Contents

1 Class Summaries

1.1 Domain

The Domain class handles information about a boundary defined in polar coordinates with respect to an origin and rotation. Additionally, this class handles various boundary-specific computations used in the RiemannSurface class.

1.1.1 Construction

The Domain class exists as a general superclass of its many specialized subclasses. While it is possible to construct any domain directly from this class, to maintain accuracy and efficiency of usage it is much better to instead construct any of its subclasses.

Note that all subclasses of Domain take Name-Values arguments.

Note also that no construction of Domain takes arguments for the origin and rotation of the domain. These properties must be set after construction.

`Domain`	Default instance of Domain. It is recommended to instead use the circleDomain subclass.
`Domain(handle)`	Domain from a function handle. Derivatives and several methods are implemented numerically.
`circleDomain(radius)`	Circular domain.
`ellipseDomain(radiusA, radiusB)`	Elliptical domain.
`cosineDomain(radius, cycles, amplitude)`	
`polygonDomain(radius, sides)`	Regular polygon domain.
`smoothDomain(radius, sides, bevelRadius)`	Regular polygon domain whose function is C^2.

1.1.2 Properties

`originX`	The X part of the origin vector.
`originY`	The Y part of the origin vector.
`theta`	The counterclockwise rotation of the domain.
`exitInterpType`	The interpolation method used in the exitInterp method.

1.1.3 Method Summary

Boundary Functions

`instance.bdr(th)`

Evaluates the boundary function at the angles th.

`instance.dbdr(th)`

Evaluates the first derivative of the boundary function at the angles th.

`instance.ddbdr(th)`

Evaluates the second derivative of the boundary function at the angles th.

Computers

`instance.getMinRadius`

Returns the minimum radius of the boundary function.

`instance.getBoundingBox`

Returns the lower left and upper right corners of the boundary's axis aligned bounding box relative to the origin.

Plotters

`instance.plot`

Plots the boundary onto an available figure using instance.bdr.

Misc

`Domain.mustBeDomain(obj)`

Errors if the passed argument is not an instance of Domain or its subclasses.

1.1.4 Detailed Method Description

1.2 Metric

The Metric class stores the conformal factor of a conformally Euclidean metric and its derivatives for use in the RiemannSurface class.

1.2.1 Construction

The Metric class exists as a general superclass of its many specialized subclasses. While it is possible to construct any metric directly from this class, to maintain accuracy and efficiency of usage it is much better to instead construct any of its subclasses.

Note that all subclasses of Metric take Name-Values arguments.

`Metric`	Default instance of Metric. It is recommended to instead use the euclidMetric subclass.
`Metric(handle)`	
`euclidMetric`	Euclidean metric.
`sphereMetric(radius)`	Metric with constant positive curvature.
`hyperbolicMetric(radius)`	Metric with constant negative curvature.
`constcurveMetric(kappa)`	Metric with constant curvature. $4\kappa = R^{-2}$

1.2.2 Method Summary

Blah

`instance.lg(x,y)`

Evaluates the natural log of the g function at the points (x,y).

1.2.3 Detailed Method Description

1.3 RiemannSurface

The RiemannSurface class stores instances of the Domain and Metric classes and handles computations dealing with geodesics and the geodesic X-Ray transform.

1.3.1 Construction

1.3.2 Properties

1.3.3 Method Summary

1.3.4 Detailed Method Description

2 Additional Examples

2.1 X-Ray Transform

2.1.1 Forward Problem

2.1.2 Representing Functions from Data

2.1.3 Inversion

2.1.4 Plotting

2.2 Writing New Metric and Domain Subclasses

3 Navigating Code

3.1 Naming Convention

4 Additional Notes on Numerical Implementations and Optimizations

4.1 Interpolated Results

4.1.1 Boundary Exit Interpolation

4.1.2 Search Interpolation

4.2 IsInside Method

4.3 Class Methods vs. Anonymous Functions