Discretisation of 2D domains, bounded by NURBS curves

Author: Andrej Kolar-Požun
Mentor: Marjetka Knez

29. november 2023

NURBS

- NURBS - Non Uniform Rational B-Splines.

NURBS

- NURBS - Non Uniform Rational B-Splines.
- B-Splines $N_{k,p}(\xi)$ basis for polynomial splines with prescribed smoothness.

NURBS

- NURBS - Non Uniform Rational B-Splines.
- B-Splines $N_{k,p}(\xi)$ basis for polynomial splines with prescribed smoothness.
- B-Splines + Control points $\rightarrow$ B-Spline Curve (special case - Bézier curve).

NURBS

- NURBS - Non Uniform Rational B-Splines.
- B-Splines $N_{k,p}(\xi)$ basis for polynomial splines with prescribed smoothness.
- B-Splines $+$ Control points $\rightarrow$ B-Spline Curve (special case - Bézier curve).
- $C(\xi) = \frac{\sum_{k=1}^{n} \mathbf{P}_k w_k N_{k,p}(\xi)}{\sum_{k=1}^{n} w_k N_{k,p}(\xi)}$.

NURBS

- NURBS - Non Uniform Rational B-Splines.
- B-Splines $N_{k,p}(\xi)$ basis for polynomial splines with prescribed smoothness.
- B-Splines + Control points $\rightarrow$ B-Spline Curve (special case - Bézier curve).
- $C(\xi) = \frac{\sum_{k=1}^{n} \mathbf{P}_k w_k N_{k,p}(\xi)}{\sum_{k=1}^{n} w_k N_{k,p}(\xi)}$.
- Multipatch NURBS are also common.

Problem description

- $\partial\Omega = C(I)$ - simple and closed curve.

Problem description

- $\partial\Omega = C(I)$ - simple and closed curve.
- Goal - discretise Ω. Obtain $X \subset \Omega$.

Problem description

- $\partial\Omega = C(I)$ - simple and closed curve.
- Goal - discretise Ω. Obtain $X \subset \Omega$.
- Motivation - Meshless methods and Monte Carlo.

Problem description

- $\partial\Omega = C(I)$ - simple and closed curve.
- Goal - discretise Ω. Obtain $X \subset \Omega$.
- Motivation - Meshless methods and Monte Carlo.
- Requirement 1: Algorithm takes N or h as input.

Problem description

- $\partial\Omega = C(I)$ - simple and closed curve.
- Goal - discretise Ω. Obtain $X \subset \Omega$.
- Motivation - Meshless methods and Monte Carlo.
- Requirement 1: Algorithm takes N or h as input.
- Requirement 2: Quasiuniformness, quasirandomness.

inRS

- inRS[1] algorithm for calculating indicator function of Ω.

[1] Sommariva, Vianello inRS: Implementing the indicator function of NURBS-shaped planar domains.

inRS

- inRS[1] algorithm for calculating indicator function of Ω.
- $C(\xi) = \frac{\sum_{k=1}^{n} \mathbf{P}_k w_k N_{k,p}(\xi)}{\sum_{k=1}^{n} w_k N_{k,p}(\xi)}$.

[1] Sommariva, Vianello inRS: Implementing the indicator function of NURBS-shaped planar domains.

inRS

- inRS[1] algorithm for calculating indicator function of Ω.
- $C(\xi) = \frac{\sum_{k=1}^{n} \mathbf{P}_k w_k N_{k,p}(\xi)}{\sum_{k=1}^{n} w_k N_{k,p}(\xi)}$.
- Main idea - try reduce to the polygon case.

[1] Sommariva, Vianello inRS: Implementing the indicator function of NURBS-shaped planar domains.

inRS

- inRS[1] algorithm for calculating indicator function of Ω.
- $C(\xi) = \frac{\sum_{k=1}^{n} \mathbf{P}_k w_k N_{k,p}(\xi)}{\sum_{k=1}^{n} w_k N_{k,p}(\xi)}$.
- Main idea - try reduce to the polygon case.
- Divide I into I_k such that C restricted to it is a pair of rational functions.

[1] Sommariva, Vianello inRS: Implementing the indicator function of NURBS-shaped planar domains.

inRS

- inRS[1] algorithm for calculating indicator function of Ω.
- $C(\xi) = \frac{\sum_{k=1}^{n} \mathbf{P}_k w_k N_{k,p}(\xi)}{\sum_{k=1}^{n} w_k N_{k,p}(\xi)}$.
- Main idea - try reduce to the polygon case.
- Divide I into I_k such that C restricted to it is a pair of rational functions.
- For each I_k, calculate the roots of the derivatives of the appropriate rational function.

[1] Sommariva, Vianello inRS: Implementing the indicator function of NURBS-shaped planar domains.

inRS

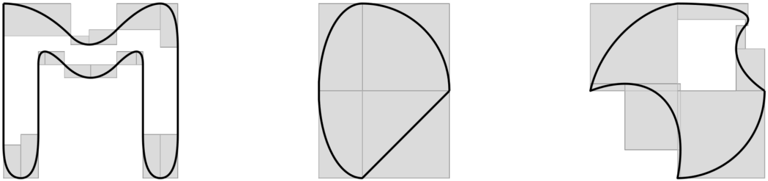
- inRS[1] algorithm for calculating indicator function of Ω.
- $C(\xi) = \frac{\sum_{k=1}^{n} \mathbf{P}_k w_k N_{k,p}(\xi)}{\sum_{k=1}^{n} w_k N_{k,p}(\xi)}$.
- Main idea - try reduce to the polygon case.
- Divide I into I_k such that C restricted to it is a pair of rational functions.
- For each I_k, calculate the roots of the derivatives of the appropriate rational function.
- For each $I_{k,j}$ find $a_{k,j} = \min_{I_{k,j}} \alpha_k(\xi)$

[1] Sommariva, Vianello inRS: Implementing the indicator function of NURBS-shaped planar domains.

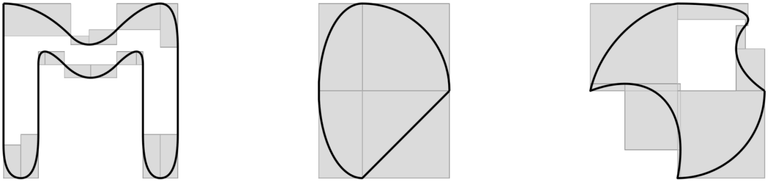
inRS

- inRS[1] algorithm for calculating indicator function of Ω.
- $C(\xi) = \frac{\sum_{k=1}^{n} \mathbf{P}_k w_k N_{k,p}(\xi)}{\sum_{k=1}^{n} w_k N_{k,p}(\xi)}$.
- Main idea - try reduce to the polygon case.
- Divide I into I_k such that C restricted to it is a pair of rational functions.
- For each I_k, calculate the roots of the derivatives of the appropriate rational function.
- For each $I_{k,j}$ find $a_{k,j} = \min_{I_{k,j}} \alpha_k(\xi)$
- Cover $\partial\Omega$ by monotone boxes $\mathcal{B}_{k,j} = [a_{k,j}, b_{k,j}] \times [c_{k,j}, d_{k,j}]$.

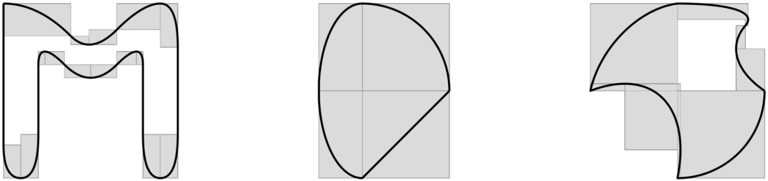
[1] Sommariva, Vianello inRS: Implementing the indicator function of NURBS-shaped planar domains.



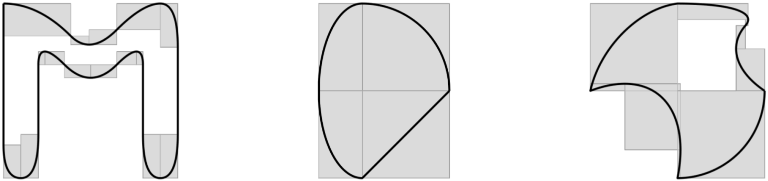
- If (x_0, y_0) outside global bounding box $\rightarrow$ also outside Ω.



- If (x_0, y_0) outside global bounding box $\rightarrow$ also outside Ω.
- If (x_0, y_0) outside monotone boxes - polygon case.

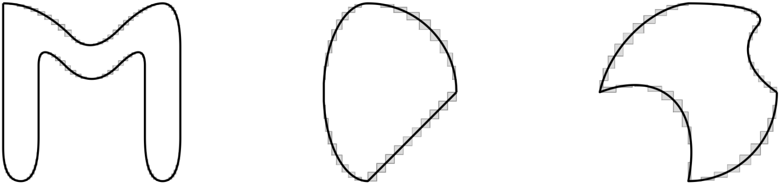


- If (x_0, y_0) outside global bounding box $\rightarrow$ also outside Ω.
- If (x_0, y_0) outside monotone boxes - polygon case.
- If inside one of the monotone boxes - polynomial solve.

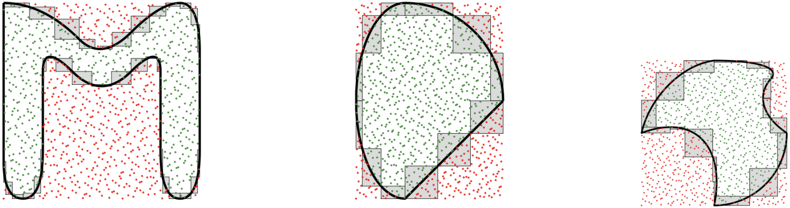


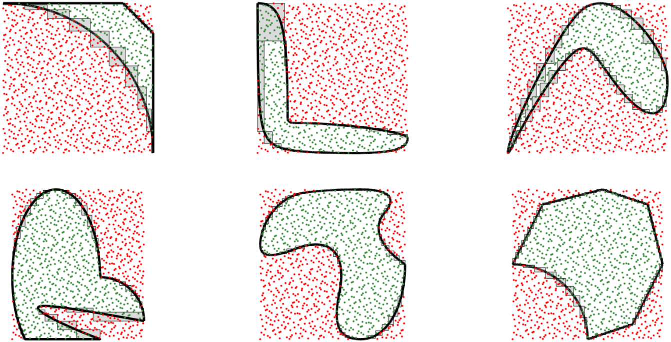
- degenerate cases and multipatch.

- degenerate cases and multipatch.



- Halton nodes + rejection sampling





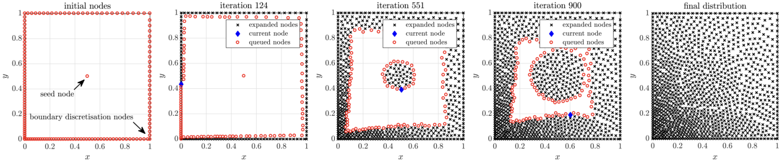
DIVG

- DIVG[2] - Dimension Independent Variable Density node Generation.

[2]Slak, Kosec: On generation of node distributions for meshless PDE discretizations

DIVG

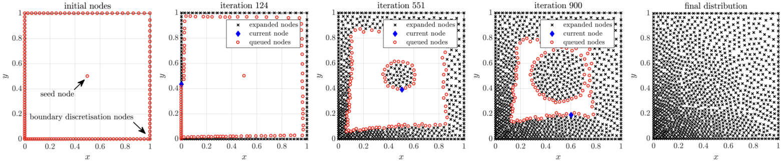
- DIVG[2] - Dimension Independent Variable Density node Generation.



[2]Slak, Kosec: On generation of node distributions for meshless PDE discretizations

DIVG

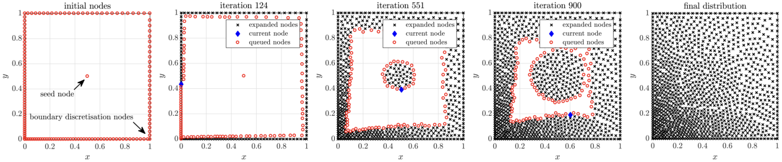
- DIVG[2] - Dimension Independent Variable Density node Generation.
- q - queue of active points, initially filled with seed points. Algorithm runs as long as it's non-empty.



[2]Slak, Kosec: On generation of node distributions for meshless PDE discretizations

DIVG

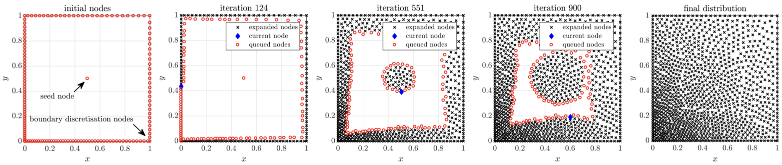
- DIVG[2] - Dimension Independent Variable Density node Generation.
- q - queue of active points, initially filled with seed points. Algorithm runs as long as it's non-empty.
- At each step, pop q to get a point x_0, use it to generate new candidates.



[2]Slak, Kosec: On generation of node distributions for meshless PDE discretizations

DIVG

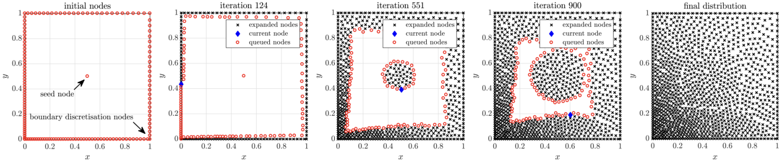
- DIVG[2] - Dimension Independent Variable Density node Generation.
- q - queue of active points, initially filled with seed points. Algorithm runs as long as it's non-empty.
- At each step, pop q to get a point x_0, use it to generate new candidates.
- Candidate is accepted if it's inside the domain and if it is far enough from already accepted points.



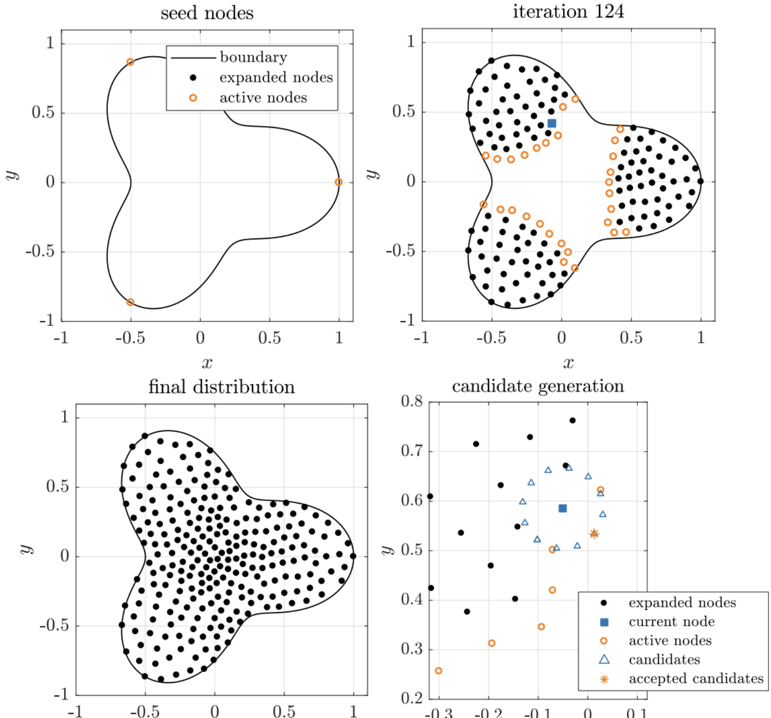
[2] Slak, Kosec: On generation of node distributions for meshless PDE discretizations

DIVG

- DIVG[2] - Dimension Independent Variable Density node Generation.
- q - queue of active points, initially filled with seed points. Algorithm runs as long as it's non-empty.
- At each step, pop q to get a point x_0, use it to generate new candidates.
- Candidate is accepted if it's inside the domain and if it is far enough from already accepted points.
- Algorithm is efficient with the help of a kd-tree.



[2] Slak, Kosec: On generation of node distributions for meshless PDE discretizations

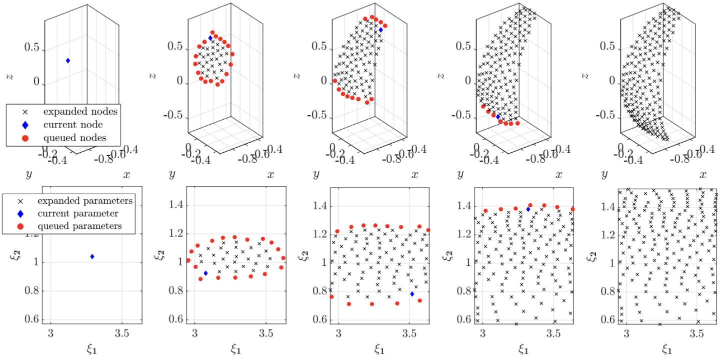


seed nodes
boundary
expanded nodes
active nodes
iteration 124
final distribution
candidate generation
expanded nodes
current node
active nodes
candidates
accepted candidates
x
y

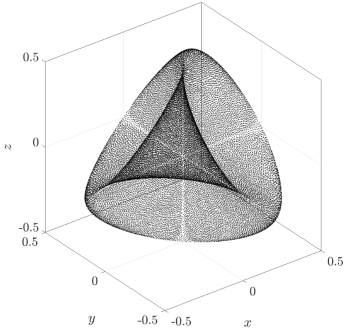
- sDIVG[3] idea - use DIVG in the parameter space.

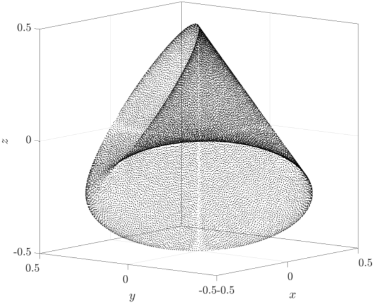
[3] Duh, Kosec, Slak: Fast variable density node generation on parametric surfaces with application to mesh-free methods

- sDIVG[3] idea - use DIVG in the parameter space.



[3]Duh, Kosec, Slak: Fast variable density node generation on parametric surfaces with application to mesh-free methods

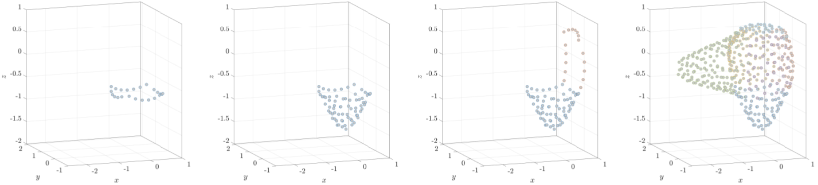

0.5
0
-0.5
z
0.5
0
-0.5
y
-0.5
0
0.5
x


0.5
0
-0.5
z
0.5
0
-0.5
y
-0.5
0
0.5
x

- NURBS-DIVG[4]

[4] Duh, Shankar, Kosec: Discretization of non-uniform rational B-spline (NURBS) models for meshless isogeometric analysis

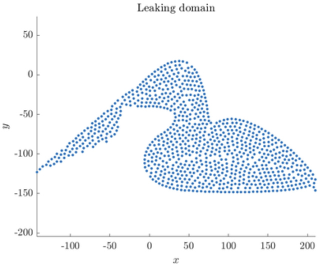
- NURBS-DIVG[4]
- Multipatch - Discretise each patch seperately with sDIVG.



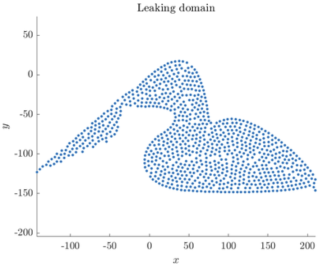
[4] Duh, Shankar, Kosec: Discretization of non-uniform rational B-spline (NURBS) models for meshless isogeometric analysis

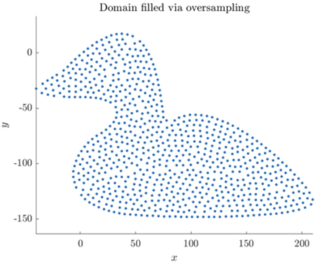
- Interior check - $(\mathbf{p} - \mathbf{x}) \cdot \mathbf{n} > 0$

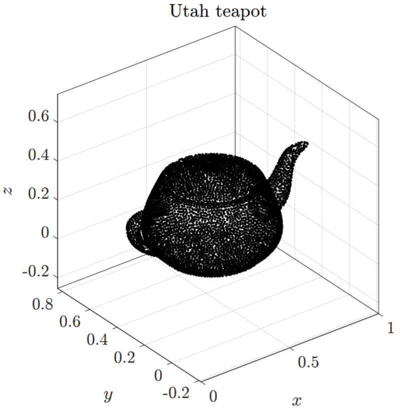
- Interior check - $(\mathbf{p} - \mathbf{x}) \cdot \mathbf{n} > 0$



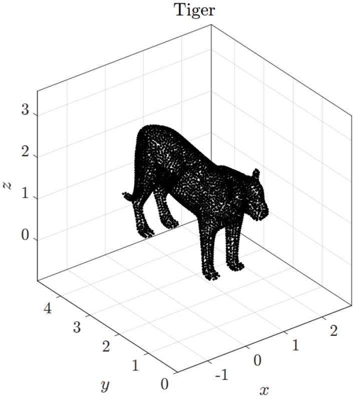
- Interior check - $(\mathbf{p} - \mathbf{x}) \cdot \mathbf{n} > 0$
- Improve accuracy with supersampling.



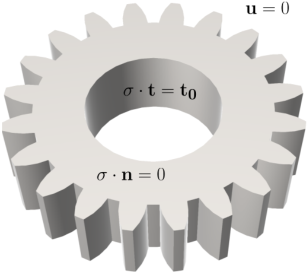


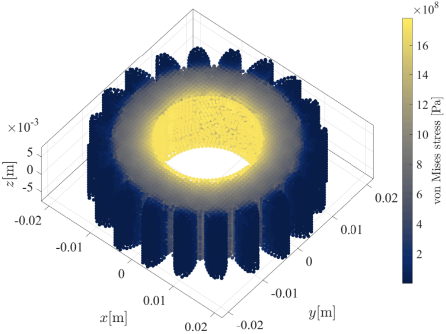


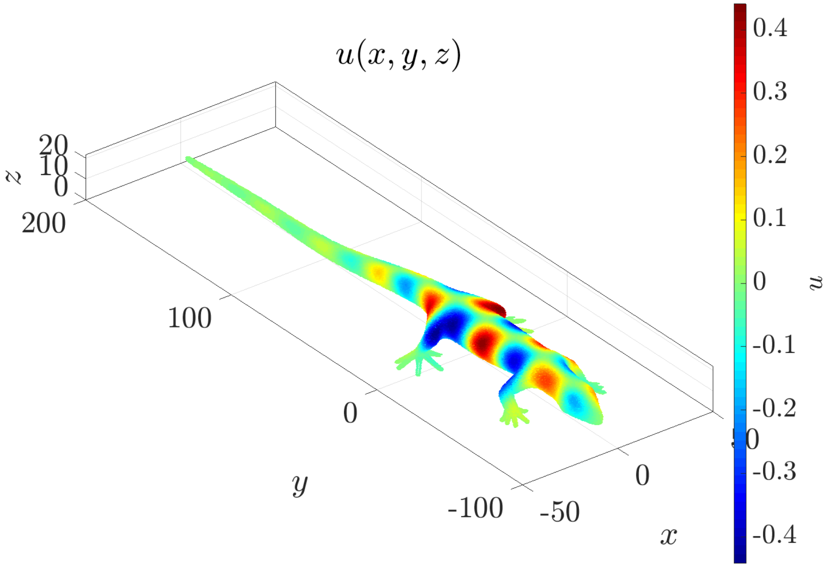
Utah teapot
z
0.6
0.4
0.2
0
-0.2
y
0.8
0.6
0.4
0.2
0
-0.2
x
0
0.5
1



Tiger
z
3
2
1
0
y
4
3
2
1
0
x
-1
0
1
2

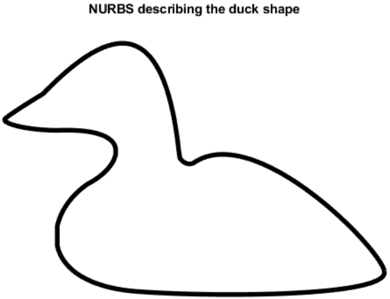

$\mathbf{u} = 0$
$\sigma \cdot \mathbf{t} = \mathbf{t_0}$
$\sigma \cdot \mathbf{n} = 0$

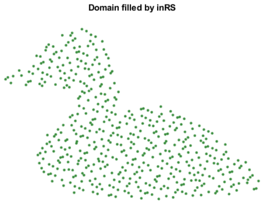

$\times 10^{-3}$
5
0
-5
z[m]
-0.02
-0.01
0
0.01
0.02
x[m]
-0.02
-0.01
0
0.01
0.02
y[m]
$\times 10^{8}$
16
14
12
10
8
6
4
2
von Mises stress [Pa]

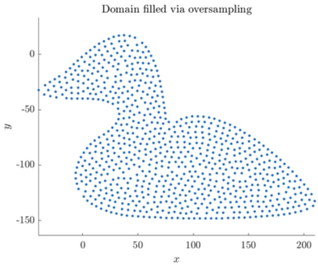


$u(x, y, z)$
z
20
10
0
200
100
0
-100
y
-50
0
50
x
0.4
0.3
0.2
0.1
0
-0.1
-0.2
-0.3
-0.4
u

Comparison


NURBS describing the duck shape


Domain filled by inRS


Domain filled via oversampling
0
-50
-100
-150
y
0
50
100
150
200
x