Approximating Implicit Curves on Triangulations with Affine Arithmetic

Afonso Paiva
ICMC-USP

with
Filipe C. Nascimento (ICMC-USP), Luiz Henrique de Figueiredo (IMPA)
and Jorge Stolfi (UNICAMP)

Overview

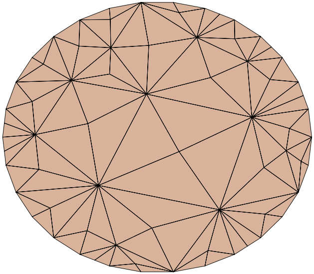
Problem Setup

Given a planar triangulation $\mathcal{T}$ and $f : \mathbb{R}^2 \to \mathbb{R}$, compute a *robust* adaptive polygonal approximation of the curve given implicitly by f on $\mathcal{T}$: $\mathcal{C} = \{(x, y) \in \mathcal{T} : f(x, y) = 0\}$.

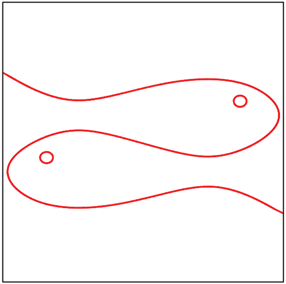
Overview

Problem Setup

Given a planar triangulation $\mathcal{T}$ and $f : \mathbb{R}^2 \to \mathbb{R}$, compute a *robust* adaptive polygonal approximation of the curve given implicitly by f on $\mathcal{T}$: $\mathcal{C} = \{(x, y) \in \mathcal{T} : f(x, y) = 0\}$.



$+$



$= ?$

$\mathcal{C} = f^{-1}(0)$

Possible Solution?

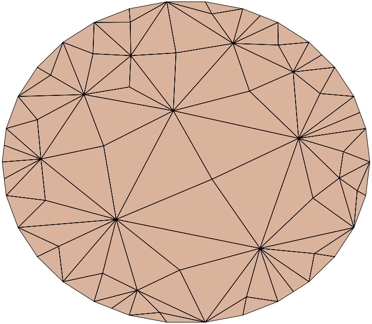
- Curve location:

Possible Solution?

- Curve location: intersection between $\mathcal{C}$ and the triangles of $\mathcal{T}$

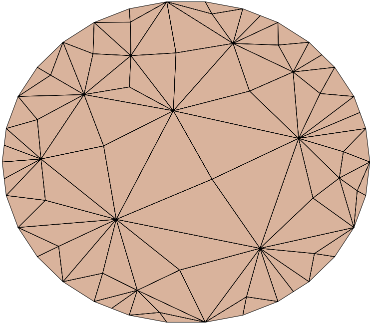
Possible Solution?

- Curve location: intersection between $\mathcal{C}$ and the triangles of $\mathcal{T}$



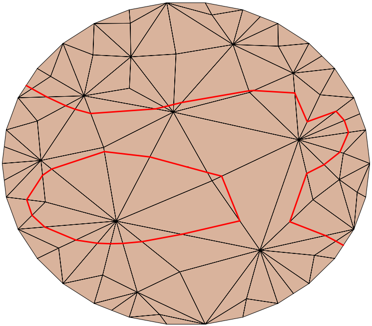
Possible Solution?

- Curve location: intersection between $\mathcal{C}$ and the triangles of $\mathcal{T}$
  What criteria?



Possible Solution?

- Curve location: intersection between $\mathcal{C}$ and the triangles of $\mathcal{T}$
  What criteria?

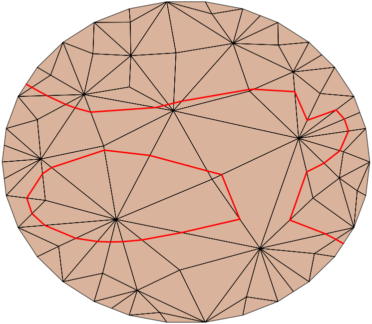


Marching Triangles

$\# \triangle = 101$

Possible Solution?

- Curve location: intersection between $\mathcal{C}$ and the triangles of $\mathcal{T}$
  What criteria?
- Mesh refinement:

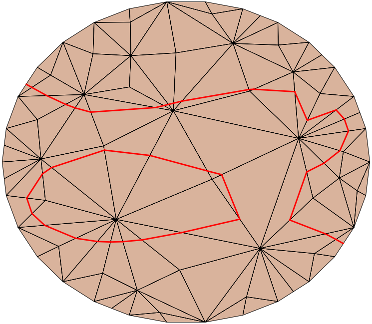


Marching Triangles

$\#\triangle = 101$

Possible Solution?

- Curve location: intersection between $\mathcal{C}$ and the triangles of $\mathcal{T}$
  What criteria?
- Mesh refinement: small triangles $\Rightarrow$ more details

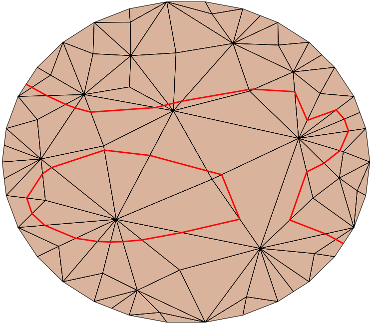


Marching Triangles

$\#\triangle = 101$

Possible Solution?

- Curve location: intersection between $\mathcal{C}$ and the triangles of $\mathcal{T}$
  What criteria?
- Mesh refinement: small triangles $\Rightarrow$ more details
  How small? How efficient?

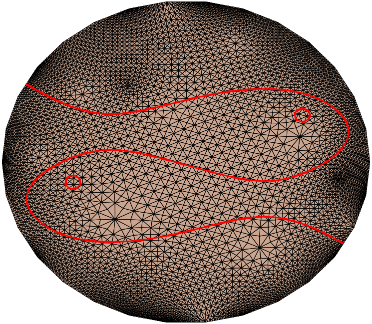


Marching Triangles

$\#\triangle = 101$

Possible Solution?

- Curve location: intersection between $\mathcal{C}$ and the triangles of $\mathcal{T}$
  What criteria?
- Mesh refinement: small triangles $\Rightarrow$ more details
  How small? How efficient?

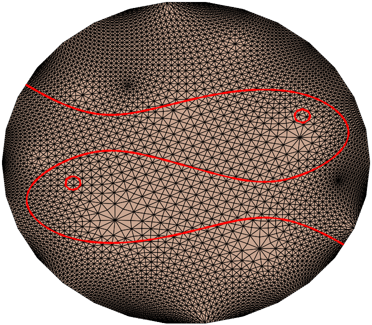


Marching Triangles

$\#\triangle = 12928$

Possible Solution?

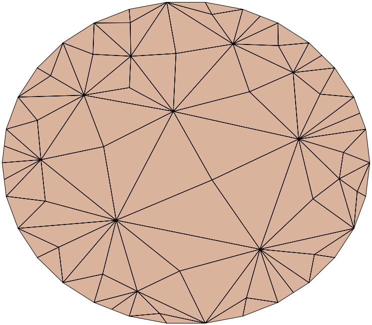
- Curve location: intersection between $\mathcal{C}$ and the triangles of $\mathcal{T}$
  What criteria? Our goal: spatial adaptation!
- Mesh refinement: small triangles $\Rightarrow$ more details
  How small? How efficient? Our goal: geometric adaptation!



Marching Triangles

$\#\triangle = 12928$

level 0

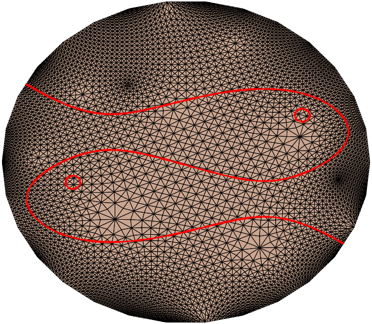


Our Method

$\#\triangle = 101$

Possible Solution?

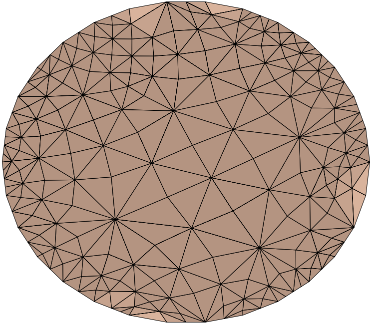
- Curve location: intersection between $\mathcal{C}$ and the triangles of $\mathcal{T}$
  What criteria? Our goal: spatial adaptation!
- Mesh refinement: small triangles $\Rightarrow$ more details
  How small? How efficient? Our goal: geometric adaptation!



Marching Triangles

$\#\triangle = 12928$

level 1

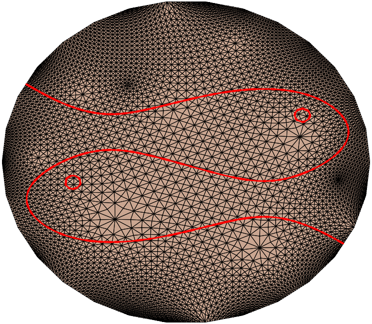


Our Method

$\#\triangle = 376$

Possible Solution?

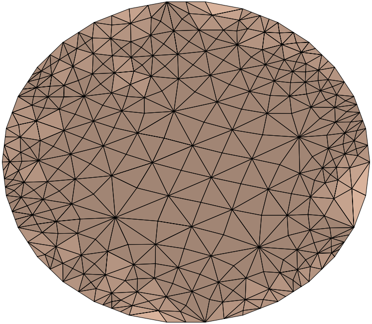
- Curve location: intersection between $\mathcal{C}$ and the triangles of $\mathcal{T}$
  What criteria? Our goal: spatial adaptation!
- Mesh refinement: small triangles $\Rightarrow$ more details
  How small? How efficient? Our goal: geometric adaptation!



Marching Triangles

$\#\triangle = 12928$

level 2

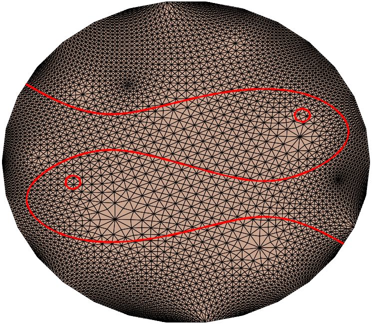


Our Method

$\#\triangle = 612$

Possible Solution?

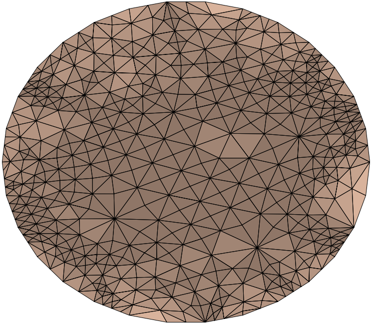
- Curve location: intersection between $\mathcal{C}$ and the triangles of $\mathcal{T}$
  What criteria? Our goal: spatial adaptation!
- Mesh refinement: small triangles $\Rightarrow$ more details
  How small? How efficient? Our goal: geometric adaptation!



Marching Triangles

$\#\triangle = 12928$

level 3

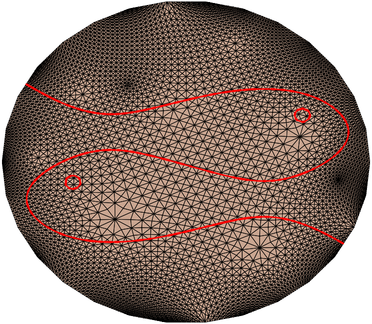


Our Method

$\#\triangle = 930$

Possible Solution?

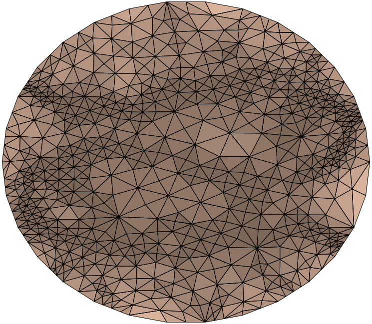
- Curve location: intersection between $\mathcal{C}$ and the triangles of $\mathcal{T}$
  What criteria? Our goal: spatial adaptation!
- Mesh refinement: small triangles $\Rightarrow$ more details
  How small? How efficient? Our goal: geometric adaptation!



Marching Triangles

$\#\triangle = 12928$

level 4

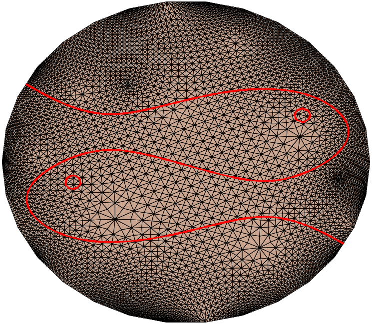


Our Method

$\#\triangle = 1314$

Possible Solution?

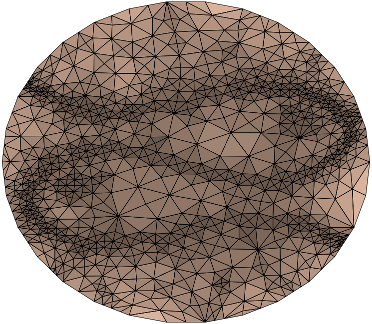
- Curve location: intersection between $\mathcal{C}$ and the triangles of $\mathcal{T}$
  What criteria? Our goal: spatial adaptation!
- Mesh refinement: small triangles $\Rightarrow$ more details
  How small? How efficient? Our goal: geometric adaptation!



Marching Triangles

$\#\triangle = 12928$

level 5

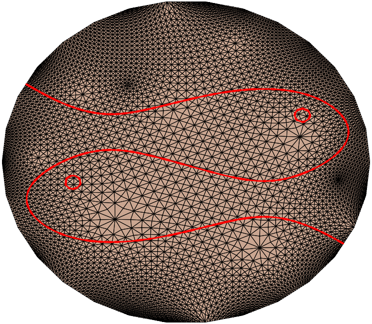


Our Method

$\#\triangle = 1759$

Possible Solution?

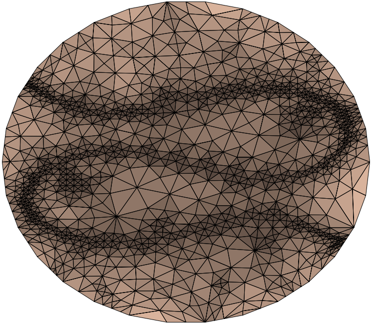
- Curve location: intersection between $\mathcal{C}$ and the triangles of $\mathcal{T}$
  What criteria? Our goal: spatial adaptation!
- Mesh refinement: small triangles $\Rightarrow$ more details
  How small? How efficient? Our goal: geometric adaptation!



Marching Triangles

$\#\triangle = 12928$

level 6

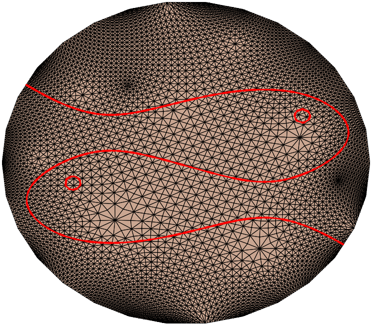


Our Method

$\#\triangle = 2431$

Possible Solution?

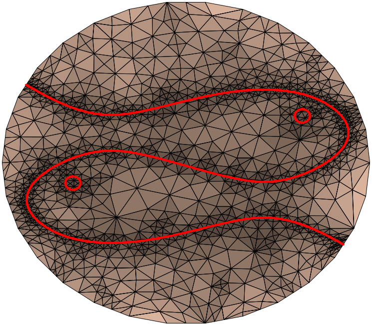
- Curve location: intersection between $\mathcal{C}$ and the triangles of $\mathcal{T}$
  What criteria? Our goal: spatial adaptation!
- Mesh refinement: small triangles $\Rightarrow$ more details
  How small? How efficient? Our goal: geometric adaptation!



Marching Triangles

$\#\triangle = 12928$

level 6



Our Method

$\#\triangle = 2431$

Numerical Tools

- Numerical oracles

Numerical Tools

- Numerical oracles
  - Is this triangle away from the curve?

Numerical Tools

- Numerical oracles
  - Is this triangle away from the curve?
  - Is the curve approximately flat inside the triangle?

Numerical Tools

- Numerical oracles
  - Is this triangle away from the curve?
  - Is the curve approximately flat inside the triangle?
- Self-validated arithmetic methods

Numerical Tools

- Numerical oracles
  - Is this triangle away from the curve?
  - Is the curve approximately flat inside the triangle?

- Self-validated arithmetic methods
  - Robust interval estimative for f with guarantee certificate:

$$F(X) \supseteq f(X) = \{f(x, y) : (x, y) \in X\}$$

Numerical Tools

- Numerical oracles
  - Is this triangle away from the curve?
  - Is the curve approximately flat inside the triangle?

- Self-validated arithmetic methods
  - Robust interval estimative for f with guarantee certificate:

$$F(X) \supseteq f(X) = \{f(x, y) : (x, y) \in X\}$$

  - $0 \notin F(X) \Rightarrow$ cell X is away from curve

Numerical Tools

- Numerical oracles
  - Is this triangle away from the curve?
  - Is the curve approximately flat inside the triangle?
- Self-validated arithmetic methods
  - Robust interval estimative for f with guarantee certificate:

$$F(X) \supseteq f(X) = \{f(x,y) : (x,y) \in X\}$$

  - $0 \notin F(X) \Rightarrow$ cell X is away from curve
  - Interval arithmetic (IA) *and* affine arithmetic (AA)

Affine Arithmetic

- Introduced by Comba and Stolfi in SIBGRAPI'93

Affine Arithmetic

- Introduced by Comba and Stolfi in SIBGRAPI'93
- Represents a quantity z with an *affine form*:

$$\hat{z} = z_0 + z_1\varepsilon_1 + z_2\varepsilon_2 + \cdots + z_n\varepsilon_n$$

where $z_i \in \mathbb{R}$ and the *noise symbols* $\varepsilon_i \in [-1, 1]$ represent independent sources of uncertainty

Affine Arithmetic

- Introduced by Comba and Stolfi in SIBGRAPI'93
- Represents a quantity z with an *affine form*:

  $$\hat{z} = z_0 + z_1\varepsilon_1 + z_2\varepsilon_2 + \cdots + z_n\varepsilon_n$$

  where $z_i \in \mathbb{R}$ and the *noise symbols* $\varepsilon_i \in [-1, 1]$ represent independent sources of uncertainty
- We can compute arbitrary formulas on affine forms

Affine Arithmetic

- Introduced by Comba and Stolfi in SIBGRAPI'93
- Represents a quantity z with an *affine form*:

$$\hat{z} = z_0 + z_1\varepsilon_1 + z_2\varepsilon_2 + \cdots + z_n\varepsilon_n$$

  where $z_i \in \mathbb{R}$ and the *noise symbols* $\varepsilon_i \in [-1, 1]$ represent independent sources of uncertainty
- We can compute arbitrary formulas on affine forms
- Good alternative to replace IA in graphics applications

Affine Arithmetic

- Introduced by Comba and Stolfi in SIBGRAPI'93
- Represents a quantity z with an *affine form*:

$$\hat{z} = z_0 + z_1\varepsilon_1 + z_2\varepsilon_2 + \cdots + z_n\varepsilon_n$$

  where $z_i \in \mathbb{R}$ and the *noise symbols* $\varepsilon_i \in [-1, 1]$ represent independent sources of uncertainty
- We can compute arbitrary formulas on affine forms
- Good alternative to replace IA in graphics applications
  - AA has ability to handle correlations

Affine Arithmetic

- Introduced by Comba and Stolfi in SIBGRAPI'93
- Represents a quantity z with an *affine form*:

$$\hat{z} = z_0 + z_1\varepsilon_1 + z_2\varepsilon_2 + \cdots + z_n\varepsilon_n$$

  where $z_i \in \mathbb{R}$ and the *noise symbols* $\varepsilon_i \in [-1, 1]$ represent independent sources of uncertainty
- We can compute arbitrary formulas on affine forms
- Good alternative to replace IA in graphics applications
  - AA has ability to handle correlations
  - AA provides tighter interval estimative

Affine Arithmetic

- Introduced by Comba and Stolfi in SIBGRAPI'93
- Represents a quantity z with an *affine form*:

$$\hat{z} = z_0 + z_1\varepsilon_1 + z_2\varepsilon_2 + \cdots + z_n\varepsilon_n$$

  where $z_i \in \mathbb{R}$ and the *noise symbols* $\varepsilon_i \in [-1, 1]$ represent independent sources of uncertainty
- We can compute arbitrary formulas on affine forms
- Good alternative to replace IA in graphics applications
  - AA has ability to handle correlations
  - AA provides tighter interval estimative
  - AA provides additional geometric information

Affine Arithmetic

- Introduced by Comba and Stolfi in SIBGRAPI'93
- Represents a quantity z with an *affine form*:

$$\hat{z} = z_0 + z_1\varepsilon_1 + z_2\varepsilon_2 + \cdots + z_n\varepsilon_n$$

  where $z_i \in \mathbb{R}$ and the *noise symbols* $\varepsilon_i \in [-1, 1]$ represent independent sources of uncertainty
- We can compute arbitrary formulas on affine forms
- Good alternative to replace IA in graphics applications
  - AA has ability to handle correlations
  - AA provides tighter interval estimative
  - AA provides additional geometric information
  - Good AA libraries in C/C++ available

Intervals in Affine Arithmetic

- AA algorithms can input and output intervals

Intervals in Affine Arithmetic

- AA algorithms can input and output intervals
- AA form $\Rightarrow$ IA form

Intervals in Affine Arithmetic

- AA algorithms can input and output intervals
- AA form $\Rightarrow$ IA form
  - $\hat{z} = z_0 + z_1\varepsilon_1 + z_2\varepsilon_2 + \cdots + z_n\varepsilon_n$

Intervals in Affine Arithmetic

- AA algorithms can input and output intervals
- AA form $\Rightarrow$ IA form
  - $\hat{z} = z_0 + z_1\varepsilon_1 + z_2\varepsilon_2 + \cdots + z_n\varepsilon_n \Rightarrow z \in [\hat{z}] := [z_0 - \delta, z_0 + \delta]$ where $\delta = |z_1| + \cdots + |z_n|$

Intervals in Affine Arithmetic

- AA algorithms can input and output intervals
- AA form $\Rightarrow$ IA form
  - $\hat{z} = z_0 + z_1\varepsilon_1 + z_2\varepsilon_2 + \cdots + z_n\varepsilon_n \Rightarrow z \in [\hat{z}] := [z_0 - \delta, z_0 + \delta]$ where $\delta = |z_1| + \cdots + |z_n|$
- IA form $\Rightarrow$ AA form

Intervals in Affine Arithmetic

- AA algorithms can input and output intervals
- AA form $\Rightarrow$ IA form
  - $\hat{z} = z_0 + z_1\varepsilon_1 + z_2\varepsilon_2 + \cdots + z_n\varepsilon_n \Rightarrow z \in [\hat{z}] := [z_0 - \delta, z_0 + \delta]$
    where $\delta = |z_1| + \cdots + |z_n|$
- IA form $\Rightarrow$ AA form
  - $z \in [a, b]$

Intervals in Affine Arithmetic

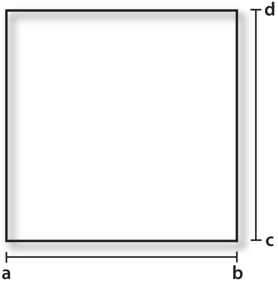
- AA algorithms can input and output intervals
- AA form $\Rightarrow$ IA form
  - $\hat{z} = z_0 + z_1\varepsilon_1 + z_2\varepsilon_2 + \cdots + z_n\varepsilon_n \Rightarrow z \in [\hat{z}] := [z_0 - \delta, z_0 + \delta]$
    where $\delta = |z_1| + \cdots + |z_n|$
- IA form $\Rightarrow$ AA form
  - $z \in [a, b] \Rightarrow \hat{z} = z_0 + z_1\varepsilon_1$ where
    $z_0 = (a + b)/2$
    $z_1 = (b - a)/2$

Bounding Iimplicit Curves with Strips on □

On axis-aligned rectangles:

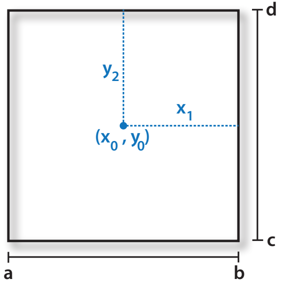
Bounding Iimplicit Curves with Strips on □

On axis-aligned rectangles: we need to evaluate $f(\square)$ with AA



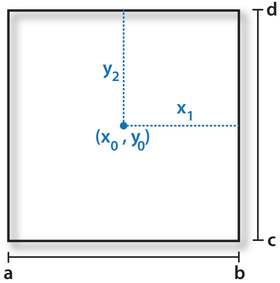
Bounding Iimplicit Curves with Strips on □

On axis-aligned rectangles: we need to evaluate $f(\square)$ with AA



Bounding Iimplicit Curves with Strips on □

On axis-aligned rectangles: we need to evaluate $f(\square)$ with AA

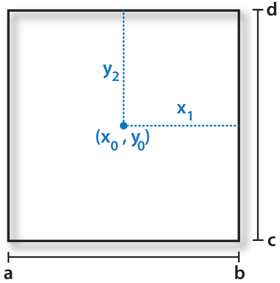


$$\hat{x} = x_0 + x_1\varepsilon_1, \quad x_0 = (a+b)/2, \quad x_1 = (b-a)/2$$

$$\hat{y} = y_0 + y_2\varepsilon_2, \quad y_0 = (c+d)/2, \quad y_2 = (d-c)/2$$

Bounding Iimplicit Curves with Strips on □

On axis-aligned rectangles: we need to evaluate $f(\square)$ with AA



$$\hat{x} = x_0 + x_1\varepsilon_1, \quad x_0 = (a+b)/2, \quad x_1 = (b-a)/2$$

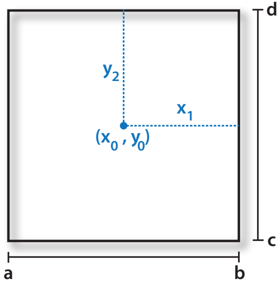
$$\hat{y} = y_0 + y_2\varepsilon_2, \quad y_0 = (c+d)/2, \quad y_2 = (d-c)/2$$

AA form of f

$$\hat{f} = f_0 + f_1\varepsilon_1 + f_2\varepsilon_2 + f_3\varepsilon_3 + \cdots + f_n\varepsilon_n$$

Bounding Iimplicit Curves with Strips on □

On axis-aligned rectangles: we need to evaluate $f(\square)$ with AA



$$\hat{x} = x_0 + x_1\varepsilon_1, \quad x_0 = (a+b)/2, \quad x_1 = (b-a)/2$$

$$\hat{y} = y_0 + y_2\varepsilon_2, \quad y_0 = (c+d)/2, \quad y_2 = (d-c)/2$$

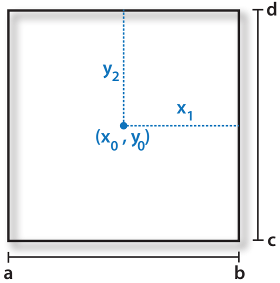
AA form of f

$$\hat{f} = f_0 + f_1\varepsilon_1 + f_2\varepsilon_2 + f_3\varepsilon_3 + \cdots + f_n\varepsilon_n$$

$\varepsilon_3, \ldots, \varepsilon_n$ are noise symbols related to non-affine operations

Bounding Iimplicit Curves with Strips on □

On axis-aligned rectangles: we need to evaluate $f(\square)$ with AA



$$\hat{x} = x_0 + x_1\varepsilon_1, \quad x_0 = (a+b)/2, \quad x_1 = (b-a)/2$$

$$\hat{y} = y_0 + y_2\varepsilon_2, \quad y_0 = (c+d)/2, \quad y_2 = (d-c)/2$$

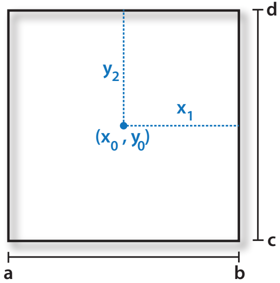
AA form of f

$$\hat{f} = f_0 + f_1\varepsilon_1 + f_2\varepsilon_2 + f_3\varepsilon_3 + \cdots + f_n\varepsilon_n$$

higher-order terms can be condensed $\Rightarrow f_3 = |f_3| + \cdots + |f_n|$

Bounding Iimplicit Curves with Strips on $\square$

On axis-aligned rectangles: we need to evaluate $f(\square)$ with AA



$$\hat{x} = x_0 + x_1\varepsilon_1, \quad x_0 = (a+b)/2, \quad x_1 = (b-a)/2$$

$$\hat{y} = y_0 + y_2\varepsilon_2, \quad y_0 = (c+d)/2, \quad y_2 = (d-c)/2$$

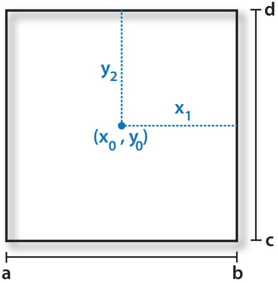
AA form of f

$$\hat{f} = f_0 + f_1\varepsilon_1 + f_2\varepsilon_2 + f_3\varepsilon_3$$

higher-order terms can be condensed $\Rightarrow f_3 = |f_3| + \cdots + |f_n|$

Bounding Iimplicit Curves with Strips on □

On axis-aligned rectangles: we need to evaluate $f(\square)$ with AA



$$\hat{x} = x_0 + x_1\varepsilon_1, \quad x_0 = (a+b)/2, \quad x_1 = (b-a)/2$$

$$\hat{y} = y_0 + y_2\varepsilon_2, \quad y_0 = (c+d)/2, \quad y_2 = (d-c)/2$$

AA form of f

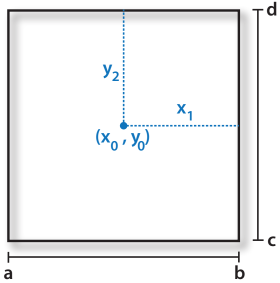
$$\hat{f} = f_0 + f_1\varepsilon_1 + f_2\varepsilon_2 + f_3\varepsilon_3$$

Spatial criteria

$$0 \notin [\hat{f}(\square)] \Rightarrow \texttt{discard}(\square)$$

Bounding Iimplicit Curves with Strips on $\square$

On axis-aligned rectangles: we need to evaluate $f(\square)$ with AA



$$\hat{x} = x_0 + x_1\varepsilon_1, \quad x_0 = (a+b)/2, \quad x_1 = (b-a)/2$$

$$\hat{y} = y_0 + y_2\varepsilon_2, \quad y_0 = (c+d)/2, \quad y_2 = (d-c)/2$$

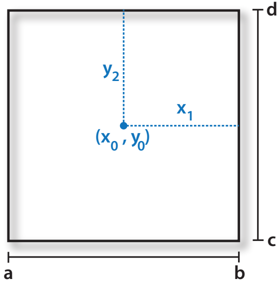
AA form of f

$$\hat{f} = f_0 + f_1\varepsilon_1 + f_2\varepsilon_2 + f_3\varepsilon_3$$

Geometric bounds using the AA form of $\hat{f}$

Bounding Iimplicit Curves with Strips on □

On axis-aligned rectangles: we need to evaluate $f(\square)$ with AA



$$\hat{x} = x_0 + x_1\varepsilon_1, \quad x_0 = (a+b)/2, \quad x_1 = (b-a)/2$$

$$\hat{y} = y_0 + y_2\varepsilon_2, \quad y_0 = (c+d)/2, \quad y_2 = (d-c)/2$$

AA form of f

$$\hat{f} = f_0 + f_1\varepsilon_1 + f_2\varepsilon_2 + f_3\varepsilon_3$$

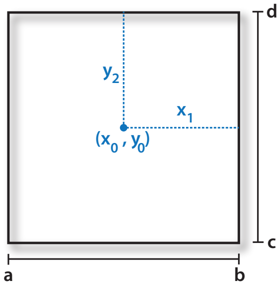
Geometric bounds using the AA form of $\hat{f}$

the graph of $z = f(x, y)$ over $\square$ is sandwiched between the planes:

$$z = f_0 + f_1\varepsilon_1 + f_2\varepsilon_2 \pm f_3$$

Bounding Iimplicit Curves with Strips on □

On axis-aligned rectangles: we need to evaluate $f(\square)$ with AA



$$\varepsilon_1 = \frac{x - x_0}{x_1} \qquad \varepsilon_2 = \frac{y - y_0}{y_2}$$

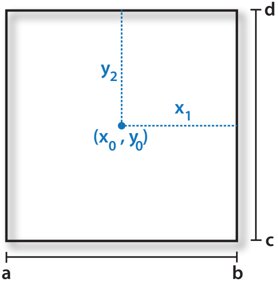
Geometric bounds using the AA form of $\hat{f}$

the graph of $z = f(x, y)$ over □ is sandwiched between the planes:

$$z = f_0 + f_1\varepsilon_1 + f_2\varepsilon_2 \pm f_3$$

Bounding Iimplicit Curves with Strips on $\square$

On axis-aligned rectangles: we need to evaluate $f(\square)$ with AA



$$\varepsilon_1 = \frac{x - x_0}{x_1} \qquad \varepsilon_2 = \frac{y - y_0}{y_2}$$

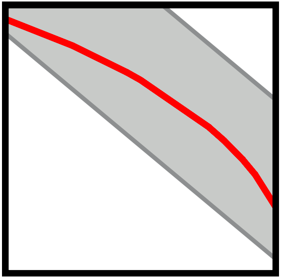
Geometric bounds using the AA form of $\hat{f}$

z in cartesian coordinates:

$$z = f_0 + \frac{f_1}{x_1}(x - x_0) + \frac{f_2}{y_2}(y - y_0) \pm f_3$$

Bounding Iimplicit Curves with Strips on $\Box$

On axis-aligned rectangles: we need to evaluate $f(\Box)$ with AA



$$\varepsilon_1 = \frac{x - x_0}{x_1} \qquad \varepsilon_2 = \frac{y - y_0}{y_2}$$

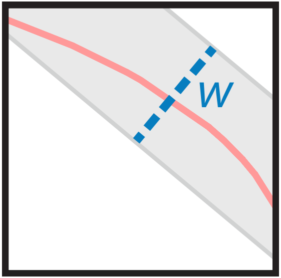
Geometric bounds using the AA form of $\hat{f}$

f is zero inside the strip defined by the two parallel lines:

$$0 = f_0 + \frac{f_1}{x_1}(x - x_0) + \frac{f_2}{y_2}(y - y_0) \pm f_3$$

Bounding Iimplicit Curves with Strips on □

On axis-aligned rectangles: we need to evaluate $f(\square)$ with AA



The width between the lines

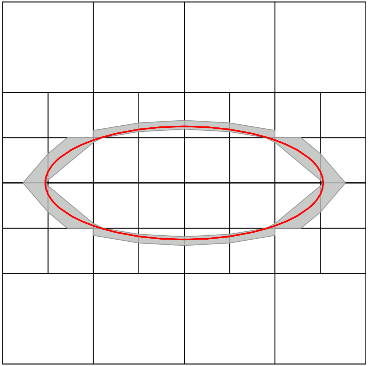
$$w = \frac{2f_3}{\sqrt{\left(\frac{f_1}{x_1}\right)^2 + \left(\frac{f_2}{y_2}\right)^2}}$$

Geometric bounds using the AA form of $\hat{f}$

f is zero inside the strip defined by the two parallel lines:

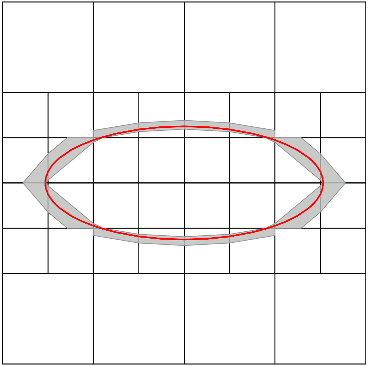
$$0 = f_0 + \frac{f_1}{x_1}(x - x_0) + \frac{f_2}{y_2}(y - y_0) \pm f_3$$

Bounding Implicit Curves with Strips on □



$$\frac{x^2}{6} + y^2 = 1$$

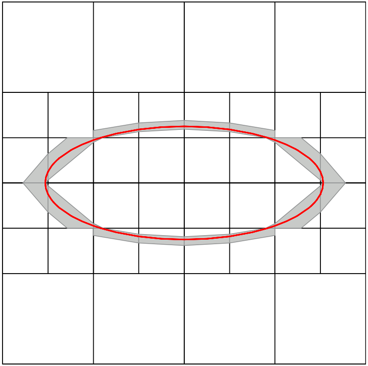
Bounding Implicit Curves with Strips on □



wide strips $\Rightarrow$ high curvature

$$\frac{x^2}{6} + y^2 = 1$$

Bounding Implicit Curves with Strips on $\square$



$$\frac{x^2}{6} + y^2 = 1$$

wide strips $\Rightarrow$ high curvature

Geometric criteria

$w > \mathit{threshold} \Rightarrow$ `subdivide(□)`

Comparing with IA: method proposed by Lopes *et al.* in SIBGRAPI 2001

Bounding Implicit Curves with Strips on □

Comparing with IA: method proposed by Lopes *et al.* in SIBGRAPI 2001

- requires the evaluation ∇f using IA and automatic differentiation

Bounding Implicit Curves with Strips on □

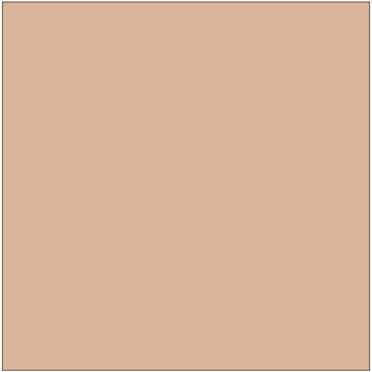
Comparing with IA: method proposed by Lopes *et al.* in SIBGRAPI 2001

- requires the evaluation ∇f using IA and automatic differentiation
- adaptive quadtree

Bounding Implicit Curves with Strips on □

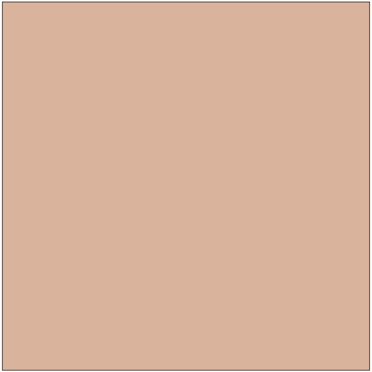
Comparing with IA: method proposed by Lopes *et al.* in SIBGRAPI 2001

- requires the evaluation ∇f using IA and automatic differentiation
- adaptive quadtree



IA

level 0

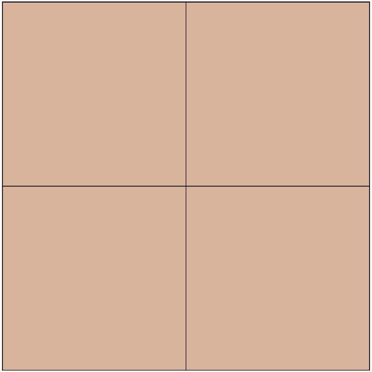


AA

Bounding Implicit Curves with Strips on $\square$

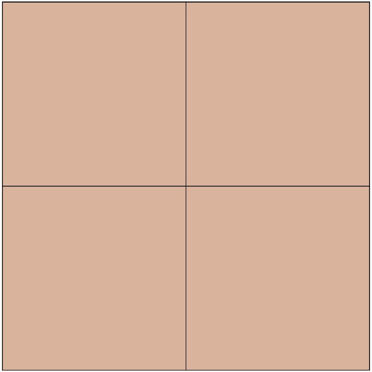
Comparing with IA: method proposed by Lopes *et al.* in SIBGRAPI 2001

- requires the evaluation ∇f using IA and automatic differentiation
- adaptive quadtree



IA

level 1

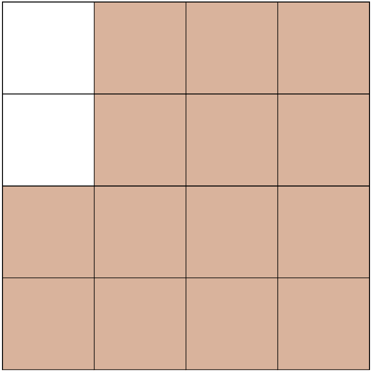


AA

Bounding Implicit Curves with Strips on □

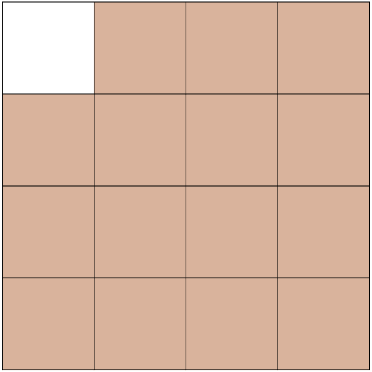
Comparing with IA: method proposed by Lopes *et al.* in SIBGRAPI 2001

- requires the evaluation ∇f using IA and automatic differentiation
- adaptive quadtree



IA

level 2

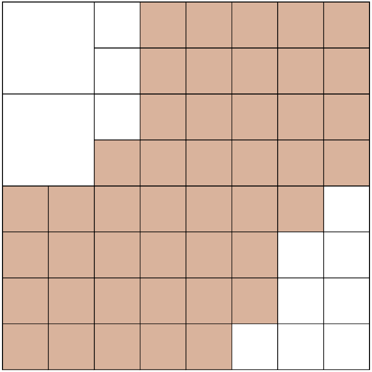


AA

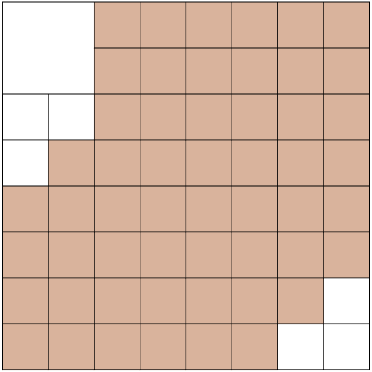
Bounding Implicit Curves with Strips on □

Comparing with IA: method proposed by Lopes *et al.* in SIBGRAPI 2001

- requires the evaluation ∇f using IA and automatic differentiation
- adaptive quadtree



IA



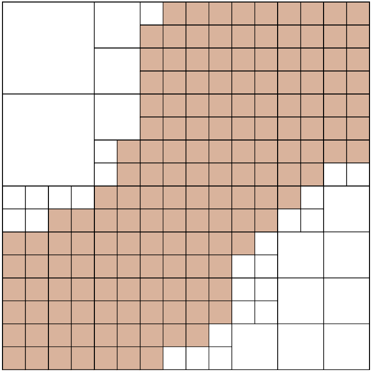
AA

level 3

Bounding Implicit Curves with Strips on □

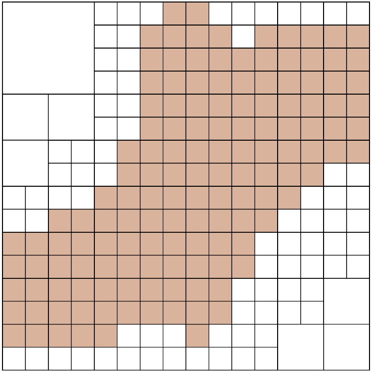
Comparing with IA: method proposed by Lopes *et al.* in SIBGRAPI 2001

- requires the evaluation ∇f using IA and automatic differentiation
- adaptive quadtree



IA

level 4

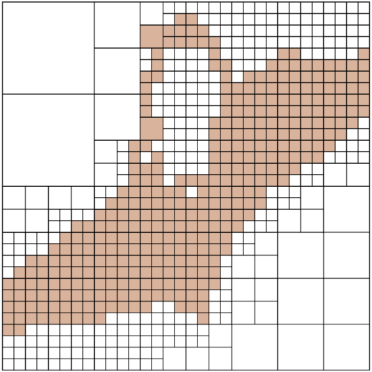


AA

Bounding Implicit Curves with Strips on □

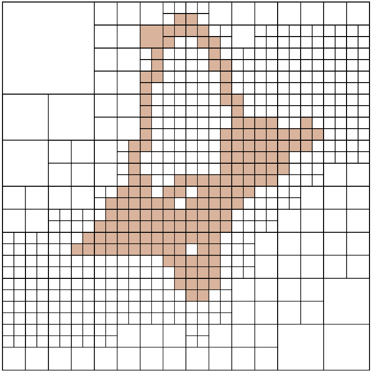
Comparing with IA: method proposed by Lopes *et al.* in SIBGRAPI 2001

- requires the evaluation ∇f using IA and automatic differentiation
- adaptive quadtree



IA

level 5

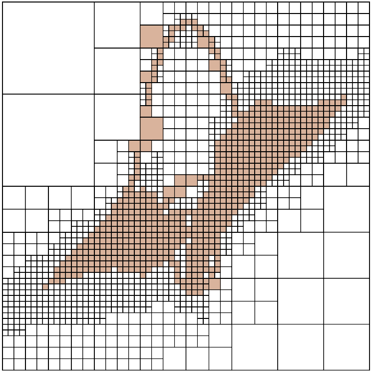


AA

Bounding Implicit Curves with Strips on $\square$

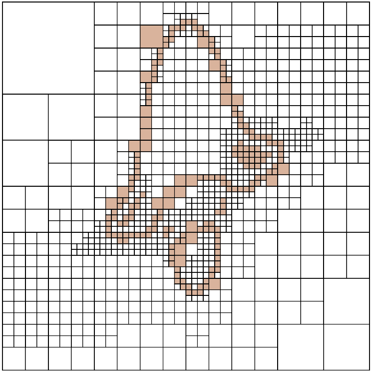
Comparing with IA: method proposed by Lopes *et al.* in SIBGRAPI 2001

- requires the evaluation ∇f using IA and automatic differentiation
- adaptive quadtree



IA

level 6

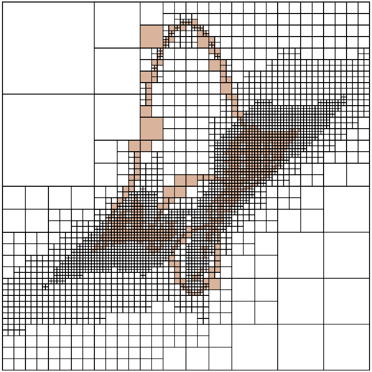


AA

Bounding Implicit Curves with Strips on □

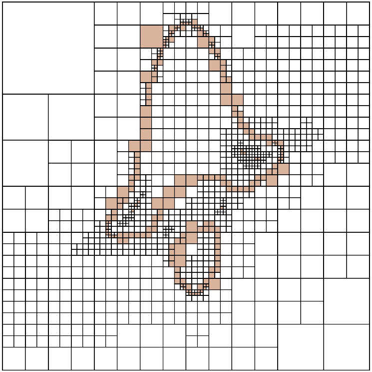
Comparing with IA: method proposed by Lopes *et al.* in SIBGRAPI 2001

- requires the evaluation ∇f using IA and automatic differentiation
- adaptive quadtree



IA

level 7

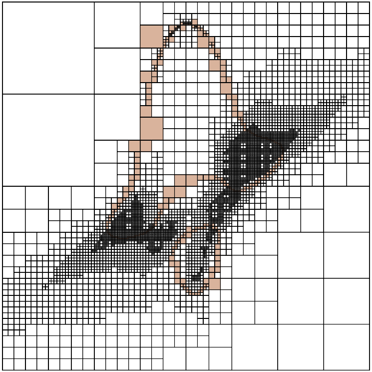


AA

Bounding Implicit Curves with Strips on □

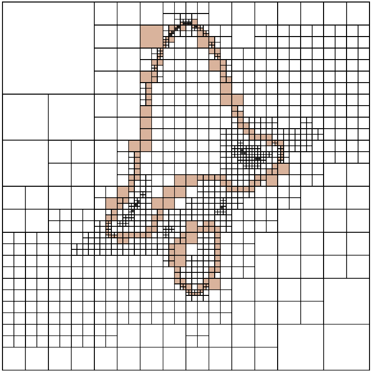
Comparing with IA: method proposed by Lopes *et al.* in SIBGRAPI 2001

- requires the evaluation ∇f using IA and automatic differentiation
- adaptive quadtree



IA

level 8

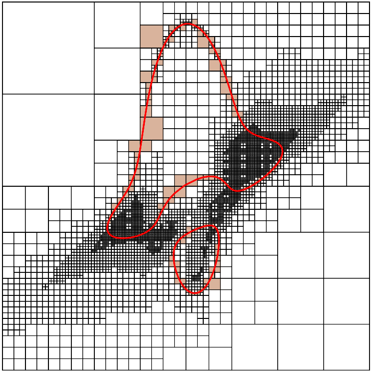


AA

Bounding Implicit Curves with Strips on □

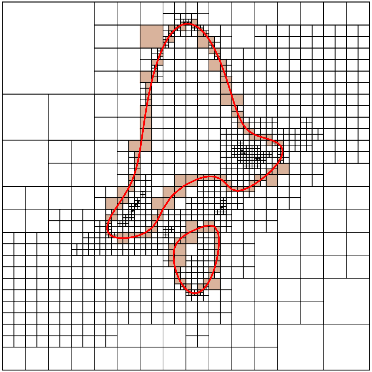
Comparing with IA: method proposed by Lopes *et al.* in SIBGRAPI 2001

- requires the evaluation ∇f using IA and automatic differentiation
- adaptive quadtree



IA

level 8

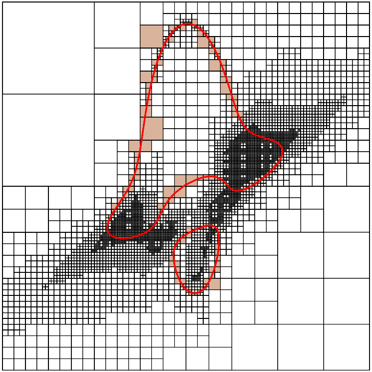


AA

Bounding Implicit Curves with Strips on $\square$

Comparing with IA: method proposed by Lopes *et al.* in SIBGRAPI 2001

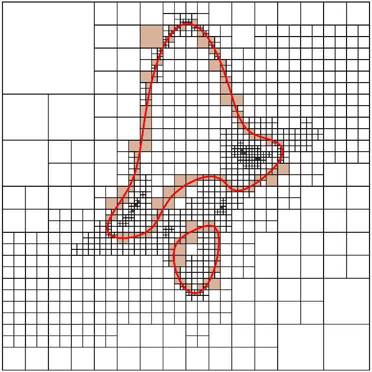
- requires the evaluation ∇f using IA and automatic differentiation
- adaptive quadtree



IA

#cells visited: 6997

level 8



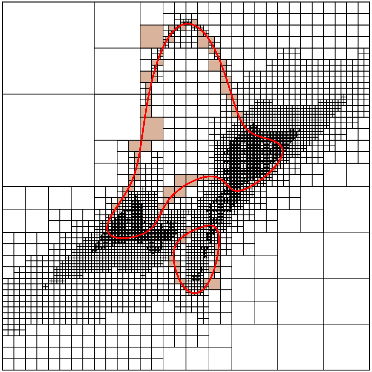
AA

#cells visited: 1697

Bounding Implicit Curves with Strips on □

Comparing with IA: method proposed by Lopes *et al.* in SIBGRAPI 2001

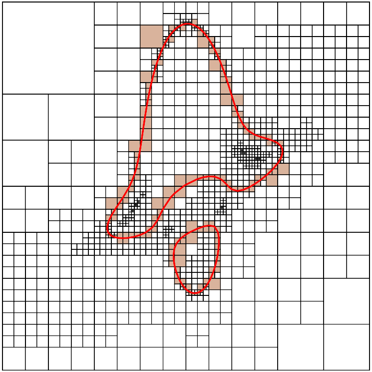
- requires the evaluation ∇f using IA and automatic differentiation
- adaptive quadtree



IA

#leaves: 341

level 8



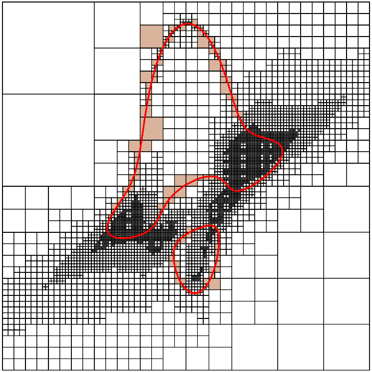
AA

#leaves: 221

Bounding Implicit Curves with Strips on □

Comparing with IA: method proposed by Lopes *et al.* in SIBGRAPI 2001

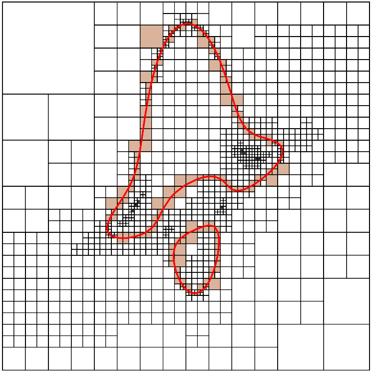
- requires the evaluation ∇f using IA and automatic differentiation
- adaptive quadtree



IA

CPU time: 394 msec

level 8



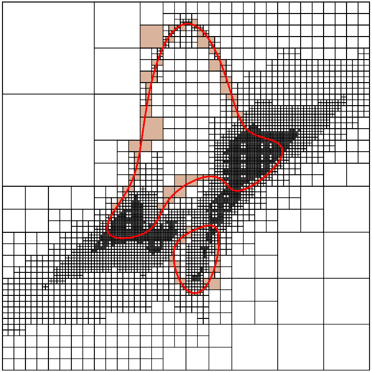
AA

CPU time: 139 msec

Bounding Implicit Curves with Strips on □

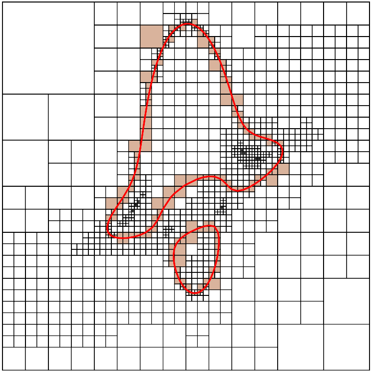
Comparing with IA: method proposed by Lopes *et al.* in SIBGRAPI 2001

- requires the evaluation ∇f using IA and automatic differentiation
- adaptive quadtree



IA
linear convergence

level 8



AA
quadratic convergence

Bounding Implicit Curves with Strips on $\Diamond$

On parallelograms:

Bounding Implicit Curves with Strips on $\Diamond$

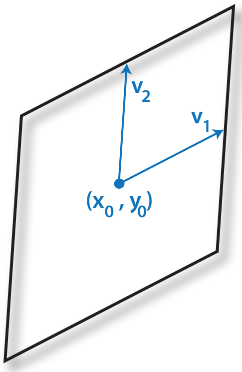
On parallelograms:

evaluate $f(\Diamond)$ with AA $\Rightarrow$ write ε_1 and ε_2 in terms of x and y

Bounding Implicit Curves with Strips on $\lozenge$

On parallelograms:

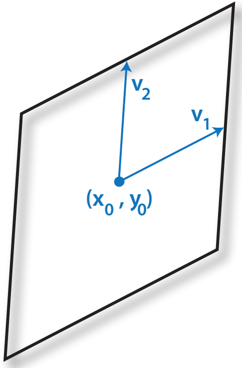
evaluate $f(\lozenge)$ with AA $\Rightarrow$ write ε_1 and ε_2 in terms of x and y



Bounding Implicit Curves with Strips on $\Diamond$

On parallelograms:

evaluate $f(\Diamond)$ with AA $\Rightarrow$ write ε_1 and ε_2 in terms of x and y

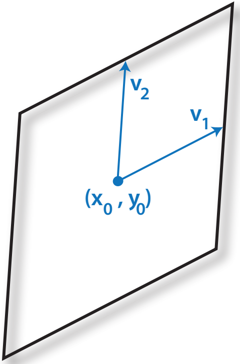


$$v_1 = (x_1, y_1) \qquad v_2 = (x_2, y_2)$$

Bounding Implicit Curves with Strips on $\Diamond$

On parallelograms:

evaluate $f(\Diamond)$ with AA $\Rightarrow$ write ε_1 and ε_2 in terms of x and y



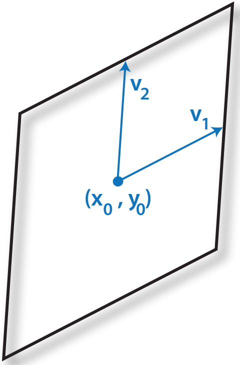
$$v_1 = (x_1, y_1) \qquad v_2 = (x_2, y_2)$$

$$\hat{x} = x_0 + x_1\varepsilon_1 + x_2\varepsilon_2 \qquad \hat{y} = y_0 + y_1\varepsilon_1 + y_2\varepsilon_2$$

Bounding Implicit Curves with Strips on $\Diamond$

On parallelograms:

evaluate $f(\Diamond)$ with AA $\Rightarrow$ write ε_1 and ε_2 in terms of x and y



$$v_1 = (x_1, y_1) \qquad v_2 = (x_2, y_2)$$

$$\hat{x} = x_0 + x_1\varepsilon_1 + x_2\varepsilon_2 \qquad \hat{y} = y_0 + y_1\varepsilon_1 + y_2\varepsilon_2$$

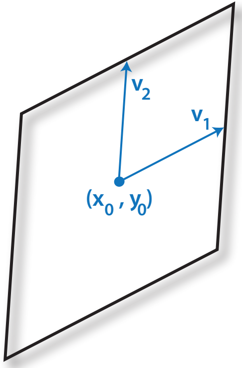
In matrix form

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x_0 \\ y_0 \end{bmatrix} + \begin{bmatrix} x_1 & x_2 \\ y_1 & y_2 \end{bmatrix} \cdot \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \end{bmatrix}$$

Bounding Implicit Curves with Strips on $\Diamond$

On parallelograms:

evaluate $f(\Diamond)$ with AA $\Rightarrow$ write ε_1 and ε_2 in terms of x and y



$$v_1 = (x_1, y_1) \qquad v_2 = (x_2, y_2)$$

$$\hat{x} = x_0 + x_1\varepsilon_1 + x_2\varepsilon_2 \qquad \hat{y} = y_0 + y_1\varepsilon_1 + y_2\varepsilon_2$$

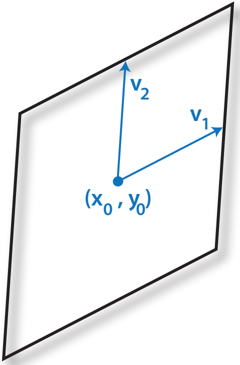
In matrix form

$$\begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \end{bmatrix} = \begin{bmatrix} x_1 & x_2 \\ y_1 & y_2 \end{bmatrix}^{-1} \cdot \begin{bmatrix} x - x_0 \\ y - y_0 \end{bmatrix}$$

Bounding Implicit Curves with Strips on $\Diamond$

On parallelograms:

evaluate $f(\Diamond)$ with AA $\Rightarrow$ write ε_1 and ε_2 in terms of x and y



$$v_1 = (x_1, y_1) \qquad v_2 = (x_2, y_2)$$

$$\hat{x} = x_0 + x_1\varepsilon_1 + x_2\varepsilon_2 \qquad \hat{y} = y_0 + y_1\varepsilon_1 + y_2\varepsilon_2$$

In matrix form

$$\begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \end{bmatrix} = \begin{bmatrix} x_1 & x_2 \\ y_1 & y_2 \end{bmatrix}^{-1} \cdot \begin{bmatrix} x - x_0 \\ y - y_0 \end{bmatrix}$$

the matrix is invertible $\iff$ the parallelogram is not degenerate

Bounding Implicit Curves with Strips on $\triangle$

On triangles:

Bounding Implicit Curves with Strips on $\triangle$

On triangles: replace the evaluation of $f(\triangle) \Rightarrow f(\Diamond)$ with AA

Bounding Implicit Curves with Strips on $\triangle$

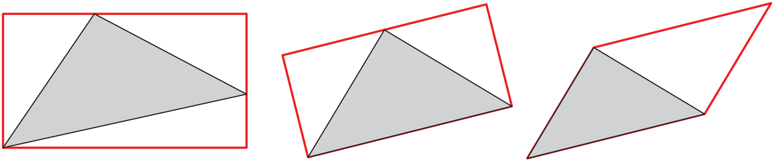
On triangles: replace the evaluation of $f(\triangle) \Rightarrow f(\Diamond)$ with AA

- include a triangle into a parallelogram

Bounding Implicit Curves with Strips on $\triangle$

On triangles: replace the evaluation of $f(\triangle) \Rightarrow f(\Diamond)$ with AA

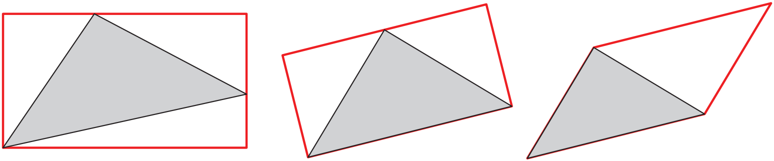
- include a triangle into a parallelogram



Bounding Implicit Curves with Strips on $\triangle$

On triangles: replace the evaluation of $f(\triangle) \Rightarrow f(\Diamond)$ with AA

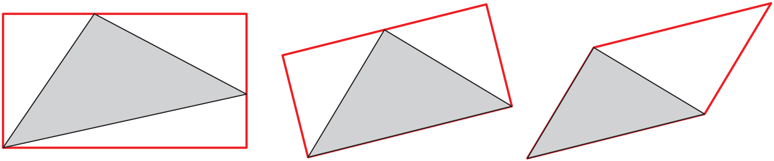
- include a triangle into a parallelogram
  - evaluate f outside of its domain



Bounding Implicit Curves with Strips on $\triangle$

On triangles: replace the evaluation of $f(\triangle) \Rightarrow f(\lozenge)$ with AA

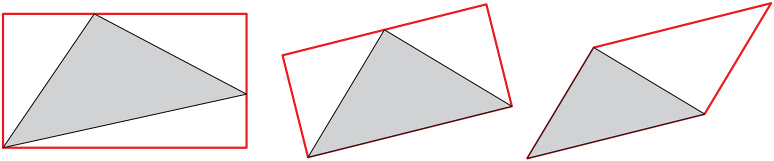
- include a triangle into a parallelogram
  - evaluate f outside of its domain
  - it does not work for surfaces



Bounding Implicit Curves with Strips on $\triangle$

On triangles: replace the evaluation of $f(\triangle) \Rightarrow f(\Diamond)$ with AA

- include a triangle into a parallelogram
  - evaluate f outside of its domain
  - it does not work for surfaces

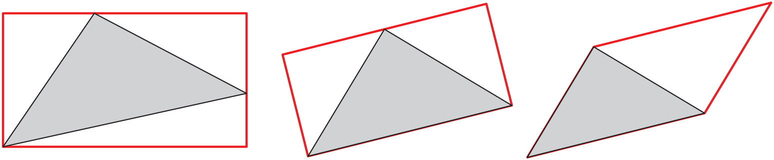


- split a triangle in three parallelograms

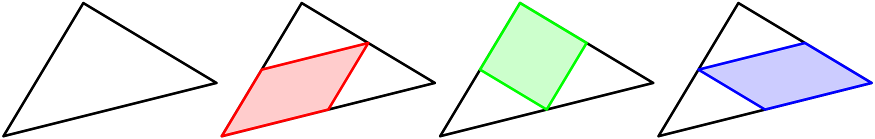
Bounding Implicit Curves with Strips on $\triangle$

On triangles: replace the evaluation of $f(\triangle) \Rightarrow f(\Diamond)$ with AA

- include a triangle into a parallelogram
  - evaluate f outside of its domain
  - it does not work for surfaces



- split a triangle in three parallelograms



Our Adaptive Method

procedure *Explore*($\triangle$)
  $\Diamond_1, \Diamond_2, \Diamond_3 \leftarrow$ *Parallelograms*($\triangle$)
  $\hat{f}_i \leftarrow f(\Diamond_i)$ with AA
  if $0 \in [\hat{f}_i]$ for some i **then**
    $w_i \leftarrow$ width of $\hat{f}$ in $\Diamond_i$
    if $w_i \leq \epsilon_{user}$, for all i **then**
      Approximate($\triangle$)
    else
      $\triangle_i \leftarrow$ *Subdivide*($\triangle$)
      for each i, *Explore*($\triangle_i$)
    end
  end
end

Our Adaptive Method

procedure *Explore*$(\triangle)$

$\Diamond_1, \Diamond_2, \Diamond_3 \leftarrow$ *Parallelograms*$(\triangle)$

$\hat{f}_i \leftarrow f(\Diamond_i)$ with AA

if $0 \in [\hat{f}_i]$ for some i **then**

$w_i \leftarrow$ width of $\hat{f}$ in $\Diamond_i$

if $w_i \leq \epsilon_{user}$, for all i **then**

Approximate$(\triangle)$

else

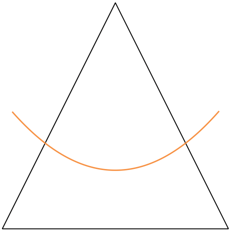
$\triangle_i \leftarrow$ *Subdivide*$(\triangle)$

for each i, *Explore*$(\triangle_i)$

end

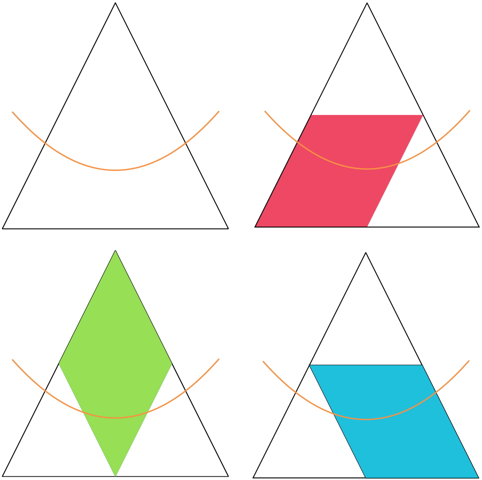
end

end



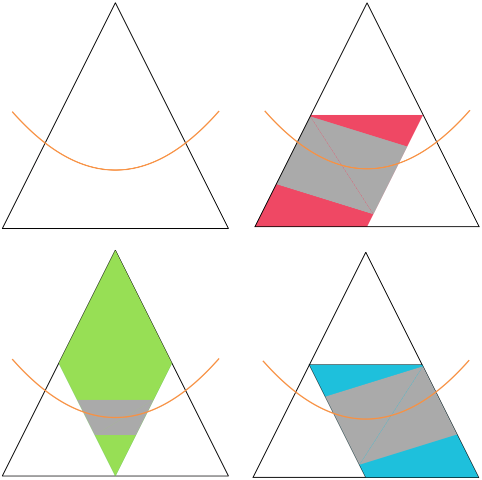
Our Adaptive Method

procedure $\mathit{Explore}(\triangle)$
  $\lozenge_1, \lozenge_2, \lozenge_3 \leftarrow \mathit{Parallelograms}(\triangle)$
  $\hat{f}_i \leftarrow f(\lozenge_i)$ with AA
  if $0 \in [\hat{f}_i]$ for some i **then**
    $w_i \leftarrow$ width of $\hat{f}$ in $\lozenge_i$
    if $w_i \leq \epsilon_{user}$, for all i **then**
      $\mathit{Approximate}(\triangle)$
    else
      $\triangle_i \leftarrow \mathit{Subdivide}(\triangle)$
      for each i, $\mathit{Explore}(\triangle_i)$
    end
  end
end



Our Adaptive Method

procedure $\mathit{Explore}(\triangle)$
  $\Diamond_1, \Diamond_2, \Diamond_3 \leftarrow \mathit{Parallelograms}(\triangle)$
  $\hat{f}_i \leftarrow f(\Diamond_i)$ with AA
  if $0 \in [\hat{f}_i]$ for some i **then**
    $w_i \leftarrow$ width of $\hat{f}$ in $\Diamond_i$
    if $w_i \leq \epsilon_{user}$, for all i **then**
      $\mathit{Approximate}(\triangle)$
    else
      $\triangle_i \leftarrow \mathit{Subdivide}(\triangle)$
      for each i, $\mathit{Explore}(\triangle_i)$
    end
  end
end



Our Adaptive Method

procedure *Explore*$(\triangle)$

$\Diamond_1, \Diamond_2, \Diamond_3 \leftarrow$ *Parallelograms*$(\triangle)$

$\hat{f}_i \leftarrow f(\Diamond_i)$ with AA

if $0 \in [\hat{f}_i]$ for some i **then**

$w_i \leftarrow$ width of $\hat{f}$ in $\Diamond_i$

if $w_i \leq \epsilon_{user}$, for all i **then**

Approximate$(\triangle)$

else

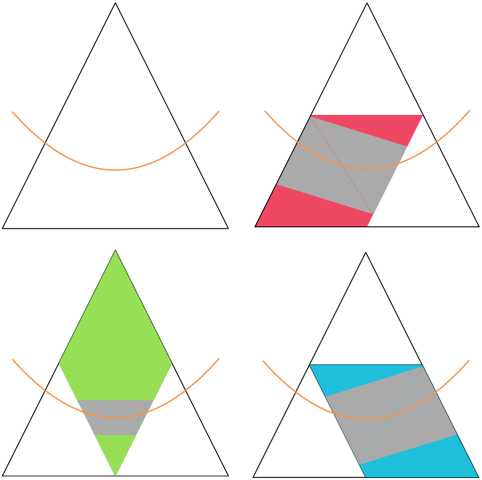
$\triangle_i \leftarrow$ *Subdivide*$(\triangle)$

for each i, *Explore*$(\triangle_i)$

end

end

end



Our Adaptive Method

procedure *Explore*$(\triangle)$

$\Diamond_1, \Diamond_2, \Diamond_3 \leftarrow$ *Parallelograms*$(\triangle)$

$\hat{f}_i \leftarrow f(\Diamond_i)$ with AA

if $0 \in [\hat{f}_i]$ for some i **then**

$w_i \leftarrow$ width of $\hat{f}$ in $\Diamond_i$

if $w_i \leq \epsilon_{user}$, for all i **then**

Approximate$(\triangle)$

else

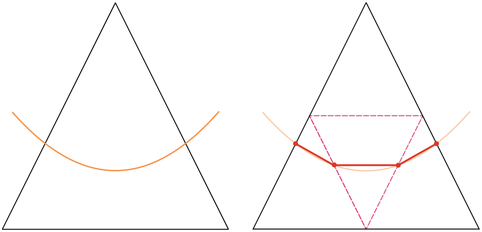
$\triangle_i \leftarrow$ *Subdivide*$(\triangle)$

for each i, *Explore*$(\triangle_i)$

end

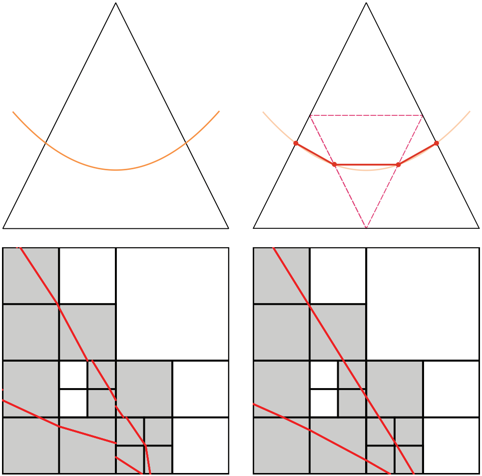
end

end



Our Adaptive Method

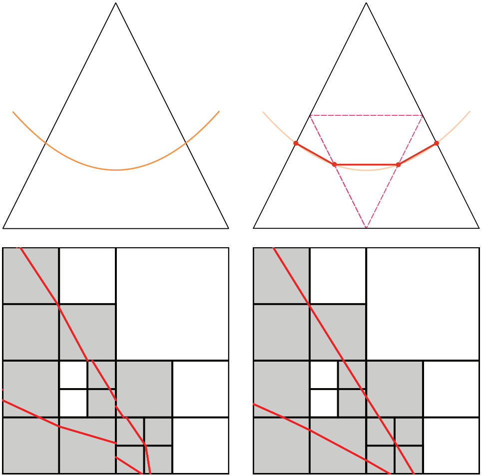
procedure $\mathit{Explore}(\triangle)$
    $\lozenge_1, \lozenge_2, \lozenge_3 \leftarrow \mathit{Parallelograms}(\triangle)$
    $\hat{f}_i \leftarrow f(\lozenge_i)$ with AA
    if $0 \in [\hat{f}_i]$ for some i **then**
        $w_i \leftarrow$ width of $\hat{f}$ in $\lozenge_i$
        if $w_i \leq \epsilon_{user}$, for all i **then**
            $\mathit{Approximate}(\triangle)$
        else
            $\triangle_i \leftarrow \mathit{Subdivide}(\triangle)$
            for each i, $\mathit{Explore}(\triangle_i)$
        end
    end
end



linear interpolation   bissection method

Our Adaptive Method

procedure $\mathit{Explore}(\triangle)$
  $\lozenge_1, \lozenge_2, \lozenge_3 \leftarrow \mathit{Parallelograms}(\triangle)$
  $\hat{f}_i \leftarrow f(\lozenge_i)$ with AA
  if $0 \in [\hat{f}_i]$ for some i **then**
    $w_i \leftarrow$ width of $\hat{f}$ in $\lozenge_i$
    if $w_i \leq \epsilon_{user}$, for all i **then**
      $\mathit{Approximate}(\triangle)$
    else
      $\triangle_i \leftarrow \mathit{Subdivide}(\triangle)$
      for each i, $\mathit{Explore}(\triangle_i)$
    end
  end
end



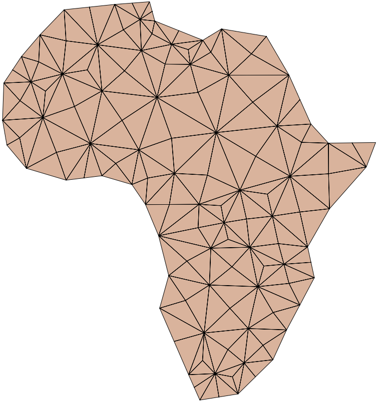
linear interpolation   bissection method

Our Adaptive Method

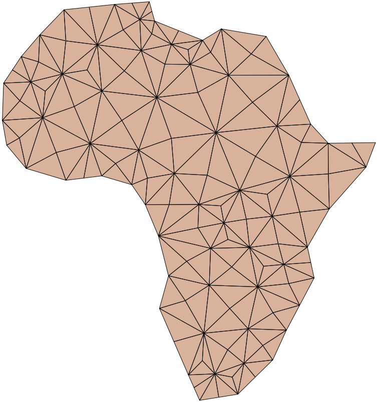
Our method does not care what mesh subdivision method is used

Our Adaptive Method

Our method does not care what mesh subdivision method is used



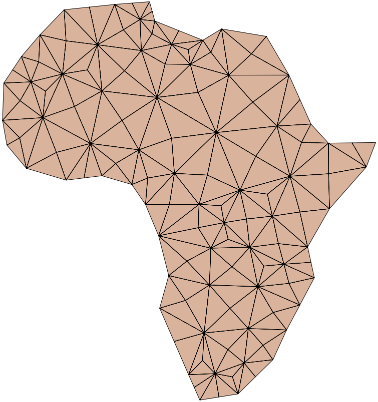
triangle soup



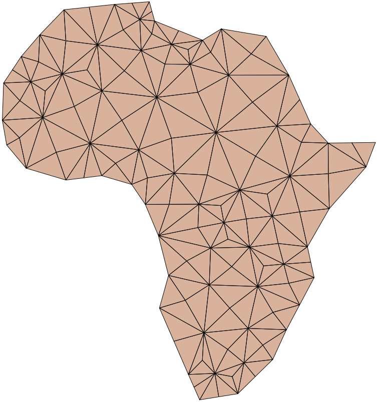
mesh with connecticity

Our Adaptive Method

Our method does not care what mesh subdivision method is used



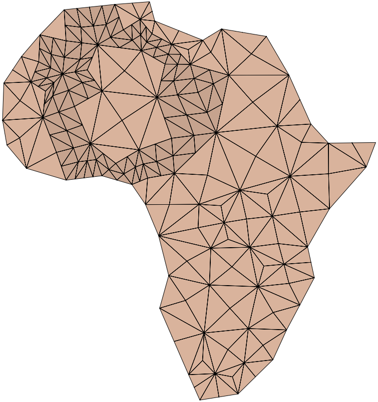
triangle soup
midpoint splitting



mesh with connecticity
$\sqrt{3}$, J_1^a, 4-8 meshes, ...

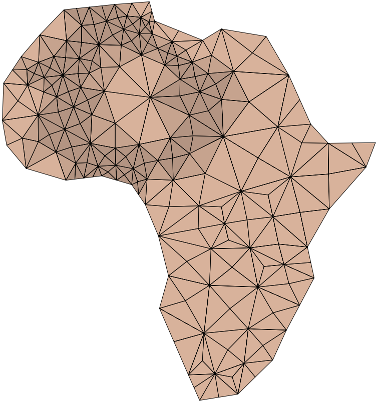
Our Adaptive Method

Our method does not care what mesh subdivision method is used



triangle soup

$\#\triangle = 193$

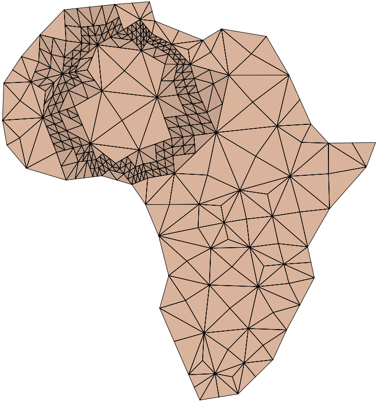


mesh with connecticity

$\#\triangle = 193$

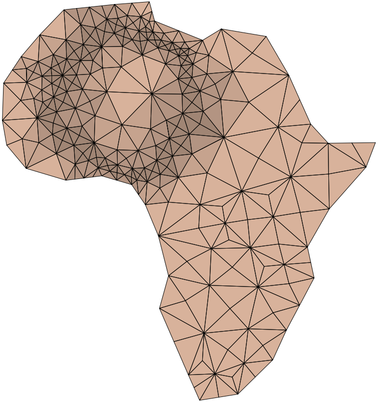
Our Adaptive Method

Our method does not care what mesh subdivision method is used



triangle soup

$\#\triangle = 307$

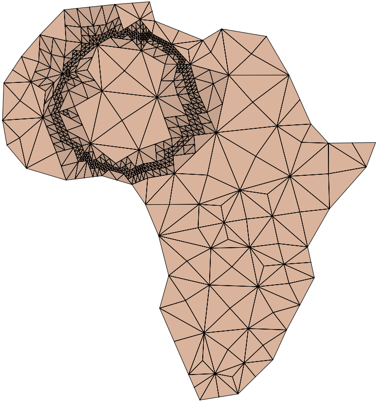


mesh with connecticity

$\#\triangle = 325$

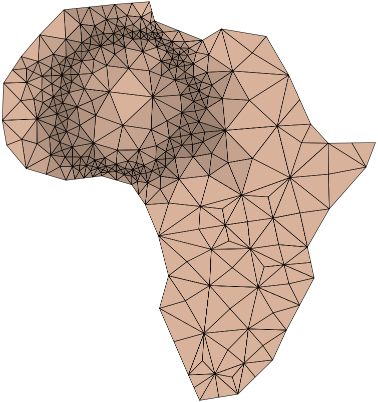
Our Adaptive Method

Our method does not care what mesh subdivision method is used



triangle soup

$\#\triangle = 512$

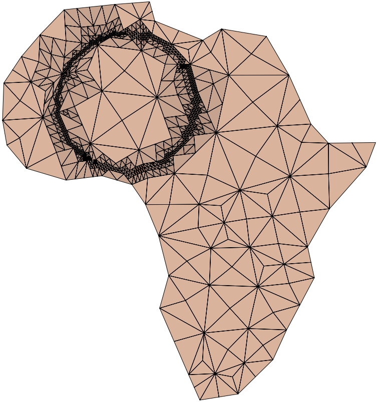


mesh with connecticity

$\#\triangle = 427$

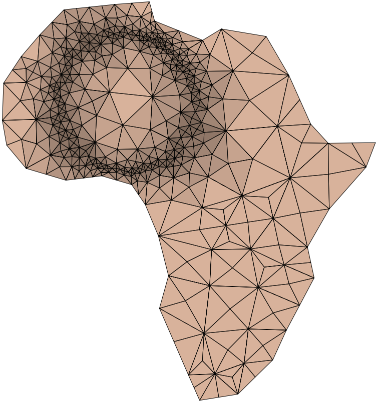
Our Adaptive Method

Our method does not care what mesh subdivision method is used



triangle soup

$\#\triangle = 922$

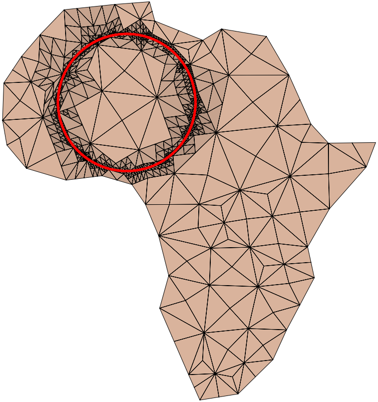


mesh with connecticity

$\#\triangle = 574$

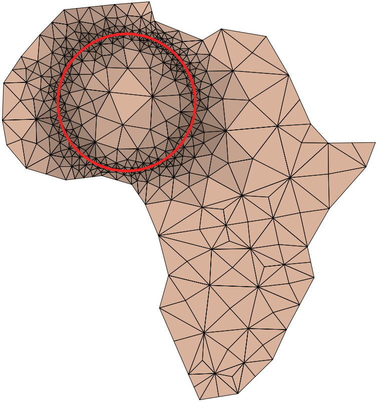
Our Adaptive Method

Our method does not care what mesh subdivision method is used



triangle soup

$\#\triangle = 1384$

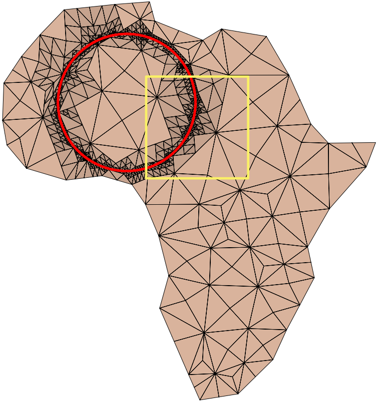


mesh with connecticity

$\#\triangle = 779$

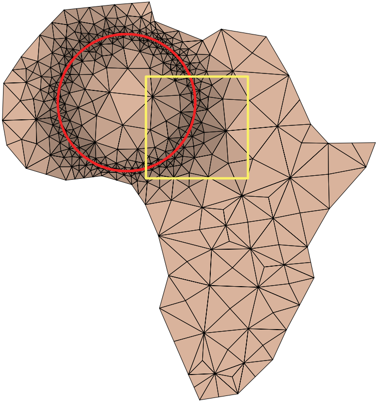
Our Adaptive Method

Our method does not care what mesh subdivision method is used



triangle soup

$\# \triangle = 1384$

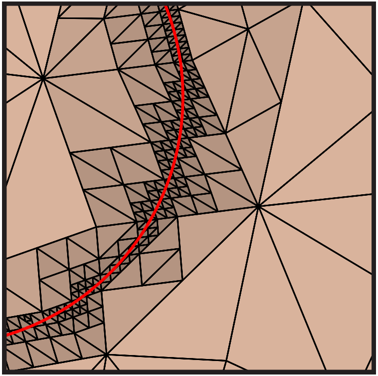


mesh with connecticity

$\# \triangle = 779$

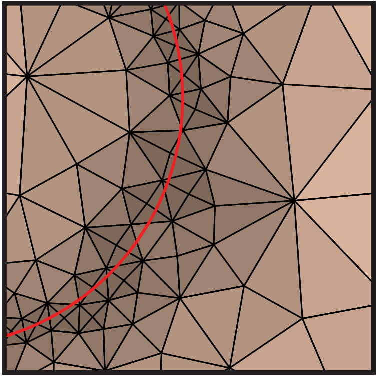
Our Adaptive Method

Our method does not care what mesh subdivision method is used



triangle soup

$\#\triangle = 1384$

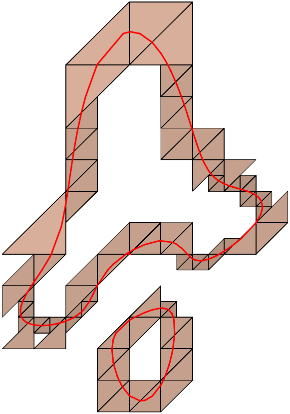


mesh with connecticity

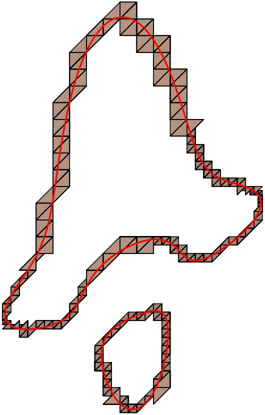
$\#\triangle = 779$

Our Adaptive Method

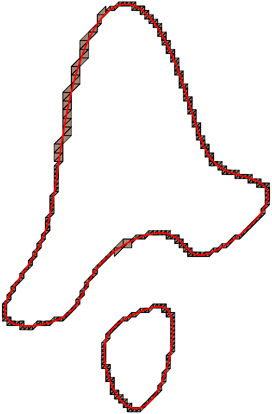
The effect of the geometric criteria on the curve in a triangular quadtree



$\epsilon_{user} = 1$

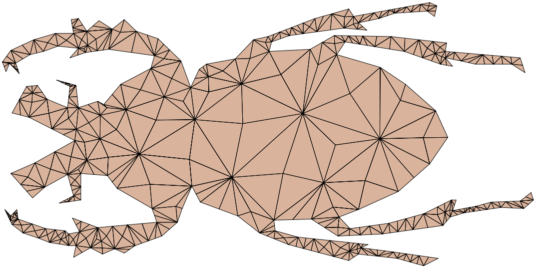


$\epsilon_{user} = 0.1$



$\epsilon_{user} = 0.01$

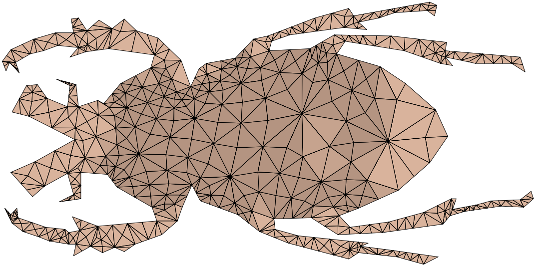
Results



level 0

$$(x+1)^3(1-x)-4y^4=0$$

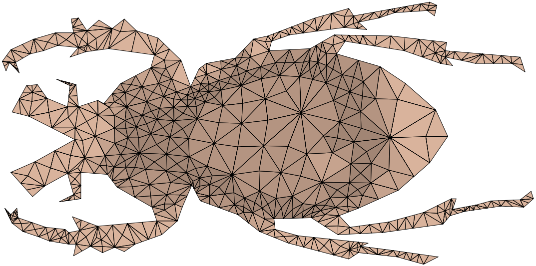
Results



level 1

$$(x+1)^3(1-x) - 4y^4 = 0$$

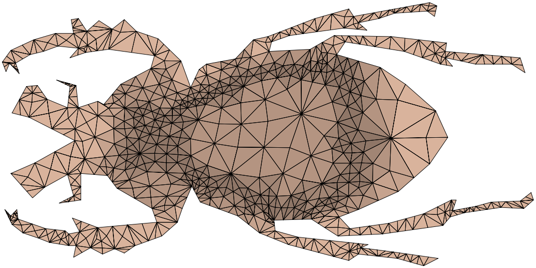
Results



level 2

$$(x+1)^3(1-x)-4y^4=0$$

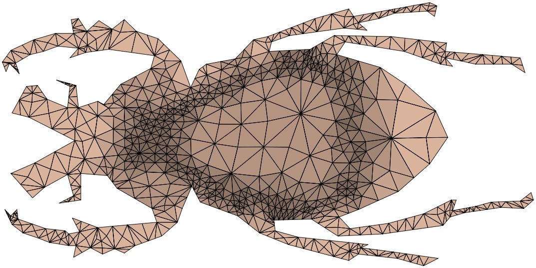
Results



level 3

$$(x+1)^3(1-x) - 4y^4 = 0$$

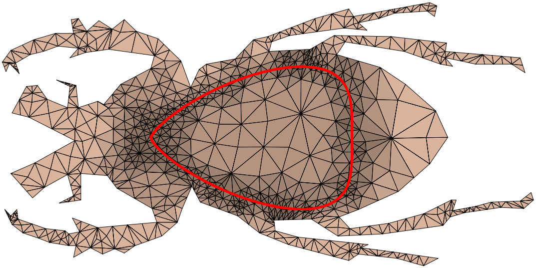
Results



level 4

$$(x+1)^3(1-x) - 4y^4 = 0$$

Results



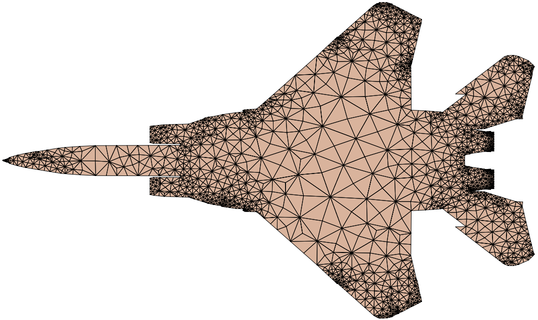
level 4

$\#\triangle_{in} = 940$
$\#\triangle_{out} = 1771$

CPU time = 280 msec

$$(x+1)^3(1-x) - 4y^4 = 0$$

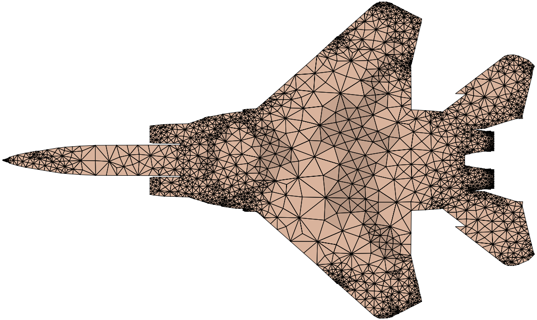
Results



level 0

$$x^3 + x - y^2 = 0$$

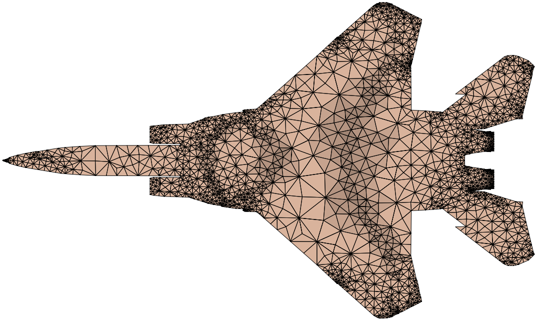
Results



level 1

$$x^3 + x - y^2 = 0$$

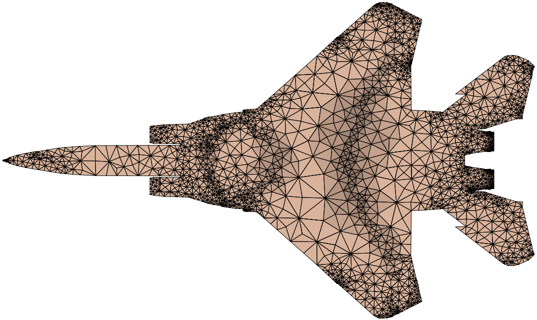
Results



level 2

$$x^3 + x - y^2 = 0$$

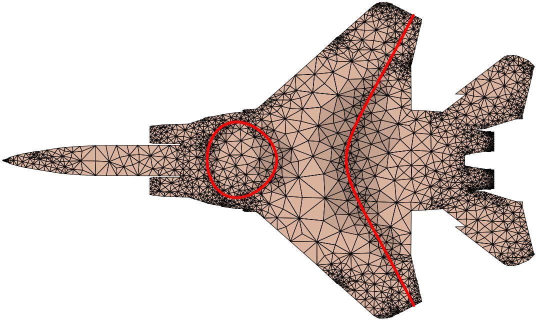
Results



level 3

$$x^3 + x - y^2 = 0$$

Results



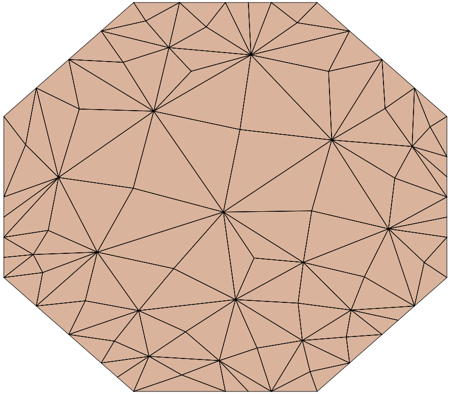
level 3

$\#\triangle_{in} = 3530$
$\#\triangle_{out} = 4142$

CPU time = 333 msec

$x^3 + x - y^2 = 0$

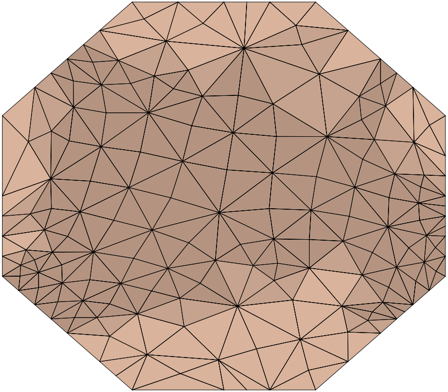
Results



level 0

$$y^2(0.75^2 - x^2) - (x^2 + 1.5y - 0.75^2)^2 = 0$$

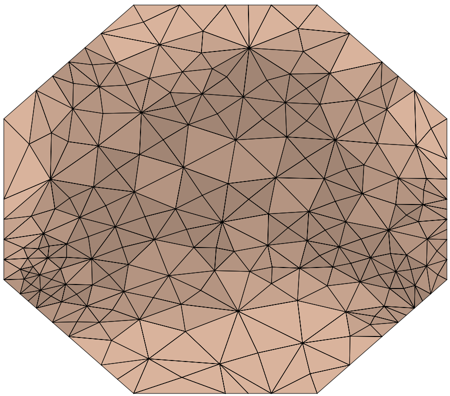
Results



level 1

$$y^2(0.75^2 - x^2) - (x^2 + 1.5y - 0.75^2)^2 = 0$$

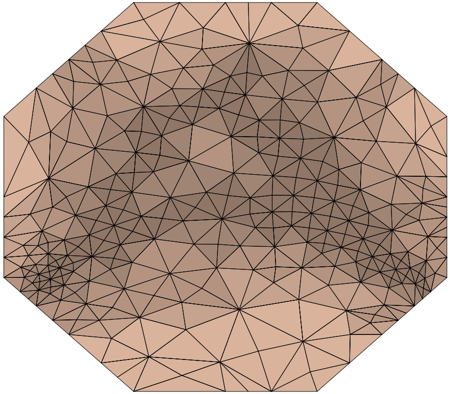
Results



level 2

$$y^2(0.75^2 - x^2) - (x^2 + 1.5y - 0.75^2)^2 = 0$$

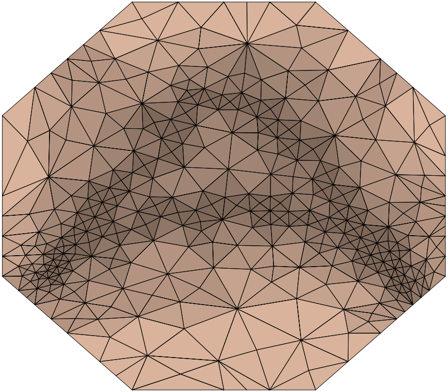
Results



level 3

$$y^2(0.75^2 - x^2) - (x^2 + 1.5y - 0.75^2)^2 = 0$$

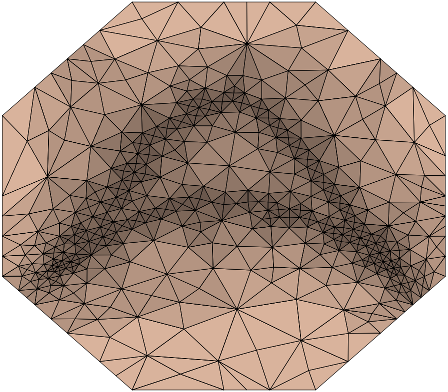
Results



level 4

$$y^2(0.75^2 - x^2) - (x^2 + 1.5y - 0.75^2)^2 = 0$$

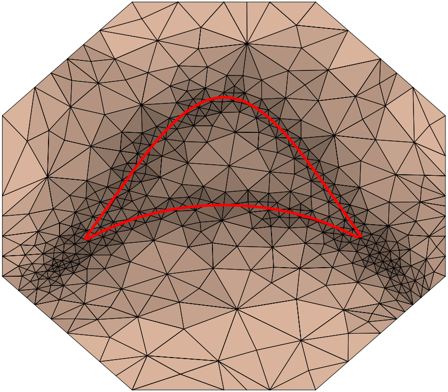
Results



level 5

$$y^2(0.75^2 - x^2) - (x^2 + 1.5y - 0.75^2)^2 = 0$$

Results



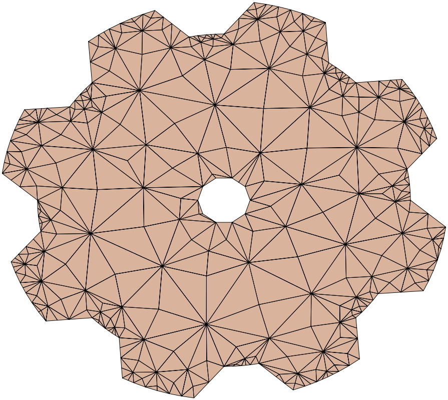
level 5

$\#\triangle_{in} = 126$
$\#\triangle_{out} = 1168$

CPU time = 123 msec

$$y^2(0.75^2 - x^2) - (x^2 + 1.5y - 0.75^2)^2 = 0$$

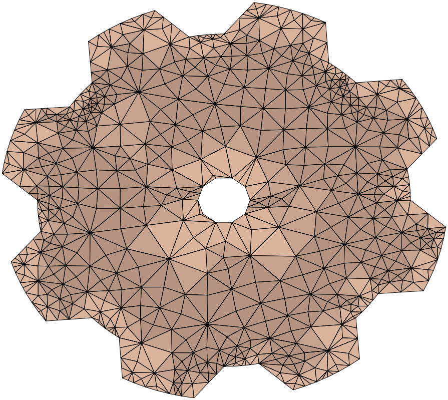
Results



level 0

$$(x^2 + y^2 - 1)^3 - x^2y^3 = 0$$

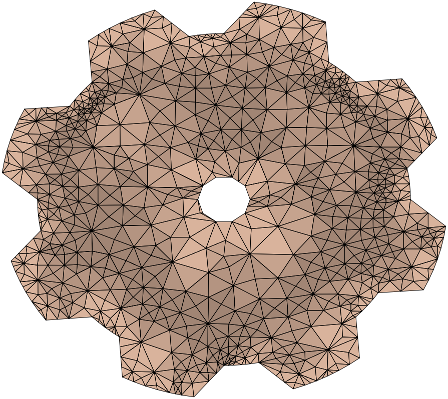
Results



level 1

$$(x^2 + y^2 - 1)^3 - x^2 y^3 = 0$$

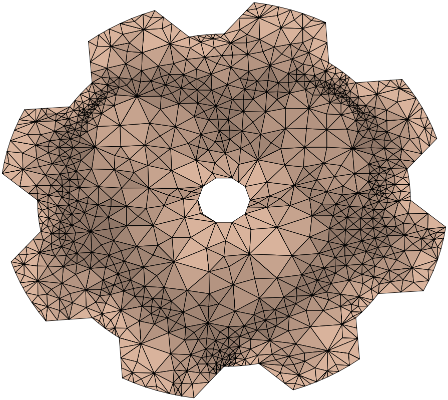
Results



level 2

$$(x^2 + y^2 - 1)^3 - x^2y^3 = 0$$

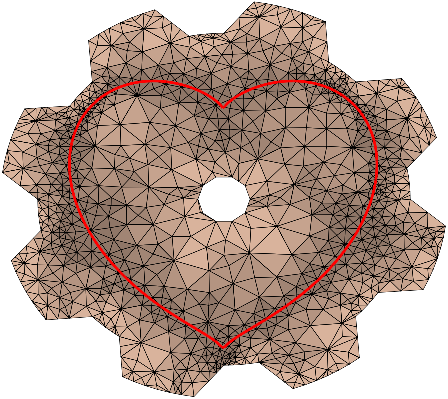
Results



level 3

$$(x^2 + y^2 - 1)^3 - x^2y^3 = 0$$

Results



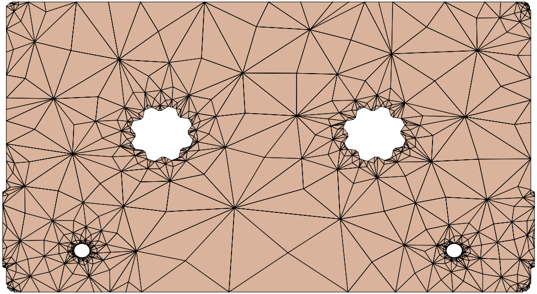
$$(x^2 + y^2 - 1)^3 - x^2y^3 = 0$$

level 3

$\#\triangle_{in} = 1424$
$\#\triangle_{out} = 3298$

CPU time = 547 msec

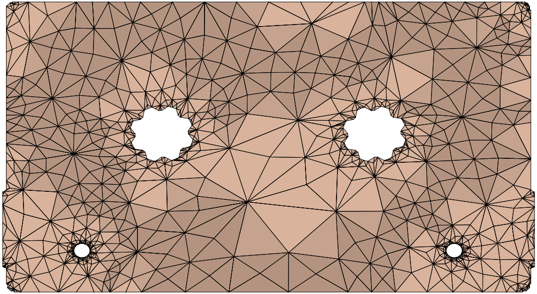
Results



level 0

$$(y - x^2 + 1)^4 + (x^2 + y^2)^4 - 1 = 0$$

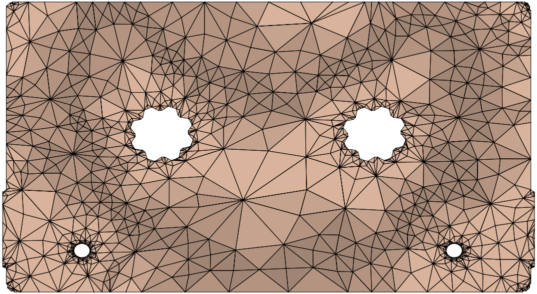
Results



level 1

$$(y - x^2 + 1)^4 + (x^2 + y^2)^4 - 1 = 0$$

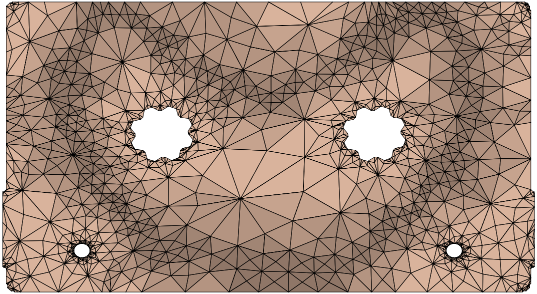
Results



level 2

$$(y - x^2 + 1)^4 + (x^2 + y^2)^4 - 1 = 0$$

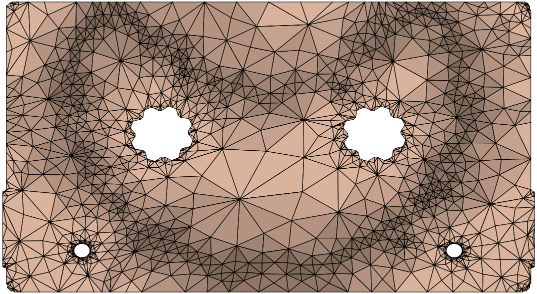
Results



level 3

$$(y - x^2 + 1)^4 + (x^2 + y^2)^4 - 1 = 0$$

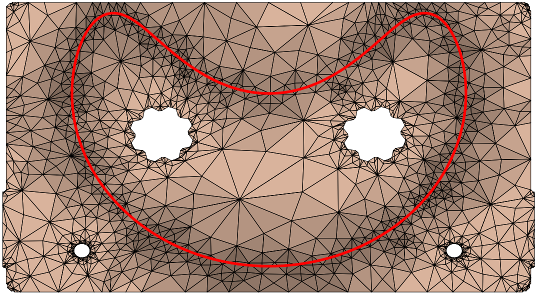
Results



level 4

$$(y - x^2 + 1)^4 + (x^2 + y^2)^4 - 1 = 0$$

Results



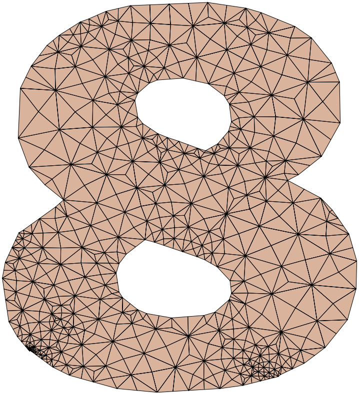
level 4

$\#\triangle_{in} = 1006$
$\#\triangle_{out} = 2134$

CPU time = 391 msec

$$(y - x^2 + 1)^4 + (x^2 + y^2)^4 - 1 = 0$$

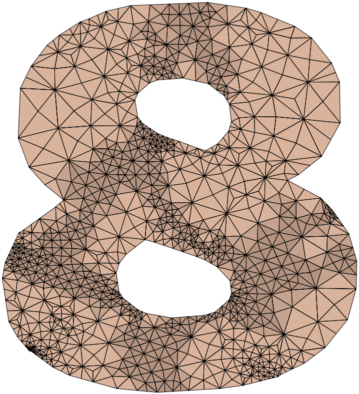
Results



level 0

$$(xy + \cos(x+y))(xy + \sin(x+y)) = 0$$

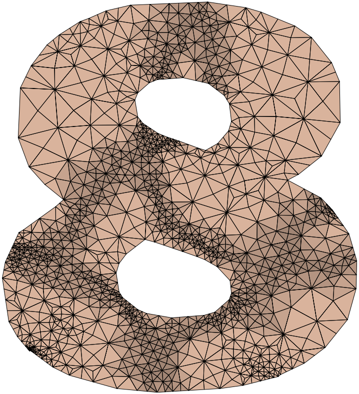
Results



level 1

$$(xy + \cos(x + y))(xy + \sin(x + y)) = 0$$

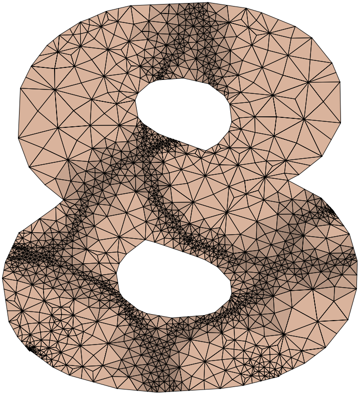
Results



level 2

$$(xy + \cos(x + y))(xy + \sin(x + y)) = 0$$

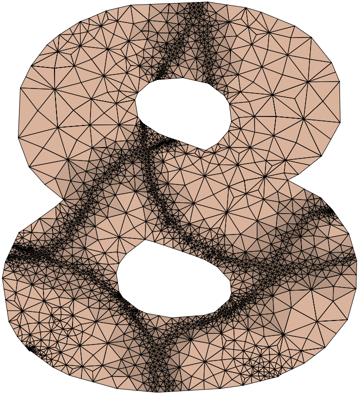
Results



level 3

$$(xy + \cos(x+y))(xy + \sin(x+y)) = 0$$

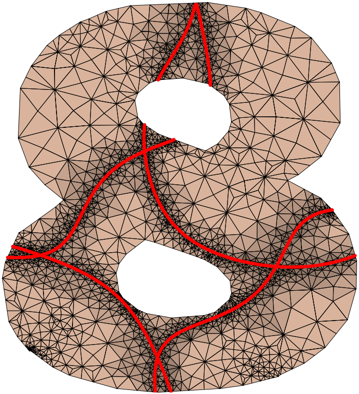
Results



level 4

$$(xy + \cos(x + y))(xy + \sin(x + y)) = 0$$

Results



level 4

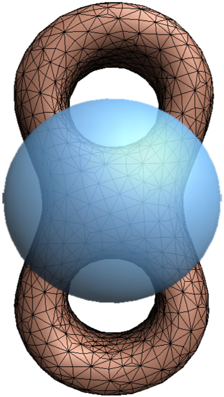
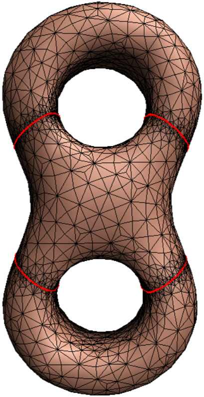
$\#\triangle_{in} = 1032$
$\#\triangle_{out} = 3897$

CPU time $= 454$ msec

$$(xy + \cos(x+y))(xy + \sin(x+y)) = 0$$

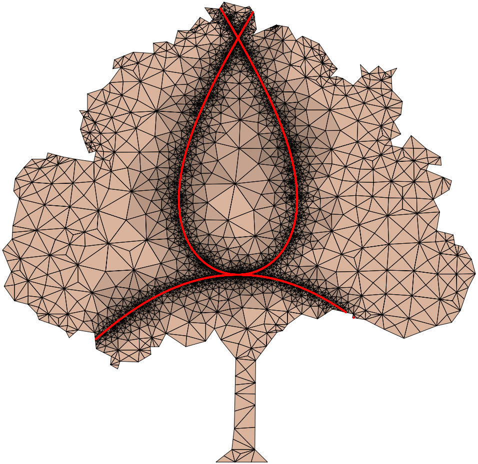
Work in Progress

Implicit curves on surfaces

curve given implicitly by $x^2 + y^2 + z^2 = 1$ on bitorus mesh

Approximating Implicit Curves on Triangulations with Affine Arithmetic



Thanks!