OpenConMap: A Matlab Toolbox for Mapping the Interior of the Unit Circle to the Exterior of Simple Closed Curves

——User's Guide

Developer: Kai He
kai.hee@foxmail.com

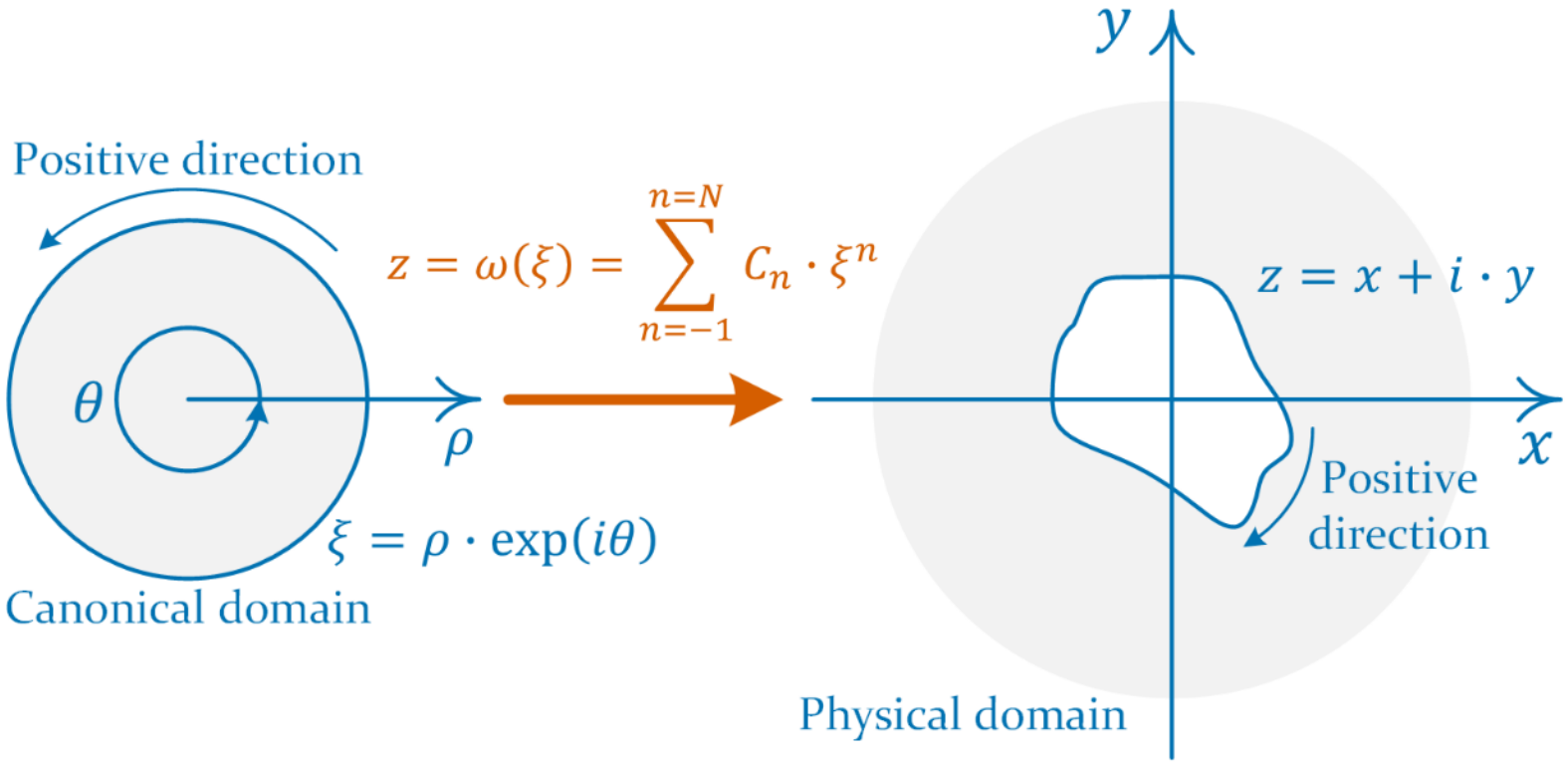
2024.2.1

Version Description

Version	Date	Update
1	2024.2.1	-
1.1	2024.5.25	Added more examples.

Introduction

This toolbox solves the conformal mapping function $\omega(\xi)$ that maps the interior of the unit circle (Canonical domain) on the ξ-plane to the exterior of an irregular simple closed curve (Physical domain) on the z-plane. The toolbox is particularly suitable for situations where boundaries contain curves.



Condition

- Schwarz–Christoffel Toolbox needs to be installed.

  Path: https://github.com/tobydriscoll/sc-toolbox

  User's Guide: https://tobydriscoll.net/project/sc-toolbox/

- Matlab version: 2023a (or later). For previous versions, it may also be applicable, but I do not guarantee this.

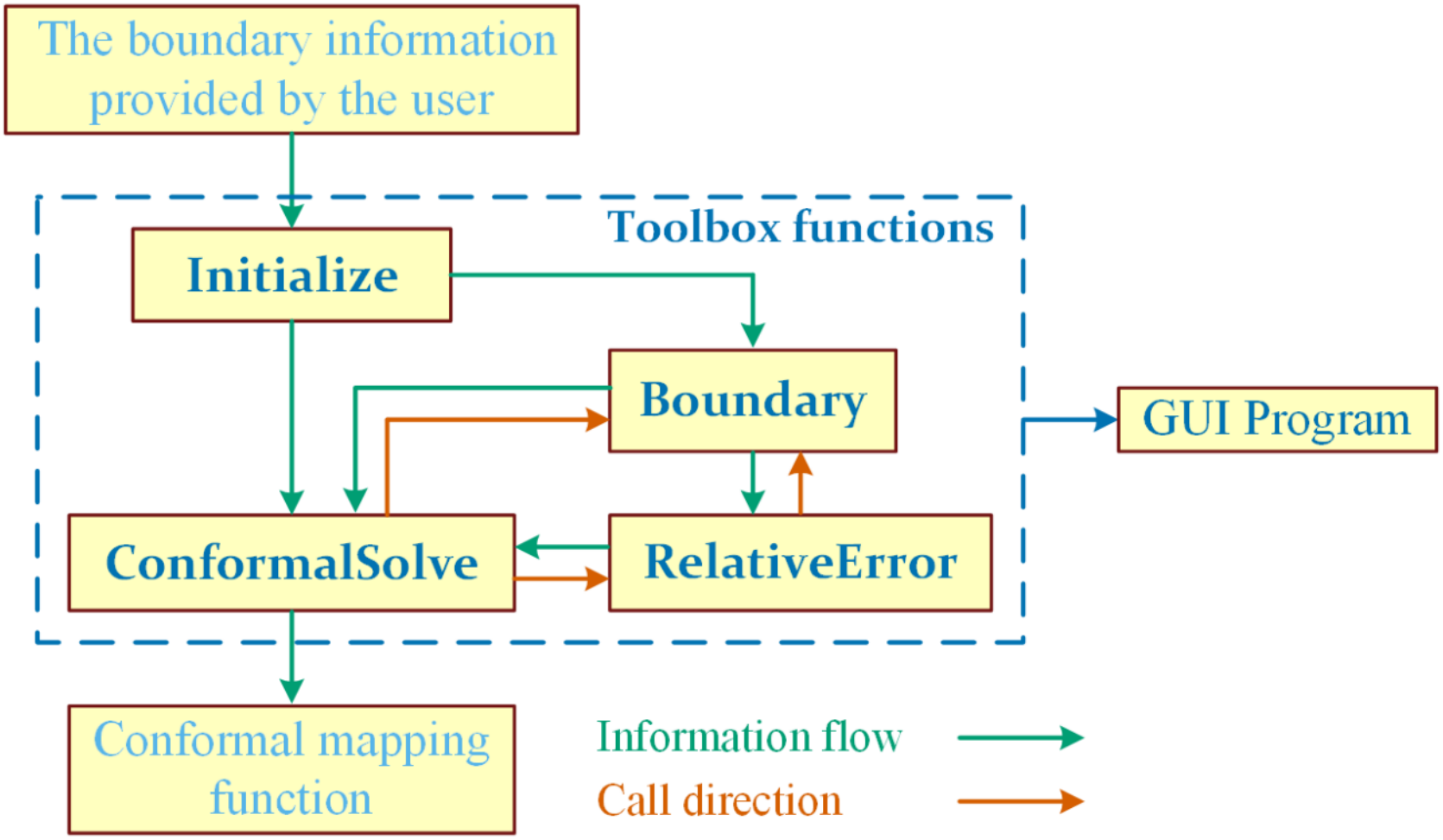
Architecture

At present, the architecture of OpenConMap toolbox is very simple, containing only 4 functions. I will expand the functionality of OpenConMap toolbox in the future. so, some functions may seem a bit cumbersome. All functions are as follows:

- z = Initialize(F, X, N)
- z = Boundary(s, p, filename)
- err = RelativeError(C)
- [C, err] = ConformalSolve(z, N1, N2, N3, Lambda, rate)

The toolbox also includes some examples to demonstrate the usage of the toolbox.

Architecture



Initialize

The function **Initialize** is used to convert user provided information into data that can be used by the algorithm. Three sets of data were generated in total. The first and second sets of data are automatically saved in the current directory with file names *Boundary_data1.mat* and *Boundary_data2.mat*, which are used by the function **Boundary**. The third set of data is the vertices of the polygon approximation method.

Syntax:

- z = Initialize(F, X)
- z = Initialize(F, X, N)

Initialize — Input Arguments

The boundary is usually described by multiple functions (usually including at least two functions). Write these functions in the form of anonymous functions in a counterclockwise direction into variable F, where F is a cell. Write the starting points of these functions in complex form into variable X. The first function in F is arbitrary, but the point in X must correspond to the function in F.

Remarks:

- For different F and X, the solution of the conformal mapping function obtained is different. And the solution effect may also vary.
- F only accepts functions with x or y as variables. When a certain boundary can be represented as either y=f(x) or x=g(y), these two representation methods are equivalent.

Initialize — Input Arguments

For **Examples 1**, the following writing methods are all correct.

- ```
  F={@(y) 0.5*(1-y.^2).^0.5, @(x) -1, @(y) -1, @(x) x+1};
  X=[i, -i, -1-i, -1];
  ```
- ```
  F={@(x) -1, @(y) -1, @(x) x+1, @(y) 0.5*(1-y.^2).^0.5};
  X=[-i, -1-i, -1, i];
  ```
- ```
  F={@(y) -1, @(x) x+1, @(y) 0.5*(1-y.^2).^0.5, @(x) -1};
  X=[-1-i, -1, i, -i];
  ```
- ```
  F={@(x) x+1, @(y) 0.5*(1-y.^2).^0.5, @(x) -1, @(y) -1};
  X=[-1, i, -i, -1-i];
  ```

For **Examples 1**, the solution effect of the following two representation methods is the same.

- ```
  F={@(y) 0.5*(1-y.^2).^0.5, @(x) -1, @(y) -1, @(x) x+1};
  X=[i, -i, -1-i, -1];
  ```
- ```
  F={@(y) 0.5*(1-y.^2).^0.5, @(x) -1, @(y) -1, @(y) y-1};
  X=[i, -i, -1-i, -1];
  ```

Initialize — Polygon approximation method

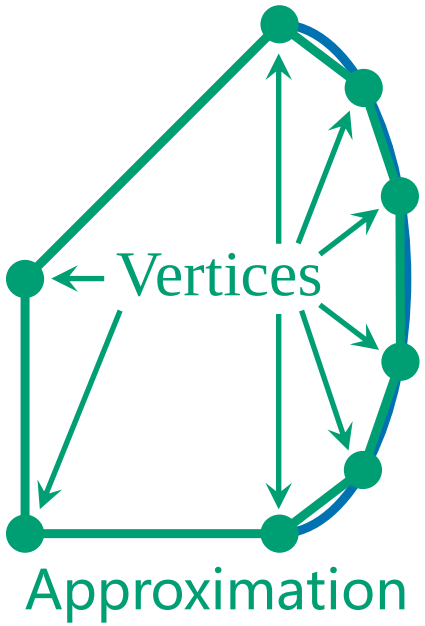
Using appropriate methods to estimate the coefficient C_n reasonably can greatly improve the convergence speed and accuracy of the iteration process. The algorithm uses polygon approximation method for initial value estimation. The principle of polygon approximation is to select an appropriate number of points on the boundary, which form a polygon. The conformal mapping function of this polygon boundary can be solved using the Schwarz-Christoffel mapping formula. Consider this mapping function as the initial value. The number of vertices in this polygon is N, and it is appropriate for this value to be greater than the number of starting points. Polygon vertices include points in X. The remaining points are distributed at positions with larger curve curvature. z is the coordinates of the vertexs of the polygon.

Initialize — Polygon approximation method

The specific location of vertices is determined as follows:

$$\int_{\Gamma} \kappa \mathrm{d}s = (N - M) \cdot \int_{\mathrm{A}}^{\mathrm{B}} \kappa \mathrm{d}s$$

where, κ represents curvature, N is the number of vertices in a polygon, M is the number of points in X. A and B represent any two adjacent vertices, except for the points in X. When N is not specified, N is the number of points in X.



Boundary

The function **Boundary** calculates the points on the boundary corresponding to the independent variable s to describe the shape of the boundary. Boundary provides two modes for describing the shape of boundaries. The first method takes the normalized arc length parameter as the independent variable, and the second method takes the polar angle of the corresponding point on the boundary as the independent variable. When using the first method, the starting point of the boundary is the first point in the **Initialize** parameter X. The normalization method is to divide the arc length from the point on the boundary to the starting point by the length of the boundary. When using the second method, the starting point of the boundary is the intersection of the boundary and the positive x-axis. Both of these calculation methods take counterclockwise as the positive direction.

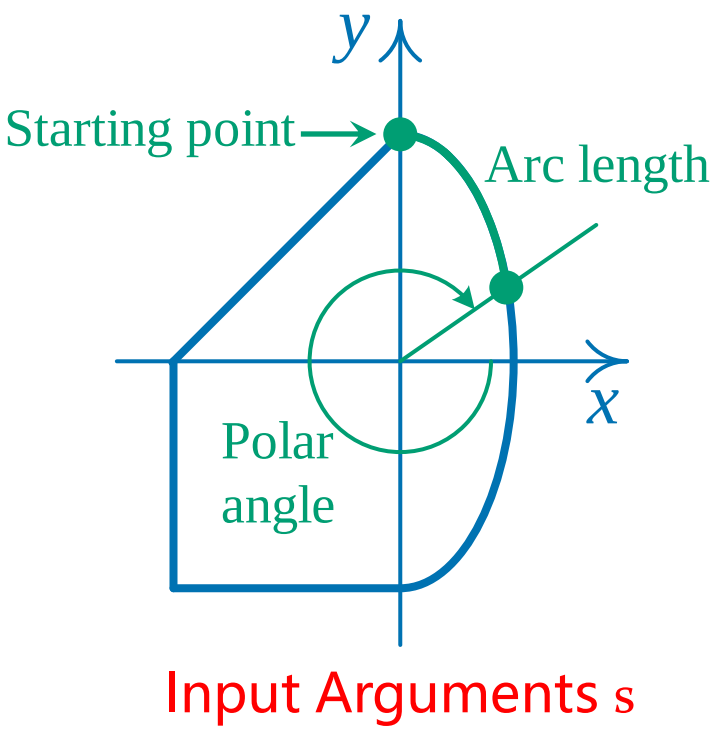
Boundary — Syntax & Output Arguments

Syntax:

- z = Boundary(s)
- z = Boundary(s, p)
- z = Boundary(s, p, name)

Output Arguments:

The function **Boundary** has only one output argument, which is the coordinate of the point corresponding to the independent variable s, and is a complex number.



Boundary — Input Arguments

The function **Boundary** has three input parameters, with the first parameter being the independent variable s, which takes the normalized arc length or polar angle as its value, depending on the calculation mode. The second parameter specifies the calculation mode, with values of C or P, and the default value is C. When the value is C, calculate the coordinates of the point based on the normalized arc length. When the value is P, calculate the coordinates of the point based on the polar angle. The third parameter is a file name, and by default, the value is determined by the parameter p. When setting the file name, the Boundary function calculates the position of points on the boundary based on the data provided in this file.

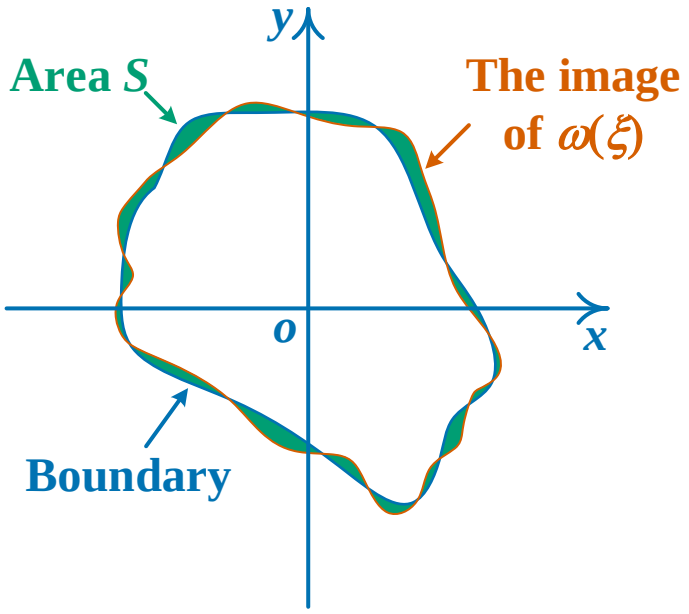
RelativeError

The function **RelativeError** is used to estimate the accuracy of the conformal mapping function obtained from the solution. The input parameter is the coefficient C of the Laurent series. The relative error of the coefficient C as the output parameter is defined as:

$$\text{err} = \frac{S}{L}$$

where, S is the area enclosed by the image of $\omega(\xi)$ and the boundary, and L is the length of the boundary.

Syntax: err = RelativeError(C)



Relative error

ConformalSolve

The function **ConformalSolve** is the main solution function. The detailed description and selection method of these parameters can be found in reference — Iterative algorithm for the conformal mapping function from the exterior of a roadway to the interior of a unit circle.

ConformalSolve

Syntax: [C, err] = ConformalSolve(z, N1, N2, N3, Lambda, rate)

Input Arguments:

- z: the vertices of polygons in polygon approximation method.
- N1: the maximum number of iterations of the algorithm.
- N2: the order of Laurent series.
- N3: the number of corresponding points is recommended to be 3-8 times the order.
- Lambda: a coefficient between 0 and 1, which is related to the convergence speed of the algorithm.
- rate: when the rate of error change is less than rate, the algorithm converges.

Output Arguments:

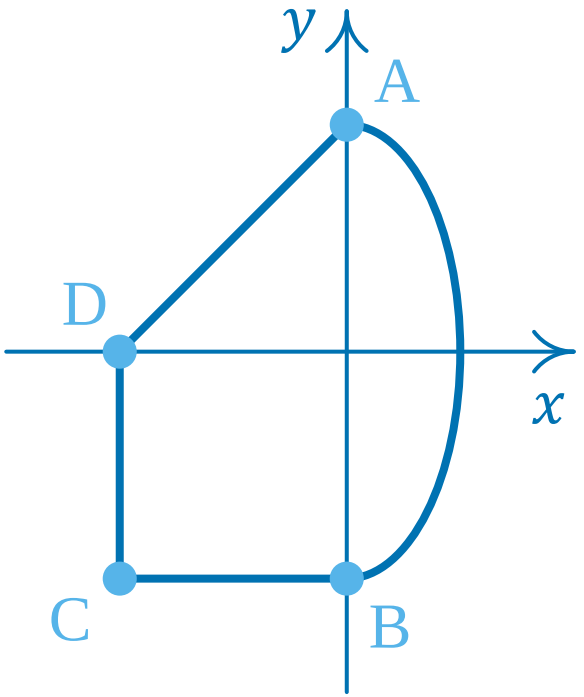
- C: The solution obtained by the algorithm.
- err: Relative error during algorithm iteration process.

Examples 1

The boundary of **Examples 1** has some special characteristics. This includes both straight and curved segments. At the same time, it also includes line segments that can only be represented by x or y, as well as line segments that can be represented by both x and y. This example illustrates that the algorithm is applicable to different boundary types.

Segment	Expression
AB	$x = 0.5 \cdot \sqrt{1 - y^2}$
BC	$y = -1$
CD	$x = -1$
DA	$y = x + 1$ or $x = y - 1$

Point	Coordinate
A	$0 + i$
B	$0 - i$
C	$-1 - i$
D	-1



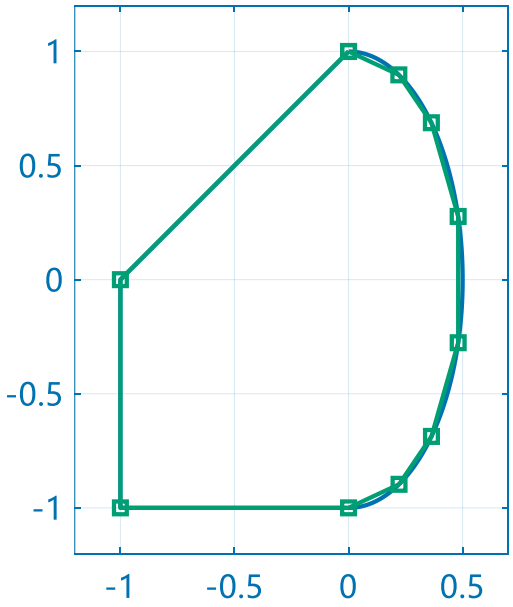
Boundary shape

Examples 1

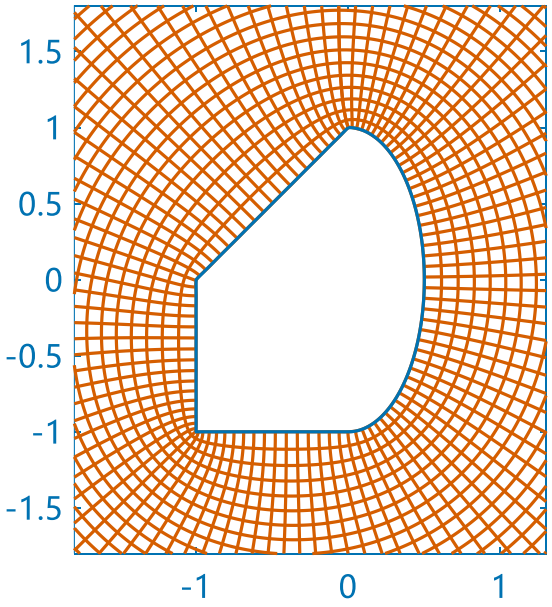
Core Code:

```
F={@(y) 0.5*(1-y.^2).^0.5, @(x) -1, @(y) -1, @(x) x+1};
X=[i, -i, -1-i, -1];
z = Initialize(F, X, 10);
[C, err] = ConformalSolve(z, 50, 30, 200, 0.6, 1e-3);
```

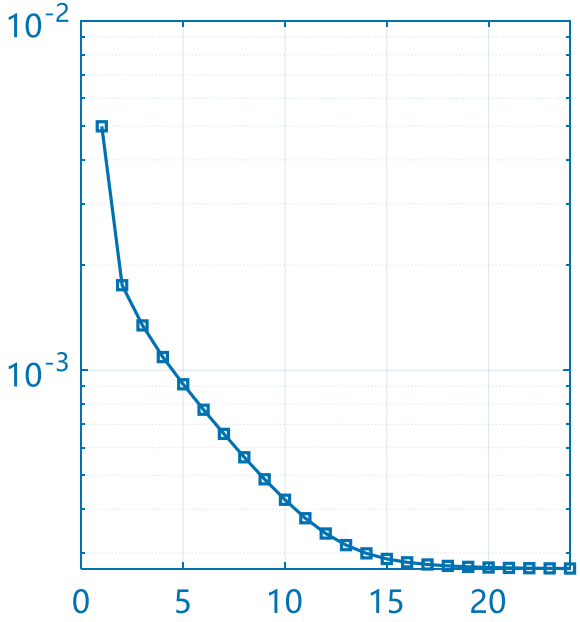
Results:



Polygon

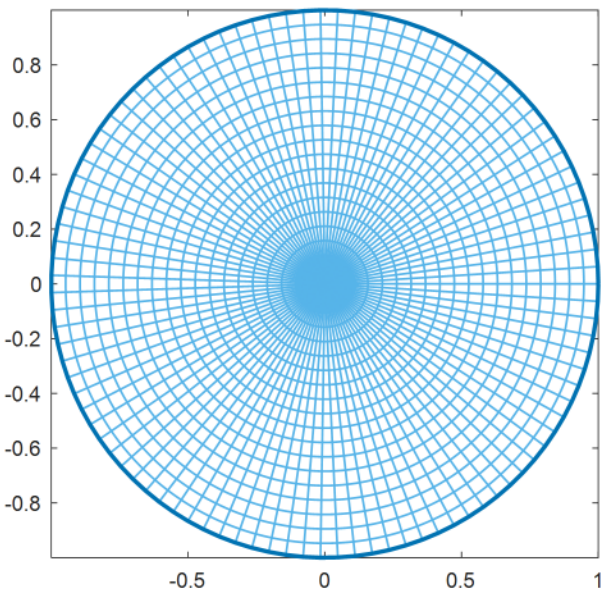


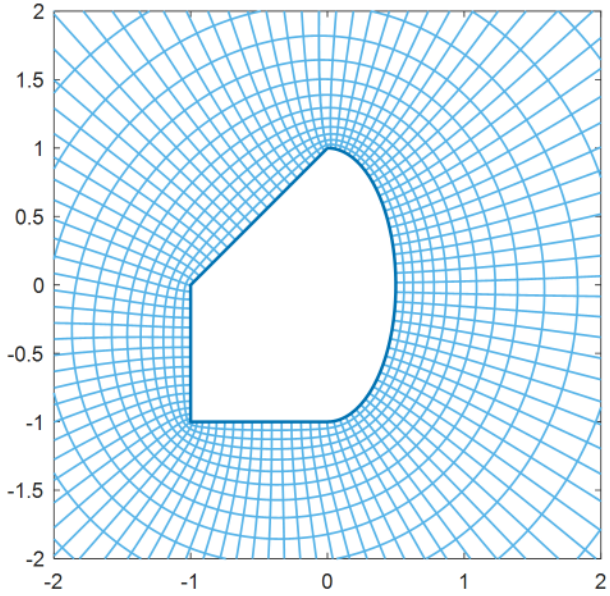
Mapping

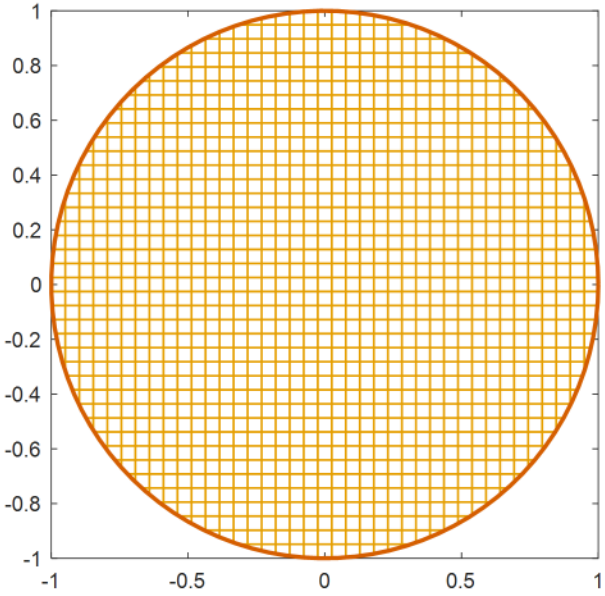


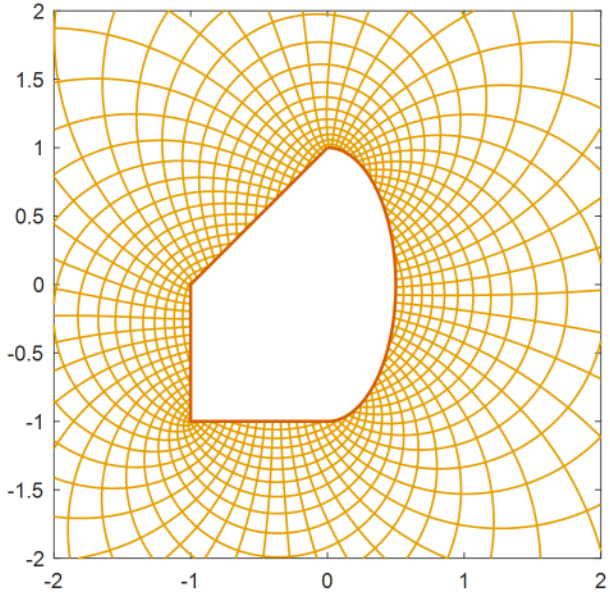
Relative error

Examples 1


0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-0.5
0
0.5
1

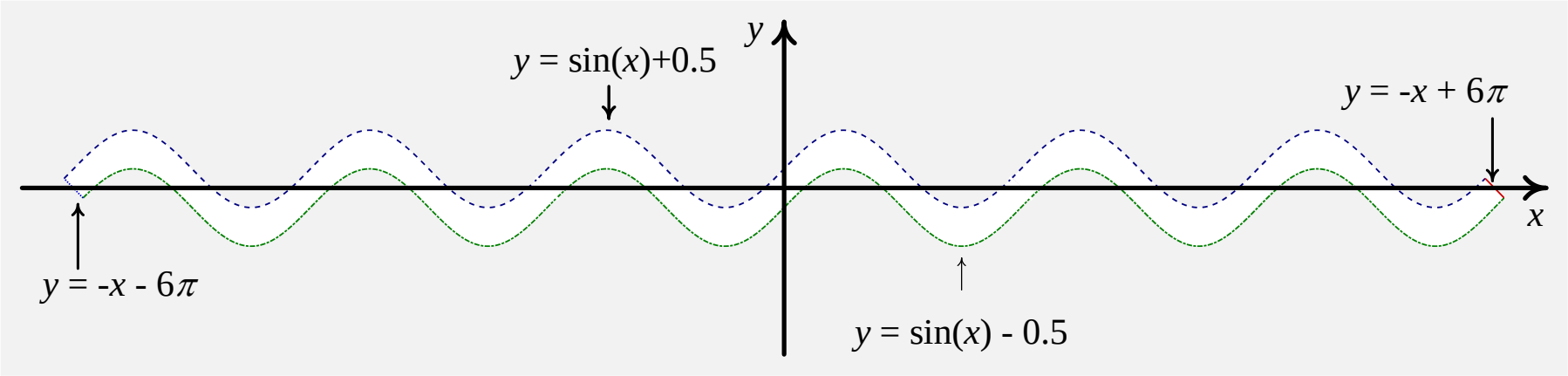

2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-2
-1
0
1
2


1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-1
-0.5
0
0.5
1


2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-2
-1
0
1
2

Examples 2

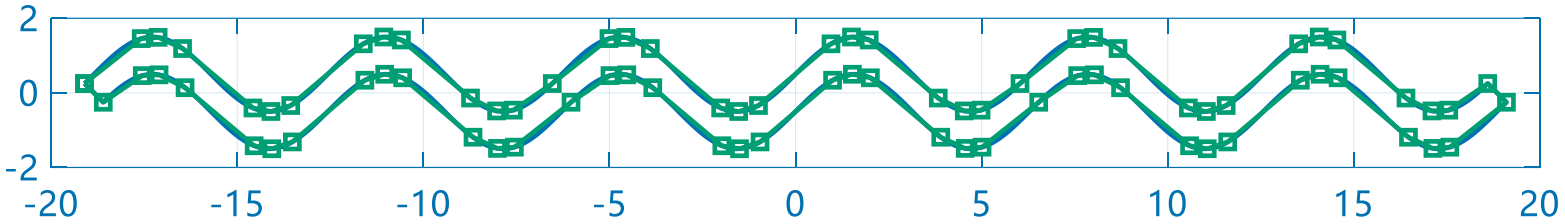
Examples 2 shows a curved narrow orifice. This example indicates that this toolbox can solve some more complex hole shapes.



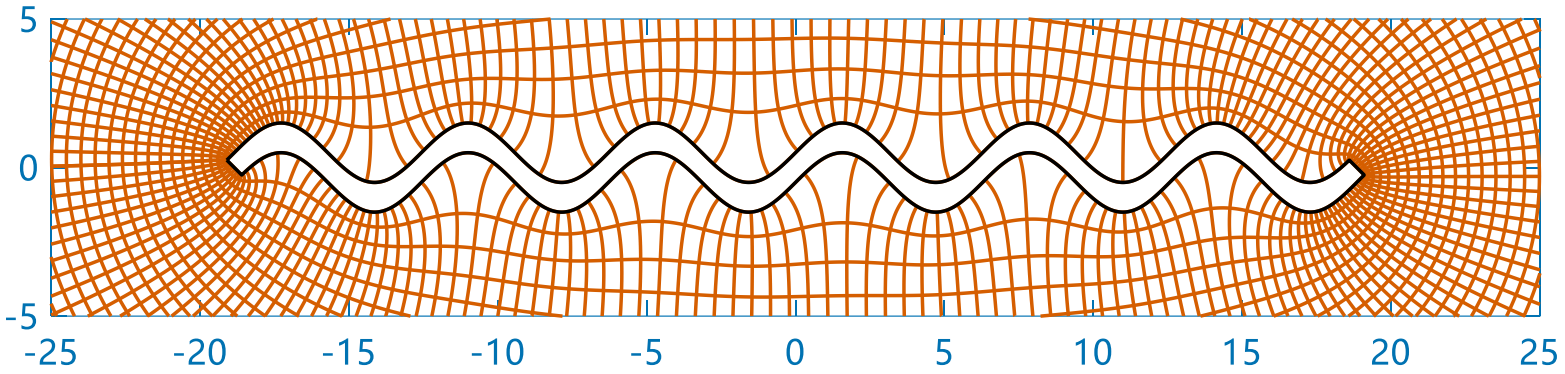
Main Code:

```
F={@(x) sin(x)+0.5, @(x) -x+6*pi, @(x) sin(x)-0.5, @(x) -x-6*pi};
X=[-19.100874546062855 + 0.251318624524098i
    18.598237297014663 + 0.251318624524097i
    19.100874546062855 - 0.251318624524098i
   -18.598237297014663 - 0.251318624524096i]';
z=BoundaryInitial(F,X,80);
[C,epsilon]=ConformalSolve(z,1000,500,4000,0.6,1e-4);
```

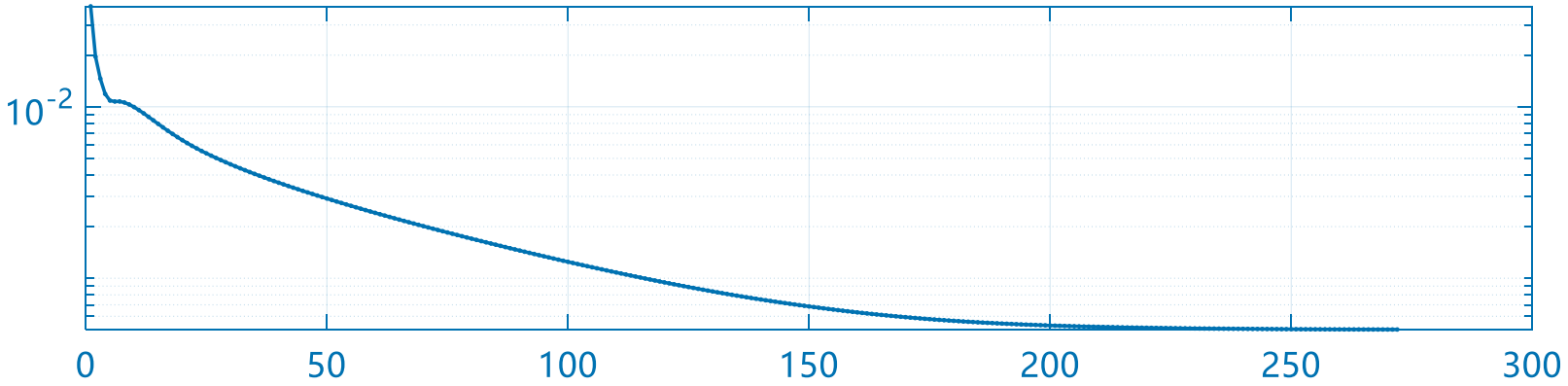
Examples 2 — Results



Polygon



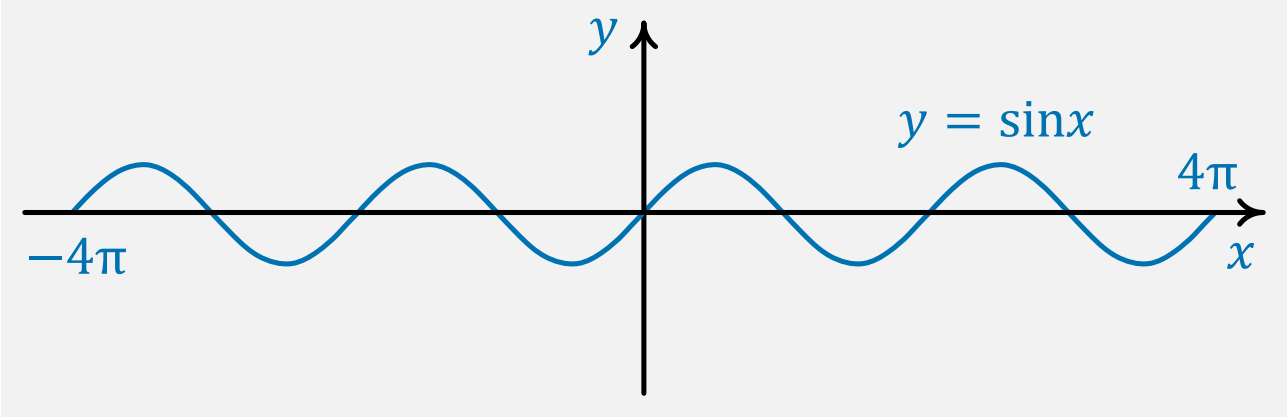
Mapping



Relative error

Examples 3

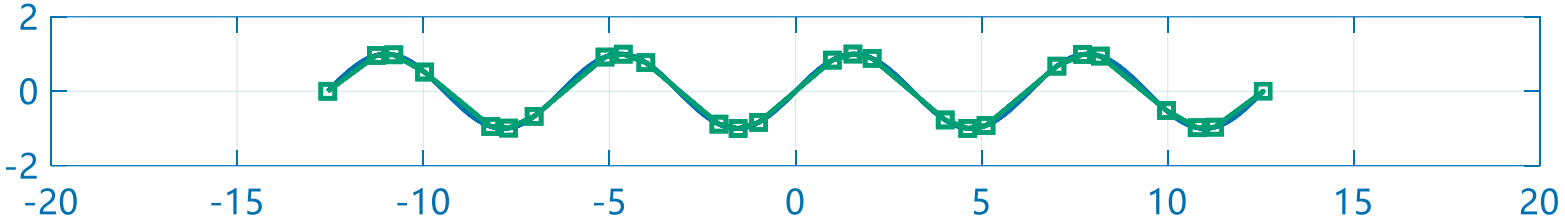
Examples 3 shows a curved narrow gap. This example demonstrates that this toolbox can be used to solve conformal mapping functions for gaps without area, which plays an important role in solving the strength factor of cracks.



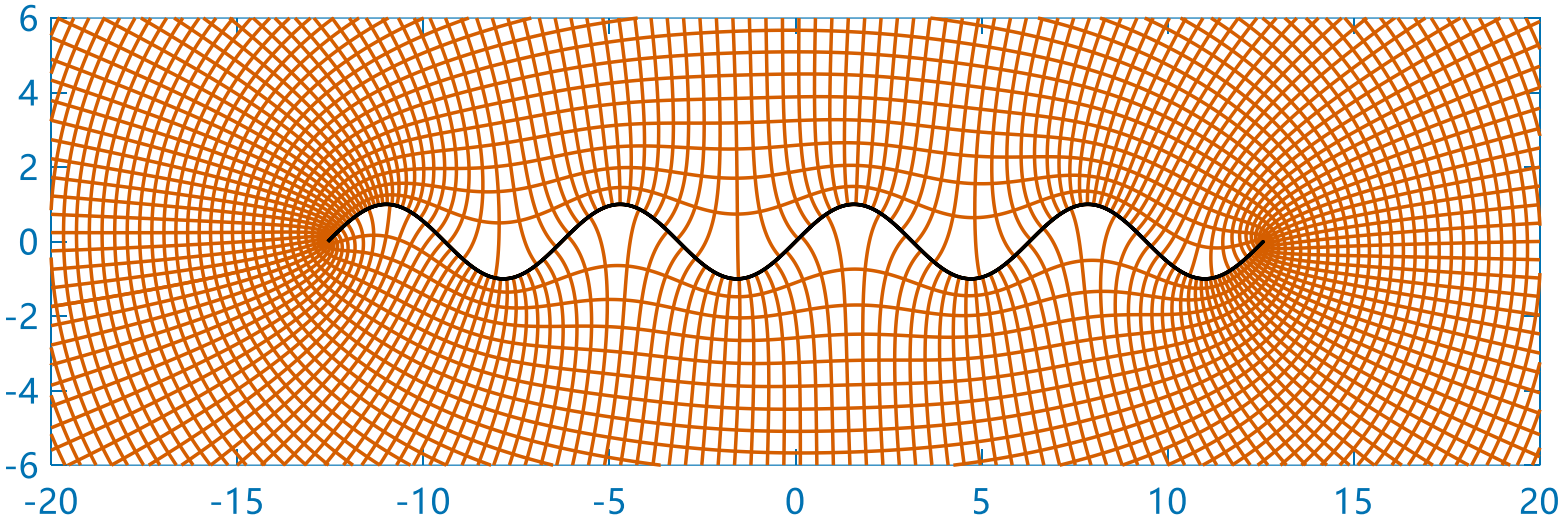
Main Code:

```
F={@(x) sin(x), @(x) sin(x)};
X=[-4*pi,4*pi];
z=BoundaryInitial(F,X,80);
[C,epsilon]=ConformalSolve(z,2000,500,4000,0.6,1e-4);
```

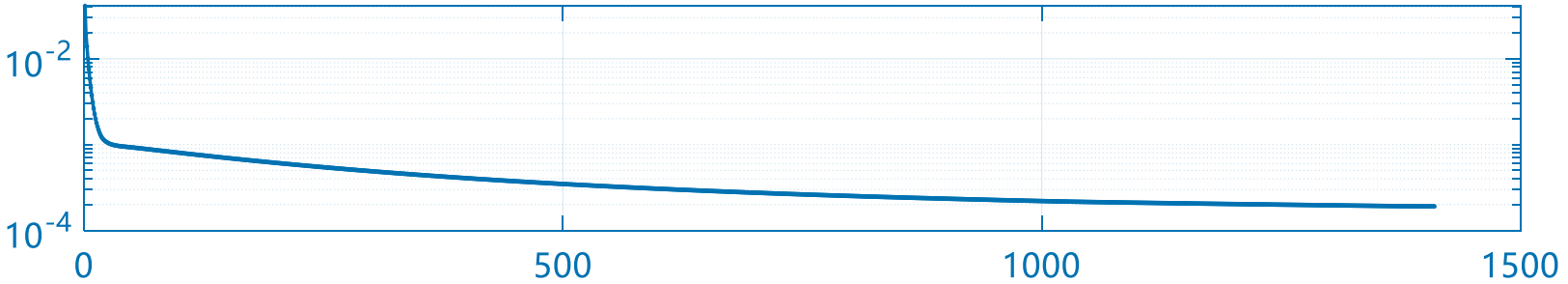
Examples 3 — Results



Polygon

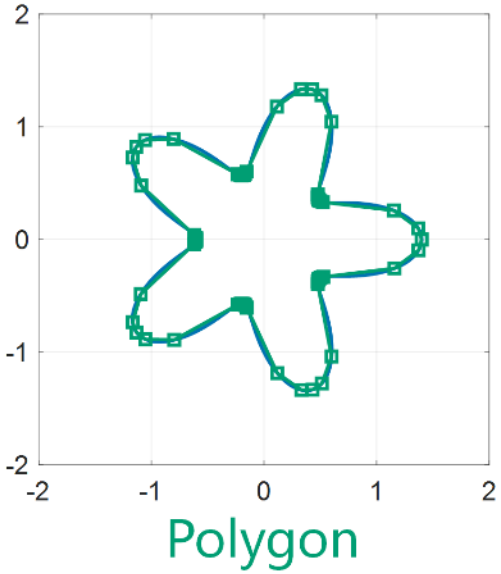


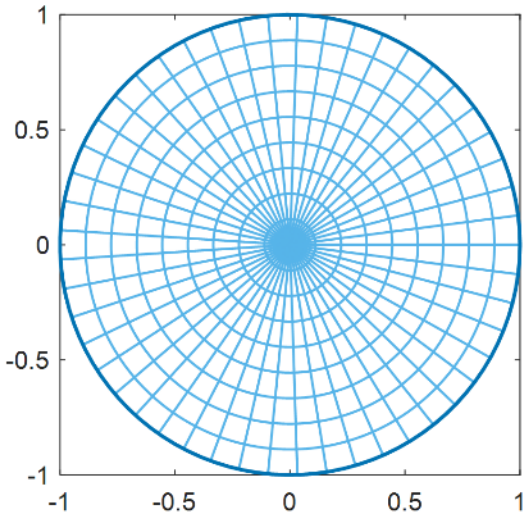
Mapping

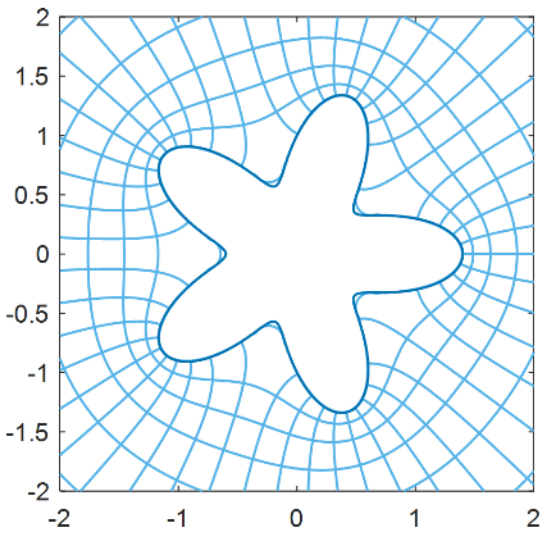


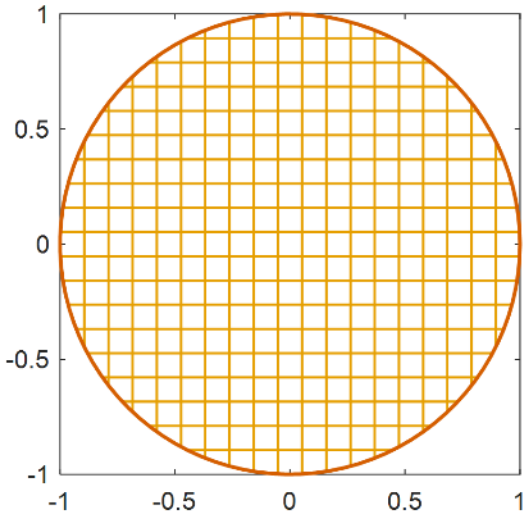
Relative error

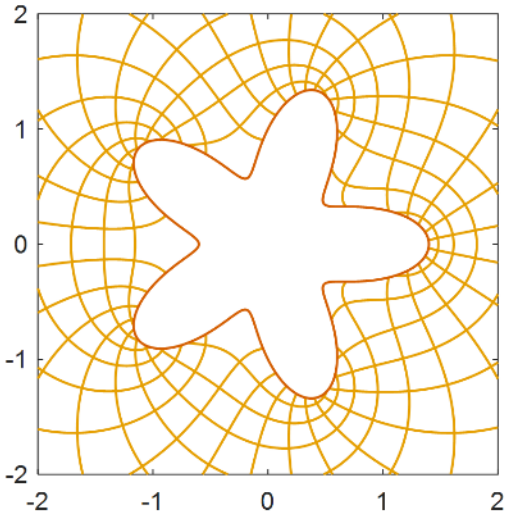
Examples 4



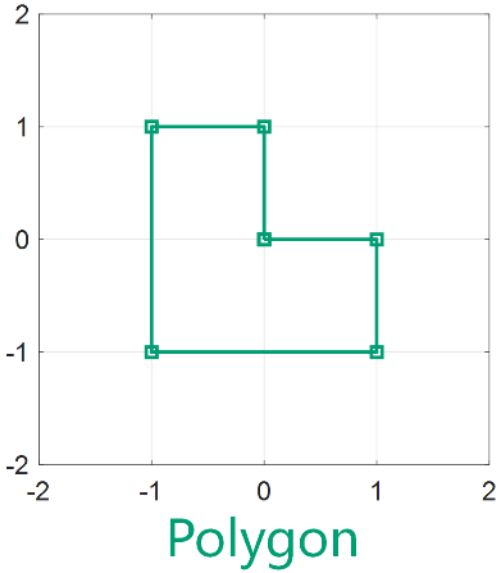


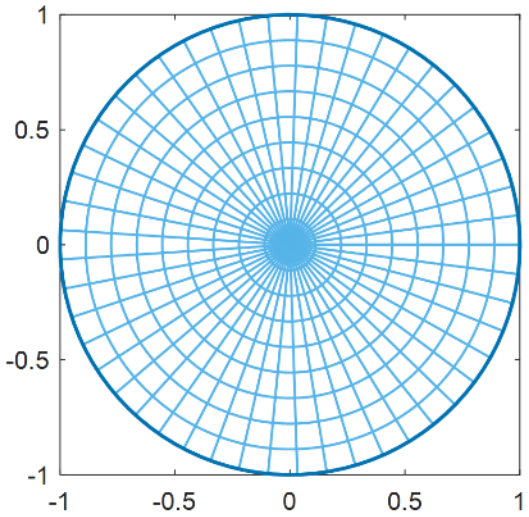


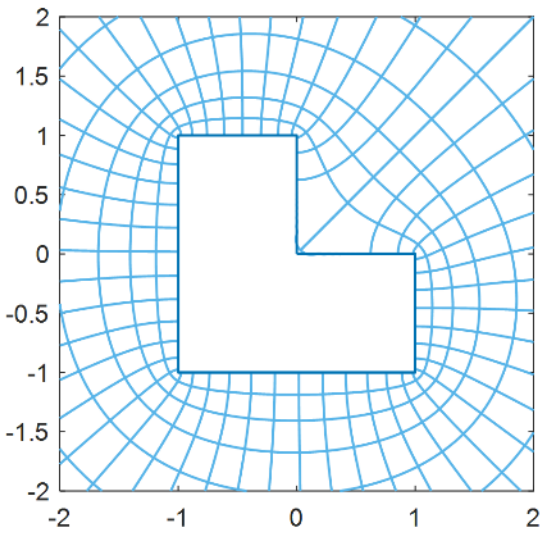


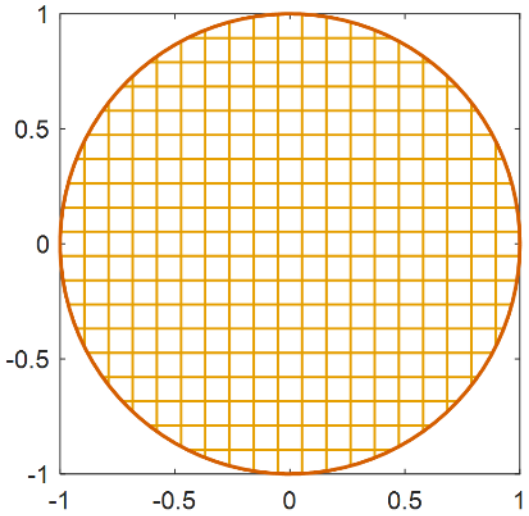


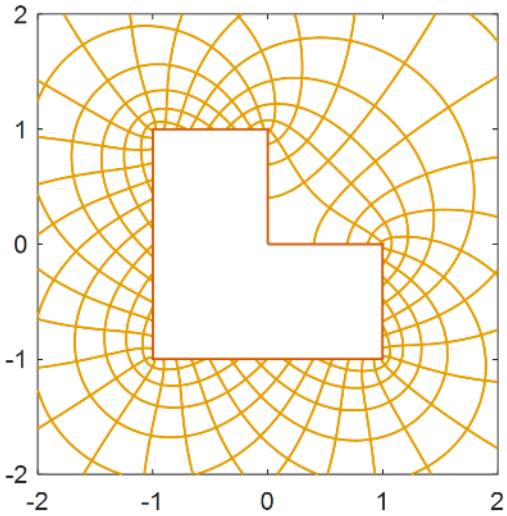
Examples 5



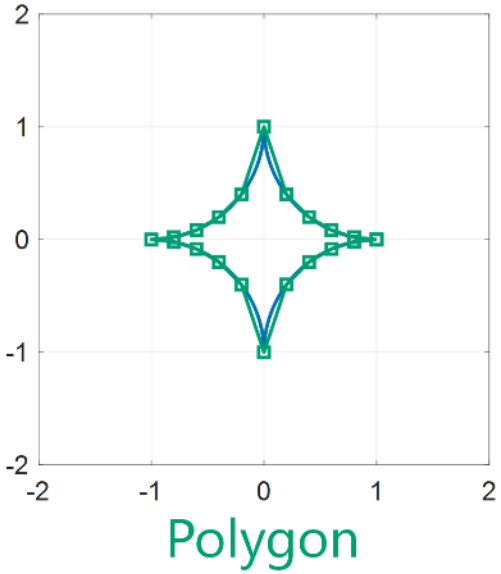


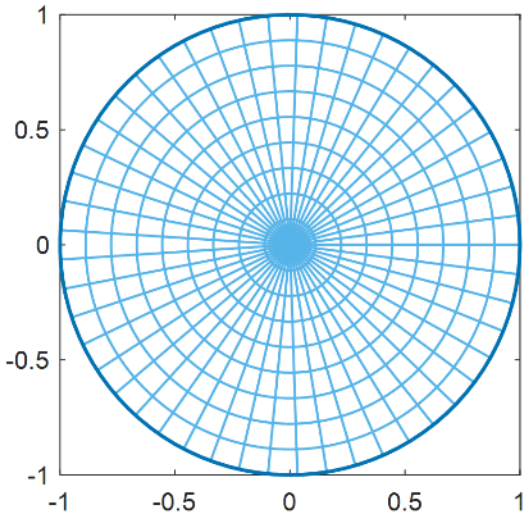


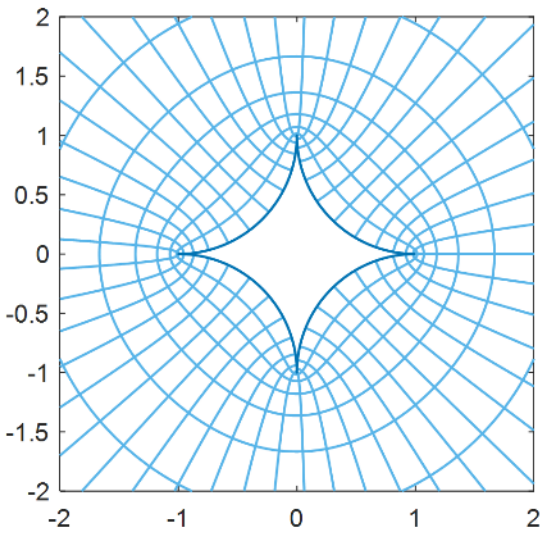


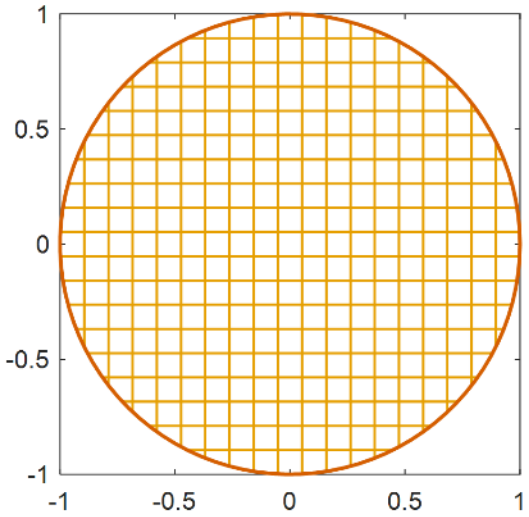


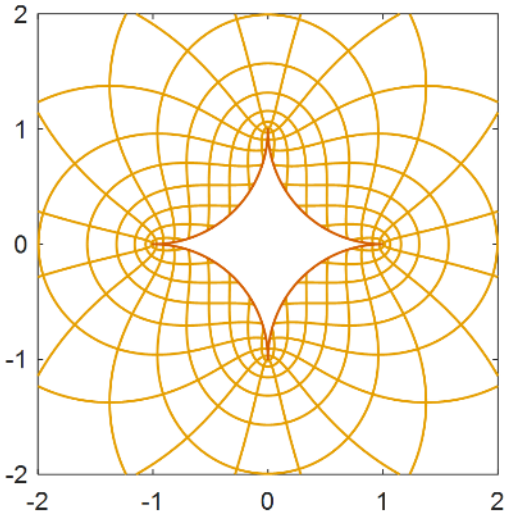
Examples 6



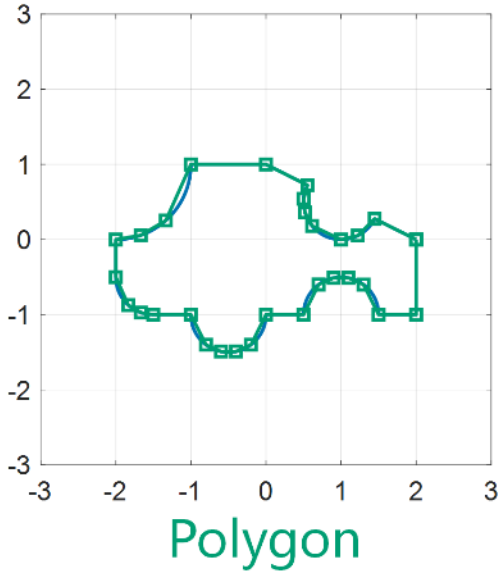


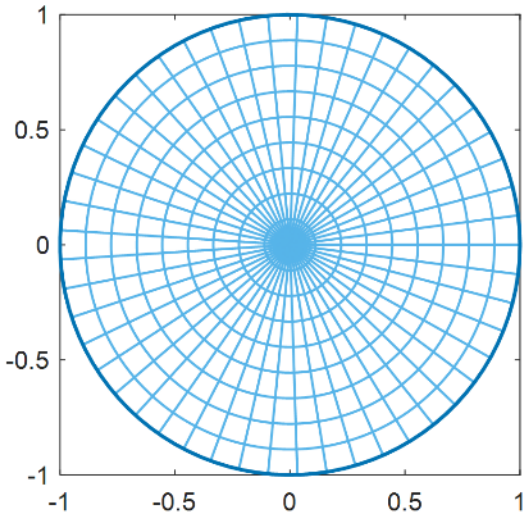


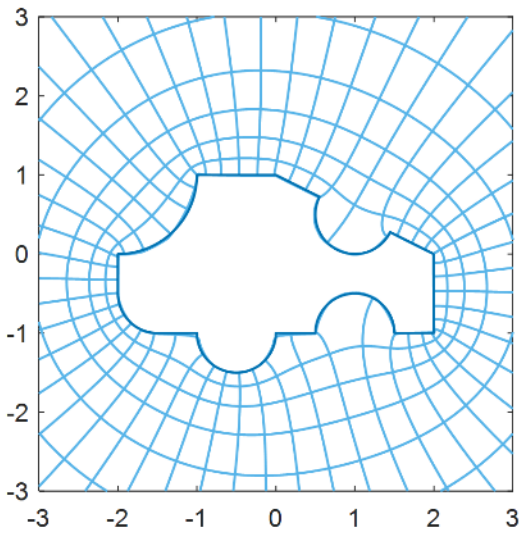


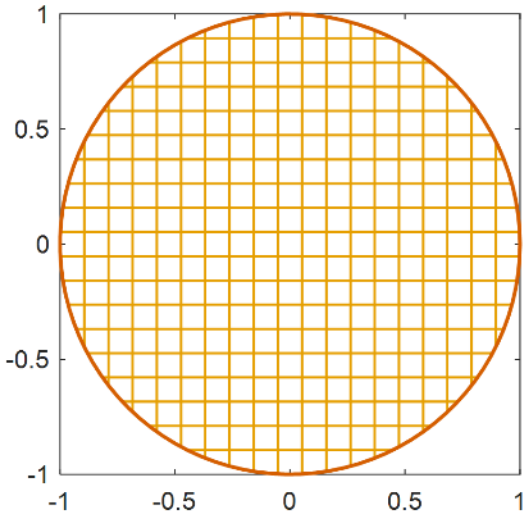


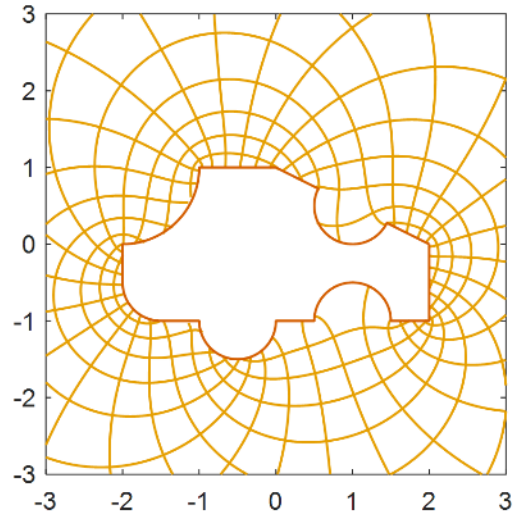
Examples 7



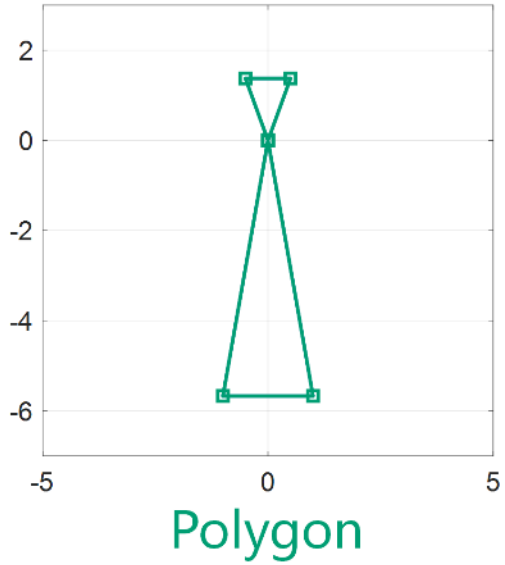




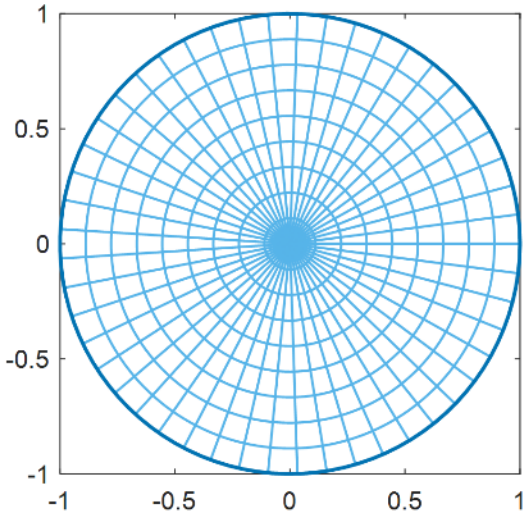


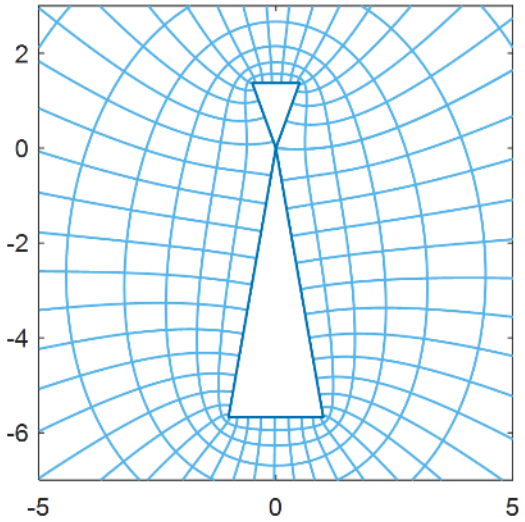


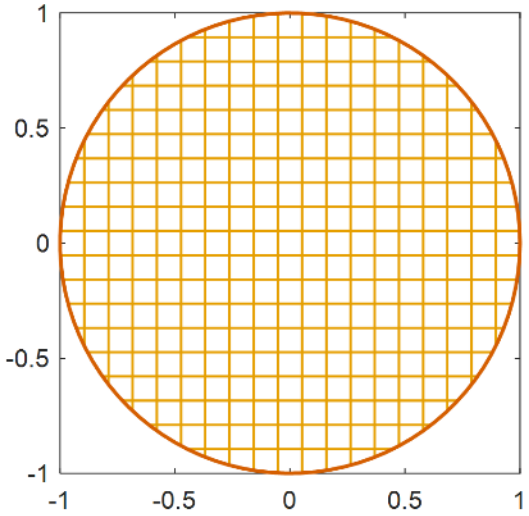
Examples 8

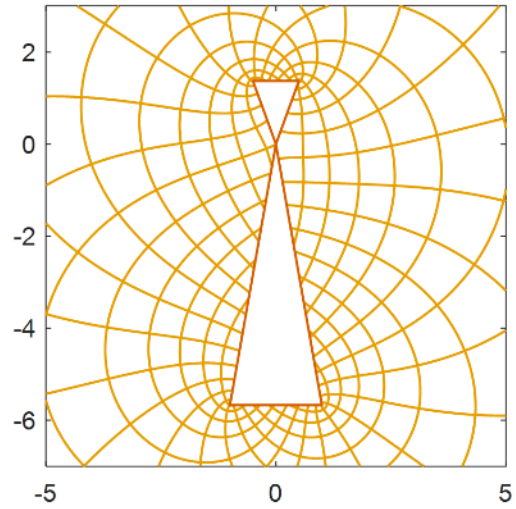


Polygon

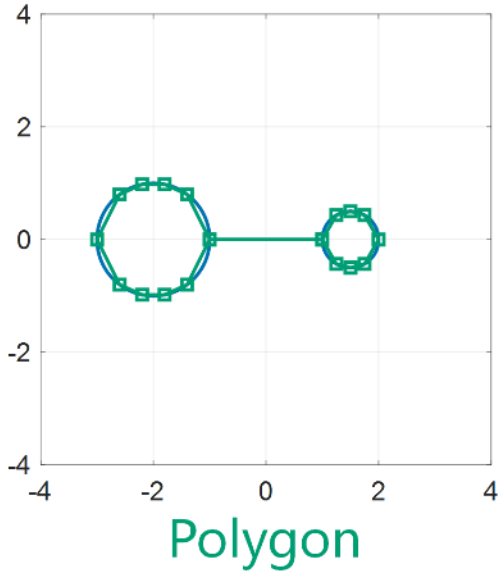


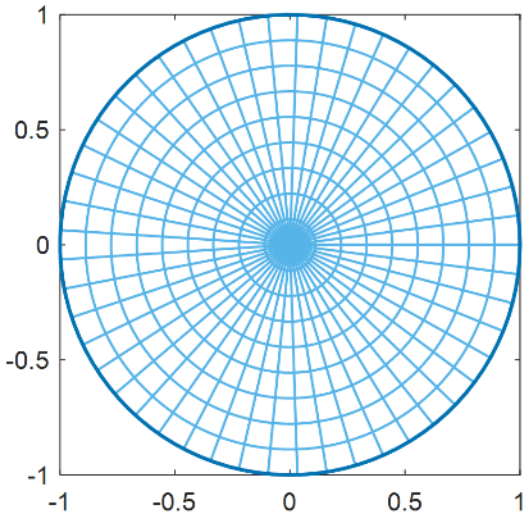


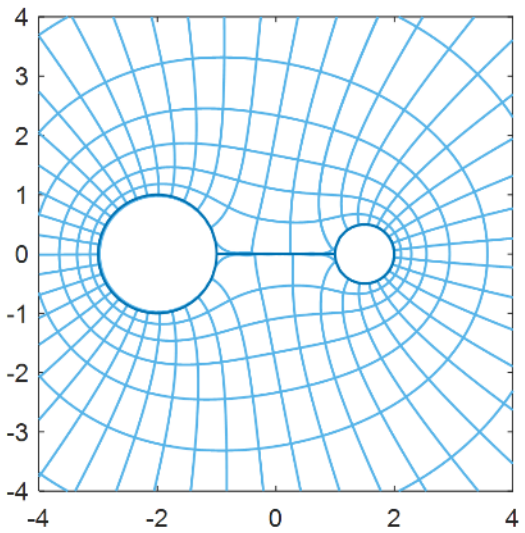


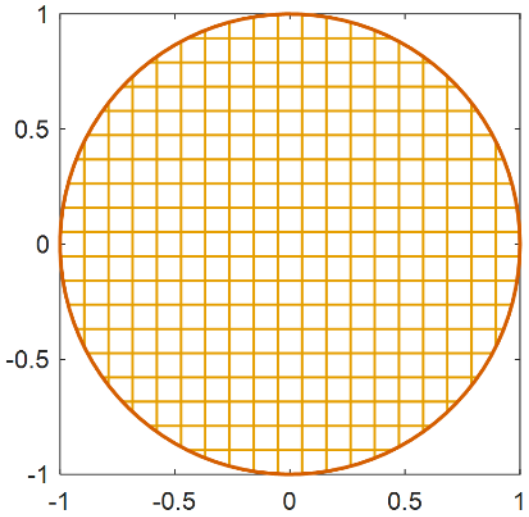


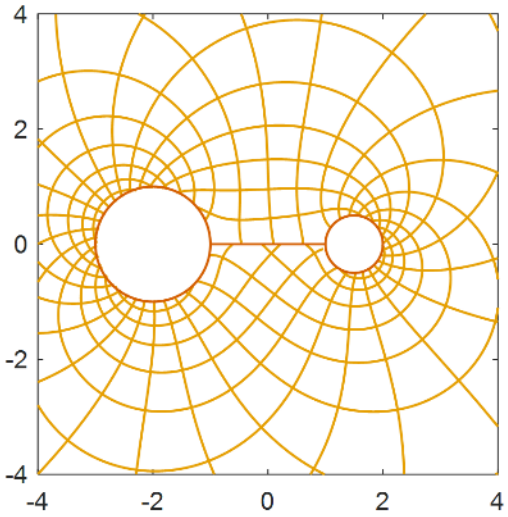
Examples 9



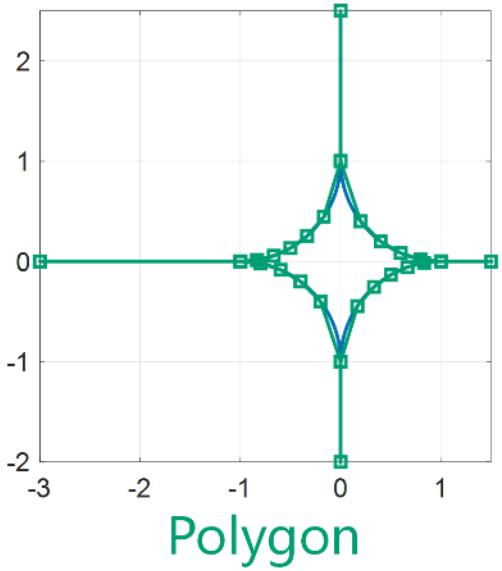


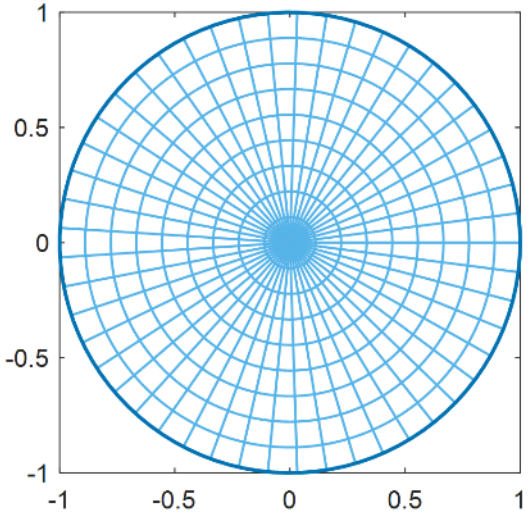


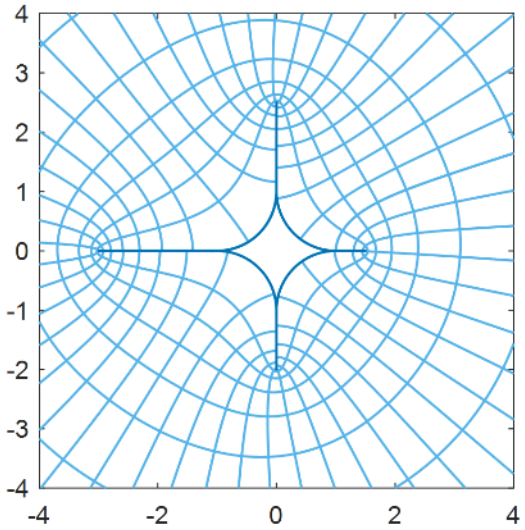


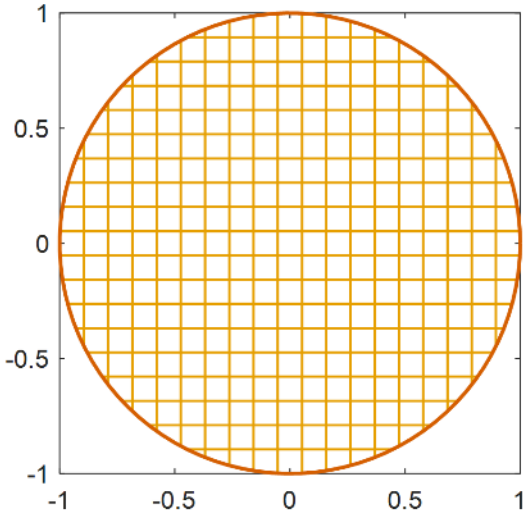


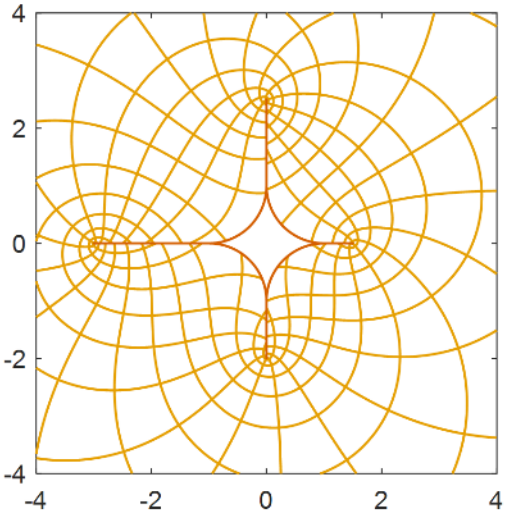
Examples 10



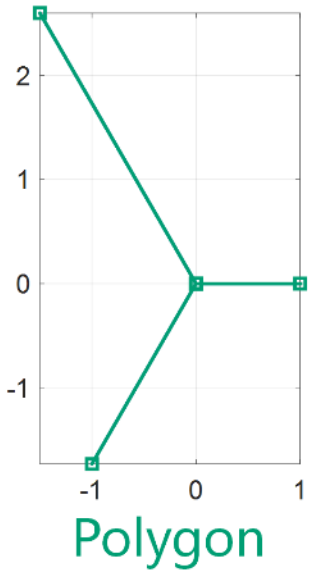


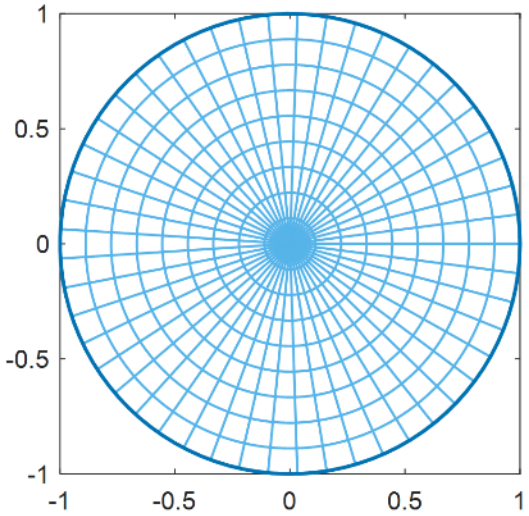


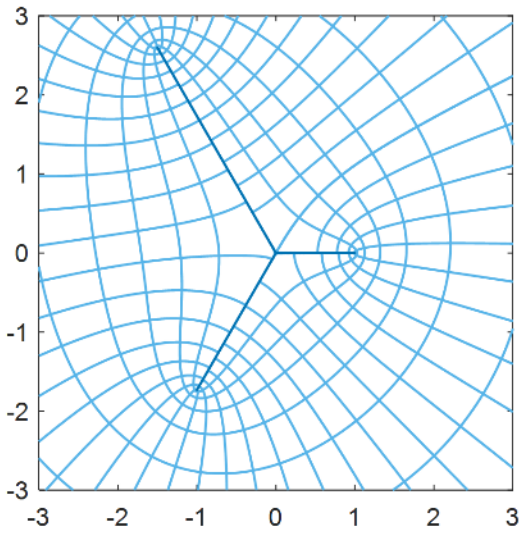


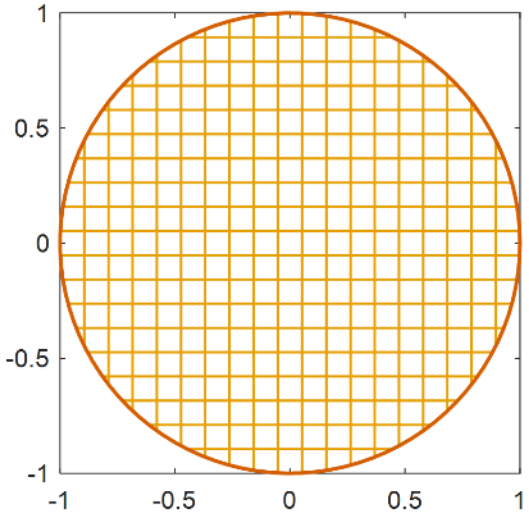


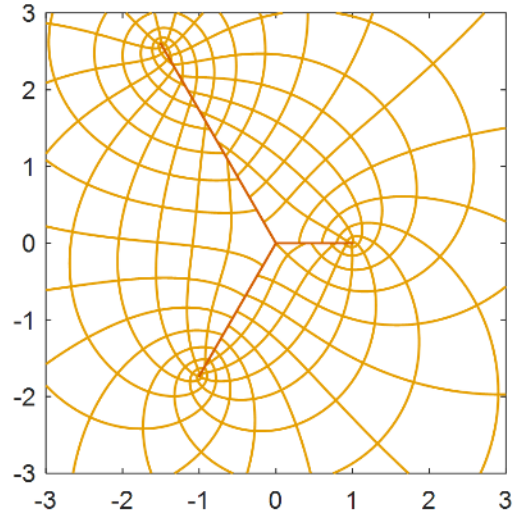
Examples 11


2
1
0
-1
-1
0
1
Polygon


1
0.5
0
-0.5
-1
-1
-0.5
0
0.5
1


3
2
1
0
-1
-2
-3
-3
-2
-1
0
1
2
3


1
0.5
0
-0.5
-1
-1
-0.5
0
0.5
1


3
2
1
0
-1
-2
-3
-3
-2
-1
0
1
2
3

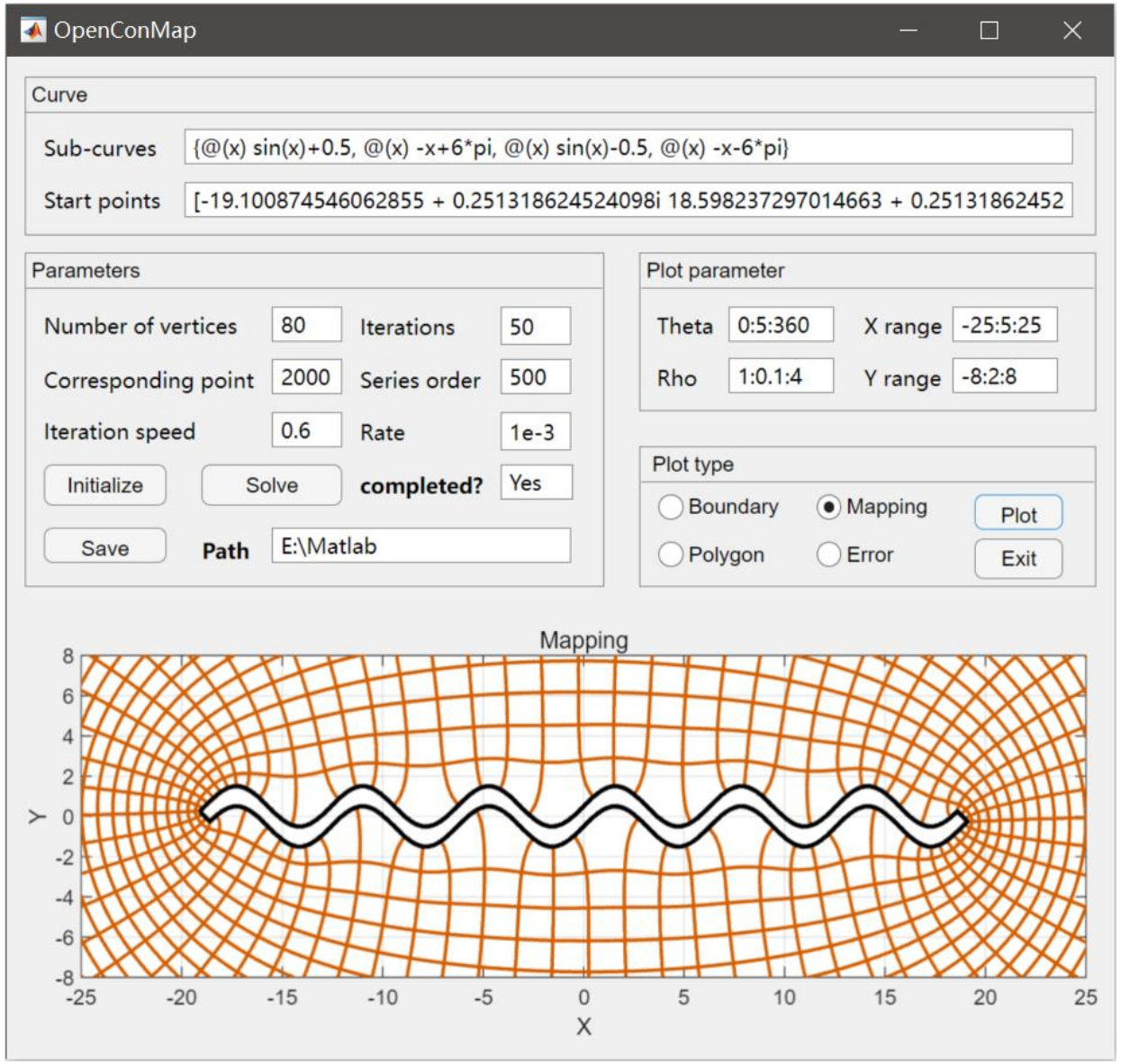
GUI Program

Fill in the information and parameters in the GUI. Click the **Initialize** button to initialize boundary information. Then, in the plot type window, select the **Boundary** option and click the **Plot** button to draw the shape of the boundary, and dynamically observe the direction of the boundary. After confirming that the boundary shape and direction are correct, click the **Solve** button to solve conformal mapping function.

After solving, in the **Plot type** window, select different drawing options to draw corresponding images and observe the solution effect. During the drawing process, the drawing parameters can be modified in the **Plot parameter** window to facilitate observation of the solution effect.

After confirming that the solution is correct, you can click the **Save** button to save the Laurent series coefficients obtained from the solution in the specified directory, which is named "Series_coefficent.xls".

GUI Program



GUI Program Window

Related publication

Users may refer to my papers for more information about the detailed principles and implementations of the algorithms in OpenConMap. If you feel **OpenConMap** helps, please cite the following paper to make it known by more people.

- Kai He, Jucai Chang, Dongdong Pang, Bingjun Sun, Zhiqiang Yin, Dong Li. Iterative algorithm for the conformal mapping function from the exterior of a roadway to the interior of a unit circle. *Arch Appl Mech* 92, 971–991 (2022). https://doi.org/10.1007/s00419-021-02087-w

Compared to the discussion in the paper, I have extended and improved the functions in the toolbox. So, the functionality and usage of these functions are slightly different from the discussion in the corresponding sections of the paper.

Enjoy it

1 February 2024, Guangzhou, China

kai.hee@foxmail.com