Exhaustive Interval-based 2-D Shape Registration Under Similarity Transformation

SCAN 2025: 20th International Symposium on Scientific Computing, Computer Arithmetic, and Verified Numerical Computations
26^{th} September 2025, Oldenburg, Germany

Verlein Radwan[(1)], Simon Rohou[(2)], Gilles Trombettoni[(1)]

(1) LIRMM, Montpellier, France

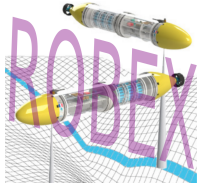
(2) ENSTA, Lab-STICC, Brest, France

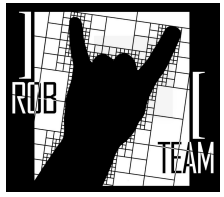












Introduction

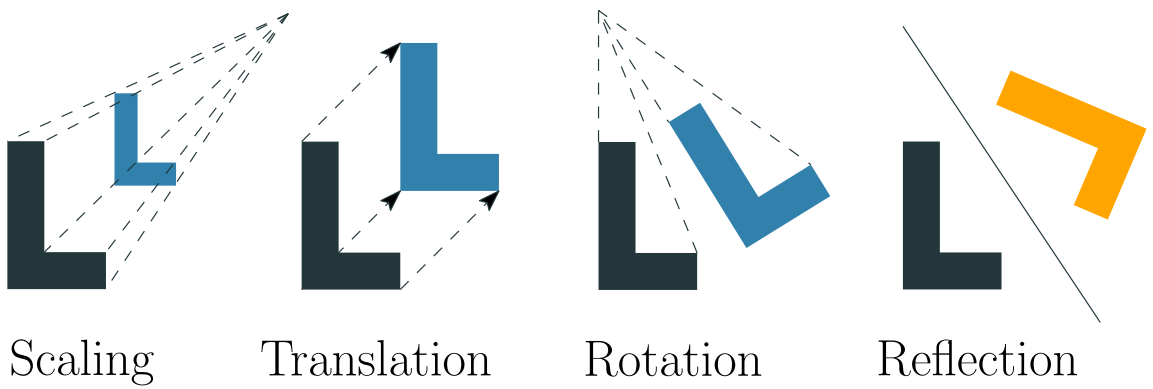
Similarity transformations (Euclidean geometry)

Two objects are **similar** if:

- they have the same shape
- or if one has the same shape as the mirror image of the other



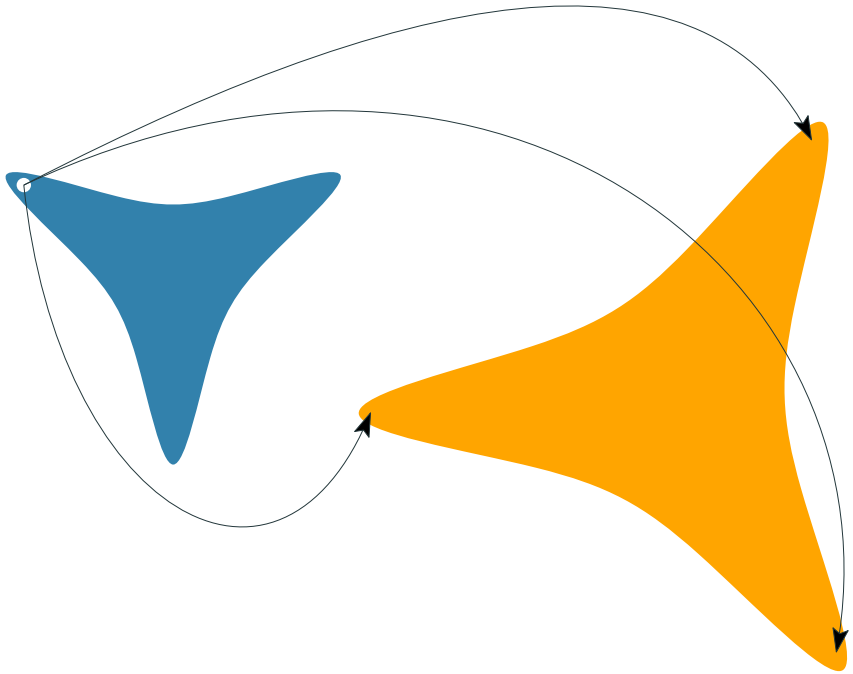
Similar objects



Multiple solutions

Exhaustive approximation of similarities:

- depending on the sets, there may be several possible similarity transformations:



- because the sets are bounded, these multiple solutions can only come from rotational or reflection symmetries.

Set-membership approximation

Given two sets $\mathbb{A}$ and $\mathbb{B}$, define the set of feasible transformations:

$$\mathbb{P} = \left\{ \mathbf{p} \in \mathbb{R}^3 \times [0, 2\pi] \mid \mathbf{f_p}(\mathbb{A}) = \mathbb{B} \right\}. \tag{1}$$

$\mathbb{P}$ is not exactly computable, but an over-estimation $\mathbb{P}^+ \supset \mathbb{P}$ is computationally affordable.

Set-membership approximation

Given two sets $\mathbb{A}$ and $\mathbb{B}$, define the set of feasible transformations:

$$\mathbb{P} = \left\{ \mathbf{p} \in \mathbb{R}^3 \times [0, 2\pi] \mid \mathbf{f_p}(\mathbb{A}) = \mathbb{B} \right\}. \tag{1}$$

$\mathbb{P}$ is not exactly computable, but an over-estimation $\mathbb{P}^+ \supset \mathbb{P}$ is computationally affordable.

Using homogeneous coordinates, the classical formulation is:

$$\begin{pmatrix} \mathbb{B} \\ 1 \end{pmatrix} = \left(\begin{array}{cc|c} & k\mathrm{R}_\theta & \mathbf{t} \\ \hline 0 & 0 & 1 \end{array}\right) \begin{pmatrix} \mathbb{A} \\ 1 \end{pmatrix} \tag{2}$$

with $k \in \mathbb{R}$ for scaling, $\theta \in \mathbb{R}$ for rotation and $\mathbf{t} \in \mathbb{R}^2$ for translation.

However, compute $\mathbb{P}^+ \in \mathcal{P}\left(\mathbb{R}^4\right)$ with a classical branch-and-bound algorithm is costly, due to the exponential complexity of these branching methods.

Procrustes



(a) *British Museum (Vase E 84).*



(b) *Caricature. Berliner Wespen, 1878*

Procrustes



(a) *British Museum (Vase E 84).*



(b) *Caricature. Berliner Wespen, 1878*

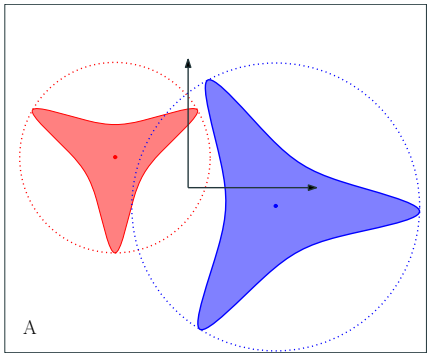
Procrustes analysis: process of performing a shape-preserving Euclidean transformation to a set of shapes.

$\Rightarrow$ removes variations in translation, rotation and scaling across the dataset

$\Rightarrow$ move the sets into a common frame of reference

Procrustes analysis on shapes

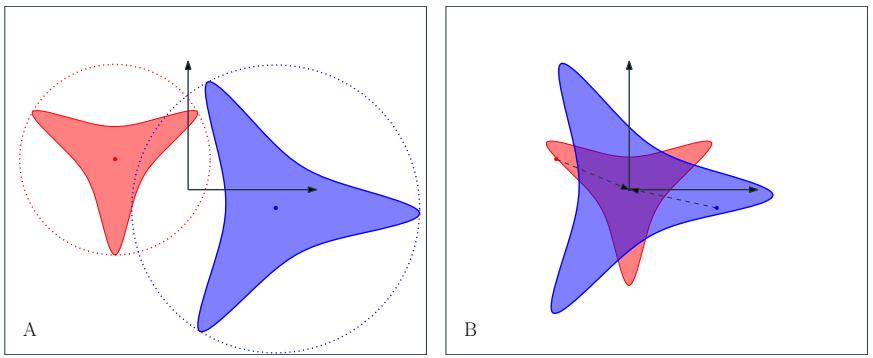
Key idea (Procrustes approach): compute the smallest enclosing circles, to then easily deal with translation, scaling and rotation.



A. Smallest enclosing circle computation: center $\mathbf{c}$ and radius r

Procrustes analysis on shapes

Key idea (Procrustes approach): compute the smallest enclosing circles, to then easily deal with translation, scaling and rotation.

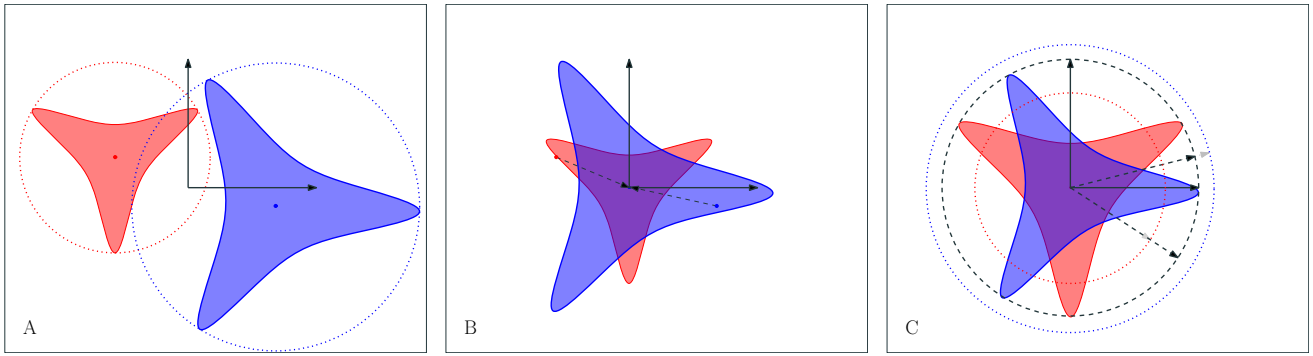


A. Smallest enclosing circle computation: center $\mathbf{c}$ and radius r

B. Centering $\Longrightarrow$ translation $\mathbf{t}$

Procrustes analysis on shapes

Key idea (Procrustes approach): compute the smallest enclosing circles, to then easily deal with translation, scaling and rotation.



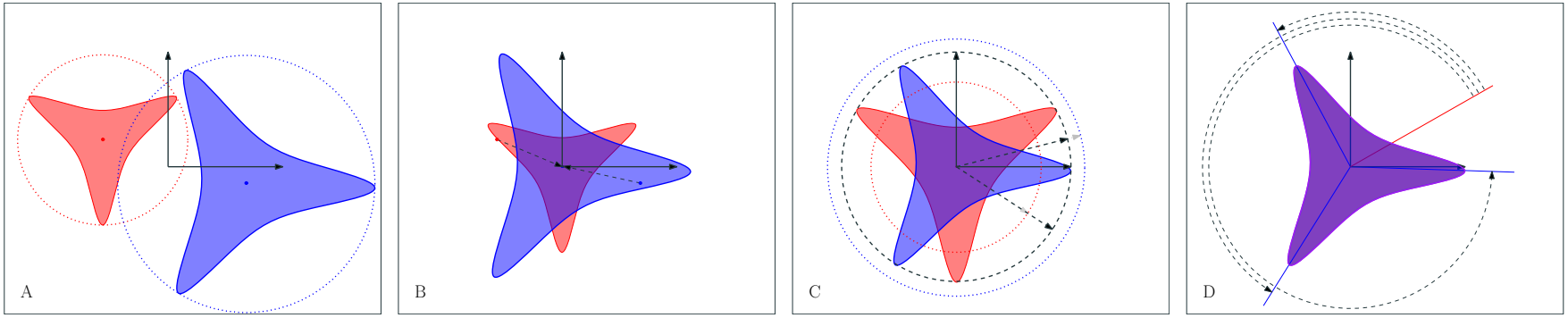
A. Smallest enclosing circle computation: center $\mathbf{c}$ and radius r

B. Centering $\Longrightarrow$ translation $\mathbf{t}$

C. Normalization $\Longrightarrow$ scaling k

Procrustes analysis on shapes

Key idea (Procrustes approach): compute the smallest enclosing circles, to then easily deal with translation, scaling and rotation.



A. Smallest enclosing circle computation: center $\mathbf{c}$ and radius r

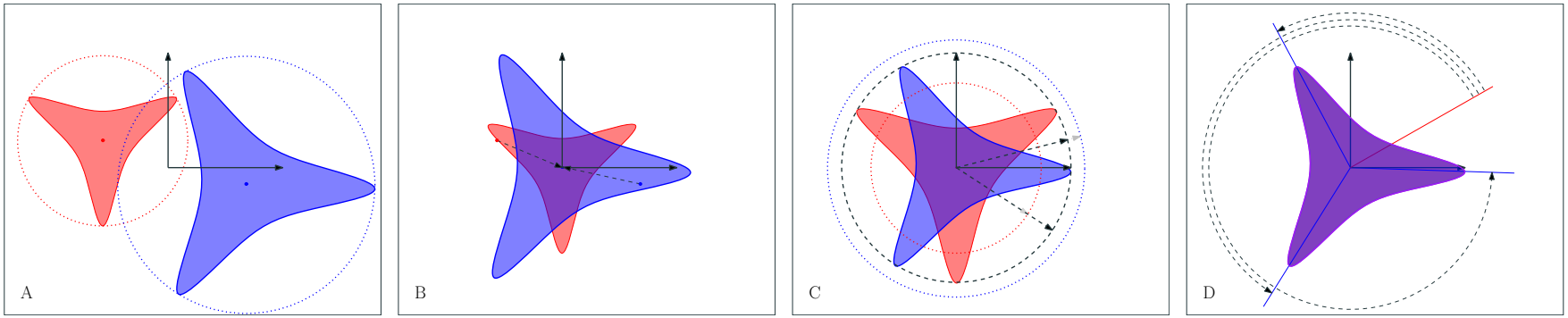
B. Centering $\Longrightarrow$ translation $\mathbf{t}$

C. Normalization $\Longrightarrow$ scaling k

D. Rotational mapping $\Longrightarrow$ rotations $\{\theta_1, \theta_2, \dots\} = \Theta$

Procrustes analysis on shapes

Key idea (Procrustes approach): compute the smallest enclosing circles, to then easily deal with translation, scaling and rotation.



A. Smallest enclosing circle computation: center $\mathbf{c}$ and radius r

B. Centering $\Longrightarrow$ translation $\mathbf{t}$

C. Normalization $\Longrightarrow$ scaling k

D. Rotational mapping $\Longrightarrow$ rotations $\{\theta_1, \theta_2, \dots\} = \Theta$

We transform the problem into a rotational mapping

$\Longrightarrow$ final branching for Θ performed in only one dimension

Contractors and Separators

Contractors

Definition

A contractor $\mathcal{C}$, associated with a constraint $\mathcal{L}$, is a mapping from $\mathbb{IR}^n$ to $\mathbb{IR}^n$ such that:

$$(i) \quad \forall [\mathbf{x}] \in \mathbb{IR}^n,\ \mathcal{C}([\mathbf{x}]) \subseteq [\mathbf{x}], \qquad \textit{(contraction)}$$

$$(ii) \quad \mathcal{L}(\mathbf{x}) \wedge (\mathbf{x} \in [\mathbf{x}]) \implies \mathbf{x} \in \mathcal{C}([\mathbf{x}]). \qquad \textit{(consistency)}$$

L. Jaulin, M. Kieffer, O. Didrit, and W. Eric. Applied Interval Analysis. Springer London, London. 2001

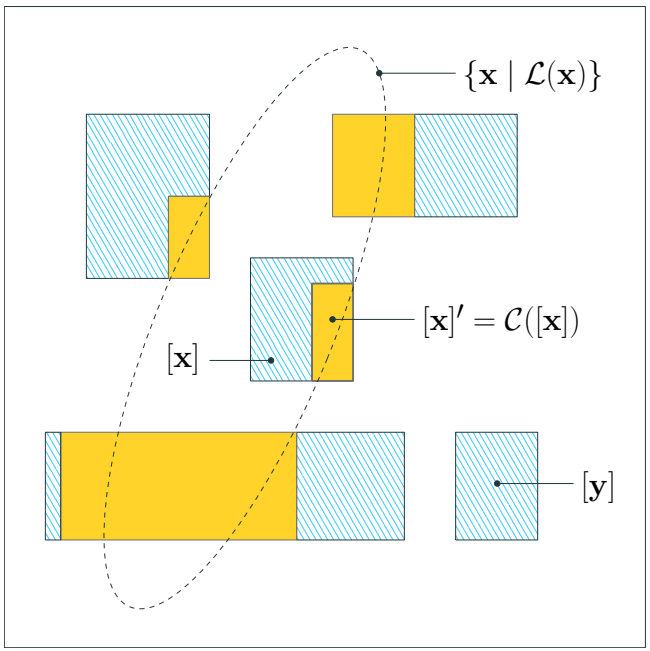


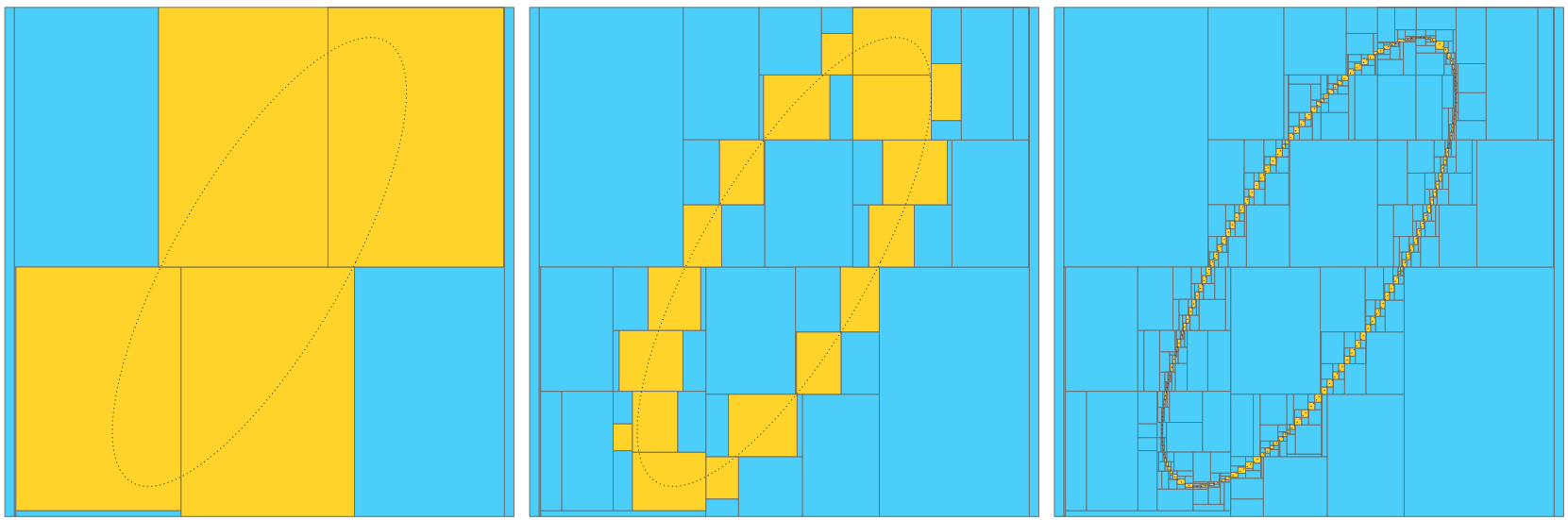
Illustration of contractions.
Several boxes are shown before and after contraction. The blue components depict the parts of the boxes that have been removed by the operator.

Revealing sets associated with contractors

Consider an ellipse constraint:

$$\mathcal{L}_{\text{ell}}: \quad ax_1^2 + bx_2^2 + cx_1x_2 + dx_1 + ex_2 + f = 0 \tag{3}$$

Illustration of an associated contractor $\mathcal{C}_{\text{ell}}$ involved in a paver:



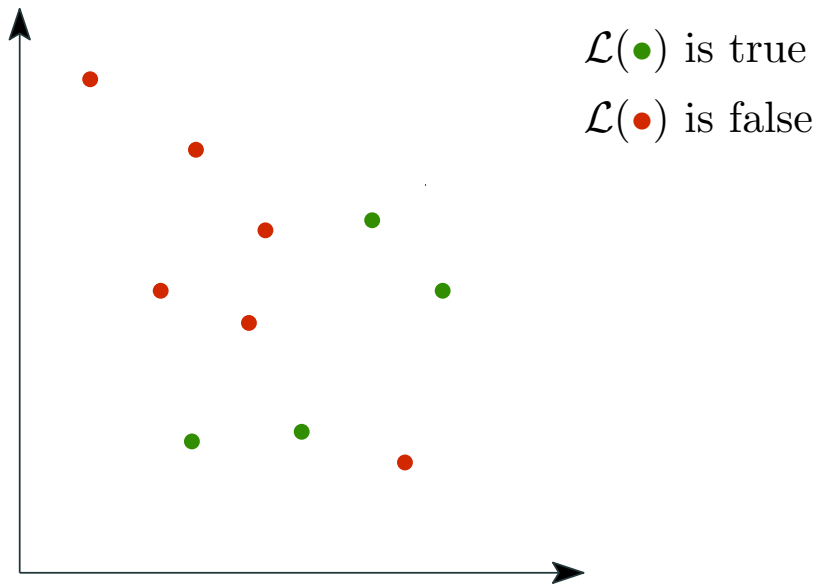
Paving the operator $\mathcal{C}_{ell}$ with several precisions $\epsilon \in \{4, 1, 0.1\}$.

Equivalence contractors $\longleftrightarrow$ sets

Let us denote $\mathrm{set}(\mathcal{L})$, the set associated with a constraint $\mathcal{L}$:

$$\mathrm{set}(\mathcal{L}) \quad = \quad \{\mathbf{x} \in \mathbb{R}^n \mid \mathcal{L}(\mathbf{x}) \text{ is true}\} \tag{4}$$

(5)

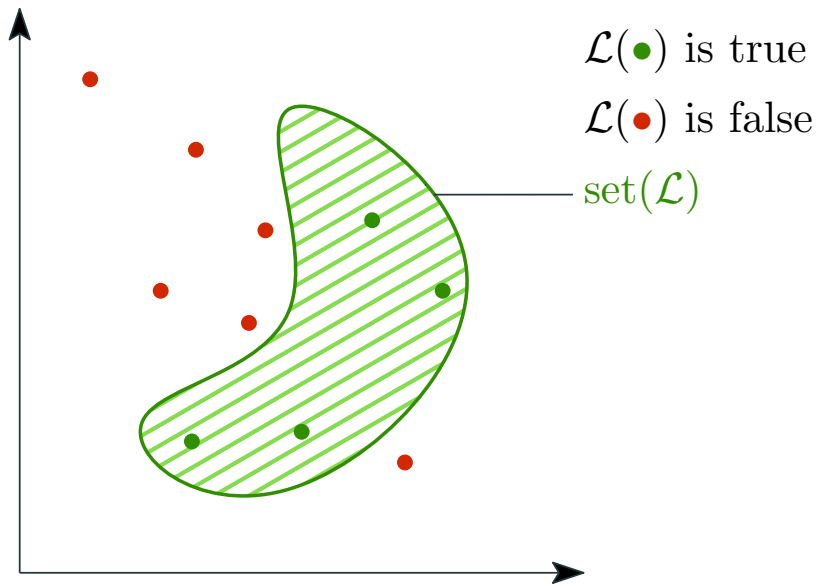


Equivalence contractors $\longleftrightarrow$ sets

Let us denote $\mathrm{set}(\mathcal{L})$, the set associated with a constraint $\mathcal{L}$:

$$\mathrm{set}(\mathcal{L}) \quad = \quad \{\mathbf{x} \in \mathbb{R}^n \mid \mathcal{L}(\mathbf{x}) \text{ is true}\} \tag{4}$$

(5)

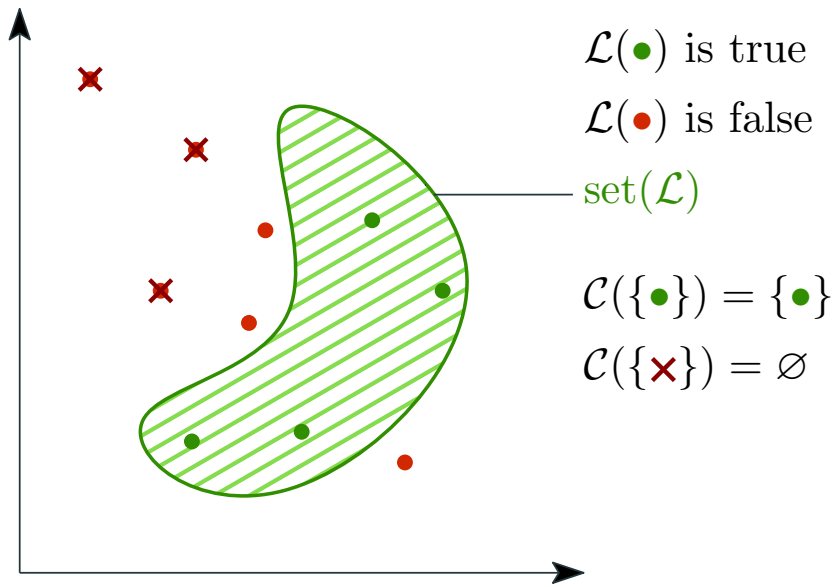


Equivalence contractors $\longleftrightarrow$ sets

Let us denote $\mathrm{set}(\mathcal{L})$, the set associated with a constraint $\mathcal{L}$:

$$\mathrm{set}(\mathcal{L}) = \{\mathbf{x} \in \mathbb{R}^n \mid \mathcal{L}(\mathbf{x}) \text{ is true}\} \tag{4}$$

$$\mathrm{set}(\mathcal{C}) = \{\mathbf{x} \in \mathbb{R}^n \mid \mathcal{C}(\{\mathbf{x}\}) = \{\mathbf{x}\}\} \tag{5}$$

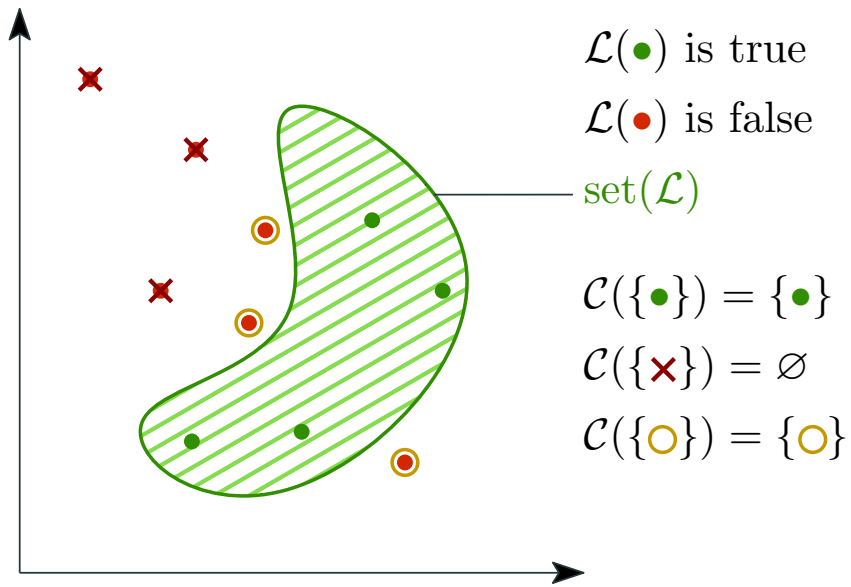


Equivalence contractors $\longleftrightarrow$ sets

Let us denote $\mathrm{set}(\mathcal{L})$, the set associated with a constraint $\mathcal{L}$:

$$\mathrm{set}(\mathcal{L}) = \{\mathbf{x} \in \mathbb{R}^n \mid \mathcal{L}(\mathbf{x}) \text{ is true}\} \tag{4}$$

$$\mathrm{set}(\mathcal{C}) = \{\mathbf{x} \in \mathbb{R}^n \mid \mathcal{C}(\{\mathbf{x}\}) = \{\mathbf{x}\}\} \tag{5}$$

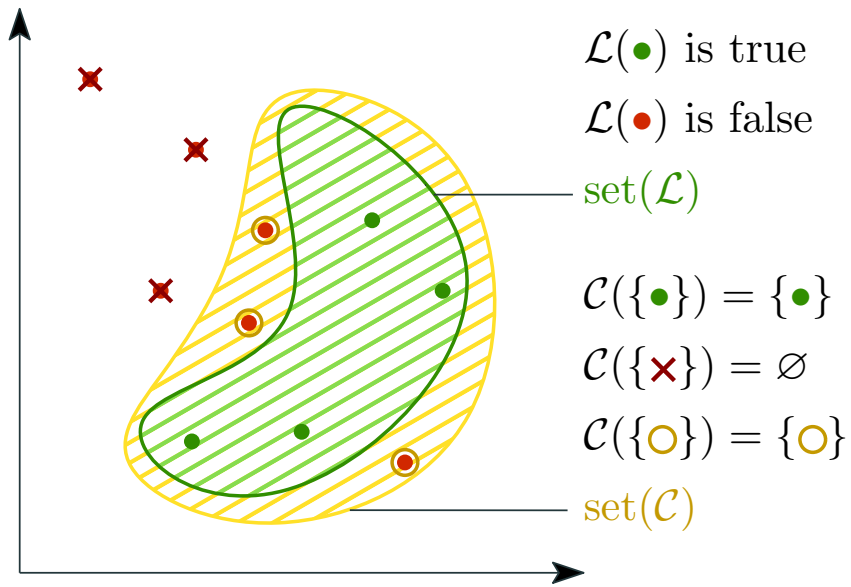


Equivalence contractors $\longleftrightarrow$ sets

Let us denote $\mathrm{set}(\mathcal{L})$, the set associated with a constraint $\mathcal{L}$:

$$\mathrm{set}(\mathcal{L}) = \{\mathbf{x} \in \mathbb{R}^n \mid \mathcal{L}(\mathbf{x}) \text{ is true}\} \quad (4)$$

$$\mathrm{set}(\mathcal{C}) = \{\mathbf{x} \in \mathbb{R}^n \mid \mathcal{C}(\{\mathbf{x}\}) = \{\mathbf{x}\}\} \quad (5)$$



If $\mathcal{C}$ is a contractor for the constraint $\mathcal{L}$, then:

$$\mathrm{set}(\mathcal{L}) \subseteq \mathrm{set}(\mathcal{C}) \quad (6)$$

Equivalence contractors $\longleftrightarrow$ sets

Since [Chabert2009], **a contractor can be seen as a set** *equivalent* to the associated constraint:

$$\mathrm{set}(\mathcal{L}) \sim \mathcal{C} \tag{7}$$

G. Chabert and L. Jaulin. *Contractor Programming*. *Artificial Intelligence*, 173:1079–1100, 2009.

Consequently, it becomes possible to use classical set operators on contractors in order to implement logical operations or relational algebra.

Example:

$$\begin{aligned}
\mathcal{L}_3 &= \mathcal{L}_1 \wedge \mathcal{L}_2 && (8)\\
\mathrm{set}\,(\mathcal{L}_3) &= \mathrm{set}\,(\mathcal{L}_1 \wedge \mathcal{L}_2) && (9)\\
&= \mathrm{set}\,(\mathcal{L}_1) \cap \mathrm{set}\,(\mathcal{L}_2) && (10)\\
&\subseteq \mathrm{set}\,(\mathcal{C}_1) \cap \mathrm{set}\,(\mathcal{C}_2) && (11)\\
&\sim \mathcal{C}_1 \cap \mathcal{C}_2 && (12)
\end{aligned}$$

Separators (as operators)

Definition [Desrochers2018]

A separator $\mathcal{S}_{\mathbb{X}}$, associated with a set $\mathbb{X} \subset \mathbb{R}^n$, is an application such that:

$$\mathcal{S} : \mathbb{IR}^n \to \mathbb{IR}^n \times \mathbb{IR}^n \tag{13}$$

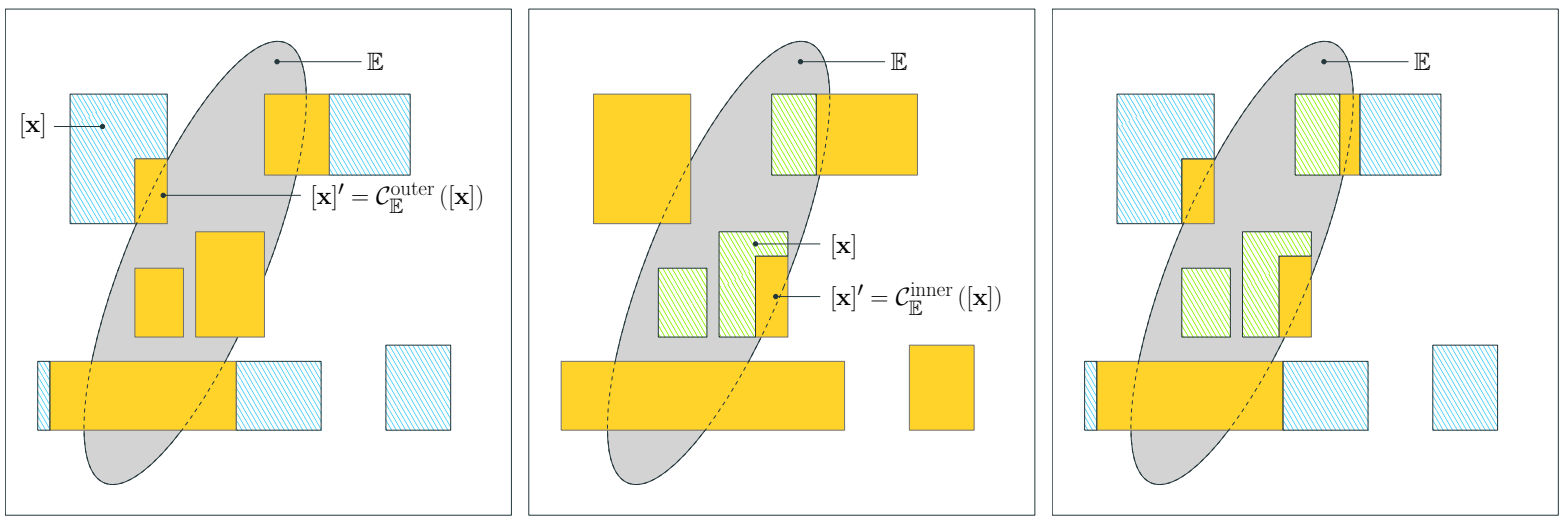
$$[\mathbf{x}] \mapsto ([\mathbf{x}_{in}], [\mathbf{x}_{out}]) \tag{14}$$

B. Desrochers. SLAM in unstuctured environments, a set-membership approach. PhD thesis, 2018.
L. Jaulin and B. Desrochers. Introduction to the Algebra of Separators [..]. Eng. Appl. of Artificial Intelligence, 2014.

A separator can also be seen as **a pair of contractors** $\{\mathcal{C}_{in}, \mathcal{C}_{out}\}$.

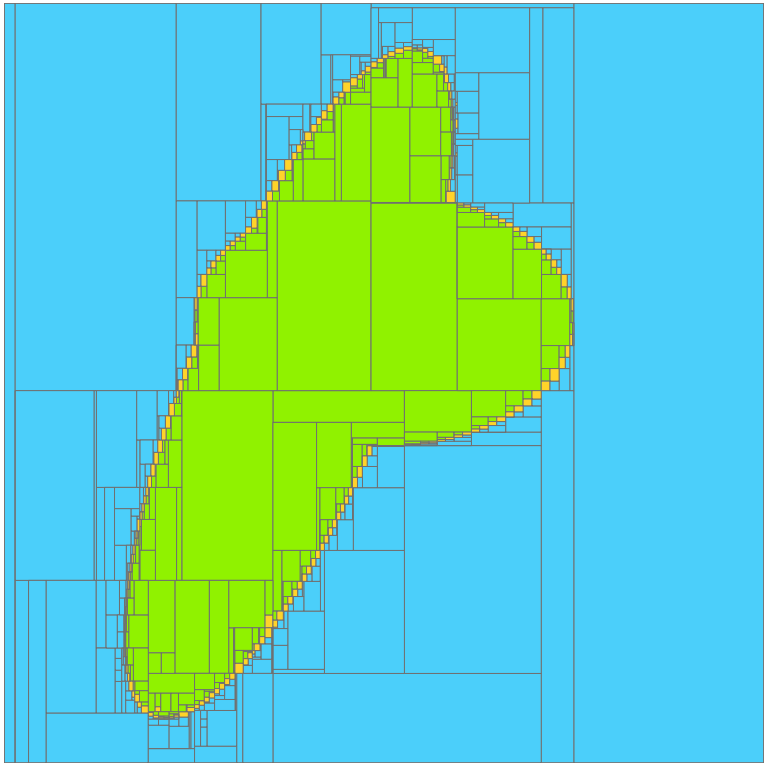
A set $\mathbb{X}$ is consistent with the separator $\mathcal{S}$ if:

$$\mathbb{X} \sim \mathcal{C}_{out} \quad \text{and} \quad \overline{\mathbb{X}} \sim \mathcal{C}_{in} \tag{15}$$

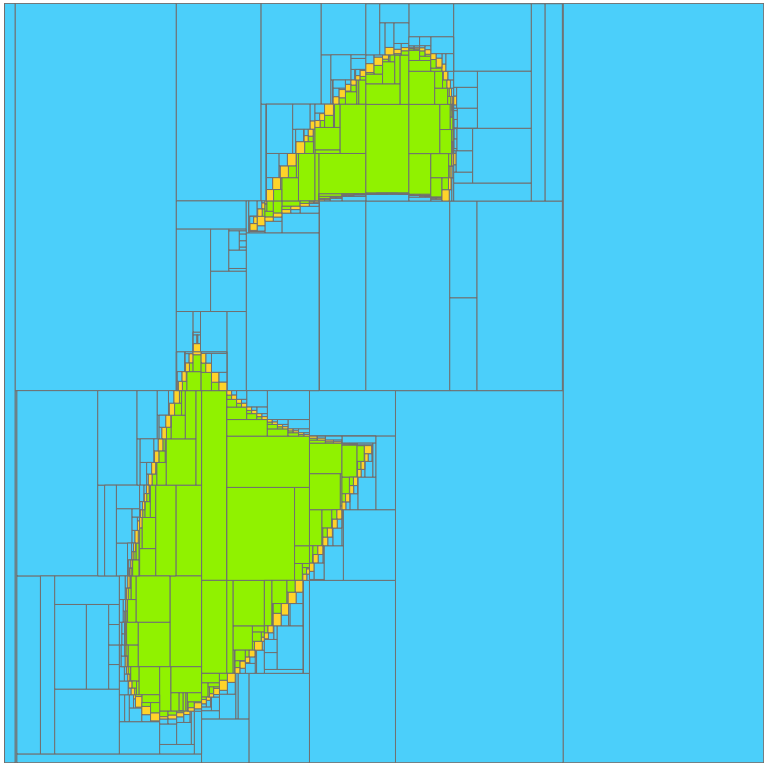


Separators (as sets)

As with contractors, we can use **Separator programming**:



(a) *Paving approximation of a constraint $\mathcal{E}_1 \vee \mathcal{E}_2$ represented by the separator $\left(\mathcal{S}_{\mathbb{E}_1} \cup \mathcal{S}_{\mathbb{E}_2}\right)$.*



(b) *Paving approximation of a constraint $\mathcal{E}_1 \wedge \neg\mathcal{E}_2$ represented by the separator$\left(\mathcal{S}_{\mathbb{E}_1} \cap \overline{\mathcal{S}_{\mathbb{E}_2}}\right)$.*

Example of Codac implementation in Python.
http://codac.io/v2

Codac Python code associated with the previous slide, Fig. (b):

```
from codac import *

x,p = VectorVar(2), VectorVar(5)

e = AnalyticFunction([x,p],
      sqr(((x[0]-p[2])*cos(p[4])+(x[1]-p[3])*sin(p[4]))/p[0])
    + sqr(((x[0]-p[2])*sin(p[4])-(x[1]-p[3])*cos(p[4]))/p[1]))
e1 = AnalyticFunction([x], e(x,Vector([3,1,-1,0,PI/3.])))
e2 = AnalyticFunction([x], e(x,Vector([2,1,0,0.5,0])))

S1 = SepInverse(e1,[-oo,1])
S2 = SepInverse(e2,[-oo,1])
S = S1 & ~S2

draw_while_paving([[-4,3],[-3,3]], S, 0.05)
```

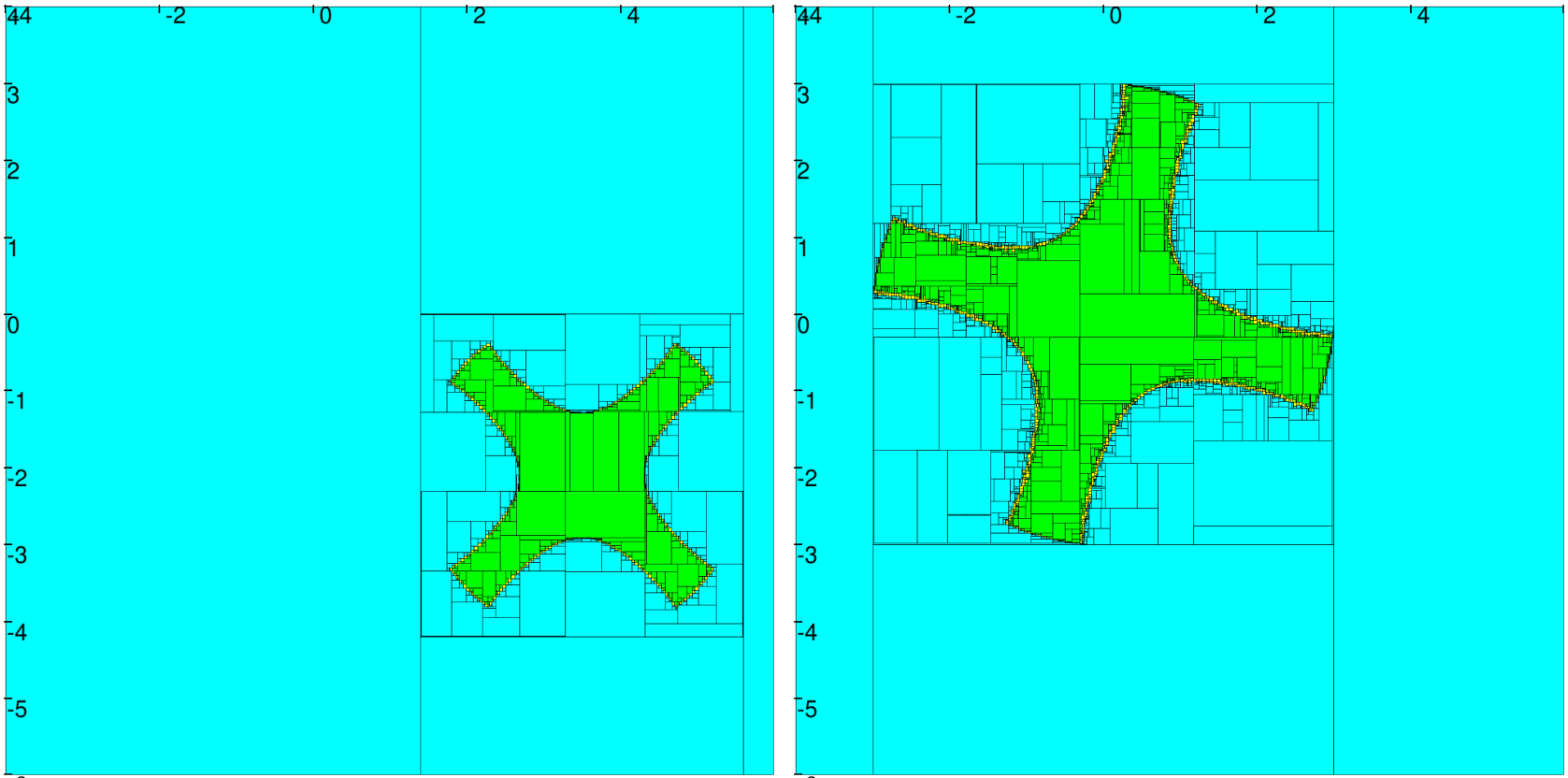
S. Rohou, B. Desrochers and F. Le Bars. The Codac Library: A Catalog of Domains and Contractors.
Acta Cybernetica, 26(4), 871-887, 2014.

Reliable approximation of transformations using separators

Provided sets

Provided sets are:

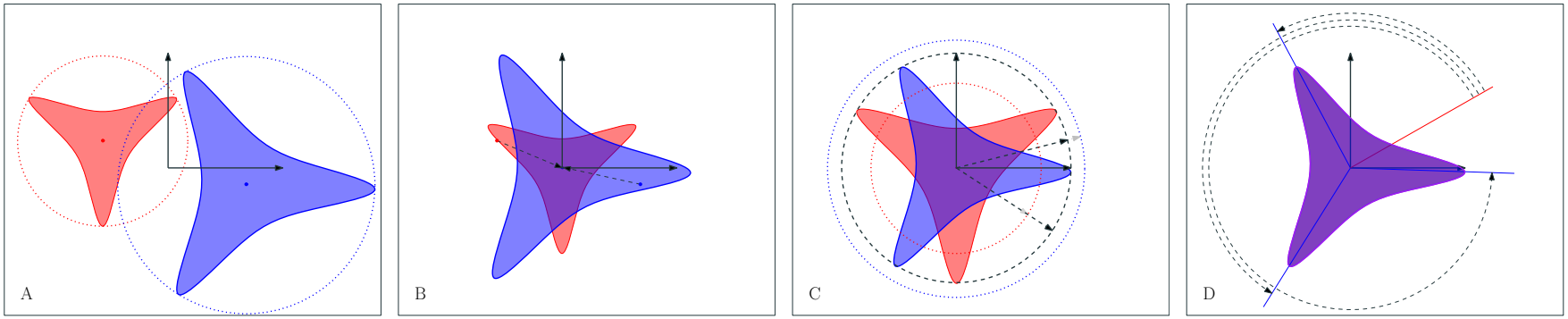
- bounded sets $\mathbb{A}$ and $\mathbb{B}$ of $\mathbb{R}^2$
- described by separators $\mathcal{S}_{\mathbb{A}}$ and $\mathcal{S}_{\mathbb{B}}$, revealed into pavers:



(a) *Paving separator $\mathcal{S}_{\mathbb{A}}$.* ***(b)*** *Paving separator $\mathcal{S}_{\mathbb{B}}$.*

Procrustes-like set-membership method

Recall the strategy:



Operations:

A1. $\mathbf{c}^{\mathbb{A}} = \texttt{MinEnclosingCircle}(\mathbb{A})$

A2. $\mathbf{c}^{\mathbb{B}} = \texttt{MinEnclosingCircle}(\mathbb{B})$

B1,C1. $\mathbb{A}_N = \left(\mathbb{A} - \mathbf{c}^{\mathbb{A}}_{1,2}\right) / c^{\mathbb{A}}_3$

B2,C2. $\mathbb{B}_N = \left(\mathbb{B} - \mathbf{c}^{\mathbb{B}}_{1,2}\right) / c^{\mathbb{B}}_3$

D. $\mathbb{B}_N = \mathbf{R}_\theta \cdot \mathbb{A}_N$

$k = c^{\mathbb{B}}_3 / c^{\mathbb{A}}_3$

$\mathbf{t} = \mathbf{c}^{\mathbb{B}}_{1,2} - k\mathbf{R}_\theta \cdot \mathbf{c}^{\mathbb{A}}_{1,2}$

Step A, minimum enclosing circle of a set

Consider $g : \mathbb{R}^3 \times \mathbb{R}^2 \to \mathbb{R}$:

$$g(\mathbf{c}, \mathbf{x}) = (x_1 - c_1)^2 + (x_2 - c_2)^2 - c_3^2. \tag{16}$$

The definition of the unique circle $\mathbf{c}^* \in \mathbb{R}^3$ circumscribed about a set $\mathbb{X} \in \mathcal{P}(\mathbb{R}^2)$ is given by:

$$\mathbf{c}^* = \min_{c_3} \{ \underbrace{\mathbf{c} \mid \forall \mathbf{x} \in \mathbb{X}, g(\mathbf{c}, \mathbf{x}) \leqslant 0}_{\mathbb{C}} \}, \tag{17}$$

where $\mathbf{c}^*_{1,2}$ is the center of this circle and c_3^* its radius.

Step A, minimum enclosing circle of a set

Consider $g : \mathbb{R}^3 \times \mathbb{R}^2 \to \mathbb{R}$:

$$g(\mathbf{c}, \mathbf{x}) = (x_1 - c_1)^2 + (x_2 - c_2)^2 - c_3^2. \tag{16}$$

The definition of the unique circle $\mathbf{c}^* \in \mathbb{R}^3$ circumscribed about a set $\mathbb{X} \in \mathcal{P}(\mathbb{R}^2)$ is given by:

$$\mathbf{c}^* = \min_{c_3}\{ \underbrace{\mathbf{c} \mid \forall \mathbf{x} \in \mathbb{X}, g(\mathbf{c}, \mathbf{x}) \leqslant 0}_{\mathbb{C}} \}, \tag{17}$$

where $\mathbf{c}^*_{1,2}$ is the center of this circle and c_3^* its radius.

How to compute $\mathbb{C} = \{\mathbf{c} \mid \forall \mathbf{x} \in \mathbb{X}, g(\mathbf{c}, \mathbf{x}) \leqslant 0\}$?
How to implement the quantifier $\forall$ with set operators?

Step A, minimum enclosing circle of a set

Quantifier $\exists$ amounts to a projection, for instance:

$$\mathbb{Z}_1 = \mathrm{proj}_{\mathbb{X}_1}(\mathbb{Z}) = \{\mathbf{x} \in \mathbb{X}_1 \mid \exists \mathbf{y}, (\mathbf{x}, \mathbf{y}) \in \mathbb{Z}\}\,. \tag{18}$$

Step A, minimum enclosing circle of a set

Quantifier $\exists$ amounts to a projection, for instance:

$$\mathbb{Z}_1 = \text{proj}_{\mathbb{X}_1}(\mathbb{Z}) = \{\mathbf{x} \in \mathbb{X}_1 \mid \exists \mathbf{y}, (\mathbf{x}, \mathbf{y}) \in \mathbb{Z}\}\,. \tag{18}$$

Rewritting (De Morgan's laws):

$$\begin{aligned} \mathbb{C} &= \{\mathbf{c} \mid \forall \mathbf{x} \in \mathbb{X}, g(\mathbf{c}, \mathbf{x}) \leqslant 0\} && (19)\\ &= \{\mathbf{c} \mid \neg(\exists \mathbf{x} \in \mathbb{X}, g(\mathbf{c}, \mathbf{x}) > 0)\}\,. && (20) \end{aligned}$$

Step A, minimum enclosing circle of a set

Quantifier $\exists$ amounts to a projection, for instance:

$$\mathbb{Z}_1 = \text{proj}_{\mathbb{X}_1}(\mathbb{Z}) = \{\mathbf{x} \in \mathbb{X}_1 \mid \exists \mathbf{y}, (\mathbf{x}, \mathbf{y}) \in \mathbb{Z}\}\,. \tag{18}$$

Rewritting (De Morgan's laws):

$$\begin{aligned} \mathbb{C} &= \{\mathbf{c} \mid \forall \mathbf{x} \in \mathbb{X}, g(\mathbf{c}, \mathbf{x}) \leqslant 0\} && (19)\\ &= \{\mathbf{c} \mid \neg(\exists \mathbf{x} \in \mathbb{X}, g(\mathbf{c}, \mathbf{x}) > 0)\}\,. && (20) \end{aligned}$$

Towards separator programming:

$$\begin{aligned} \mathbb{G} &= \{(\mathbf{c} \times \mathbf{x}) \mid g(\mathbf{c}, \mathbf{x}) > 0\} \\ \mathbb{C} &= \overline{\text{proj}_{1,2,3}(\mathbb{G} \cap (\mathbb{R}^3 \times \mathbb{X}))} \end{aligned}$$

Step A, minimum enclosing circle of a set

Quantifier $\exists$ amounts to a projection, for instance:

$$\mathbb{Z}_1 = \text{proj}_{\mathbb{X}_1}(\mathbb{Z}) = \{\mathbf{x} \in \mathbb{X}_1 \mid \exists \mathbf{y}, (\mathbf{x}, \mathbf{y}) \in \mathbb{Z}\}\,. \tag{18}$$

Rewritting (De Morgan's laws):

$$\mathbb{C} = \{\mathbf{c} \mid \forall \mathbf{x} \in \mathbb{X}, g(\mathbf{c}, \mathbf{x}) \leqslant 0\} \tag{19}$$

$$= \{\mathbf{c} \mid \neg(\exists \mathbf{x} \in \mathbb{X}, g(\mathbf{c}, \mathbf{x}) > 0)\}\,. \tag{20}$$

Towards separator programming:

$$\left.\begin{array}{rl} \mathbb{G} = & \{(\mathbf{c} \times \mathbf{x}) \mid g(\mathbf{c}, \mathbf{x}) > 0\} \\ \mathbb{C} = & \overline{\text{proj}_{1,2,3}(\mathbb{G} \cap (\mathbb{R}^3 \times \mathbb{X}))} \end{array}\right\} \Rightarrow \left\{\begin{array}{l} \mathcal{S}_1 := \mathcal{S}_{\text{inv}}(g, [0, \infty]) \\ \mathcal{S}_2 := \mathcal{S}_1 \cap (\mathcal{S}_{\mathbb{R}^3} \times \mathcal{S}_{\mathbb{X}}) \\ \mathcal{S}_3 := \mathcal{S}_{\text{proj}_{1,2,3}}(\mathcal{S}_2) \\ \mathcal{S}_{\mathbb{C}} := \overline{\mathcal{S}_3} \end{array}\right.$$

Step A, minimum enclosing circle of a set

From $\mathbb{C}$ (separator $\mathcal{S}_{\mathbb{C}}$), approximate the smallest circle $\mathbf{c}^*$. Recall:

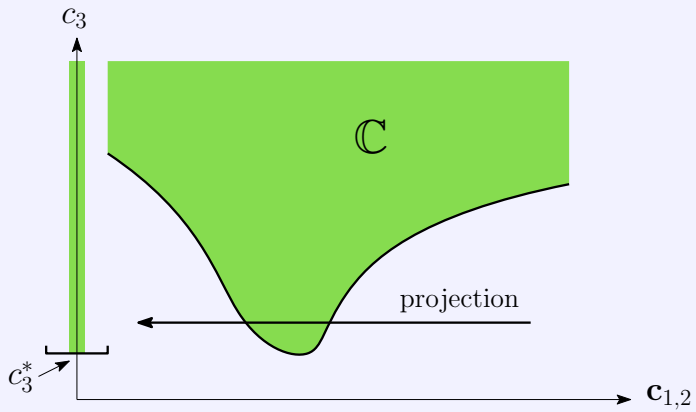
$$\mathbf{c}^* = \min_{c_3}\{ \underbrace{\mathbf{c} \mid \forall \mathbf{x} \in \mathbb{X}, g(\mathbf{c}, \mathbf{x}) \leqslant 0}_{\mathbb{C}} \} \tag{21}$$

Step A, minimum enclosing circle of a set

From $\mathbb{C}$ (separator $\mathcal{S}_{\mathbb{C}}$), approximate the smallest circle $\mathbf{c}^*$. Recall:

$$\mathbf{c}^* = \min_{c_3}\{ \underbrace{\mathbf{c} \mid \forall \mathbf{x} \in \mathbb{X}, g(\mathbf{c}, \mathbf{x}) \leqslant 0}_{\mathbb{C}} \} \tag{21}$$

The min can be computed in a set-theoretic way:



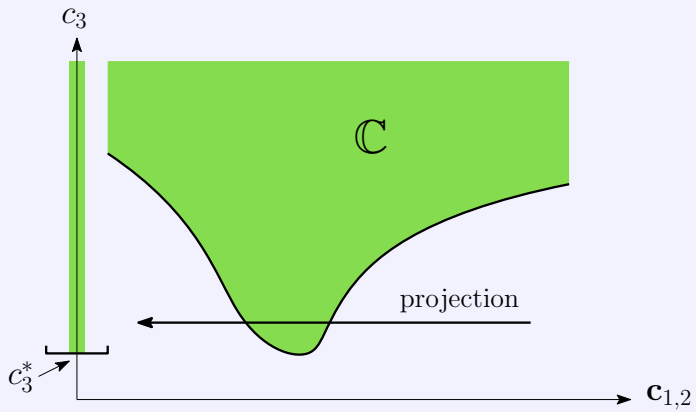
Minimizing amounts to projecting. The set $\mathbb{C}$ corresponds to the epigraph of a function to be minimized.

Step A, minimum enclosing circle of a set

From $\mathbb{C}$ (separator $\mathcal{S}_{\mathbb{C}}$), approximate the smallest circle $\mathbf{c}^*$. Recall:

$$\mathbf{c}^* = \min_{c_3}\{ \underbrace{\mathbf{c} \mid \forall \mathbf{x} \in \mathbb{X}, g(\mathbf{c}, \mathbf{x}) \leqslant 0}_{\mathbb{C}} \} \tag{21}$$

The min can be computed in a set-theoretic way:



Minimizing amounts to projecting. The set $\mathbb{C}$ corresponds to the epigraph of a function to be minimized.

Using separators:

$$\mathcal{S}_4 := \text{proj}_3(\mathcal{S}_{\mathbb{C}}) \tag{22}$$

$$\mathcal{C}_{c_3^*} := \text{inner}(\mathcal{S}_4) \cap \text{outer}(\mathcal{S}_4) \tag{23}$$

We obtain a contractor $\mathcal{C}_{c_3^*}$ for the radius of the smallest enclosing circle.
A contractor for the center $\mathbf{c}_{1,2}^*$ is obtained in a second step.

Example of Codac implementation in Python.
http://codac.io/v2

Codac code for approximating the radius $[r]$ of the smallest enclosing circle of a set described by a separator $\mathbb{S}_{\mathbb{X}}$:

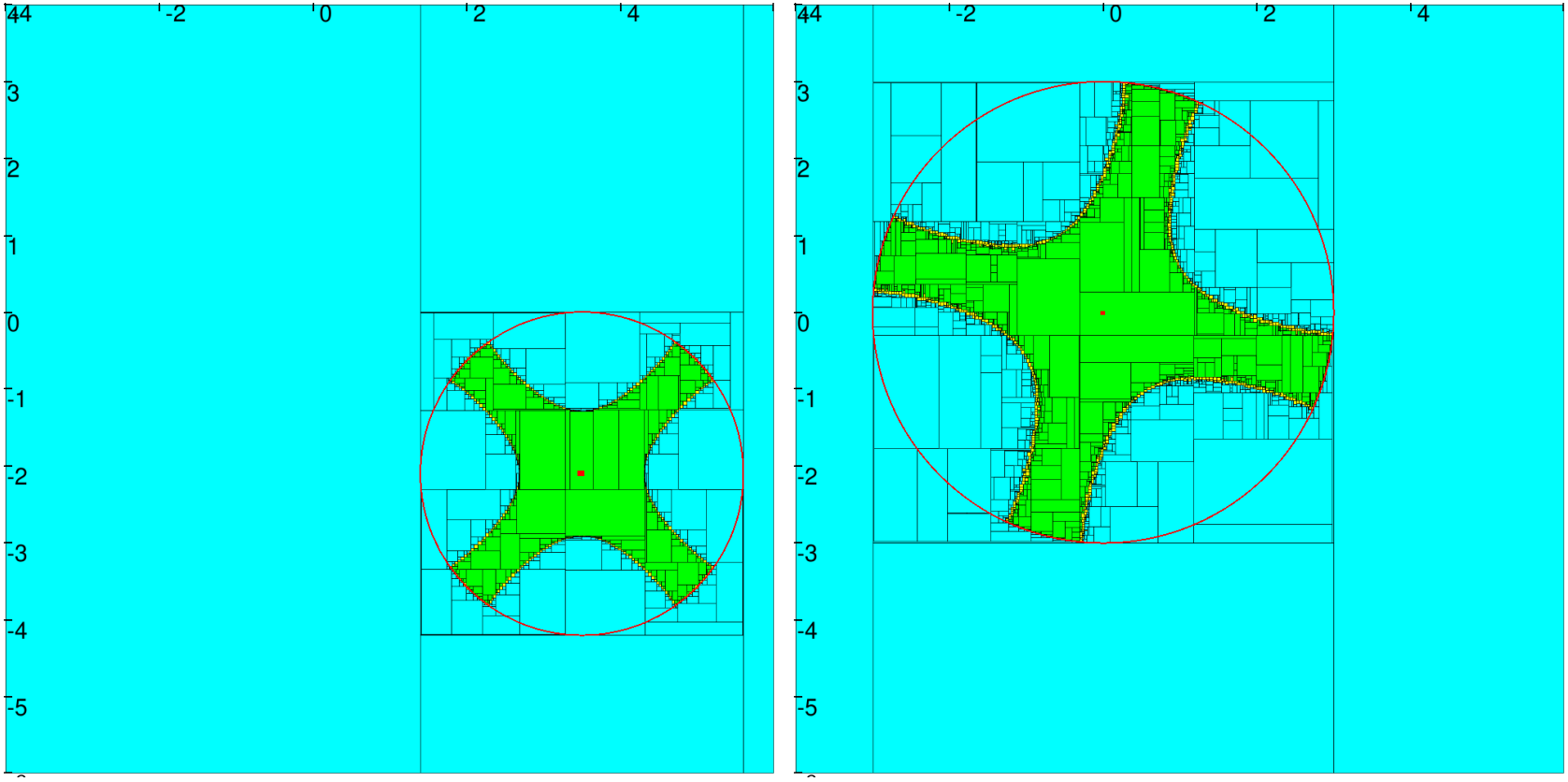
```
from codac import *

k = VectorVar(5)
g = AnalyticFunction(
  [k], sqr(k[3]-k[0])+sqr(k[4]-k[1])-sqr(k[2])
)

Sx = ... # given a set X described by a separator
S1 = SepInverse(g, [0,oo])                          # dim 5
f_proj = AnalyticFunction([k], [k[3],k[4]])
S2 = S1 & SepInverse(f_proj, Sx)                    # dim 5
S3 = SepProj(S2, [0,1,2])                           # dim 3
Sc = ~S3                                            # dim 3
S4 = SepProj(Sc, [0])
Sc3 = S4 & ~S4                                      # dim 1

p = pave([[0,oo]], Sc3, 1e-2)
r = hull(p.boxes(PavingInOut.outer))
```

Step A: results

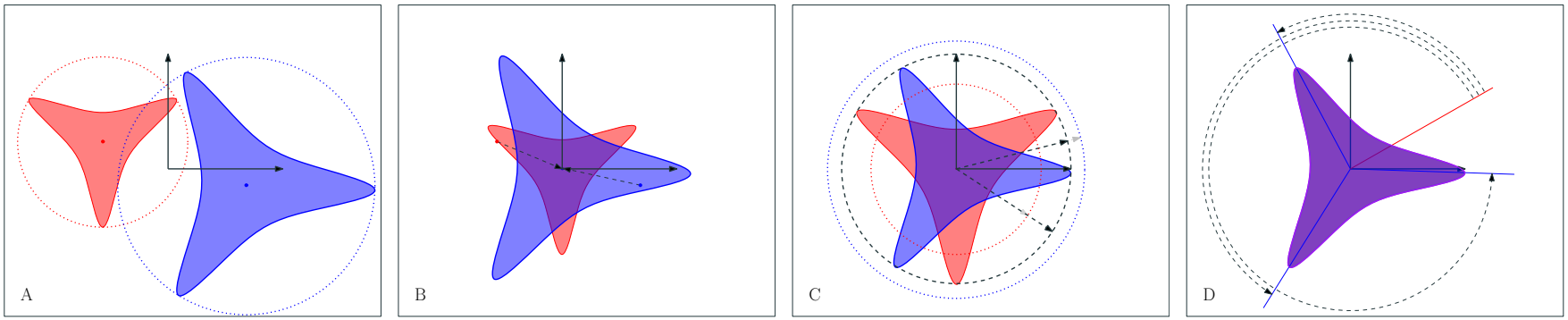


(a) *Smallest enclosing circles of* $\mathbb{A}$.

(b) *Smallest enclosing circles of* $\mathbb{B}$.

Steps B – C: translation and scaling

Recall the strategy:



Operations:

A1. $\mathbf{c}^{\mathbb{A}} = \texttt{MinEnclosingCircle}(\mathbb{A})$

A2. $\mathbf{c}^{\mathbb{B}} = \texttt{MinEnclosingCircle}(\mathbb{B})$

B1,C1. $\mathbb{A}_N = \left(\mathbb{A} - \mathbf{c}^{\mathbb{A}}_{1,2}\right) / c^{\mathbb{A}}_3$

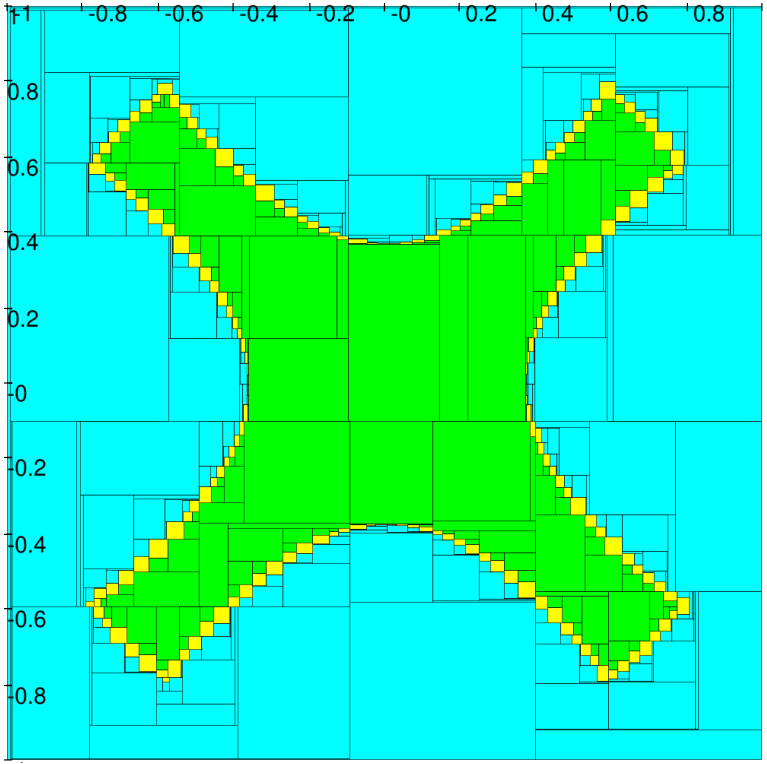
B2,C2. $\mathbb{B}_N = \left(\mathbb{B} - \mathbf{c}^{\mathbb{B}}_{1,2}\right) / c^{\mathbb{B}}_3$

D. $\mathbb{B}_N = \mathbf{R}_\theta \cdot \mathbb{A}_N$

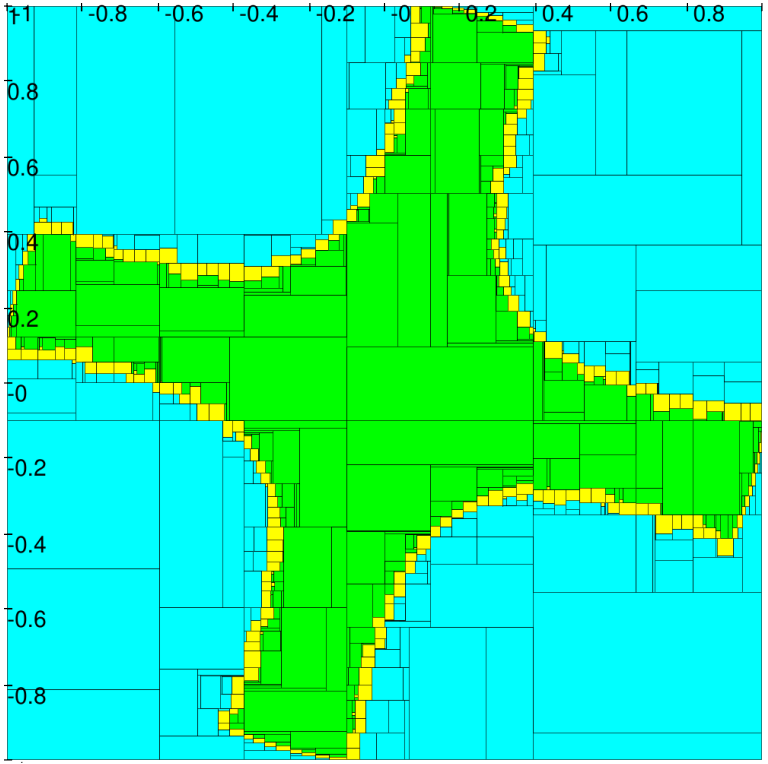
$k = c^{\mathbb{B}}_3 / c^{\mathbb{A}}_3$

$\mathbf{t} = \mathbf{c}^{\mathbb{B}}_{1,2} - k\mathbf{R}_\theta \cdot \mathbf{c}^{\mathbb{A}}_{1,2}$

Steps B – C: results



(a) *Translation and scaling of* $\mathbb{A}$.



(b) *Translation and scaling of* $\mathbb{B}$.

Step D: rotational mapping $\boxed{\mathbb{B}_N = \mathbf{R}_\theta \cdot \mathbb{A}_N}$

Recall that we want to approximate Θ:

$$\Theta := \{\theta \mid \mathbb{B}_N = \mathbf{R}_\theta \cdot \mathbb{A}_N\}\,. \tag{24}$$

Due to the numerical errors of Steps A (enclosure of $\mathbf{c}^*$), only outer approximations of $\mathbb{A}_N$ and $\mathbb{B}_N$ are available:

$$\mathbb{A}_N \in [\mathbb{A}_N^-, \mathbb{A}_N^+] \quad \text{and} \quad \mathbb{B}_N \in [\mathbb{B}_N^-, \mathbb{B}_N^+] \tag{25}$$

Step D: rotational mapping $\boxed{\mathbb{B}_N = \mathbf{R}_\theta \cdot \mathbb{A}_N}$

Recall that we want to approximate Θ:

$$\Theta := \{\theta \mid \mathbb{B}_N = \mathbf{R}_\theta \cdot \mathbb{A}_N\}\,. \tag{24}$$

Due to the numerical errors of Steps A (enclosure of $\mathbf{c}^*$), only outer approximations of $\mathbb{A}_N$ and $\mathbb{B}_N$ are available:

$$\mathbb{A}_N \in [\mathbb{A}_N^-, \mathbb{A}_N^+] \quad \text{and} \quad \mathbb{B}_N \in [\mathbb{B}_N^-, \mathbb{B}_N^+] \tag{25}$$

Therefore, Step D is dealt with a double inclusion:

$$\left.\begin{aligned} \mathbb{B}_N &\supseteq \mathbf{R}_\theta \cdot \mathbb{A}_N \\ \mathbb{A}_N &\supseteq \mathbf{R}_{-\theta} \cdot \mathbb{B}_N \end{aligned}\right\} \implies \mathbb{B}_N^+ \supseteq \mathbb{B}_N \supseteq \mathbf{R}_\theta \cdot \mathbb{A}_N \supseteq \mathbf{R}_\theta \cdot \mathbb{A}_N^- \tag{26}$$

Step D: rotational mapping $\boxed{\mathbb{B}_N = \mathbf{R}_\theta \cdot \mathbb{A}_N}$

Recall that we want to approximate Θ:

$$\Theta := \{\theta \mid \mathbb{B}_N = \mathbf{R}_\theta \cdot \mathbb{A}_N\}. \tag{24}$$

Due to the numerical errors of Steps A (enclosure of $\mathbf{c}^*$), only outer approximations of $\mathbb{A}_N$ and $\mathbb{B}_N$ are available:

$$\mathbb{A}_N \in [\mathbb{A}_N^-, \mathbb{A}_N^+] \quad \text{and} \quad \mathbb{B}_N \in [\mathbb{B}_N^-, \mathbb{B}_N^+] \tag{25}$$

Therefore, Step D is dealt with a double inclusion:

$$\left.\begin{array}{l} \mathbb{B}_N \supseteq \mathbf{R}_\theta \cdot \mathbb{A}_N \\ \mathbb{A}_N \supseteq \mathbf{R}_{-\theta} \cdot \mathbb{B}_N \end{array}\right\} \implies \mathbb{B}_N^+ \supseteq \mathbb{B}_N \supseteq \mathbf{R}_\theta \cdot \mathbb{A}_N \supseteq \mathbf{R}_\theta \cdot \mathbb{A}_N^- \tag{26}$$

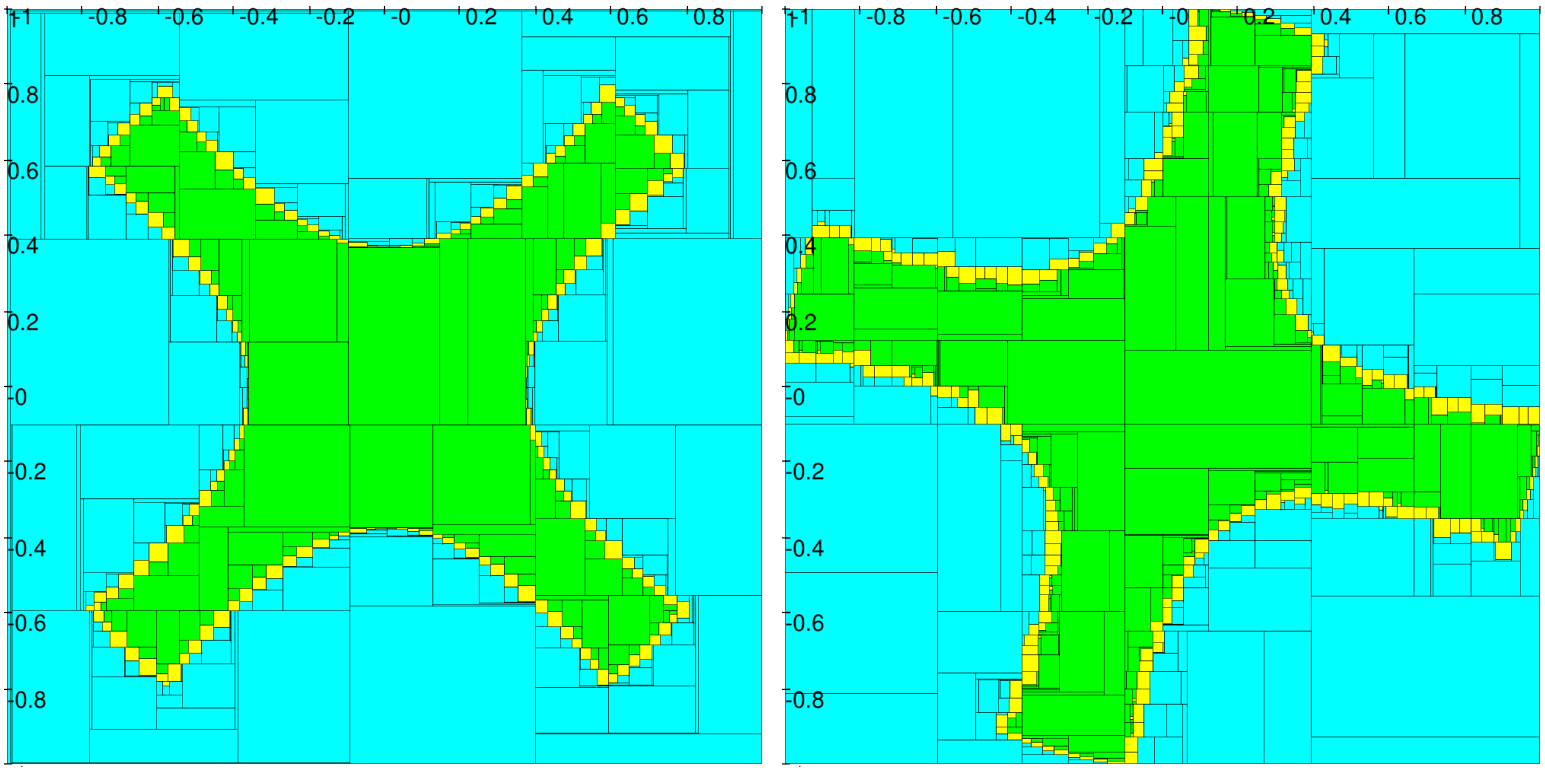
$$\boxed{\mathbb{B}_N = \mathbf{R}_\theta \cdot \mathbb{A}_N \implies \mathbb{B}_N^+ \supseteq \mathbf{R}_\theta \cdot \mathbb{A}_N^-} \tag{27}$$

Step D: polar coordinates

Sets are transformed into **polar coordinates**.

Easier to compute the last step: in polar coordinates, the rotation is a sum:

$$\mathrm{R}_\theta \cdot \begin{pmatrix} \rho \\ \phi \end{pmatrix} = \begin{pmatrix} \rho \\ \phi + \theta \end{pmatrix}. \tag{28}$$



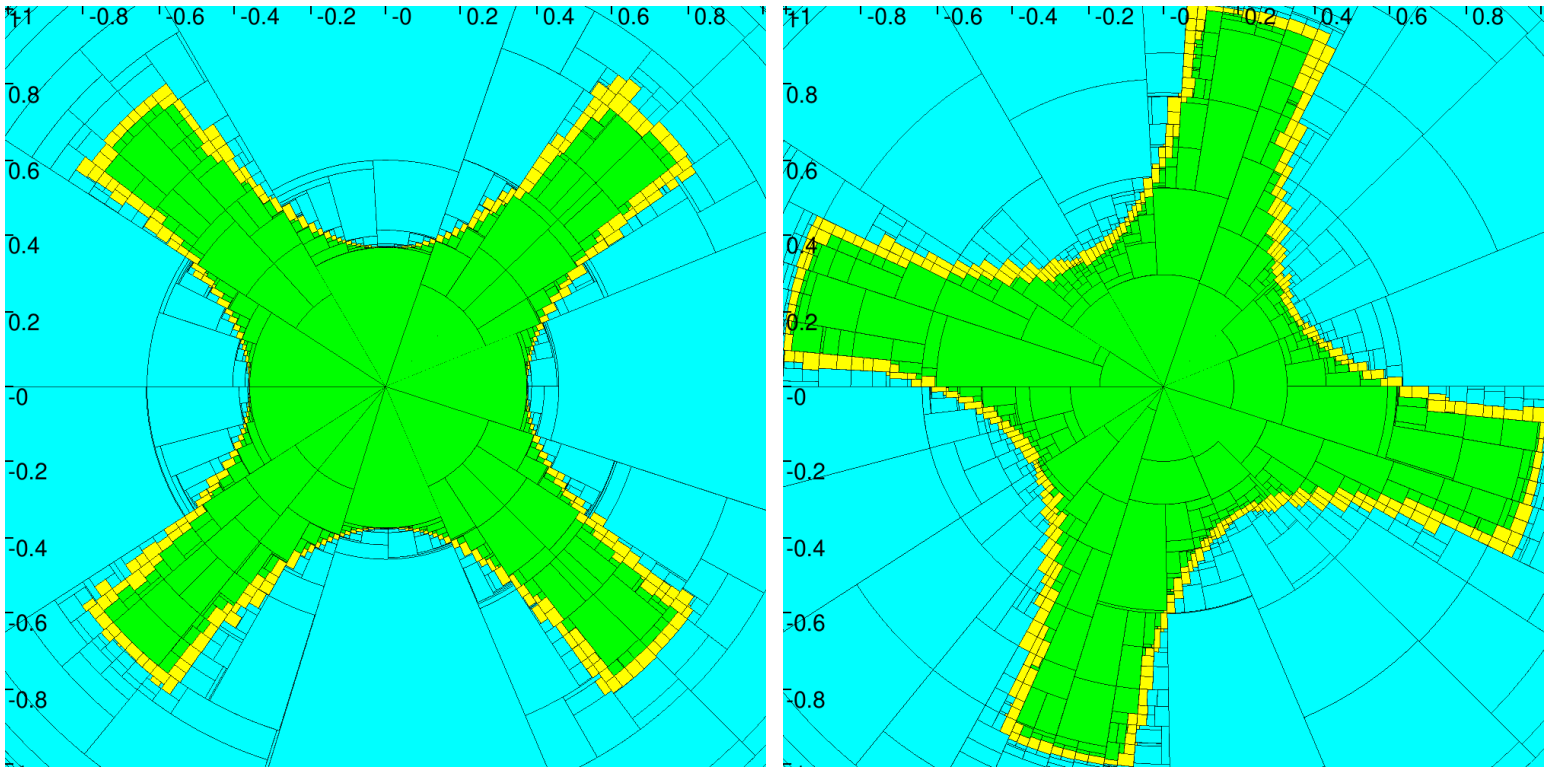
(a) *Translation and scaling of* $\mathbb{A}$. ***(b)*** *Translation and scaling of* $\mathbb{B}$.

Step D: polar coordinates

Sets are transformed into **polar coordinates**.

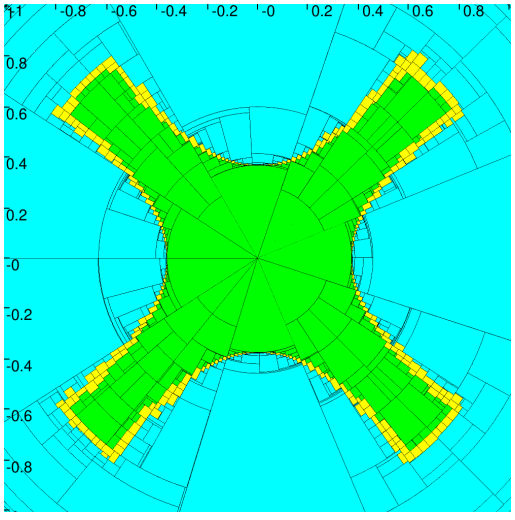
Easier to compute the last step: in polar coordinates, the rotation is a sum:

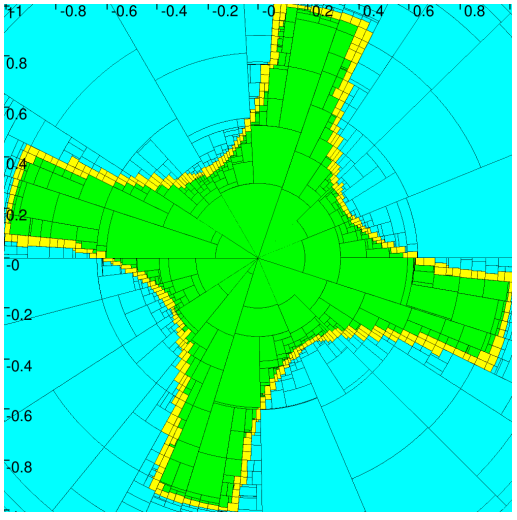
$$\mathrm{R}_\theta \cdot \begin{pmatrix} \rho \\ \phi \end{pmatrix} = \begin{pmatrix} \rho \\ \phi + \theta \end{pmatrix}. \tag{28}$$

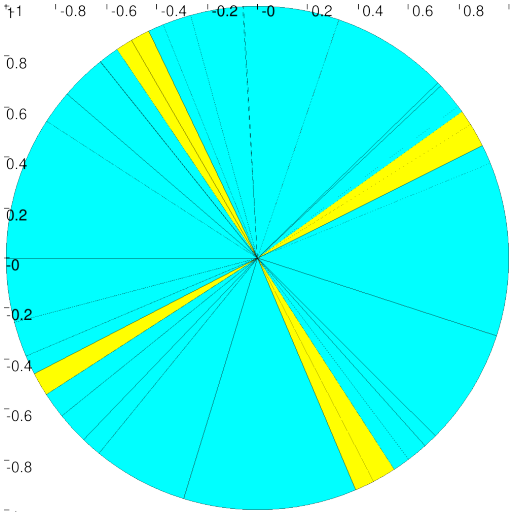


(a) *Translation and scaling of* $\mathbb{A}$. ***(b)*** *Translation and scaling of* $\mathbb{B}$.

Step D: results







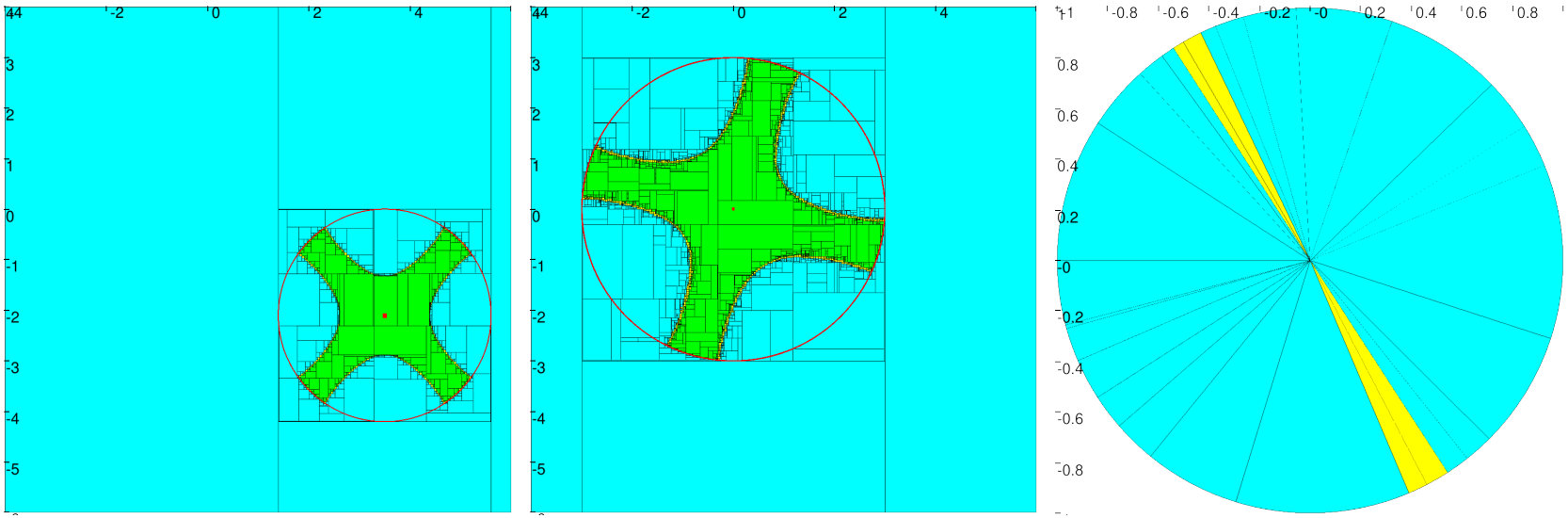
$\mathbb{A}^{-}_{N,polar}$ $\ominus$ $\mathbb{B}^{+}_{N,polar}$ $\supset$ Θ

The symmetries related to the 4 arms of the cross have been approximated.

Singular examples

Stretched cross

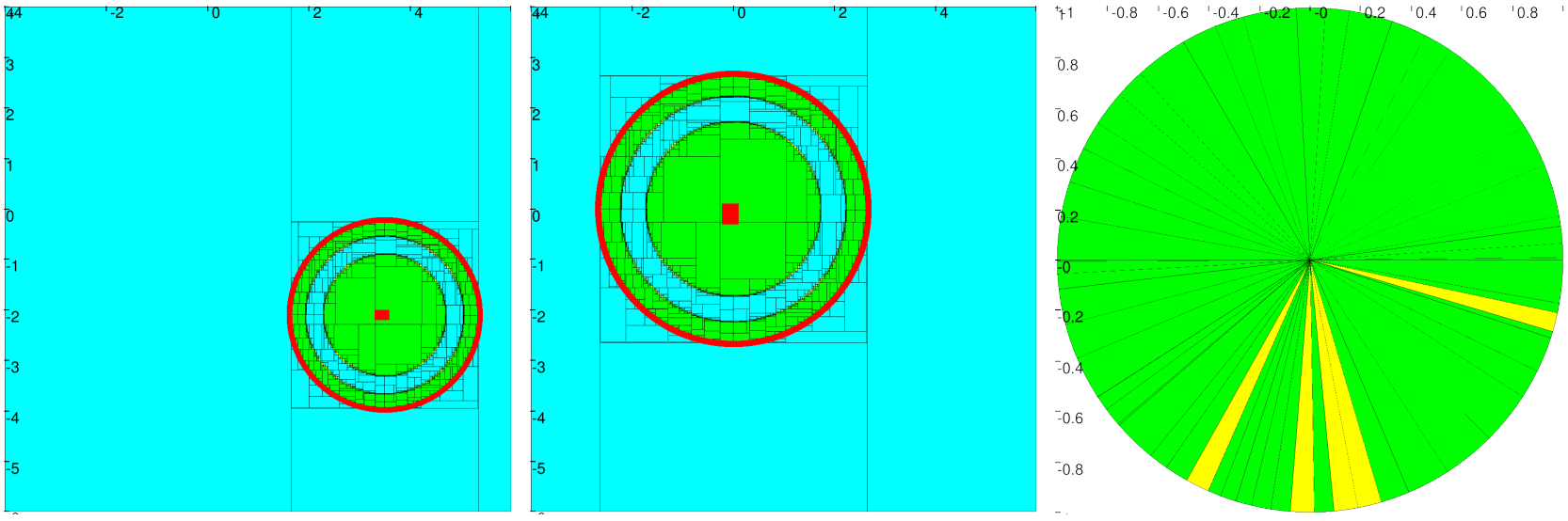
The arms of the cross are identical in pairs, but not all four are the same.



$\Longrightarrow$ the method is reliable: it avoids false similarity transformations.

Rings

Example of infinite possibilities of transformations:



$\Longrightarrow$ the method is exhaustive: no possible transformation can be lost.

Conclusion

Separators combined with a Procrustes approach:

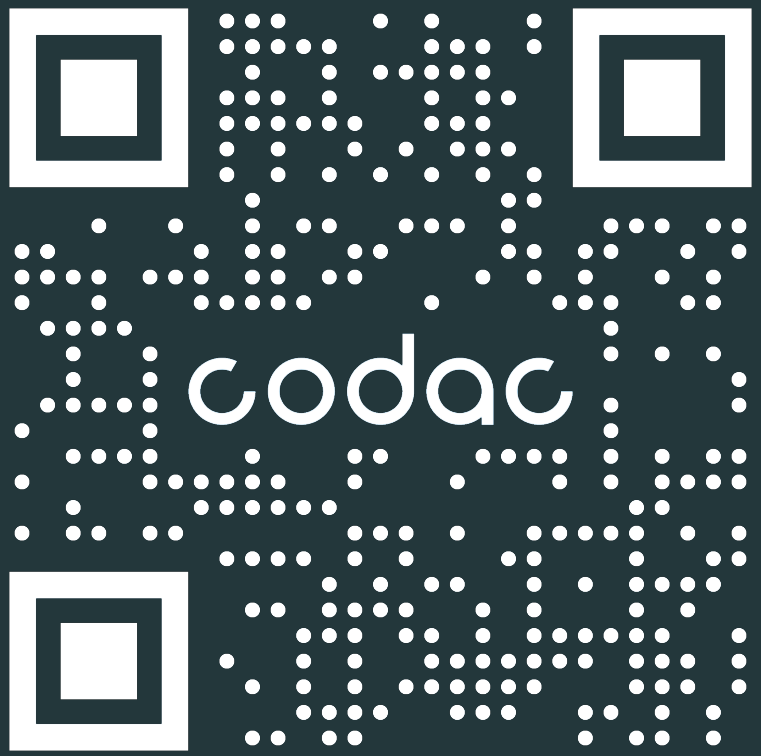
- difficult problem: 4 parameters
- reliability and exhaustiveness
- separator approach: elegance and direct translation of set operations

Limits:

- our method only applies for bounded sets

Perspectives:

- deal with thick sets as inputs: $[\mathbb{A}]$ and $[\mathbb{B}]$
- deal with 3d shapes (and so, 2d rotations)



http://codac.io/v2

Appendice

Results (timeout: 1 hour)

Cross:

Prec.	radii			centers			rotations			Total
	time	# ctr	unc. size	time	# ctr	unc. size	time	# ctr	unc. size	time
0.1	2.80s	6	4.3%	0.60s	74	13.1%	20.7s	105	49.0%	24.1s
0.01	15.3s	2	0.40%	1.45s	100	1.75%	31.4s	173	6.0%	48.1s
0.001	90.7s	6	0.043%	1.77s	100	0.10%	102.3s	173	0.40%	196.5s

Stretched Cross:

Prec.	radii			centers			rotations			Total
	time	# ctr	unc. size	time	# ctr	unc. size	time	# ctr	unc. size	time
0.1	6.80s	10	5.2%	0.58s	76	15.4%	18.7s	109	53.3%	26.1s
0.01	32.5s	10	0.51%	1.57s	82	2.89%	35.8s	113	3.37%	69.9s
0.001	110s	8	0.040%	4.02s	108	0.21%	18.5s	251	1.02%	132.4s

Disk and ring:

Prec.	radii			centers			rotations			Total
	time	# ctr	unc. size	time	# ctr	unc. size	time	# ctr	unc. size	time
0.1	3.48s	4	4.78%	0.63s	38	15.0%	45.4s	163	100%	49.6s
0.01	40.7s	4	0.46%	3.53s	38	0.87%	3484s	1895	100%	3528s
0.001	262s	4	0.046%	23.13s	42	0.087%		timeout		∞

Projection of a separator

Also, the projection of a separator is a separator.

