mathcal{I}, \mathcal{R})$	Sets of curves or intervals	$\mathcal{I}(\mathbb{R}) = \{\boldsymbol{t} = [\underline{t}, \overline{t}] \big\vert \underline{t} \le \overline{t}\}$
Bold-italic $(\boldsymbol{a}, \boldsymbol{A})$	Sets	$\boldsymbol{A} = \{A \in \mathbb{C} \big\vert \Re(A) = 1\}$
Italic (a, A)	Numbers	$A \in \boldsymbol{A} = \{t + it \big\vert t \in \boldsymbol{t}\}$
Lowercase $(a, \boldsymbol{a})$	Real numbers and sets	$a \in \boldsymbol{a} \subset \mathbb{R}$
Uppercase $(A, \boldsymbol{A})$	Complex numbers and sets	$A \in \boldsymbol{A} \subset \mathbb{C}$
Sans-sherif (n, N)	Natural numbers and sets	$\mathsf{n} \in \{1..\mathsf{N}\} \subset \mathbb{N}, P_{2\mathsf{n}-1} = O_{\mathsf{n}} + r_{\mathsf{n}} e^{i\overline{\varphi}_{\mathsf{n}}}$
Under- and overlined $(\underline{t}, \overline{t})$	Infimum and supremum	$[\underline{t}, \overline{t}] \ni t, \underline{t} \le t \le \overline{t}$
Fraktur $(\Re, \Im)$	Real and imaginary part	$A = \Re(A) + i\Im(A) = A^{\Re} + iA^{\Im}$

Table 1 Typesetting in mathematical expressions

At the beginning of each section we provide a summary of its findings. Detailed derivations and illustrative examples of the proofs are provided in Online Resource 1. For the sake of brevity, the typesetting of symbols in equations carry specific meanings, which is summarized in Table 1.

2 Complex interval subspaces

In this section, we define the metric space of complex intervals. We also define some of its notable subspaces with finite data representations: the primitive intervals (rectangular, polar, and circular) that are characterized by their various real-valued projections (Fig. 1), and the geometrical intervals (polygonal and polyarcular) that are characterized by their boundaries (Fig. 2). We show that primitive intervals can be approximated by polygonal intervals and perfectly described by polyarcular intervals (Fig. 3).

2.1 Complex intervals

Complex intervals are all subsets of the complex plane $\mathbb{C}$, which we will consider our universe. Since there is no common definition for complex intervals, for this discussion we assume that all of them are connected and bounded sets. As a useful simplification, we also assume that all connected and bounded intervals can be described by a simple, piece-wise smooth, closed boundary curve.

Definition 2.1 A simple curve is the continuous image of a closed real interval in the complex plane with no self-intersections. The set of piecewise smooth simple curves on the complex plane is

$$\mathcal{O}(\mathbb{C}) = \{\boldsymbol{\Gamma} = \Gamma(\boldsymbol{t}) \big\vert \Gamma : \boldsymbol{t} \to \mathbb{C}, \text{piecewise smooth}\},$$

where $\boldsymbol{t}$ is a real interval.

Definition 2.2 The set $\mathring{\mathcal{O}}(\mathbb{C}) \subset \mathcal{O}(\mathbb{C})$ of piecewise smooth Jordan curves consists of piecewise smooth simple curves with matching endpoints: $\Gamma(\underline{t}) = \Gamma(\overline{t})$.

Definition 2.3 A complex set belongs to the set $\mathcal{I}(\mathbb{C})$ of complex intervals if it is bounded by a piecewise smooth Jordan curve:

$$\boldsymbol{A} \in \mathcal{I}(\mathbb{C}) \iff \partial \boldsymbol{A} \in \mathring{\mathcal{O}}(\mathbb{C}).$$

Remark 2.1 In practice, intervals with holes – such as the annulus $A = [1, 2]e^{i[-\pi,\pi]}$ – can be handled by narrow cuts leading to the holes. Such curves are called weakly simple (Villafuerte and Wiederhold, 2022).

The complex plane can be considered a 2-dimensional Euclidean space $\mathbb{R}^2$ endowed with a multiplication. As the complex space is naturally a metric space with the Euclidean distance, the set $2^{\mathbb{C}}$ of all its subsets can be endowed with a Hausdorff distance. This distance makes the various subsets of complex intervals metric subspaces with the inherited Hausdorff metric.

Definition 2.4 The Hausdorff distance of two complex sets $\boldsymbol{A}, \boldsymbol{B} \subset \mathbb{C}$ is

$$d(\boldsymbol{A}, \boldsymbol{B}) = \inf \left(\varepsilon \geq 0 \middle| \boldsymbol{A} \oplus \boldsymbol{D}(0, \varepsilon) \supset \boldsymbol{B} \text{ and } \boldsymbol{B} \oplus \boldsymbol{D}(0, \varepsilon) \supset \boldsymbol{A} \right),$$

where $\boldsymbol{D}(0, \varepsilon)$ denotes the closed ε-disk around the origin.

This is an equivalent reformulation of to the standard definition of the Hausdorff distance, in the language of Minkowski operations.

2.2 Primitive intervals

The foundation stone of primitive complex intervals is the real interval.

Definition 2.5 The set of closed real intervals is

$$\mathcal{I}(\mathbb{R}) = \left\{ \boldsymbol{t} = [\underline{t}, \overline{t}] \middle| \underline{t} \leq \overline{t} \right\}, \quad [\underline{t}, \overline{t}] = \{ x \in \mathbb{R} \big| \underline{t} \leq x \leq \overline{t} \}.$$

The three most common primitive complex interval types are the rectangular, the polar (Boche, 1965) and the circular (Gargantini and Henrici, 1971) intervals. Illustrative examples can be found in Figure 1.

Definition 2.6 A rectangular interval

$$\boldsymbol{a} + \boldsymbol{b}i = \{ Z \in \mathbb{C} \mid \Re(Z) \in \boldsymbol{a}, \Im(Z) \in \boldsymbol{b} \}$$

(with $\boldsymbol{a}, \boldsymbol{b} \in \mathcal{I}(\mathbb{R})$) is the Cartesian product of two real intervals. The subspace of rectangular intervals is denoted by $\mathcal{R}(\mathbb{C})$.

Definition 2.7 A polar interval is defined by the polar product of two real intervals, that is

$$\boldsymbol{r} e^{i\boldsymbol{\varphi}} = \{ Z \in \mathbb{C} \mid |Z| \in \boldsymbol{r}, \angle Z \in \boldsymbol{\varphi} \}$$

with $\boldsymbol{r}, \boldsymbol{\varphi} \in \mathcal{I}(\mathbb{R})$. The subspace of polar intervals is denoted by $\mathcal{P}(\mathbb{C})$.

Definition 2.8 A circular interval is a closed disk in the complex plane, that is

$$O + [0, r] \cdot e^{i[-\pi,\pi]} = \{ Z \in \mathbb{C} \mid |Z - O| \leq r \}$$

with center $O \in \mathbb{C}$ and radius $r \in \mathbb{R}$. The set of these intervals is denoted by $\mathcal{C}(\mathbb{C})$.

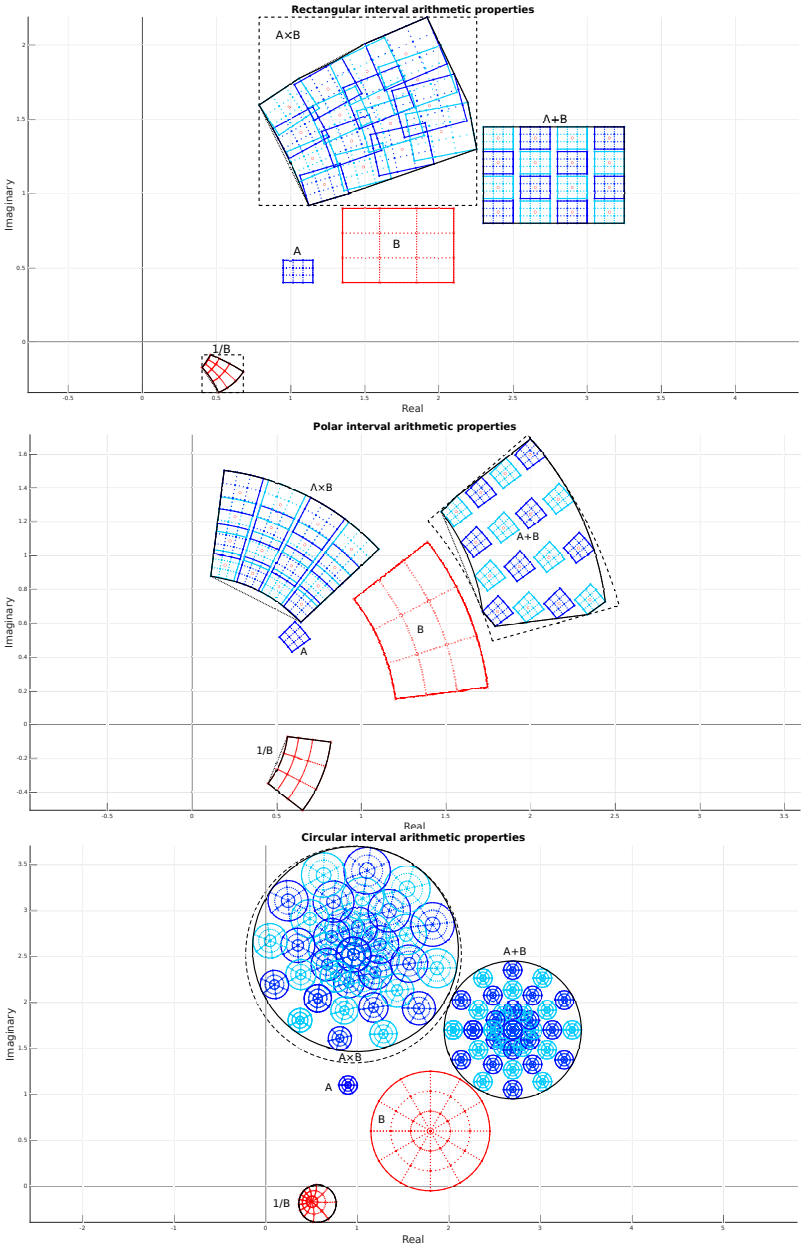

Fig. 1 Sum, product, and reciprocal of primitive intervals. Solid light lines indicate the boundaries of the operand intervals $\boldsymbol{A}$ and $\boldsymbol{B}$ and their translated and scale-rotated copies. Circles indicate internal points of operand $\boldsymbol{B}$ used to translate and scale-rotate operand $\boldsymbol{A}$. Filled dots indicate internal points of both operands. Black lines indicate the boundaries of the operation results $\boldsymbol{A} + \boldsymbol{B}$, $\boldsymbol{A} \times \boldsymbol{B}$ and $1/\boldsymbol{B}$ with solid style for the polyarcular type, dotted style for the convex polygonal type, and dashed style for the primitive type.

It can be shown that the boundaries of rectangular, polar, and circular intervals are rectangles, annular sectors, and circles, respectively, which are all piecewise smooth closed curves. Therefore, all primitive intervals are elements of the complex interval subspace.

Lemma 2.1 *The boundaries of primitive intervals consist of edges and arcs.*

Corollary 2.1.1 *All primitive intervals are complex intervals.*

$$\mathcal{R}(\mathbb{C}) \cup \mathcal{P}(\mathbb{C}) \cup \mathcal{C}(\mathbb{C}) \subset \mathcal{I}(\mathbb{C})$$

An illustration of the subspace hierarchy can be found in Figure 3.

2.3 Geometric intervals

Polygon interval arithmetic (PIA) was introduced by Ohta et al (1990) to represent uncertainty in robust control systems. Since a polygon can approximate any simple closed curve with a finite dataset, it can also represent any complex interval with arbitrary precision limited only by computational constraints.

Definition 2.9 An edge between $P_1, P_2 \in \mathbb{C}$ is

$$\Gamma = \bar{\Gamma}(P_1, P_2) : [0, 1] \to \mathbb{C}, \quad t \mapsto (1 - t)P_1 + tP_2.$$

The set of edges is $\bar{\mathcal{O}}(\mathbb{C}) \subset \mathcal{O}(\mathbb{C})$.

Definition 2.10 A closed polygonal curve can be given as

$$\Gamma : [0, \mathsf{n}] \to \mathbb{C}, \quad \Gamma(t) = \Gamma_{\mathsf{n}}(t) \text{ for } t \in [\mathsf{n} - 1, \mathsf{n}]$$

with

$$\Gamma_{\mathsf{n}}(t) = \bar{\Gamma}(P_{\mathsf{n}}, P_{\mathsf{n}+1})(t - (\mathsf{n} - 1)), \quad P_{\mathsf{N}+1} = P_1,$$

for $\mathsf{n} \in \{1..\mathsf{N}\}$. The set of closed polygonal curves is $\mathring{\bar{\mathcal{O}}}(\mathbb{C}) \subset \mathring{\mathcal{O}}(\mathbb{C})$.

Definition 2.11 The polygonal interval subspace $\mathcal{G}(\mathbb{C})$ is a collection of all complex intervals bounded by polygonal curves.

$$\boldsymbol{A} \in \mathcal{G}(\mathbb{C}) \subset \mathcal{I}(\mathbb{C}) \iff \partial \boldsymbol{A} \in \mathring{\bar{\mathcal{O}}}$$

The boundaries of primitive intervals can be represented as ordered sets of straight edges ($\bar{\Gamma}$) and circular arcs ($\breve{\Gamma}$). Polygons can, however, represent only the edges precisely, while they need to approximate arcs with a number of edges.

Remark 2.2 Polygonal intervals are complex intervals. All rectangular intervals are polygonal intervals, but polar and circular intervals are not.

$$\mathcal{R}(\mathbb{C}) \subset \mathcal{G}(\mathbb{C}), \mathcal{P}(\mathbb{C}) \not\subset \mathcal{G}(\mathbb{C}), \mathcal{C}(\mathbb{C}) \not\subset \mathcal{G}(\mathbb{C}),$$

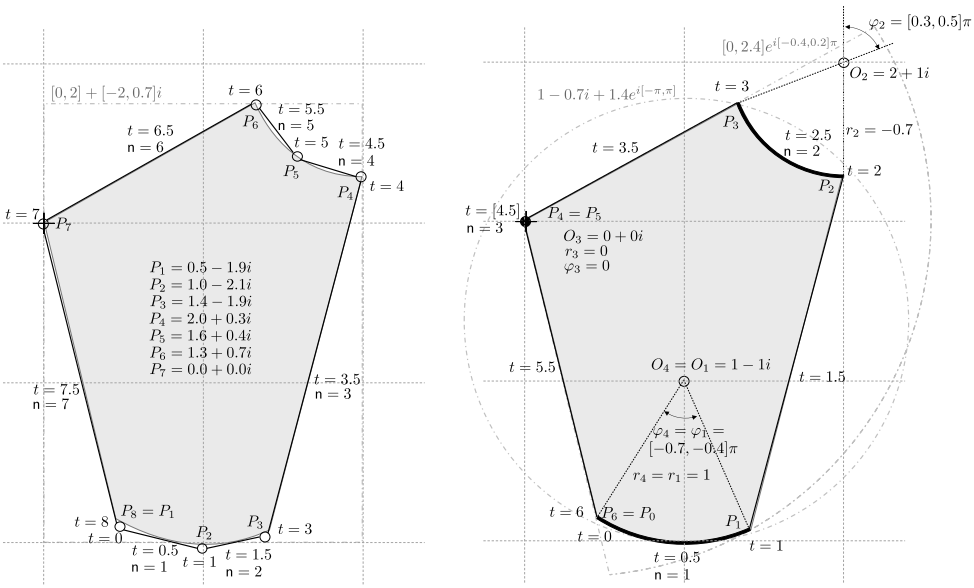

Fig. 2 Example of a complex interval represented by various complex interval types. The gray area identifies the complex interval; solid black lines identify the polygonal and polyarcular boundary curves. The grey dash-dotted lines indicate the boundaries of inclusive rectangular, polar and circular interval type objects.

It comes as a natural conclusion that a curve consisting of edges and arcs could precisely represent all primitive intervals. The polyarcular curve is a direct extension of the polygonal curve, where instead of complex-valued vertices, we use arcs with a complex-valued center point, a radius and an argument interval each to define a curve consisting of the defining arcs and implicit edges (Fig. 2.) Based on this curve, we propose a new interval subspace that contains all the primitive complex interval subspaces and has the same or better approximation capability for arbitrary complex intervals as the polygonal intervals. Polyarcular intervals are elements of the complex interval subspace by definition 2.3.

Definition 2.12 An arc is

$$\Gamma = \breve{\Gamma}(O, r, \boldsymbol{\varphi}) : [0,1] \to \mathbb{C}, \quad t \mapsto O + re^{(1-t)i\underline{\varphi} + ti\overline{\varphi}},$$

given $O \in \mathbb{C}, r \in \mathbb{R}, \boldsymbol{\varphi} \in \mathcal{I}(\mathbb{R})$. The set of arcs is $\breve{\mathcal{O}}(\mathbb{C}) \subset \mathcal{O}(\mathbb{C})$.

Remark 2.3 When arcs are used to form polyarcular curves, the radius value determines whether the arc will be convex ($r > 0$), concave ($r < 0$) or just a point ($r = 0$). To maintain the required counterclockwise point order in the concave curve segments, the following modified arc function must be used:

$$t \mapsto P + re^{(\frac{1}{2}-t')i\underline{\varphi} + (\frac{1}{2}+t')i\overline{\varphi}}, \quad t' = \operatorname{sgn}(r)\left(\frac{1}{2} - t\right).$$

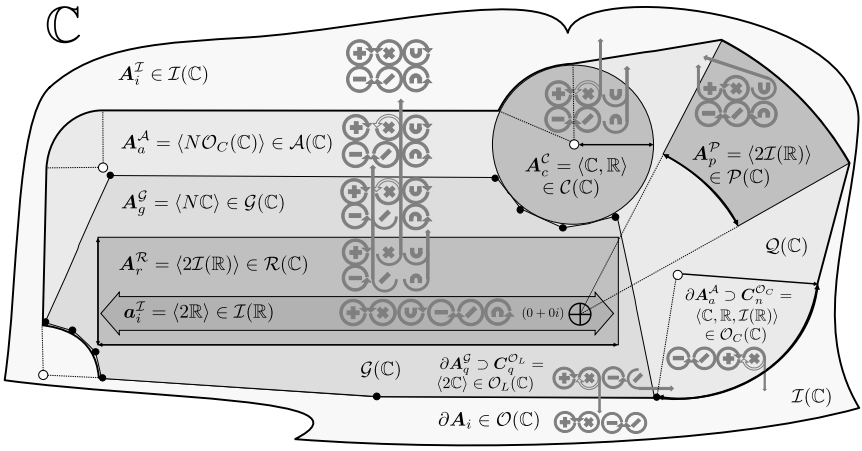

Fig. 3 Graphical summary of complex interval subspaces, types and operations. We figure illustrates a number of complex intervals from the following subspaces: rectangular ($\mathcal{R}$), polar ($\mathcal{P}$), circular ($\mathcal{C}$), polygonal ($\mathcal{G}$), polyarcular ($\mathcal{A}$). In the equation the interval symbol with the type designation is followed by the amount and type of stored parameters, and finally the notation of the narrowest subspace to which it belongs. The $\oplus$ and $\otimes$ indicate the binary addition and multiplication operations; $\ominus$ and $\oslash$ indicate the unary negative and reciprocal operations; $\cup$ and $\cap$ indicate the binary union and intersection operations. Arrows around operators indicate whether the result of an operation is in the same subspace or is outside of it. Hollow arrows indicate significant special cases (trivial exceptions, such as division by zero and union of non-intersecting intervals are not indicated).

Definition 2.13 A polyarcular curve can be given as

$$\Gamma : [0, 2\mathsf{N}] \to \mathbb{C}, \quad \Gamma(t) = \begin{cases} \breve{\Gamma}_{\mathsf{n}}(t - 2\mathsf{n} + 2)) & \text{for } t \in [2\mathsf{n} - 2, 2\mathsf{n} - 1], \\ \bar{\Gamma}_{\mathsf{n}}(t - 2\mathsf{n} + 1) & \text{for } t \in [2\mathsf{n} - 1, 2\mathsf{n}], \end{cases}$$

for $\mathsf{n} \in \{1..\mathsf{N}\}$, with alternating arc and edge pieces (either can be a single point if necessary). Precisely

$$\begin{aligned} \breve{\Gamma}_{\mathsf{n}}(t) &= \breve{\Gamma}(O_{\mathsf{n}}, r_{\mathsf{n}}, \boldsymbol{\varphi}_{\mathsf{n}})(t), \\ \bar{\Gamma}_{\mathsf{n}}(t) &= \bar{\Gamma}(P_{2\mathsf{n}-1}, P_{2\mathsf{n}})(t), \end{aligned}$$

satisfying

$$P_{2\mathsf{n}-1} = O_{\mathsf{n}} + r_{\mathsf{n}} e^{i\overline{\varphi}_{\mathsf{n}}}, \quad P_{2\mathsf{n}} = O_{\mathsf{n}+1} + r_{\mathsf{n}+1} e^{i\underline{\varphi}_{\mathsf{n}+1}},$$

with $O_{\mathsf{N}+1} = O_1$, $r_{\mathsf{N}+1} = r_1$, $\boldsymbol{\varphi}_{\mathsf{N}+1} = \boldsymbol{\varphi}_1$, $P_{2\mathsf{N}} = P_0$.

The set of polyarcular curves is $\mathring{\breve{\mathcal{O}}}(\mathbb{C}) \subset \mathring{\mathcal{O}}(\mathbb{C})$.

Definition 2.14 The polyarcular interval subspace $\mathcal{A}(\mathbb{C})$ is a collection of all complex intervals bounded by polyarcular curves.

$$\boldsymbol{A} \in \mathcal{A}(\mathbb{C}) \subset \mathcal{I}(\mathbb{C}) \iff \partial\boldsymbol{A} \in \mathring{\breve{\mathcal{O}}}$$

Corollary 2.1.2 *All primitive and polygonal intervals are polyarcular intervals.*

$$(\mathcal{R}(\mathbb{C}) \cup \mathcal{P}(\mathbb{C}) \cup \mathcal{C}(\mathbb{C})) \cup \mathcal{G}(\mathbb{C}) \subset \mathcal{A}(\mathbb{C})$$

$\boldsymbol{A}, \boldsymbol{B} \in$	$\mathcal{R}(\mathbb{C})$	$\mathcal{P}(\mathbb{C})$	$\mathcal{C}(\mathbb{C})$	$\mathcal{G}(\mathbb{C})$	$\mathcal{A}(\mathbb{C})$
$\boldsymbol{A} + \boldsymbol{B}$	$\mathcal{R}(\mathbb{C})$	$\mathcal{A}(\mathbb{C})$	$\mathcal{C}(\mathbb{C})$	$\mathcal{G}(\mathbb{C})$	$\mathcal{A}(\mathbb{C})$
$\boldsymbol{A} \times \boldsymbol{B}$	$\mathcal{I}(\mathbb{C})$	$\mathcal{P}(\mathbb{C})$	$\mathcal{I}(\mathbb{C})$	$\mathcal{I}(\mathbb{C})$	$\mathcal{I}(\mathbb{C})^*$
$-\boldsymbol{A}$	$\mathcal{R}(\mathbb{C})$	$\mathcal{P}(\mathbb{C})$	$\mathcal{C}(\mathbb{C})$	$\mathcal{G}(\mathbb{C})$	$\mathcal{A}(\mathbb{C})$
$\boldsymbol{A}^{-1}$	$\mathcal{A}(\mathbb{C})$	$\mathcal{P}(\mathbb{C})$	$\mathcal{C}(\mathbb{C})$	$\mathcal{A}(\mathbb{C})$	$\mathcal{A}(\mathbb{C})$
$\boldsymbol{A} \cap \boldsymbol{B}$	$\mathcal{R}(\mathbb{C})$	$\mathcal{P}(\mathbb{C})$	$\mathcal{A}(\mathbb{C})$	$\mathcal{G}(\mathbb{C})$	$\mathcal{A}(\mathbb{C})$
$\boldsymbol{A} \cup \boldsymbol{B}$	$\mathcal{G}(\mathbb{C})$	$\mathcal{A}(\mathbb{C})$	$\mathcal{A}(\mathbb{C})$	$\mathcal{G}(\mathbb{C})$	$\mathcal{A}(\mathbb{C})$

Table 2 Arithmetic properties of complex interval subspaces. The highlighted text indicates when the operation points outside the subspace. (* In the special case where one operand is a member of polar intervals, the result is polyarcular, which has a relevance in our case study in Section 5.)

An illustration of the subspace hierarchy can be found in Figure 3.

3 Arithmetic properties

In this section, we consider basic arithmetic and set operations applied to complex intervals. The aim is to find out whether the results of these operations belong to the same subspace as the operands, which is important for the computational implementation. Although there are existing arithmetic algorithms for primitive complex intervals based on their unique definitions (Boche, 1965; Gargantini and Henrici, 1971; Candau et al, 2006), we focus on their boundaries in order to make this arithmetic framework general across the different subspaces. We are highly relying on the Minkowski algebra structure and Matheron (1975), as well as on geometric algebra studies of Farouki et al (2000, 2001). We show that by Gauss map matching we can identify the subset of either operand that contributes to a particular element in the result interval (Fig. 4). We present three methods for the analysis of the result boundary of unary and binary operations on complex intervals (Figures 5, 6 and 7). Then we apply these methods to straight edges and circular arcs, which constitute the boundary segments of all the complex interval representations considered in this paper. We show that the arithmetic properties of the newly proposed polyarcular interval is better than those of the primitive intervals and the polygonal interval (Boche, 1965; Gargantini and Henrici, 1971; Petkovic and Petkovic, 1998; Candau et al, 2006; Ohta et al, 1990; Ohta, 2000) (Table 2, Figures 1 and 3).

3.1 Element-wise operations

Arithmetic operations on intervals are typically defined by the Minkowski algebra, which is the collection of pointwise operations on sets. Minkowski addition and multiplication are both closed binary operations ($\mathcal{I}(\mathbb{C}) \times_c \mathcal{I}(\mathbb{C}) \to \mathcal{I}(\mathbb{C})$, where $\times_c$ indicates the Cartesian product), and the element-wise negation is a closed unary operation ($\mathcal{I}(\mathbb{C}) \to \mathcal{I}(\mathbb{C})$). This means that the results of the sum, product and negative of complex intervals are also complex intervals. However, the reciprocal is only a partial unary operation because it maps the intervals that include the complex zero outside the subspace ($\mathcal{I}(\mathbb{C}) \to \mathcal{I}(\mathbb{C} \cup \frac{1}{0})$). The element-wise subtraction and division can be defined as the combination of

the operations mentioned above and consequently they are closed and partial binary operations respectively. For a graphical summary, see Figure 3.

Definition 3.1 For $\boldsymbol{A}, \boldsymbol{B} \in \mathcal{I}(\mathbb{C})$

$$\begin{aligned}
\boldsymbol{A} \oplus \boldsymbol{B} &= \{A + B \big| A \in \boldsymbol{A}, B \in \boldsymbol{B}\}, \\
\boldsymbol{A} \otimes \boldsymbol{B} &= \{A \times B \big| A \in \boldsymbol{A}, B \in \boldsymbol{B}\}, \\
-\boldsymbol{A} &= \{-A \big| A \in \boldsymbol{A}\}, \\
\boldsymbol{A}^{-1} &= \{A^{-1} \big| A \in \boldsymbol{A}\}, \\
\boldsymbol{A} \ominus \boldsymbol{B} &= \{A + (-B) \big| A \in \boldsymbol{A}, B \in \boldsymbol{B}\} = \boldsymbol{A} \oplus (-\boldsymbol{B}), \\
\boldsymbol{A} \oslash \boldsymbol{B} &= \{A \times B^{-1} \big| A \in \boldsymbol{A}, B \in \boldsymbol{B}\} = \boldsymbol{A} \otimes (\boldsymbol{B}^{-1}).
\end{aligned}$$

Similarly to the real intervals, the Minkowski addition and multiplication on complex intervals are commutative and associative but not distributive (Giardina and Dougherty, 1988; Petkovic and Petkovic, 1998). The non-distributivity is also part of a larger question called the dependency problem (Dawood, 2011; Dawood and Dawood, 2019).

Theorem 3.1 *For* $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C} \in \mathcal{I}(\mathbb{C})$

$$\begin{aligned}
&\boldsymbol{A} \oplus \boldsymbol{B} = \boldsymbol{B} \oplus \boldsymbol{A}, && \boldsymbol{A} \otimes \boldsymbol{B} = \boldsymbol{B} \otimes \boldsymbol{A}, \\
&(\boldsymbol{A} \oplus \boldsymbol{B}) \oplus \boldsymbol{C} = \boldsymbol{A} \oplus (\boldsymbol{B} \oplus \boldsymbol{C}), && (\boldsymbol{A} \otimes \boldsymbol{B}) \otimes \boldsymbol{C} = \boldsymbol{A} \otimes (\boldsymbol{B} \otimes \boldsymbol{C}), \\
&\boldsymbol{A} \otimes (\boldsymbol{B} \oplus \boldsymbol{C}) \subseteq (\boldsymbol{A} \otimes \boldsymbol{B}) \oplus (\boldsymbol{A} \otimes \boldsymbol{C}).
\end{aligned}$$

Also, there is no additive and multiplicative inverse element for non-degenerate complex intervals, and therefore no inverse operations either. (Intervals consisting of a single non-zero complex number have inverses.) We only have the following trivial inclusions.

$$\begin{aligned}
&\boldsymbol{A} \oplus (-\boldsymbol{A}) \supseteq 0, \ (\boldsymbol{A} \oplus \boldsymbol{B}) \ominus \boldsymbol{B} \supseteq \boldsymbol{A}, \\
&\boldsymbol{A} \otimes \boldsymbol{A}^{-1} \supseteq 1, \quad (\boldsymbol{A} \otimes \boldsymbol{B}) \oslash \boldsymbol{B} \supseteq \boldsymbol{A}.
\end{aligned} \tag{1}$$

Without a loss of generality we can freely translate (add a complex number to) the operands of the addition or negative operations, and rotate-and-scale (multiply by a complex number) the operands of the multiplication or reciprocal operation (Farouki et al, 2001). This allows the operation to be performed on the normalized sets and then get the unnormalized result as

$$\begin{aligned}
&\boldsymbol{A} \oplus \boldsymbol{B} = (\boldsymbol{A} + U) \oplus (\boldsymbol{B} + V) - (U + V), \\
&-\boldsymbol{A} = -(\boldsymbol{A} + U) + U, \\
&\boldsymbol{A} \otimes \boldsymbol{B} = (\boldsymbol{A} \times W) \otimes (\boldsymbol{B} \times Z) \times (W^{-1} Z^{-1}), \\
&\boldsymbol{A}^{-1} = (\boldsymbol{A} \times W)^{-1} \times W.
\end{aligned} \tag{2}$$

Set operations can be easily defined by element-wise conditions. The union and intersection are both partial binary operations because the union and intersection of two disconnected complex intervals are not bounded by a Jordan curve.

3.2 Backtracking

The set operations allow the reformulation of the arithmetic operations as

$$\begin{aligned} \boldsymbol{A} \oplus \boldsymbol{B} &= \bigcup_{B \in \boldsymbol{B}} \boldsymbol{A} + B = \{Z \big| \boldsymbol{A} \cap (-\boldsymbol{B} + Z) \neq \emptyset\} \\ \boldsymbol{A} \otimes \boldsymbol{B} &= \bigcup_{B \in \boldsymbol{B}} \boldsymbol{A} \times B = \{Z \big| \boldsymbol{A} \cap (\boldsymbol{B}^{-1} \times Z) \neq \emptyset\}. \end{aligned} \tag{3}$$

Using these identities, we can identify the subset of either operand that contributes to a particular element in the result interval.

Definition 3.2 Let $\boldsymbol{A}, \boldsymbol{B} \in \mathcal{I}(\mathbb{C})$. For an element of the sum $Z \in \boldsymbol{A} \oplus \boldsymbol{B}$, we define

$$\boldsymbol{A}_Z = \boldsymbol{A} \cap (-\boldsymbol{B} + Z), \quad \boldsymbol{B}_Z = \boldsymbol{B} \cap (-\boldsymbol{A} + Z),$$

satisfying

$$\forall A \in \boldsymbol{A}_Z, \exists B \in \boldsymbol{B}_Z : A + B = Z,$$

Similarly, for a product element $Z \in \boldsymbol{A} \otimes \boldsymbol{B}$, let

$$\boldsymbol{A}_Z = \boldsymbol{A} \cap (\boldsymbol{B}^{-1} \times Z), \quad \boldsymbol{B}_Z = \boldsymbol{B} \cap (\boldsymbol{A}^{-1} \times Z),$$

that satisfies

$$\forall A \in \boldsymbol{A}_Z, \exists B \in \boldsymbol{B}_Z : A \times B = Z.$$

3.3 Boundary analysis

For an element Z of the result boundary the subsets $\boldsymbol{A}_Z, \boldsymbol{B}_Z$ of definition 3.2 will be subsets of the operand boundaries, and in many cases they consist of single points. Therefore, the result boundary of the arithmetic operations can be defined using only the boundary points of the operands (Farouki et al, 2000).

Proposition 3.2 *For* $\boldsymbol{A}, \boldsymbol{B} \in \mathcal{I}(\mathbb{C})$.

$$\begin{aligned} &\partial(\boldsymbol{A} \oplus \boldsymbol{B}) \subseteq \partial\boldsymbol{A} \oplus \partial\boldsymbol{B}, \quad \partial(-\boldsymbol{A}) = -(\partial\boldsymbol{A}), \\ &\partial(\boldsymbol{A} \otimes \boldsymbol{B}) \subseteq \partial\boldsymbol{A} \otimes \partial\boldsymbol{B}, \quad \partial(\boldsymbol{A}^{-1}) = (\partial\boldsymbol{A})^{-1}, \\ &\partial(\boldsymbol{A} \cup \boldsymbol{B}) \subseteq \partial\boldsymbol{A} \cup \partial\boldsymbol{B}, \\ &\partial(\boldsymbol{A} \cap \boldsymbol{B}) \subseteq \partial\boldsymbol{A} \cup \partial\boldsymbol{B}. \end{aligned}$$

Definition 3.3 An interval boundary segment is a regular piece of a complex interval boundary (see Definition 2.3).

$$\boldsymbol{\Gamma}_{A,\mathsf{n}} = \Gamma_{A,\mathsf{n}}(\boldsymbol{s}) \subset \partial\boldsymbol{A} \in \mathring{\mathcal{O}}$$

Definition 3.4 An interval boundary vertex is the intersection point of two interval boundary segments (see definition 3.3).

$$P_{A,\mathsf{n}} = \boldsymbol{\Gamma}_{A,\mathsf{n}-1} \cap \boldsymbol{\Gamma}_{A,\mathsf{n}}, \quad P_{A,1} = \boldsymbol{\Gamma}_{A,\mathsf{N}} \cap \boldsymbol{\Gamma}_{A,1}, \mathsf{n} \in \{1..\mathsf{N}\} \tag{4}$$

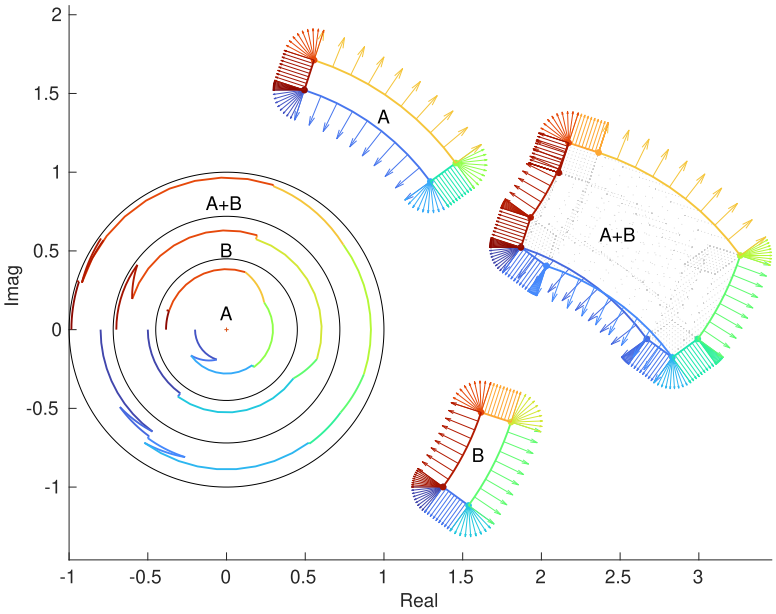

Fig. 4 Two polar intervals and their sum combined with their Gauss maps on the same plot. The three concentric curves in the unit circle shows their Gauss maps, where the argument and the lightness indicates the curve normal angle, and the radius indicates the position on the curve. The arrows show the curve normal vectors.

3.3.1 Gauss map

Gauss map matching (Farouki et al, 2000, 2001) or curve convolution (Lee et al, 1999) is a direct way to identify the operand elements that contribute to the boundary of the sum of two curves. It requires the matching of the normal arguments of the boundary curves. The resulting sum is a boundary point with the same argument as the contributing points (Fig. 4.) This method can be also applied to the boundary of the product using the logarithms of the curves, leading to an equation with the argument of the normalized curve normal.

Proposition 3.3 (Gauss map matching) *For two regular curves* $F, G : \mathbb{R} \to \mathbb{C}$

$$F(s_0) + G(t_0) \in \partial\{F(\mathbb{R}) + G(\mathbb{R})\} \implies \angle F'(s_0) = \angle G'(t_0),$$
$$F(s_0) \times G(t_0) \in \partial\{F(\mathbb{R}) \times G(\mathbb{R})\} \implies \angle \frac{F'(s_0)}{F(s_0)} = \angle \frac{G'(t_0)}{G(t_0)}.$$

Remark 3.1 The normalization of the operands shown in (2) does not affect their Gauss maps.

Let us consider the Gauss map

$$\gamma(F) : \boldsymbol{s} \to \mathbb{R}, \quad \gamma(F, s) = \angle i F'(s),$$

that returns the argument of the normal to the curve. Let us denote the argument of F by $\dot{\gamma}(F,s) = \angle F(s)$. The logarithmic Gauss map function is then defined as

$$\mathring{\gamma}(F,s) = \gamma(\log F, s) = \angle \frac{F'(s)}{F(s)} = \gamma_F - \dot{\gamma}_F$$

with a consistent choice of complex logarithm.

Let us apply the Gauss map on a curve segment $F(\boldsymbol{s})$. The result is the real set

$$\boldsymbol{\gamma}_F = \gamma(F,\boldsymbol{s}) = \{\gamma(F,s) \big| s \in \boldsymbol{s}\}.$$

The same applies to the logarithmic Gauss map.

Let us extend the Gauss map (and the logarithmic Gauss map) to a vertex P at the intersection of two neighboring curve segments $F_1(\boldsymbol{s}_1)$ and $F_2(\boldsymbol{s}_2)$. We define γ_P to be a monotonic function onto the interval $[\gamma_{F_1}(\overline{s_1}), \gamma_{F_2}(\underline{s_2})]$ with a consistent choice of the argument and logarithm functions.

We can now apply the Gauss map matching (Proposition 3.3) to this setup.

Proposition 3.4 *For two boundary segments* $\boldsymbol{\Gamma}_{A,\mathsf{n}} \subset \partial\boldsymbol{A}$ *and* $\boldsymbol{\Gamma}_{B,\mathsf{k}} \subset \partial\boldsymbol{B}$

$$\begin{aligned} \boldsymbol{\Gamma}_{A,\mathsf{n}} \oplus \boldsymbol{\Gamma}_{B,\mathsf{k}} \subset \partial(\boldsymbol{A} \oplus \boldsymbol{B}) &\implies \boldsymbol{\gamma}_{\Gamma_{A,\mathsf{n}}} \cap \boldsymbol{\gamma}_{\Gamma_{B,\mathsf{k}}} \neq \emptyset, \\ \boldsymbol{\Gamma}_{A,\mathsf{n}} \otimes \boldsymbol{\Gamma}_{B,\mathsf{k}} \subset \partial(\boldsymbol{A} \otimes \boldsymbol{B}) &\implies \mathring{\boldsymbol{\gamma}}_{\Gamma_{A,\mathsf{n}}} \cap \mathring{\boldsymbol{\gamma}}_{\Gamma_{B,\mathsf{k}}} \neq \emptyset. \end{aligned}$$

For a boundary segment $\boldsymbol{\Gamma}_{A,\mathsf{n}} \subset \partial\boldsymbol{A}$ *and a boundary vertex* $P_{B,\mathsf{k}} \in \partial\boldsymbol{B}$

$$\begin{aligned} \boldsymbol{\Gamma}_{A,\mathsf{n}} + P_{B,\mathsf{k}} \subset \partial(\boldsymbol{A} \oplus \boldsymbol{B}) &\implies \boldsymbol{\gamma}_{\Gamma_{A,\mathsf{n}}} \cap \boldsymbol{\gamma}_{P_{B,\mathsf{k}}} \neq \emptyset, \\ \boldsymbol{\Gamma}_{A,\mathsf{n}} \times P_{B,\mathsf{k}} \subset \partial(\boldsymbol{A} \otimes \boldsymbol{B}) &\implies \mathring{\boldsymbol{\gamma}}_{\Gamma_{A,\mathsf{n}}} \cap \mathring{\boldsymbol{\gamma}}_{P_{B,\mathsf{k}}} \neq \emptyset. \end{aligned}$$

For two boundary vertices $P_{A,\mathsf{n}} \in \partial\boldsymbol{A}$ *and* $P_{B,\mathsf{k}} \in \partial\boldsymbol{B}$

$$\begin{aligned} P_{A,\mathsf{n}} + P_{B,\mathsf{k}} \subset \partial(\boldsymbol{A} \oplus \boldsymbol{B}) &\implies \boldsymbol{\gamma}_{P_{A,\mathsf{n}}} \cap \boldsymbol{\gamma}_{P_{B,\mathsf{k}}} \neq \emptyset, \\ P_{A,\mathsf{n}} \times P_{B,\mathsf{k}} \subset \partial(\boldsymbol{A} \otimes \boldsymbol{B}) &\implies \mathring{\boldsymbol{\gamma}}_{P_{A,\mathsf{n}}} \cap \mathring{\boldsymbol{\gamma}}_{P_{B,\mathsf{k}}} \neq \emptyset. \end{aligned}$$

3.3.2 Arithmetic combination of polyarcular boundaries

As a consequence of Propositions 3.2 and 3.4, the arithmetic combination of two polyarcular intervals will result in a polyarcular interval iff the arithmetic combinations of the operand boundary segments with overlapping Gauss map produce only vertices, edges and arcs. The polyarcular arithmetic properties are summarized by Figure 3 and Tables 2 and 4.

Since the combination of a vertex with any other boundary element type is a simple translation for addition and a scale-rotation for multiplicative combination, the result boundary segment will be an edge or an arc. Therefore, in the following analysis, we will focus on the combinations of edges and arcs. We describe three methods using their implicit equations in the Cartesian and polar coordinate system, and their parametrizations, to determine the envelope and the parametric condition of the envelope being part of the boundary. We will denote the segments at hand as

	Parametrization $F(s)$ $G(s)$	Cartesian equation $f(x,y)$ $g(x,y)$	Cartesian boundary $s(x,y)$ $t(x,y)$	Polar equation $\mathring{f}(\rho,\theta)$ $\mathring{g}(\rho,\theta)$	Polar boundary $\mathring{s}(\rho,\theta)$ $\mathring{t}(\rho,\theta)$
$\bar{\mathcal{O}}_{0/}$	$s+ias$	$ax-y$	x	$\lvert\tan(\theta)\rvert - a$	$\rho\cos(\theta)$
$\bar{\mathcal{O}}_{0-}$	s	y	x	$\tan(\theta)$	ρ
$\bar{\mathcal{O}}_{1\lvert}$	$1+is$	$x-1$	y	$\frac{1}{\rho}-\cos(\theta)$	$\rho\sin(\theta)$
$\breve{\mathcal{O}}_0$	re^{is}	$x^2+y^2-r^2$	$\operatorname{atan2}\binom{y,}{x}$	$\rho - r$	θ
$\breve{\mathcal{O}}_1$	$re^{is}+1$	$(x-1)^2+y^2-r^2$	$\operatorname{atan2}\binom{y,}{x-1}$	$\rho^2-2\rho\cos(\theta)+1-r^2$	$\operatorname{atan2}\binom{\rho\sin(\theta),}{\rho\cos(\theta)-1}$

Table 3 Functions for the implicit and parametric equations used in the analysis of edges and arcs. The gradient of $\bar{\mathcal{O}}_0 = \bar{\mathcal{O}}(\mathbb{C}, P_1, P_2)$ is $a = \Re(P_2 - P_1)/\Im(P_2 - P_1)$. Conversion between the coordinate systems: $x(\rho,\theta) = \rho\cos(\theta)$, $y(\rho,\theta) = \rho\sin(\theta)$, $\rho(x,y) = \sqrt{x^2+y^2}$, $\theta(x,y) = \operatorname{atan2}(y,x)$.

$$\begin{aligned}\boldsymbol{\Gamma} = \boldsymbol{\Gamma}_{A,\mathsf{n}} = \{F(s)\big|s \in \boldsymbol{s}\} &= \{x+iy\big|f(x,y)=0, s(x,y)\in\boldsymbol{s}\}\\ &= \{\rho e^{i\theta}\big|\mathring{f}(\rho,\theta)=0, \mathring{s}(\rho,\theta)\in\boldsymbol{s}\},\\ \boldsymbol{\Gamma'} = \boldsymbol{\Gamma}_{B,\mathsf{k}} = \{G(t)\big|s \in \boldsymbol{t}\} &= \{x+iy\big|g(x,y)=0, t(x,y)\in\boldsymbol{t}\}\\ &= \{\rho e^{i\theta}\big|\mathring{g}(\rho,\theta)=0, \mathring{t}(\rho,\theta)\in\boldsymbol{t}\}.\end{aligned} \tag{5}$$

Applying the normalization in (2), our analysis of additive operations can be restricted to segments of zero-crossing lines ($\bar{\mathcal{O}}_{0/}$) and zero-centered circles ($\breve{\mathcal{O}}_0$). While our analysis of multiplicative operations can be restricted to segments of zero-crossing horizontal and one-crossing vertical lines ($\bar{\mathcal{O}}_{0-}$ and $\bar{\mathcal{O}}_{1\lvert}$) and zero-centered and one-centered circles ($\breve{\mathcal{O}}_0$ and $\breve{\mathcal{O}}_1$). We collected the functions of these subspaces in Table 3.

It is well published that the results of additive combinations of lines and circles are bounded by linear and circular curves (de Berg et al, 2008). However, Farouki et al. showed that multiplicative combinations of lines and circles result in envelopes (Bruce and Giblin, 1981) that are not linear or circular functions (Farouki et al, 2001). Therefore it is clear that some, but not all arithmetic combinations of polyarc curve segments yield polyarc bounded sets.

3.3.3 Mixed envelope evaluation

Applying Farouki's method (Farouki et al, 2001) we can formulate the result of a binary operation ($\bigcirc = \oplus$ or $\otimes$) as follows. Let us consider the implicit equation of $\boldsymbol{\Gamma} \subset \partial\boldsymbol{A}$ and the parametrization of $\boldsymbol{\Gamma'} \subset \partial\boldsymbol{B}$. That gives the description

$$\boldsymbol{\Gamma} \bigcirc \boldsymbol{\Gamma'} = \{(x+iy) \bigcirc G(t) \ \big|f(x,y) = 0, s(x,y) \in \boldsymbol{s}, t \in \boldsymbol{t}\}$$

of the result. Similarly to backtracking, if we consider $Z = A + B \in \boldsymbol{\Gamma} \oplus \boldsymbol{\Gamma'}$ (respectively, $Z \in \boldsymbol{\Gamma} \otimes \boldsymbol{\Gamma'}$) then we get $Z - B = Z - G(t) = A \in \boldsymbol{\Gamma}$ for some $t \in \boldsymbol{t}$ (respectively, $Z/G(t) \in \boldsymbol{\Gamma}$). This gives us an equation $\hat{h} = 0$ – in place of the equation $f = 0$ – for the result:

$$\boldsymbol{\Gamma} \bigcirc \boldsymbol{\Gamma'} = \{x+iy\big|\hat{h}(x,y,t) = 0, u(x,y,t) \in \boldsymbol{s}\big|t \in \boldsymbol{t}\}, \tag{6}$$

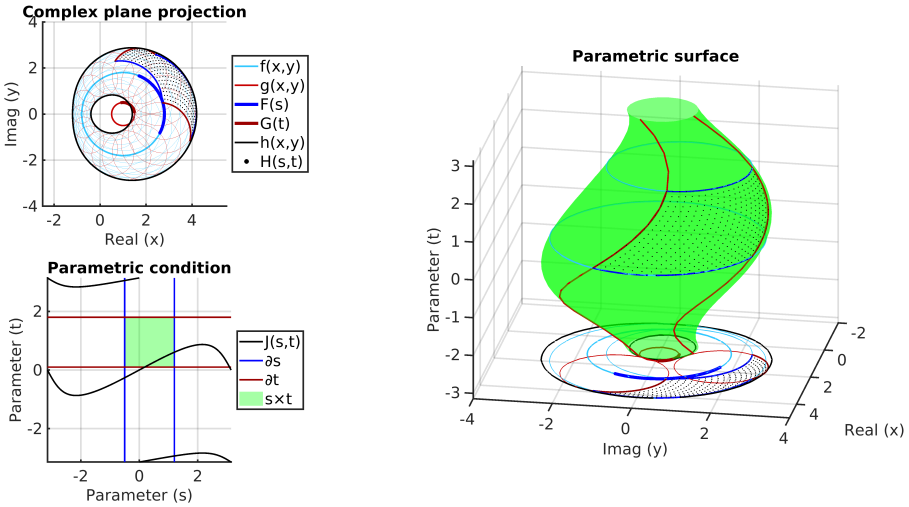

Fig. 5 Evaluation of the product of two arcs. The plot in the top left shows the sampled result and the implicit boundary curves besides the operands on the complex plane. The plot on the right shows the sampled result on its implicit surface constrained by planes over the complex plane. The plot in the bottom left shows the parametric condition of the boundary crossing the envelope. Similar figures for all edge and arc arithmetic combinations can be found in Online Resource 1.

where

$$\hat{h}(x,y,t) = \begin{cases} f\left((x+iy) - G(t)\right) = f(x - \Re(G(t)), y - \Im(G(t))) & \text{if } \bigcirc = \oplus, \\ f\left((x+iy)/G(t)\right) = \mathring{f}\left(\rho(x,y)/|G(t)|, \theta(x,y) - \angle G(t)\right) & \text{if } \bigcirc = \otimes, \end{cases}$$

where $\rho(x,y) = \sqrt{x^2+y^2}$, $\theta(x,y) = \text{atan2}(y,x)$ and $u(x,y,t)$ can be calculated from $s(x,y)$ and $\mathring{s}(x,y)$ in the same way as $\hat{h}(x,y,t)$ from $f(x,y)$ and $\mathring{f}(x,y)$.

A boundary point in this region is either an envelope point or an extremal point corresponding to one (or both) of the constraints marking the ends of the curve segments. The envelope (or discriminant) of the family defined by $\hat{h}(x,y,t) = 0$ is the set of (x,y) points for which there is a t satisfying $h = \partial h/\partial t = 0$ (Bruce and Giblin, 1981), (Bruce and Giblin, 1984, Chapter 5). Therefore,

$$\begin{aligned} \partial(\boldsymbol{\Gamma} \bigcirc \boldsymbol{\Gamma'}) \subset\ & \{x+iy \,|\, \hat{h}(x,y,t) = 0, \partial\hat{h}/\partial t = 0, u(x,y,t) \in \boldsymbol{s} \,|\, t \in \boldsymbol{t}\} \\ & \cup \{x+iy \,|\, \hat{h}(x,y,t) = 0, u(x,y,t) \in \boldsymbol{s} \,|\, t \in \{\underline{t}, \overline{t}\}\} \\ & \cup \{x+iy \,|\, \hat{h}(x,y,t) = 0, u(x,y,t) \in \{\underline{s}, \overline{s}\} \,|\, t \in \boldsymbol{t}\}. \end{aligned} \tag{7}$$

Since the constraint components are edges and arcs translated or multiplied by scalars, the only possibly non-polyarcular segment of the boundary is the envelope component. Therefore the question of this analysis is whether $\{\partial\hat{h}/\partial t = 0\}$ is a polyarcular curve, and whether the $\{u(x,y,t) \in \boldsymbol{s}, t \in \boldsymbol{t}\}$

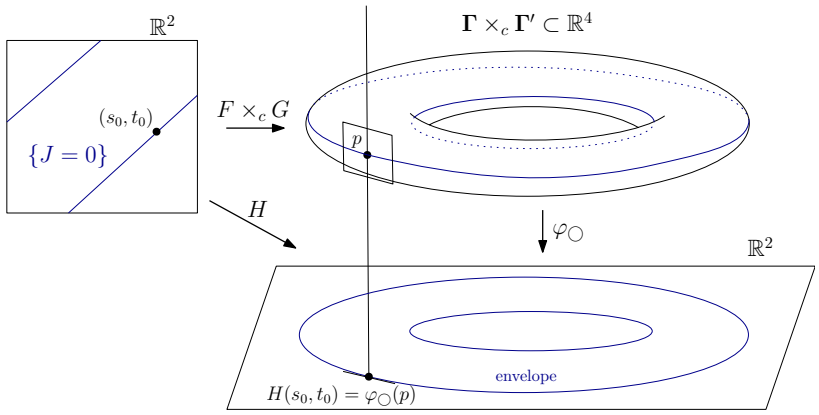

Fig. 6 Schematic summary of the parametrization of the envelope

condition is fulfilled. A demonstrative example can be found in Figure 5, and more are availabile in Online Resource 1.

3.3.4 Parametric envelope evaluation

Another method of finding the envelope is by using the parametrizations of both operands. We are aiming for a parametrization of the result of the operation. First, let us look at the two parametrizations as map from the two-dimensional parameter space to the Cartesian product $\mathbb{C} \times_c \mathbb{C} = \mathbb{C}^2 \cong \mathbb{R}^4$.

Then apply the map $\varphi_{\bigcirc} : \mathbb{C}^2 \to \mathbb{C} \cong \mathbb{R}^2$ corresponding to the complex addition or multiplication operation:

$$\begin{aligned} \varphi_{\oplus}(x, y, u, v) &= (x + u, y + v), \\ \varphi_{\otimes}(x, y, u, v) &= (xu - yv, xv + yu), \end{aligned} \tag{8}$$

where $x + iy$ and $u + iv$ are the two complex coordinates of $\mathbb{C}^2$.

Together, they form a composition

$$H : \mathbb{R}^2 \xrightarrow{F \times_c G} \mathbb{R}^4 \xrightarrow{\varphi_{\bigcirc}} \mathbb{R}^2,$$

which is the parametrization of the result of the operation.

The points of the envelope are all critical points of the real map $\varphi_{\bigcirc}$ restricted to the intermediate image $(F \times_c G)(\mathbb{R}^2) \subset \mathbb{R}^4$, or in other words critical points of the parametrization $H(s, t)$. This means that the differential $dH = d(\varphi_{\bigcirc} \circ (F \times_c G))$ of the composition is not of full rank. This situation is depicted in Fig. 6.

Hence, a point $p = F(s_0) \bigcirc G(t_0)$ can only be an envelope point if the Jacobian determinant $J(s, t) = \det(\mathrm{Jac}_H(s_0, t_0))$ of the parametrization $H : \mathbb{R}^2 \to \mathbb{R}^2$ vanish at (s_0, t_0). This gives another way to describe the superset of

the boundary with three parts – as in (7):

$$\partial(\mathbf{\Gamma} \bigcirc \mathbf{\Gamma'}) \subset \big\{F(s) \bigcirc G(t) \big| J(s,t) = 0 \text{ or } s \in \{\underline{s}, \overline{s}\} \text{ or } t \in \{\underline{t}, \overline{t}\}\big\}. \quad (9)$$

In computations we will use the following description. In the additive case

$$H_{\oplus}(s,t) = \varphi_{\oplus}(F(s), G(t)) = \big(\Re(F)(s) + \Re(G)(t), \Im(F)(s) + \Im(G)(t)\big),$$

$$J_{\oplus}(s,t) = \big|\mathrm{Jac}_{H_{\oplus}}(s,t)\big| = \begin{vmatrix} \Re(F)'(s) & \Re(G)'(t) \\ \Im(F)'(s) & \Im(G)'(t) \end{vmatrix}.$$

In the multiplicative case

$$H_{\otimes}(s,t) = \varphi_{\otimes}(F,G) = \big(\Re(F)\Re(G) - \Im(F)\Im(G), \Re(F)\Im(G) + \Im(F)\Re(G)\big),$$

$$J_{\otimes}(s,t) = \big|\mathrm{Jac}_{H_{\otimes}}\big| = \begin{vmatrix} \Re(F)'\Re(G) - \Im(F)'\Im(G) & \Re(F)\Re(G)' - \Im(F)\Im(G)' \\ \Re(F)'\Im(G) + \Im(F)'\Re(G) & \Re(F)\Im(G)' + \Im(F)\Re(G)' \end{vmatrix}.$$

In fact, this method of finding the parametrization of the envelope was already mentioned – in the latter form – and used in Farouki et al (2005).

3.3.5 Implicit envelope evaluation

A third way of describing the envelope is using the implicit equations of both operands. We aim for an implicit equation of the envelope.

Let the two operands – without boundary conditions, so far – be $\mathbf{\Gamma} = \{f(x,y) = 0\}, \mathbf{\Gamma'} = \{g(x,y) = 0\} \subset \mathbb{C}$. Their Cartesian product is $\mathbf{\Gamma} \times_c \mathbf{\Gamma'} = \{f(x,y) = 0, g(u,v) = 0\} \subset \mathbb{C}^2$. Consider the map $\varphi_{\bigcirc} : \mathbb{R}^4 \to \mathbb{R}^2$ corresponding to Minkowski addition and multiplication, respectively, as defined in (8).

Similarly to the previous method, we find the envelope as the set $\widetilde{\boldsymbol{E}}$ of critical points of the map $\varphi_{\bigcirc}$ restricted to $\mathbf{\Gamma} \times_c \mathbf{\Gamma'}$. These are the points where the gradients of $f(x,y)$, $g(u,v)$, and that of the two components of $\varphi_{\bigcirc}$ are not linearly independent. Thus, at these points the Jacobian determinant vanish:

$$\widetilde{h}(x,y,u,v) = \det \mathrm{Jac}\big(f, g, \Re(\varphi_{\bigcirc}), \Im(\varphi_{\bigcirc})\big) = 0.$$

The set $\widetilde{\boldsymbol{E}} \subset \mathbb{R}^4$ of critical points is described by the three equations $f = g = \widetilde{h} = 0$ and is a real algebraic variety of dimension at most one (Milnor, 1963).

The actual envelope is the image $E = \varphi_{\bigcirc}(\widetilde{E})$ in $\mathbb{R}^2 \cong \mathbb{C}$. In general, the image of a variety on an algebraic map may not be a variety; however, in case of a proper map, it is (Shafarevich, 2013, 5.2). Our non-trivial cases involve only proper maps. Again, we summarize the spaces involved in Fig. 7.

When we look for the defining equations of the image $E \subset \mathbb{R}^2$, we want to find (the ideal of) those functions $\psi : \mathbb{R}^2 \to \mathbb{R}$ that, when composed with $\varphi_{\bigcirc} : \mathbb{R}^2 \to \mathbb{R}^2$, the composition $\psi \circ \varphi_{\bigcirc}$ vanishes on $\widetilde{E}$. (When the map $\varphi_{\bigcirc}$ is not proper, this ideal defines a variety that contains E as a Zariski open

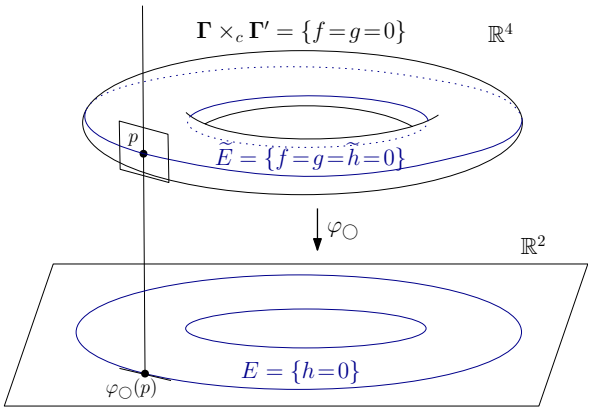

Fig. 7 Schematic summary of the implicit envelope evaluation

subset.) To compute the generators of the ideal defining E, we need Gröbner bases (Adams and Loustaunau, 1994, Chapter 1). More precisely, we find the reduced Gröbner basis of the ideal

$$I = (f, g, \widetilde{h}, \Re(\varphi_{\bigcirc}) - X, \Im(\varphi_{\bigcirc}) - Y)$$

(where X, Y are the coordinate functions of the target $\mathbb{R}^2$) with respect to a suitable elimination order. Lastly, we take those basis elements that only involve X and Y (Adams and Loustaunau, 1994, 2.4). In most of our cases, $\widetilde{E}$ and E are one-dimensional; therefore, the algorithm results in a single polynomial function that will call $h(X, Y)$. We used SageMath for the computations, the code can be found in Section S2 of Online Resource 1.

For each additive and multiplicative combination, the step-by-step computations of the implicit equation $h(x, y) = 0$ of the envelope and that of the condition $J(s, t) = 0$ describing for what parameter values the envelope is included in the boundary can be found in Online Resource 1, together with illustrations.

3.4 Properties of polyarcular intervals

3.4.1 Unary operations

Let us analyze the result of the two unary operations, negative and reciprocal, using the Cartesian and polar implicit functions. Where necessary, we map between the two coordinate systems. The corresponding computations can be found in Section S1 of Online Resource 1.

Lemma 3.5 *The negative of an edge is also an edge. The negative of an arc is also an arc.*

$+$	$\bar{\mathcal{O}}$	$\breve{\mathcal{O}}$
$\bar{\mathcal{O}}$	$\emptyset$	$\bar{\mathcal{O}}$
$\breve{\mathcal{O}}$		$\breve{\mathcal{O}}$

$\times$	$\bar{\mathcal{O}}$	$\breve{\mathcal{O}}$
$\bar{\mathcal{O}}$	$\mathcal{O}^2(\emptyset, \bar{\mathcal{O}})$	$\mathcal{O}^2(\bar{\mathcal{O}}, \breve{\mathcal{O}}, \mathrm{pt})$
$\breve{\mathcal{O}}$		$\mathcal{O}^4(\breve{\mathcal{O}})$

Table 4 Envelope subspace of polyarc segment binary combinations. pt indicates that the envelope is a point, $\bar{\mathcal{O}}$ indicates linear curves (edges), $\breve{\mathcal{O}}$ circular curves (arcs), $\mathcal{O}^2$ quadratic, and $\mathcal{O}^4$ quartic polynomial curves; $\emptyset$ indicates that there is no envelope. (Subspace symbols in brackets indicate special cases, when one or both operands are on a zero crossing line or a zero-centered circle.)

Proof Taking the negative is simply taking the reflection across the origin which takes lines into lines, and arcs into arcs. □

Lemma 3.6 *The reciprocal of an edge is an edge if the containing line is zero-crossing; otherwise it is an arc. The reciprocal of an arc is also an arc.*

Proof The reciprocal is known to be equivalent to the combination of the geometric inversion with respect to the complex 0 and the reflection through the real axis. The former takes circles and lines into circles – except for the lines through the center that are fixed – and the latter is an isometry. □

Proposition 3.7 *The negative of a polyarcular interval is a polyarcular interval.*

$$\boldsymbol{A} \in \mathcal{A}(\mathbb{C}) \implies -\boldsymbol{A} \in \mathcal{A}(\mathbb{C})$$

Proof Negation is a closed unary operation on complex intervals, and a complex interval is polyarcular if it is bounded by a polyarc curve. According to Proposition 3.2 the negative of a complex interval is bounded by the negative of the interval boundary, and in Lemma 3.5 we show that the negative of the polyarc curve segments are also polyarc curve segments. □

Proposition 3.8 *The reciprocal of a polyarcular interval is a polyarcular interval if the operand does not contain the complex zero.*

$$\boldsymbol{A} \in \mathcal{A}(\mathbb{C}), 0 \notin \boldsymbol{A} \implies \boldsymbol{A}^{-1} \in \mathcal{A}(\mathbb{C})$$

Proof Reciprocal is a partial unary operation on complex intervals, and a complex interval is polyarcular if it is bounded by a polyarc curve. According to Proposition 3.2 the reciprocal of a complex interval is bounded by the reciprocal of the interval boundary, and in Lemma 3.6 we showed that the reciprocal of the polyarc curve segments are also polyarc curve segments. □

3.4.2 Binary operations

Lemma 3.9 *The sum of two edges is a complex interval bounded by edges.*

Proof The sum of two lines has no envelope unless the two lines are identical, in which case the envelope is the line itself. Consequently, the sum of two non-parallel edges is bounded only by the operand edges translated by the endpoints of the other operand, while the sum of two parallel edges is a single edge that lies in the line containing the operands. See the derivation in Section S2.1.1 of Online Resource 1. □

Lemma 3.10 *The sum of an arc and an edge is a complex interval bounded by edges and arcs.*

Proof The sum of a circle and a line has an envelope consisting of two lines parallel to the edge. The boundary of the result contains sections of the envelope when the argument of the arc is normal to the edge. Consequently, the result boundary consists of the operand edge translated by the arc endpoints, the operand arc translated by the edge endpoints, and edges formed by sections of the envelope. See the derivation in the Section S2.1.2 Online Resource 1. □

Lemma 3.11 *The sum of two arcs is a complex interval bounded by arcs.*

Proof The sum of two circles has an envelope consisting of two circles with radii equal to the sum and difference of the radii of the operand circles. The result boundary contains sections of the envelope if the argument intervals of the two arcs overlap. Consequently, the result boundary consists of the operand arcs translated by the endpoints of the other operand and the arcs formed by sections of the envelope. See the derivation in the Section S2.1.3 of Online Resource 1. □

Lemma 3.12 *The product of two edges is a complex interval bounded by edges if at least one operand is from a zero-crossing line or if the intersection of their normalized parameter intervals is empty. Otherwise, the boundary of the result has segments of a parabola, a quadratic curve, that are generically neither edges nor arcs.*

Proof The product of two lines has an envelope consisting of a parabolic curve when none of them crosses the origin. The result boundary contains a segment of the envelope when their normalized parameter intervals overlap. If only one line crosses the origin, then there is no envelope, while if both lines cross the origin, then the product is a single line, which is also the envelope. Consequently, the result boundary consist of the operand edges translated by the endpoints of the other operand, and sections of the envelope. See the derivation in the Section S2.2.1 of Online Resource 1. □

Lemma 3.13 *The product of an edge and an arc is a complex interval bounded by edges and arcs if the edge is from a zero crossing line, or the arc is from a zero-centered circle, or the circle radius is exactly 1, or the following condition is not fulfilled for any of the parameter value pairs of their normalized parameter intervals:*

$$r + \cos(s) + \sin(s)/t = 0$$

Otherwise, the boundary of the result has segments of a hyperbola or ellipse, a quadratic curve, that are generically neither edges nor arcs.

Proof The envelope of a non-zero-centered circle and a non-zero-crossing line is a hyperbola if the circle radius is less than 1, and an ellipse if its radius is more than 1. The result boundary contains a segment of the envelope when the above condition holds for the arguments. If the circle radius equals 1, the envelope is the real line, except the real 0 and 2 points, and the result is the union of two concave, arc bounded regions touching at the real 0 and 2 points, which technically can be considered a complex interval. If the circle is zero-centered, but the line is non-zero-crossing, the envelope is a circle. If the line is zero-crossing but the circle is non-zero centered, the envelope is two lines. If both operands are zero-centered, the envelope is the point of origin. Consequently, the result boundary consist of the operand arc and edge scale-rotated by the endpoints of the other operand, and sections of the envelope. See the derivation in the Section S2.2.2 of Online Resource 1. □

Lemma 3.14 *The product of two arcs is a complex interval bounded by arcs if at least one operand is from a zero-centered circle or if the following condition is not fulfilled for any of the parameter value pairs from their normalized parameter intervals:*

$$\sin(s - t) = r_1 \sin(t) - r_2 \sin(s),$$

where r_1, r_2 denote the radii of the normalized circles.
Otherwise, the boundary of the result has segments of a Cartesian oval, a quartic curve, that are generically neither edges nor arcs.

Proof The product of two non-zero centered circles has a quartic envelope, a Cartesian oval. The result boundary contains sections of it if the above condition holds for the arguments. If only one arc is zero-centered, then the envelope consists of two rescaled copies of the zero-centered circle, and the result boundary includes envelope segments if the normalized argument of the nonzero centered circle avoids the values 0 and π. If both arcs are zero-centered, their product is a single arc that also belongs to the boundary (similarly to the case of zero-crossing lines). Consequently, the result boundary consist of the operand arcs scale-rotated by the endpoints of the other operand, and sections of the envelope. See the derivation in the Section S2.2.3 of Online Resource 1. □

Proposition 3.15 *The sum of two polyarcular intervals is a polyarcular interval.*

$$\boldsymbol{A}, \boldsymbol{B} \in \mathcal{A}(\mathbb{C}) \implies (\boldsymbol{A} \oplus \boldsymbol{B}) \in \mathcal{A}(\mathbb{C})$$

Proof Addition is a closed binary operation on complex intervals, and a complex interval is polyarcular if it is bounded by a polyarc curve. According to Proposition 3.2, the sum of complex intervals is bounded by the sum of the interval boundaries. In equation (3.2) we show that the sum of polyarc curves is bounded by a subset of the sums of the curve segments. In lemmas 3.9, 3.10 and 3.11 we show that the sum of polyarcular curve segments is always bounded by edges and arcs. □

Proposition 3.16 *The product of two polyarcular intervals is a polyarcular interval if in each combination of curve segments that contributes to the result boundary at least one of the operands is from a zero-crossing line or zero-centered circle, or none of the results of the combinations of curve segments touch the envelope (see conditions in lemmas 3.12, 3.13 and 3.14). Otherwise, it is a complex interval.*

$$\boldsymbol{A}, \boldsymbol{B} \in \mathcal{A}(\mathbb{C}) \implies (\boldsymbol{A} \otimes \boldsymbol{B}) \in \begin{cases} \mathcal{A}(\mathbb{C}) & \textit{conditions above} \\ \mathcal{I}(\mathbb{C}) & \textit{otherwise} \end{cases}$$

Proof Multiplication is a closed binary operation on complex intervals, and a complex interval is polyarcular if it is bounded by a polyarc curve. According to Proposition 3.2 the product of complex intervals is bounded by the product of the interval boundaries. In equation (3.2) we show that the product of polyarc curves is bounded by a subset of the products of the curve segments. In lemmas 3.12, 3.13 and 3.14 we show that in case one of the curve segments in the combination is zero-crossing or zero-centered, then the product is always bounded by edges and arcs. In the same lemma we also show that the product envelope of non-zero-crossing/centered curve segments is neither linear nor circular; therefore, the product is bounded by edges and arcs only if the envelope is not part of the boundary. □

3.4.3 Set operations

Proposition 3.17 *The union and intersection of polyarcular intervals are polyarcular intervals if the intersection of the operands is not empty.*

$$\boldsymbol{A}, \boldsymbol{B} \in \mathcal{A}(\mathbb{C}), \boldsymbol{A} \cap \boldsymbol{B} \neq \emptyset \implies \begin{cases} (\boldsymbol{A} \cup \boldsymbol{B}) \in \mathcal{A}(\mathbb{C}) \\ (\boldsymbol{A} \cap \boldsymbol{B}) \in \mathcal{A}(\mathbb{C}) \end{cases}$$

Proof Proposition 3.2 shows that the boundary of the union and the intersection of complex intervals consist of the boundary of the operands. The boundaries of polyarcular intervals consist of edges and arcs by definition, and any continous closed curve constructed from intersection polyarc curves will also consist of edges and arcs. □

4 Computational properties

In this section, we consider the data representation and computation of complex intervals. We define a data type for each complex interval representation and determine their data storage requirements. To measure the goodness of

their representation capability, we introduce the tightness measure. We then define the type-casting operation between interval types and show how it can cause a loss of tightness through arithmetic and set operations (Table 5). Finally, we show two utility processes, one for extracting the simple boundary when an operation results a self-intersecting boundary, and another for backtracking the subsets of the operands of an operation that map to a point in the result interval.

4.1 Interval types

Interval types are finite data representations of intervals, meaning that they can be identified with particular elements of the corresponding subspace using finite sets of parameter values. A type is a more restrictive class than a subspace as it determines how the complex interval is stored and manipulated in the computational environment, and therefore an interval instance can have only one type assigned to it. It is possible to represent an interval using any of the types, but if the interval is not a member of the corresponding subspace, then we assume the smallest bounding interval of the type, in which case the representation will not be tight (see the definition of tightness in Section 4.2). As a shorthand format, we indicate the type of a variable in its upper index. First, we define some utility data types.

Definition 4.1 The real type represents real numbers with the double precision floating-point data type variable (float).

Definition 4.2 The complex type represents complex numbers as two real-type variables (2 floats).

Definition 4.3 The real interval type represents a bounded real set $\boldsymbol{a} \subset \mathbb{R}$ by storing its bounds as real type variables (2 floats in total).

$$\boldsymbol{a}^{\mathcal{I}} = \mathcal{I}(\boldsymbol{a}) := \mathcal{I}(\mathbb{R}\big|\underline{a}, \overline{a}) = [\underline{a}, \overline{a}] \quad \underline{a} = \inf(\boldsymbol{a}), \overline{a} = \sup(\boldsymbol{a})$$

Definition 4.4 The edge type represents edges by storing their endpoints as two complex type variables (4 floats in total).

$$\boldsymbol{\Gamma} \in \bar{\mathcal{O}}(\mathbb{C}) \implies \boldsymbol{\Gamma}^{\bar{\mathcal{O}}} := \{\bar{\Gamma}(P_1, P_2)(t)\big|t \in [0,1]\}$$

Definition 4.5 The arc type represents arcs by storing their center point as a complex type, radius as a real type and argument as a real interval type variable (5 floats in total).

$$\boldsymbol{\Gamma} \in \breve{\mathcal{O}}(\mathbb{C}) \implies \boldsymbol{\Gamma}^{\breve{\mathcal{O}}} := \{\breve{\Gamma}(O, r, \boldsymbol{\varphi})(t)\big|t \in [0,1]\}$$

A complex interval $\boldsymbol{A} \in \mathcal{I}(\mathbb{C})$ can be represented by various complex interval types in the following way.

Definition 4.6 The rectangular interval type represents complex intervals by storing their bounds along the real and imaginary axes as real intervals (4 floats in total).

$$\boldsymbol{A}^{\mathcal{R}} = \mathcal{R}(\boldsymbol{A}) := \mathcal{R}(\mathbb{C}\big|\boldsymbol{a}, \boldsymbol{b}) = \boldsymbol{a} + i\boldsymbol{b}, \quad \boldsymbol{a} = \Re(\boldsymbol{A}), \boldsymbol{b} = \Im(\boldsymbol{A})$$

Definition 4.7 The polar interval type represents complex intervals by storing their bounds along the radial and angular axes as real intervals (4 floats in total).

$$\boldsymbol{A}^{\mathcal{P}} = \mathcal{P}(\boldsymbol{A}) := \mathcal{P}(\mathbb{C}|\boldsymbol{r}, \boldsymbol{\varphi}) = \boldsymbol{r}e^{i\boldsymbol{\varphi}}, \quad \boldsymbol{r} = |\boldsymbol{A}|, \, \boldsymbol{\varphi} = \angle \boldsymbol{A}$$

Definition 4.8 The circular interval type represents complex intervals by storing the center point and radius of the smallest bounding circle as a complex type and a real type variable respectively (3 floats in total).

$$\boldsymbol{A}^{\mathcal{C}} = \mathcal{C}(\boldsymbol{A}) := \mathcal{C}(\mathbb{C}\big|O, r) = O + [0, r]e^{[-\pi,\pi]}$$

Definition 4.9 The polygonal interval type represents complex intervals by storing the ordered set of vertices of the smallest bounding polygon of a given vertex count N as complex-type variables ($2\mathsf{N}$ floats in total).

$$\boldsymbol{A}^{\mathcal{G}} = \mathcal{G}(\boldsymbol{A}) := \mathcal{G}(\mathbb{C}\big|\{P_{\mathsf{n}}\big|\mathsf{n} \in \{1..\mathsf{N}\}\}, \quad \partial\boldsymbol{A}^{\mathcal{G}} = \{\bar{\boldsymbol{\Gamma}}_{\mathsf{n}}\big|\mathsf{n} \in \{1..\mathsf{N}\}\},$$

where P_{n} is the n^{th} vertex, and

$$\bar{\boldsymbol{\Gamma}}_{\mathsf{n}} = \{\bar{\Gamma}(P_{\mathsf{n}}, P_{\mathsf{n}+1})(t)\big|t \in [0, 1]\}$$

is the n^{th} implicit edge between the corresponding vertices (see also Definition 2.10 and Figure 2).

Definition 4.10 The polyarcular interval type represents complex intervals by storing the ordered set of arcs of the smallest bounding polyarc of a given arc count N as arc-type variables (total $5\mathsf{N}$ floats).

$$\boldsymbol{A}^{\mathcal{A}} = \mathcal{A}(\boldsymbol{A}) := \mathcal{A}(\mathbb{C}\big|\{\breve{\boldsymbol{\Gamma}}_{\mathsf{n}} \in \breve{\mathcal{O}}(\mathbb{C})\big|\mathsf{n} \in \{1..\mathsf{N}\}\}, \quad \partial\boldsymbol{A}^{\mathcal{A}} = \{\breve{\boldsymbol{\Gamma}}_{\mathsf{n}} \cup \bar{\boldsymbol{\Gamma}}_{\mathsf{n}}\big|\mathsf{n} \in \{1..\mathsf{N}\}\},$$

where

$$\breve{\boldsymbol{\Gamma}}_{\mathsf{n}} = \{\breve{\Gamma}(O_{\mathsf{n}}, r_{\mathsf{n}}, \boldsymbol{\varphi}_{\mathsf{n}})(t)\big|t \in [0, 1]\}$$

is the n^{th} arc,

$$\bar{\boldsymbol{\Gamma}}_{\mathsf{n}} = \{\bar{\Gamma}(P_{2\mathsf{n}-1}, P_{2\mathsf{n}})(t)\big|t \in [0, 1]\}$$

is the n^{th} implicit edge between the corresponding arcs, $P_{2\mathsf{n}-1}$ and $P_{2\mathsf{n}}$ are implicit vertices (see also Definition 2.13 and Figure 2).

Remark 4.1 Polyarcular curves consist of arcs defined by its data set and the implicit edges connecting the endpoints of adjacent arcs. It is possible to suppress circular

segments by setting $r_{\mathsf{n}} = 0$, while the linear segment can be suppressed by making sure that $P_{2\mathsf{n}-1} = P_{2\mathsf{n}}$. Concave arcs can be created using negative radius values.

Figure 2 gives a demonstrative example of a complex interval represented by the mentioned complex interval types.

4.2 Tightness

The size of a real interval $\boldsymbol{a} \in \mathcal{I}(\mathbb{R})$ is its length:

$$\mu(\boldsymbol{a}) = \overline{a} - \underline{a}.$$

Let us measure complex intervals with the standard area (or Lebesgue measure). According to Green's theorem the following holds (Stewart, 1999, Chapter 16) (Lang, 2012, X,1).

Proposition 4.1 *The size of a complex interval $\boldsymbol{A} \in \mathcal{I}(\mathbb{C})$ bounded by a simple, closed, piecewise smooth curve equals the following closed line integral*

$$\mu(\boldsymbol{A}) = \frac{1}{2} \oint_{\partial \boldsymbol{A}} x\,dy - y\,dx = \frac{1}{2i} \oint_{\partial \boldsymbol{A}} z^*\,dz \tag{10}$$

where $z = x + iy$ and $z^* = x - iy$.

This leads to the equation of the polyarcular interval size, where polygonal and primitive intervals represent special cases.

$$\begin{aligned}\mu(\boldsymbol{A}^{\mathcal{A}}) &= \sum_n \frac{1}{2i} \oint_{\breve{\boldsymbol{\Gamma}}_n} z^*\,dz + \sum_n \frac{1}{2i} \oint_{\bar{\boldsymbol{\Gamma}}_n} z^*\,dz \\ &= \sum_n \frac{(\overline{\varphi}_{\mathsf{n}} - \underline{\varphi}_{\mathsf{n}})r_{\mathsf{n}}^2}{2} + i\frac{O_{\mathsf{n}}^* r}{2}(e^{i\underline{\varphi}_{\mathsf{n}}} - e^{i\overline{\varphi}_{\mathsf{n}}}) \\ &\quad + \sum_n \frac{1}{4i}(|P_{2n}|^2 - |P_{2\mathsf{n}-1}|^2 + 2i(P_{2\mathsf{n}-1}^{\Re} P_{2\mathsf{n}}^{\Im} - P_{2\mathsf{n}-1}^{\Im} P_{2\mathsf{n}}^{\Re}),\end{aligned} \tag{11}$$

where $\breve{\boldsymbol{\Gamma}}_{\mathsf{n}}, \bar{\boldsymbol{\Gamma}}_{\mathsf{n}} \in \partial \boldsymbol{A}$, $\mathsf{n} \in (1..\mathsf{N})$. (See also Definition 4.10.)

Then we can define the tightness of a representation in the following way.

Definition 4.11 The tightness of a complex interval representation $\boldsymbol{A} \in \mathcal{I}(\mathbb{C})$ is the ratio of its original size and its represented size.

$$\tau(\boldsymbol{A}^{\mathcal{X}}) = \frac{\mu(\boldsymbol{A})}{\mu(\boldsymbol{A}^{\mathcal{X}})} \in [0, 1],$$

where $\mathcal{X}$ is a placeholder for one of the complex interval types.

Example 1 The tightness of a circular interval represented by the rectangular type is

$$\boldsymbol{A} \in \mathcal{C}(\mathbb{C}) \implies \tau(\boldsymbol{A}^{\mathcal{R}}) = \frac{\mu(\boldsymbol{A})}{\mu(\boldsymbol{A}^{\mathcal{R}})} = \frac{r^2\pi}{(2r)^2} = \frac{\pi}{4},$$

where r is the radius of the circular interval.

If an interval belongs to the data type's corresponding subspace, then that representation will be tight ($\tau = 1$), otherwise it will be loose ($\tau < 1$). Figure 3 offers an intuitive demonstration as a Venn diagram where each set is shaped accordingly. The representation and operation tightness of our interval types are listed in Table. 5.

Remark 4.2 In certain applications – such as algorithms for the approximation of boundary curves with polygonal or polyarcular curves – the Hausdorff distance (Definition 2.4) may be preferred over the tightness metric, because it can measure the representation error of a boundary segment, while tightness can only be applied to an entire interval.

4.3 Type casting

Changing data type can be a useful and sometimes necessary step when handling complex intervals. Binary operations, for example, are typically only defined between operands of the same type. Although it is possible and can be practical in certain cases to define binary operations between different types (e.g. a fast algorithm for determining the smallest polar interval enclosing the product of a polar and a circular interval), we don't discuss these in this paper. Therefore, type casting is necessary when two complex intervals of different types are to be combined, when using another interval type in an operation is preferred, or when the result is not in the operand subspace.

Definition 4.12 Type casting is a unary operation that transforms a finite representation of an interval into another finite representation.

$$\mathcal{Y}(\boldsymbol{A}^{\mathcal{X}}) = \mathcal{Y}(\mathcal{X}(\boldsymbol{A})),$$

where $\mathcal{X}$ and $\mathcal{Y}$ are placeholders for complex interval types.

There are three kinds of type casting. If the subspace corresponding to the source data type is a subset of the target data type's subspace, it is a widening casting and no tightness will be lost in the process; if it is the other way we talk about narrowing casting, which can result in a loss of tightness if the interval is not in the narrower subspace. If the data types are on the same level in the hierarchy we can talk about lateral casting, which typically results in a loss of tightness as these subspaces have no or very small overlapping regions.

Type	Representation of $\boldsymbol{A} \in$					Unary $(\boldsymbol{A}^{\mathcal{X}})$		Binary $\boldsymbol{A}^{\mathcal{X}} \bigcirc \boldsymbol{B}^{\mathcal{X}}$			
	$\mathcal{R}$	$\mathcal{P}$	$\mathcal{C}$	$\mathcal{G}$	$\mathcal{A}$	$-(.)$	$(.)^{-1}$	$+$	$\times$	$\cap$	$\cup$
$\mathcal{R}(\boldsymbol{A})$	$=$	$<$	$<$	$<$	$<$	$=$	$<$	$=$	$<$	$=$	$<$
$\mathcal{P}(\boldsymbol{A})$	$<$	$=$	$<$	$<$	$<$	$=$	$<$	$<$	$=$	$=$	$<$
$\mathcal{C}(\boldsymbol{A})$	$<$	$<$	$=$	$<$	$<$	$=$	$=$	$=$	$<$	$<$	$<$
$\mathcal{G}(\boldsymbol{A})$	$=$	$\approx$	$\approx$	$=$	$\approx$	$=$	$\approx$	$=$	$\approx$	$=$	$=$
$\mathcal{A}(\boldsymbol{A})$	$=$	$=$	$=$	$=$	$=$	$=$	$=$	$=$	$\approx$	$=$	$=$

Table 5 Tightness of interval representation, and unary and binary operations of complex interval types ($\mathcal{R}$: rectangular, $\mathcal{P}$: polar, $\mathcal{C}$: circular, $\mathcal{G}$: polygonal, $\mathcal{A}$: polyarcular, $\mathcal{X}$ is a placeholder). Each row represents an interval type, and the properties are ordered in columns. The first vertical block indicates how well the type can represent an interval from a given subspace (with the same symbols as the type). The second vertical block shows how well it can represent the result of a unary operation on an interval from the same subspace as the type. The third vertical block indicates how well it can represent the result of a binary operation with two intervals of the same subspace as the type. The property value $=$ indicates perfect tightness ($\tau = 1$), $<$ indicates imperfect tightness ($\tau < 1$) and $\approx$ indicates arbitrarily high tightness ($\tau \approx 1$).

$$\tau(\mathcal{Y}(\boldsymbol{A}^{\mathcal{X}})) \begin{cases} = \tau(\boldsymbol{A}^{\mathcal{X}}) & \text{if } \mathcal{X}(\mathbb{C}) \subset \mathcal{Y}(\mathbb{C}) \\ = \tau(\boldsymbol{A}^{\mathcal{X}}) & \text{if } \mathcal{X}(\mathbb{C}) \not\subset \mathcal{Y}(\mathbb{C}) \text{ and } \boldsymbol{A}^{\mathcal{X}} \in (\mathcal{X}(\mathbb{C}) \cap \mathcal{Y}(\mathbb{C})) \\ < \tau(\boldsymbol{A}^{\mathcal{X}}) & \text{otherwise} \end{cases}$$

4.4 Arithmetic operations

Some of the most important properties of interval types are the computational complexity and accuracy, which are often connected to each other, forcing a compromise between computational speed and accuracy of arithmetic operations. For example, the polygonal type provides increasing accuracy for an increasing number of vertices when representing an interval that is not a member of the polygonal subspace.

Aligned with computational arithmetic conventions, let us force all complex type operations to result in a variable of the same type, and therefore implicit type-casting is not allowed. (For example, in C++ language, the division between the integer variables 2 and 5 results in $5/2 = 2$ in which case the result is truncated to become an integer.)

$$f(\boldsymbol{A}^{\mathcal{X}}) = \mathcal{X}(f(\boldsymbol{A}^{\mathcal{X}})), \quad f(\boldsymbol{A}^{\mathcal{X}}, \boldsymbol{B}^{\mathcal{X}}) = \mathcal{X}(f(\boldsymbol{A}^{\mathcal{X}}, \boldsymbol{B}^{\mathcal{X}}))$$

In Section 3 we show that not all combinations of subspace operations result in an interval belonging to the same subspace as the operand(s). When intervals are represented as types, this means that the result has to be re-represented after the operation, which unavoidably decreases the tightness of the representation.

Example 2 The inverse of a rectangular interval is not rectangular; therefore, if we perform the inverse operation on its rectangular-type representation, the result will

be relaxed to the smallest enclosing rectangle. An accurate result can be achieved by type-casting the operand to polyarcular type and performing the inverse on that.

$$\boldsymbol{A} \in \mathcal{R}(\mathbb{C}), \boldsymbol{A}^{-1} \notin \mathcal{R}(\mathbb{C}) \implies \left(\boldsymbol{A}^{\mathcal{R}}\right)^{-1} = \mathcal{R}\left(\boldsymbol{A}^{-1}\right) \supset \boldsymbol{A}^{-1} = \left(\boldsymbol{A}^{\mathcal{A}}\right)^{-1}$$

When a function combines several intervals from a subspace that is not closed under the operations, the re-representation error can result a cumulative loss of tightness. In this case the bounds of the result representation are not necessarily touching the bounds of the result.

Example 3 The sum of polar intervals is not polar. Therefore, if the sum is performed on two polar-type operands, the result will be relaxed to the smallest enclosing polar interval. If we add a third polar interval to this relaxed sum, then the result will contain the representation error of the first operation's result and the relaxation of the second operation. This results a cascading error.

$$\begin{gathered}\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C} \in \mathcal{P}(\mathbb{C}), \quad \mathcal{P}(\boldsymbol{A}) + \mathcal{P}(\boldsymbol{B}) + \mathcal{P}(\boldsymbol{C}) \notin \mathcal{P}(\mathbb{C}) \implies \\ \boldsymbol{A}^{\mathcal{P}} + \boldsymbol{B}^{\mathcal{P}} + \boldsymbol{C}^{\mathcal{P}} = \mathcal{P}(\mathcal{P}(\boldsymbol{A} + \boldsymbol{B}) + \boldsymbol{C}) \subset \mathcal{P}(\boldsymbol{A} + \boldsymbol{B} + \boldsymbol{C}) \subset (\boldsymbol{A} + \boldsymbol{B} + \boldsymbol{C})\end{gathered}$$

Unlike Example 3, if the subspace is closed under the operation, then the combination of all the representation errors equals the representation error of the result.

Example 4 The sum of rectangular invervals is rectangular. Therefore, if we represent three nonrectangular intervals with the rectangular type, then their sum will contain only the representation error and will be the same as the rectangular representation of the interval sum. The error does not cascade.

$$\begin{gathered}\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C} \notin \mathcal{R}(\mathbb{C}), \quad \mathcal{R}(\boldsymbol{A}) + \mathcal{R}(\boldsymbol{B}) + \mathcal{R}(\boldsymbol{C}) \in \mathcal{R}(\mathbb{C}) \implies \\ \boldsymbol{A}^{\mathcal{R}} + \boldsymbol{B}^{\mathcal{R}} + \boldsymbol{C}^{\mathcal{R}} = (\mathcal{R}(\boldsymbol{A}) + \mathcal{R}(\boldsymbol{B})) + \mathcal{R}(\boldsymbol{C}) = \mathcal{R}(\boldsymbol{A} + \boldsymbol{B} + \boldsymbol{C}) \subset \boldsymbol{A} + \boldsymbol{B} + \boldsymbol{C}\end{gathered}$$

Algorithms for all basic arithmetic operations with primitive interval types are available in the literature (Petkovic and Petkovic, 1998; Moore et al, 2009; Dawood, 2011). Typically operations that yield a result in the operand's subspace are simpler and faster (e.g. rectangular addition or polar multiplication), while operations that have to re-represent the result are more complex (such as rectangular multiplication, or polar addition). For the latter group of operations, fast and loose algorithms are typically also available, which do not provide the tightest representation around the result but offer a faster calculation (e.g. circular multiplication or rectangular inverse).

The so-called Minkowski method allows performing operations on polygonal intervals using their defining vertices only. This allows the edges of the polygon to stay implicit throughout the entire process, with the exception to the reciprocal operation, which requires the approximation of convex arcs (Ohta et al, 1990). If combined with the Gauss map matching method (Farouki

et al, 2000), which is a simple intersection operation between the curve normal argument intervals of each vertex pair, we can get an algorithm with linear complexity by the number of vertices. For the addition and negative operations, this results tight bounds, while for the multiplication it replaces the concave parabolic curves with straight edges.

Polyarcular intervals are defined by a set of arcs, with the vertices and edges stored implicitly. Similarly to the reciprocal of polygons, the implicit components have to be calculated for certain operations. The negative operation, for example, requires the negation of the defining arcs only. The sum operation has to consider the Gauss map matched sum of the operand vertices and arcs, where vertices simply translate the other segments, but for two arcs the envelope segment has to be calculated. Reciprocal and multiplication operations require the evaluation of all three types of curve segments. Since the product envelope of arcs and edges are not polyarcular in general, their approximation may be necessary. However, since in many cases vertices cover most of the Gauss map, the approximated curve segments typically constitute only a small part of the total curve. The computational complexity therefore depends significantly on the operands and varies case-by-case.

4.5 Set operations

Algorithms for all basic set operations that involve primitive interval types are available in the literature (Boche, 1965; Gargantini and Henrici, 1971; Candau et al, 2006; Moore et al, 2009).

In case of polygonal and polyarcular interval types, set operations require the identification of intersections between curve segments, where they can be split, and then recombined according to the operation's logic. The intersection can be easily found at the points where the parametric equations of the segments are equal, while splitting a segment can be easily done by duplicating it and then splitting its parameter interval at the intersection point. Once ordered sets of split segments are available, inner and outer segments can be identified by finding a single extremal point in the whole set and then counting the number of intersections along the boundaries. The boundary of the union then consists of all the outer segments, and the boundary of the intersection consists of all the inner segments.

4.6 Trimming

A special case of the set operations is the trimming, which is a closed unary set operation from closed curves to simple closed curves. The addition and multiplication of the polygonal and polyarcular curve segments can result in self-intersecting boundaries when the operands are non-convex (in the multiplicative case: non-log-convex). Such a boundary is not acceptable as complex intervals have to be bounded by simple curves; therefore we have to extract the outer curve. This can be done using the trimming method described by Farouki

et al (2005), or if holes in the interval is not a concern, a simple always-turn-right rule can be used at each intersection of the counter-clock-wise oriented curve. For the sake of brevity, we will assume this step to be implicit and will not indicate in arithmetic equations.

$$\boldsymbol{\Gamma} = \{\Gamma(t), t \in \boldsymbol{t}\} \notin \mathcal{O}(\mathbb{C}), \mathcal{O}(\boldsymbol{\Gamma}) \in \mathcal{O}(\mathbb{C}) \implies \mathcal{I}(\mathbb{C}\big|\boldsymbol{\Gamma}) := \mathcal{I}(\mathbb{C}\big|\mathcal{O}(\boldsymbol{\Gamma})),$$

where $\mathcal{O}(\boldsymbol{\Gamma})$ is the trimming operation.

An example of a non-simple result boundary curve can be seen on Fig. 4.

4.7 Backtracking

As shown in Definition 3.2 it is possible to elementwise backtrack the $\mathbb{C} \times_c \mathbb{C} \to \mathbb{C}$ mapping of the Minkowski addition and multiplication operations on complex intervals. This requires the negation, translation and intersection operations for the backtracking of an addition; and it requires the reciprocal, rotate-and-scale and intersection operations for the backtracking of a multiplication. We defined all of these operations for polyarcular intervals above; therefore, the algorithms are ready to be implemented.

Backtracking can be used to verify tightness, because the points in the relaxation region of the representation have no corresponding subsets in the operand intervals. It can also be used to investigate the cause of certain outcomes, for example, to identify worst-case error patterns in tolerance analysis.

Example 5 It has been shown that the smallest rectangle enclosing the sum of two non-rectangular intervals will not be tight. If we backtrack a point in the relaxation region (the difference of the interval and its representation), the corresponding operand subsets will be empty (see Definition 3.2).

$$\boldsymbol{A}, \boldsymbol{B} \notin \mathcal{R}(\mathbb{C}) \implies \exists Z \in (\boldsymbol{A}^{\mathcal{R}} \oplus \boldsymbol{B}^{\mathcal{R}}) : \boldsymbol{A}_Z = \boldsymbol{A} \cap (-\boldsymbol{B} + Z) = \emptyset.$$

Backtracking can also be of use in the investigation of the interval dependency problem (Dawood and Dawood, 2019). When an operand interval appears more than once in a more complicated expression (such as $\boldsymbol{A} \oslash (\boldsymbol{A} \oplus \boldsymbol{B})$), the naive result evaluated assuming independent operands will be too relaxed. The backtracking of each point in this relaxation region to the instances of the repeated operand reveals that they are not valid.

Example 6 For the sake of simplicity let us consider the real valued function $\boldsymbol{a}/(\boldsymbol{a}+\boldsymbol{b})$ with the values $\boldsymbol{a} = [1, 3]$ and $\boldsymbol{b} = [2, 4]$. The correct solution is $[\underline{a}/(\underline{a}+\overline{b}), \overline{a}/(\overline{a}+\underline{b})] = [1/5, 3/5]$, however the interval arithmetic result is the relaxed interval $[\underline{a}/(\overline{a}+\overline{b}), \overline{a}/(\underline{a}+\underline{b})] = [1/7, 1]$. If we now backtrack the subsets of the $\boldsymbol{a}$ instances corresponding to the $z = 4/5$ point in the relaxation region, we get $\boldsymbol{a}_{1z} = \boldsymbol{a} \cap [(\boldsymbol{a}+\boldsymbol{b})z] = [2.40, 3.00]$ and $\boldsymbol{a}_{2z} = \boldsymbol{a} \cap [\boldsymbol{a}/z - \boldsymbol{b}] = [1, 1.75]$,

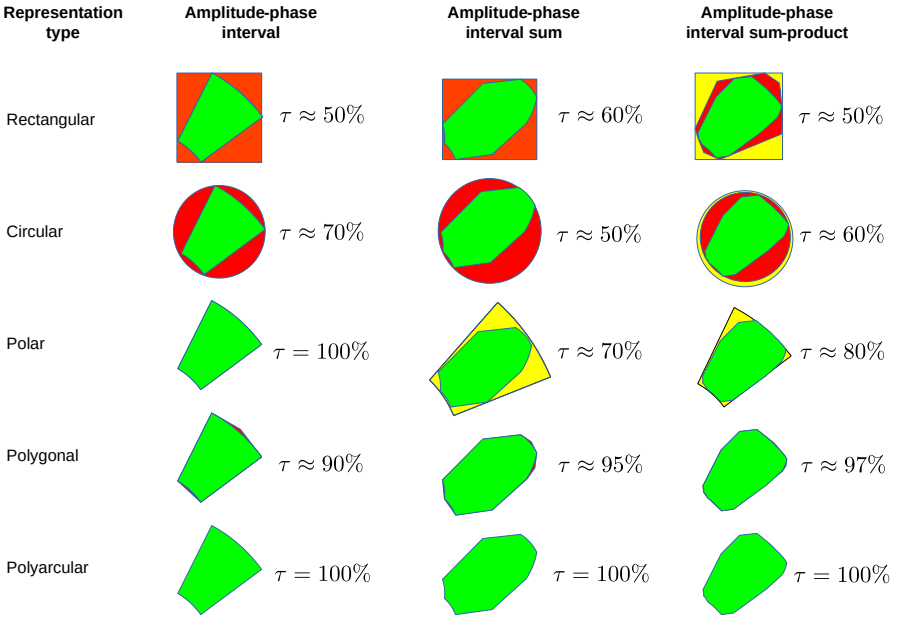

Fig. 8 Example of a sensor array tolerance analysis using various complex interval types. Light inner region is the complex interval, dark and light outer areas are the relaxation regions of the representation and operation errors respectively. (For illustration only.)

which has an empty intersection, so there is no such solution where $a_1, a_2 \in \boldsymbol{a}, b \in \boldsymbol{b},\ a_1 = a_2,\ a_1/(a_2 + b) = z$.

The location and size of the corresponding operand subsets can also provide information about the probability of an outcome if we assume a probability density function over the operand intervals (e.g. uniform).

5 Case study

In this section, we present a case study that motivated the development of the polyarcular interval type. The tolerance analysis of antenna arrays using interval analysis is an active research area within sensor array design and signal processing (He et al, 2019, 2021). We have previously analyzed the worst-case spatial response of acoustic arrays impacted by calibration errors and mutual coupling (Arnestad et al, 2023). Later, we found that given the physical model in our study, the polyarcular interval type yields a tight bound around the complex interval of the array response, which we demonstrate in the following.

Let $\{\boldsymbol{A}_{\mathsf{n}} \in \mathcal{P}(\mathbb{C}) \big| \mathsf{n} \in \{1..\mathsf{N}\}\}$ represent the combined amplitude and phase sensitivity interval of the individual elements of a sensor array, and let $\{\boldsymbol{C}_{\mathsf{m},\mathsf{n}} \in \mathcal{P}(\mathbb{C}) \big| \mathsf{m}, \mathsf{n} \in \{1..\mathsf{N}\}\}$ represent the coupling coefficient interval of each pair of elements, including the self-coupling $\boldsymbol{C}_{\mathsf{n},\mathsf{n}} = 1$. Then, assuming a

narrow-band, far-field operation, the complex response interval of the array is $\boldsymbol{B} = \sum_{\mathsf{n}} \boldsymbol{A}_{\mathsf{n}} \sum_{\mathsf{m}} \boldsymbol{C}_{\mathsf{m},\mathsf{n}}$.

Since the addition of polar intervals is not only not tight, but also computationally heavy, the literature considers using rectangular, circular, and convex polygonal representations to calculate bounds on the array response interval. While computationally very light, the usage of the rectangular and circular types will introduce a signficant loss of tightness at the type-casting and another loss of tightness though the multiplicaton

$$\tau\left(\boldsymbol{B}^{\mathcal{R} \text{ or } \mathcal{C}}\right) \leq \tau\left(\boldsymbol{C}_{\mathsf{m},\mathsf{n}}^{\mathcal{R} \text{ or } \mathcal{C}}\right) \leq 1.$$

The convex polygonal type can represent the outer convex arc of a polar interval with arbitrary precision, but it replaces the concave inner arc with an edge. One could argue that since the addition is a convexifying operation – in other words, the sums of many concave sets are approximately convex (Schneider, 1993) – the representation of the sum will be sufficiently tight in most cases. However, the product of polygonal intervals can include concave parabolic segments, which is relaxed to edges through the mupltiplication causing additional loss of tightness

$$\tau\left(\boldsymbol{B}^{\mathcal{G}}\right) \leq \tau\left(\sum_{\mathsf{m}} \boldsymbol{C}_{\mathsf{m},\mathsf{n}}^{\mathcal{G}}\right) \approx 1.$$

We found that the polyarcular type is an ideal choice for this application, as it provides perfect tightness for a limited increase in computational complexity. We showed that the sum of the polar intervals is in the polyarcular subspace: $\sum_{\mathsf{m}} \boldsymbol{C}_{\mathsf{m},\mathsf{n}}^{\mathcal{A}} \in \mathcal{A}(\mathbb{C})$, and that the product of the polyarcular intervals is polyarcular if at least one of each pair of operand segments is from a zero-crossing line or a zero-centered circle. Since the boundary segments of the polar intervals are all from zero crossing lines or zero-centered circles, the product of a polyarcular and a polar interval is in the polyarcular subspace: $\boldsymbol{A}_{\mathsf{n}}^{\mathcal{A}} \sum_{\mathsf{m}} \boldsymbol{C}_{\mathsf{m},\mathsf{n}}^{\mathcal{A}} \in \mathcal{A}(\mathbb{C})$. Finally, the sum of polyarcular intervals is polyarcular; therefore, the complex response interval is in the polyarcular subspace: $\boldsymbol{B} \in \mathcal{A}(\mathbb{C})$. This means that the polyarcular type provides perfectly tight bounds on the complex response interval:

$$\tau(\boldsymbol{B}^{\mathcal{A}}) = 1.$$

Figure 8 shows an example of such a tolarance analysis.

6 Conclusion

In this paper, we showed that all commonly used complex interval types can be represented and arithmetically combined using the polyarcular interval type with improved or equivalent tightness in return for a moderate increase in complexity. This makes the polyarcular type a valuable option for performing calculations in various cases of interval analysis. We have shown that, similar to the polygonal type, simple arithmetic operations can be performed with a computational complexity linearly dependent on the number of elements

constituting the operand boundaries. We presented a case study that served as our motivation to develop this method. It shows that the polyarcular interval type is applicable in practical design tasks.

We found no commonly used formal definition of complex intervals, so we followed suit in finding a practical definition fitting the paper's scope. While it was tempting to further discuss the general properties of complex intervals, we rather relied on the existing literature and focused on the representations in this paper. However, we could not resist trying to fill in some of the gaps in the discussion of the arithmetic properties of primitive complex intervals, which is a significantly less published area than its two neighbours: the real interval arithmetic and the geometry of plane curves.

One could argue that polygonal interval arithmetic is the most general approach to the representation of complex intervals that provides very tight bounds as it can sample the infinite boundary set with an arbitrary resolution and perform point-wise operations on the vertices. (Hence, polygonal interval arithmetic is also called the Minkowski method.) However, as we showed, the inclusiveness of the polygonal bounds is not automatically guaranteed when sampling convex curves or performing the reciprocal operation (Ohta et al, 1990). We can also argue that within the complex interval subspaces the polyarcular type is more general than the polygonal, and that perfect tightness for a fixed computational complexity can be preferred in some cases over a variable tightness depending on the interval attributes and chosen number of vertices.

While our derivations provide small theoretical progress beyond the work of Farouki et al (2001), the closed form parametric conditions allowed us to design efficient algorithms by dividing the segment-wise operations into subcases of increasing complexity. We also hope that the used method (see Section 3) will enable the analysis of other problems too.

Areas of potential further research are the derivation of additional algebraic functions and conformal mappings to widen the range of applicability in interval analytical problems, while it would also be desirable to attempt to bridge the gap between statistics and interval analysis by investigating how simple unary and binary operations work on complex probabilistic variables (inspired by the grayscale morphology in Giardina and Dougherty (1988)). We also intend to publish a polyarcular interval type code implementation in the near future in our existing repository of complex interval arithmetic codes: https://github.com/unioslo-mn/ifi-complex-interval-arithmetic/.

Supplementary information. Derivation of the arithmetic properties of edges and arcs can be found in Online Resource 1.

Acknowledgments. Special thanks to Håvard Arnestad (Department of Informatics, University of Oslo) and Tor Inge Lønmo (Kongsberg Discovery) for their assistance in the case study and the review and editing of the manuscript.

We appreciate the assistance of Andreas Austeng, Jan Egil Kirkebø, Sven Peter Näsholm (Department of Informatics, University of Oslo) and Tom Louis Lindstrøm (Department of Mathematics, University of Oslo) for useful comments on an earlier draft of this manuscript.

Many thanks to Jacob Sznajdman (Neo4j) for his advice during the concept development.

We are grateful to László Surányi for connecting the two authors and we thank Gergő Pintér (Budapest University of Technology) for the enlightening conversations.

G. Geréb acknowledge funding from the Research Council of Norway project *Element calibration of sonars and echosounders*, project number 317874.

A. Sándor acknowledges funding from the Global Teaching Fellowship Program of Central European University and the Élvonal (Frontier) Grant KKP144148 of the NKFIH.

Statements and Declarations. We have no conflicts of interest to disclose.

References

Adams WW, Loustaunau P (1994) An introduction to Gröbner bases. No. 3 in Graduate studies in mathematics, American mathematical society, Providence (R.I.)

Anselmi N, Poli L, Tenuti L, et al (2015) Tolerance analysis of planar arrays through Minkowski-based Interval Analysis. p 2502, https://doi.org/10.1109/APS.2015.7305639

Arnestad HK, Geréb G, Lønmo TIB, et al (2023) Worst-case analysis of array beampatterns using interval arithmetic. The Journal of the Acoustical Society of America https://doi.org/10.1121/10.0019715

de Berg M, Cheong O, van Kreveld M, et al (2008) Computational Geometry: Algorithms and Applications. Springer, Berlin Heidelberg

Boche R (1965) Complex interval arithmetic with some applications. Tech. Rep. LMSC4-22-66-1,, Locheed Missiles and Space

Bruce JW, Giblin P (1984) Curves and Singularities. Cambridge University Press, UK

Bruce JW, Giblin PJ (1981) What Is an Envelope? The Mathematical Gazette 65(433):186–192. https://doi.org/10.2307/3617131, URL http://www.jstor.org/stable/3617131, publisher: Mathematical Association

Candau Y, Raissi T, Ramdani N, et al (2006) Complex Interval Arithmetic Using Polar Form. Reliable Computing 12:1–20. https://doi.org/10.1007/

s11155-006-2966-7

Dawood H (2011) Theories of Interval Arithmetic: Mathematical Foundations and Applications

Dawood H, Dawood Y (2019) A Logical Formalization of the Notion of Interval Dependency: Towards Reliable Intervalizations of Quantifiable Uncertainties. Online Mathematics 1:15–36. https://doi.org/10.5281/zenodo.3234184

Dongarra JJ (1995) BLAS (Basic Linear Algebra Subprograms). URL https://netlib.org/blas/

Farouki RT, Moon HP, Ravani B (2000) Algorithms for Minkowski products and implicitly-defined complex sets. Advances in Computational Mathematics 13(3):199–229. https://doi.org/10.1023/A:1018910412112, URL https://doi.org/10.1023/A:1018910412112

Farouki RT, Moon HP, Ravani B (2001) Minkowski Geometric Algebra of Complex Sets. Geometriae Dedicata 85(1):283–315. https://doi.org/10.1023/A:1010318011860

Farouki RT, Han CY, Hass J (2005) Boundary evaluation algorithms for Minkowski combinations of complex sets using topological analysis of implicit curves. Numerical Algorithms 40(3):251–283. https://doi.org/10.1007/s11075-005-4565-9

Gargantini I, Henrici P (1971) Circular arithmetic and the determination of polynomial zeros. Numerische Mathematik 18(4):305–320. https://doi.org/10.1007/BF01404681

Giardina C, Dougherty E (1988) Morphological methods in image and signal processing

Hansen ER (1975) A generalized interval arithmetic. In: Nickel K (ed) Interval Mathematics. Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, pp 7–18, https://doi.org/10.1007/3-540-07170-9_2

He G, Gao X, Zhou H, et al (2019) Comparison of three interval arithmetic-based algorithms for antenna array pattern upper bound estimation. Electronics Letters 55(14):775–776. https://doi.org/10.1049/el.2019.1229, URL https://onlinelibrary.wiley.com/doi/abs/10.1049/el.2019.1229

He G, Gao X, Zhang R (2021) Impact Analysis and Calibration Methods of Excitation Errors for Phased Array Antennas. IEEE Access https://doi.org/10.1109/ACCESS.2021.3073222

Jaulin L, Kieffer M, Didrit O, et al (2001) Applied Interval Analysis: With Examples in Parameter and State Estimation, Robust Control and Robotics. Springer, Berlin

Kearfott RE, Musaev EA, Nesterov VM, et al (1995) Reliable Computing | Volumes and issues. URL https://link.springer.com/journal/11155/volumes-and-issues

Kreinovich V, Lauter C (2023) Interval Computations. URL https://www.cs.utep.edu/interval-comp/main.html

Lang S (2012) Calculus of Several Variables, 3rd edn. Springer, USA, oCLC: 899736081

Lee Ik, Kim MS, Elber G, et al (1999) The Minkowski Sum of 2D Curved Objects. 1998, Tel-Aviv

Matheron G (1975) Random sets and integral geometry. Wiley series in probability and mathematical statistics, Wiley, New York

Milnor JW (1963) Morse Theory. No. 51 in Annals of mathematics studies, Princeton University Press, USA

Moore R (1963) Interval arithmetic and automatic error analysis in digital computing. PhD thesis, Stanford University California

Moore RE, Kearfott RB, Cloud MJ (2009) Introduction to Interval Analysis. SIAM, Philadelphia

Ohta Y (2000) Nonconvex Polygon Interval Arithmetic as a Tool for the Analysis and Design of Robust Control Systems. Reliable Computing 6(3):247–279. https://doi.org/10.1023/A:1009926413485

Ohta Y, Gong L, Haneda H (1990) Polygon interval arithmetic and design of robust control systems. In: 29th IEEE Conference on Decision and Control, pp 1065–1067 vol.2, https://doi.org/10.1109/CDC.1990.203765

Petkovic M, Petkovic LD (1998) Complex Interval Arithmetic and Its Applications. John Wiley & Sons, Berlin

Rump SM (1999) INTLAB — INTerval LABoratory. In: Csendes T (ed) Developments in Reliable Computing. Springer Netherlands, Dordrecht, p 77–104, https://doi.org/10.1007/978-94-017-1247-7_7, URL www.tuhh.de/ti3/rump/intlab/

Schneider R (1993) Convex Bodies: The Brunn-Minkowski Theory. Cambridge University Press, Cambridge, UK

Shafarevich IR (2013) Basic Algebraic Geometry 1: Varieties in Projective Space. Springer Berlin Heidelberg, Berlin, Heidelberg

Stewart J (1999) Calculus, 4th edn. Brooks/Cole, Pacific Grove, CA

Tenuti L, Anselmi N, Rocca P, et al (2017) Minkowski Sum Method for Planar Arrays Sensitivity Analysis With Uncertain-But-Bounded Excitation Tolerances. IEEE Transactions on Antennas and Propagation 65(1):167–177. https://doi.org/10.1109/TAP.2016.2627548

Villafuerte M, Wiederhold P (2022) A Polygonal Approximation for General 4-Contours Corresponding to Weakly Simple Curves. Journal of Mathematical Imaging and Vision 64(2):161–193. https://doi.org/10.1007/s10851-021-01060-0, URL https://doi.org/10.1007/s10851-021-01060-0