# Unstructured mesh generators and a finite element solver

Dr Nikolay Kirov
Institute of Mathematics and Informatics,
Bulgarian Academy of Sciences, Sofia

This work is supported by VW-Project "Iterative Techniques for Convection-Diffusion Problems on locally refined meshes"

**Abstract.**

A brief overview of triangle and tetrahedral meshing algorithms is presented. Also included some comments about smoothing, clean-up and refinement procedures. Short descriptions of 2D and 3D mesh generation codes, Triangle and QMG, are given.

We use the streamline-diffusion finite element method for solving convection diffusion boundary valued problems on triangle and tetrahedral meshes. An implementation with piecewise linear trial functions in MATLAB environment is presented.

Two types of numerical results illustrate the mesh generators and the finite element solver – boundary layers on the unite square and unite cube, and a complicated geometry domain in 3D.

1. Automatic unstructured mesh generation
- Triangle and tetrahedral meshing
- The mesh quality
- Mesh post-processing
- Refinement

2. **Triangle** - 2D quality mesh generator

3. **QMG** - 3D mesh generator

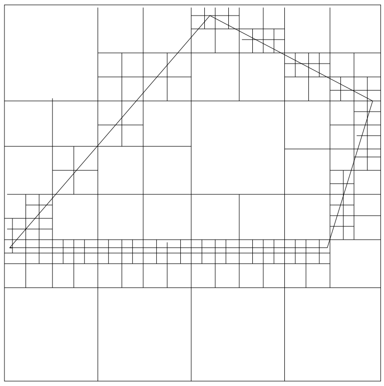
4. Finite element solver

5. Problems solving
- 2D boundary layers problems
- 3D problems

## Triangle and tetrahedral meshing

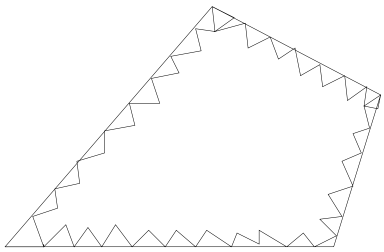
### Octree

Cubes containing the geometric model are recursively subdivided until the desired resolution is reached. Irregular cells are then created where cubes intersect the surface. Tetrahedra are generated from both the irregular cells on the boundary and the internal regular cells.



### Advancing front

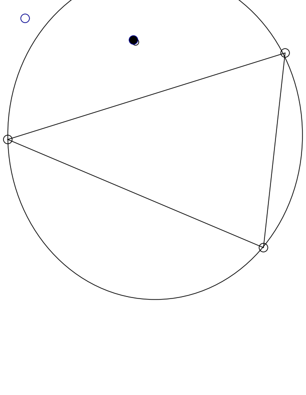
The tetrahedra are built progressively inward from the triangulated surface. An active front is maintained where new tetrahedra are formed. As the algorithm progresses, the front will advance to fill the remainder of the area with triangles. Also required intersection checks to ensure that triangles do not overlap as opposing fronts advance toward each other.

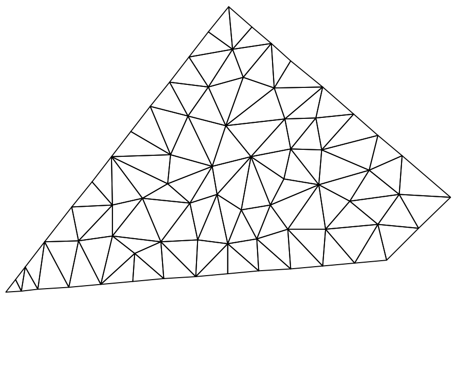


### Delaunay

The Delaunay criteria: Any node must not be contained within the circumsphere of any tetrahedra within the mesh.

Mesh the boundary of the geometry model to provide an initial set of nodes. The boundary nodes are then triangulated according to the Delaunay criterion. Nodes are then inserted incrementally into the existing mesh, redefining the triangles or tetrahedra locally as each new node is inserted to maintain the Delaunay criterion.

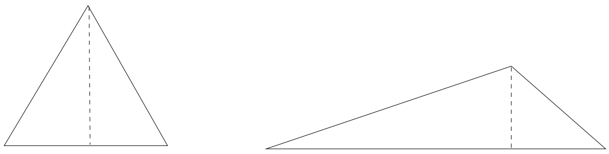




Methods:
- define nodes from a regular grid of points;
- nodes be recursively inserted at triangle or tetrahedral centroids;
- nodes at element circumcircle/circumsphere centers;
- advancing front approach to node insertion;
- point insertion along edges.

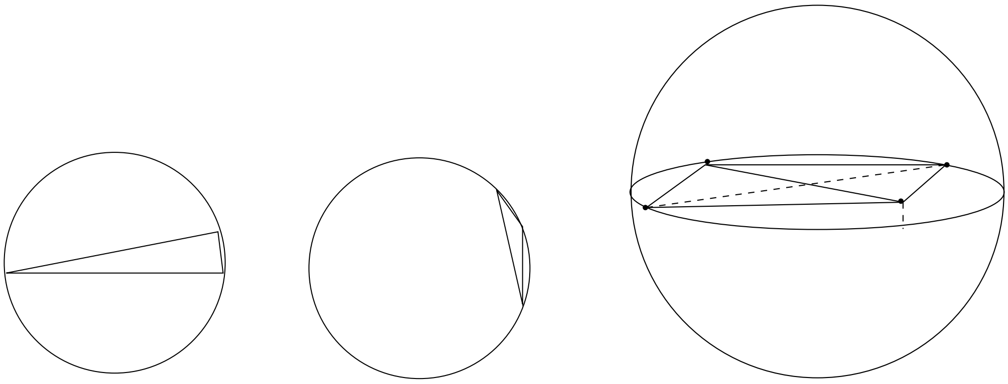
## The mesh quality

The **aspect ratio** of the triangle or tetrahedron is the length of the longest edge divided by the length of the shortest altitude.



a.r.$= 2/\sqrt{3} \approx 1.15$ a.r.$= 4$

The minimum angle $\alpha$, gives a bound of $\pi - 2\alpha$ of maximum angle and guarantees an aspect ratio between $[1/\sin\alpha]$ and $[2/\sin\alpha]$.



A skinny triangle will have a circumcircle much lager than its shortest edge. Tetrahedra can have roughly equal length edges, a reasonably sized circumsphere, and yet be arbitrary skinny.

**Definition.** A *tetrahedral shape measure* is a continuous function that evaluates the quality of a tetrahedron. It must be invariant under translation, rotation, reflection and uniform scaling of the tetrahedron. It must be maximum for the regular tetrahedron and it must be minimum for a degenerate tetrahedron. There is no local maximum other than the global maximum for a regular tetrahedron and there is no local minimum other than the global minimum for a degenerate tetrahedron. For the easy of comparison, it should be scaled to the interval $[0, 1]$, and be 1 for the regular tetrahedron and 0 for a degenerate tetrahedron.

An aspect ratio function, defining by

$$\gamma = \frac{12}{\sqrt{6}} \cdot \frac{\text{radius of inscribed sphere}}{\text{length of largest edge}}$$

is a tetrahedral shape measure but minimum dihedral angle is not (according to the definition).

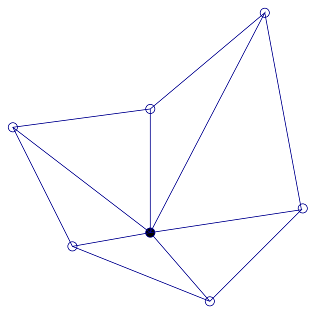
## Mesh post-processing

### Smoothing

Smoothing includes any method that adjust node locations while maintaining the element connectivity.

Methods:

• ***Laplacian smoothing*** – an internal nodes placed at the average location of any other node connected to it by an edge;



• ***optimization-based smoothing*** techniques measure the quality of the surrounding elements to a node and attempt to optimize by computing the local gradient of the element quality with respect to the node location;

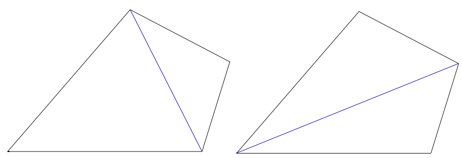
• ***physically-based methods*** – reposition nodes using simulated physically based attraction or repulsion force.

### Cleanup

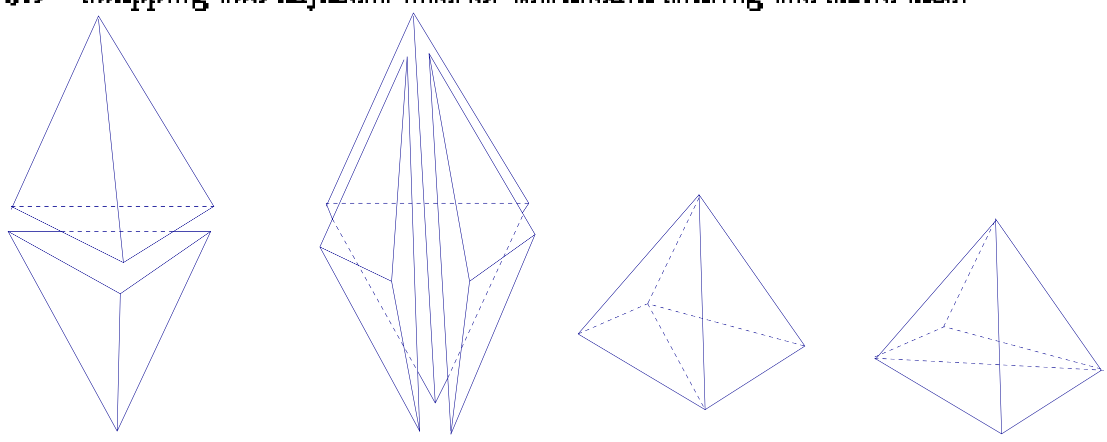
Cleanup methods improve the quality of the mesh by making local changes to the element connectivities.

***Topological improvement***

In 2D – simple diagonal swaps.



In 3D – swapping two adjacent interior tetrahedra sharing the same face.



***Topological improvement***

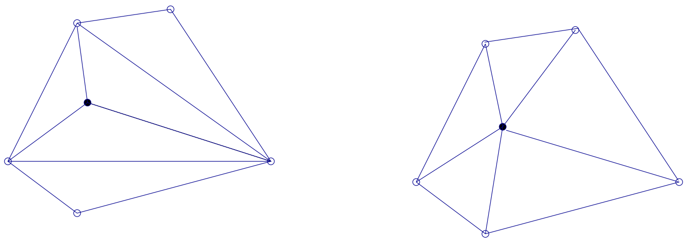
To attempt to optimize the number of edges sharing a single node (node degree).

## Refinement

Refinement effectively reduces the local element size.

***Edge bisection*** involves splitting individual edges in the triangulation.

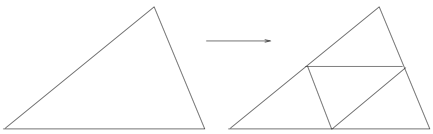
***Point insertion*** – to insert a single node at the centroid of an existing element, dividing the triangle into 3 or tetrahedron into 4.



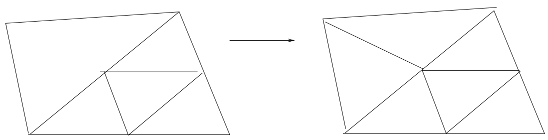
***Template*** – a specific decomposition of the triangle.

• decompose a triangle into 4 similar triangles by inserting a new node at each of its edges;

• decompose tetrahedron into 8 tetrahedra where each face of the tetrahedron has been decomposed into 4 similar triangles.



red refinement



green refinement

## Triangle

[J. R. Shewchuk, Carnegie Mellon University]

**Triangle** is a C-program for 2D mesh generation and construction of Delaunay triangulation and constrained Delaunay triangulation.

Main features:

- user-specified constraints on angles and triangle areas;
- user-specified holes and concavities;
- use of exact arithmetic to improve robustness.

**Triangle**'s input is a planar straight line graph (PSLG) defined to be a collection of vertices and segments, where the endpoints of every segment are included in the list of vertices.

```
# unite square
4  2  0  1
   1   0.0     0.0    1
   2   0.0     1.0    1
   3   1.0     1.0    1
   4   1.0     0.0    1
4  1
   1    1   2   1
   2    2   3   1
   3    3   4   1
   4    4   1   1
0
```

## QMG

[Stephan A. Vavasis, Cornell University]

The Quality Mesh Generator (**QMG**) package does finite element mesh generation in two and three dimensions. The package includes *geometric modeling software, the mesh generator* itself and a simple finite element solver. **QMG** consists of 60 MATLAB function and uses the scripting capabilities of MATLAB software package.

The **QMG** handles complicated topology. The domain can have *holes* and quite complex *internal boundaries*.

Input data have to be presented in form of a **brep**. A **brep** is a geometric object that is specified by its boundary faces. All breps must have *flat boundaries*, i.e. every element of the boundary must be a subset of a linear affine space.

Abstractly, a **brep** is an acyclic directed graph. Every node in the graph stands for a face of the brep. The term "face" refers to a vertex, edge or facet. The interior of the brep is also considered a face. Each of these faces has some information associated to it (for instance, vertices have their space coordinates associated with them). The arcs of the directed graph indicate boundary relationships. For example, an edge that is bounded by two vertices has arcs to those two vertices to indicate the bounding relation. A facet has arcs to the edges that act as its boundary.

```
# unite cube
< brep
 < 3 3
  < (
   < v0_0 (< point (0.0 0.0 0.0) >) () () >
   < v0_1 (< point (1.0 0.0 0.0) >) () () >
   < v0_2 (< point (0.0 1.0 0.0) >) () () >
   < v0_3 (< point (0.0 0.0 1.0) >) () () >
   < v0_4 (< point (1.0 1.0 0.0) >) () () >
   < v0_5 (< point (0.0 1.0 1.0) >) () () >
   < v0_6 (< point (1.0 0.0 1.0) >) () () >
   < v0_7 (< point (1.0 1.0 1.0) >) () () >
   )
   < (
    < e1_0  () (v0_0 v0_1) () >
    < e1_1  () (v0_0 v0_2) () >
    < e1_2  () (v0_0 v0_3) () >
    < e1_3  () (v0_1 v0_4) () >
    < e1_4  () (v0_1 v0_6) () >
    < e1_5  () (v0_2 v0_4) () >
    < e1_6  () (v0_2 v0_5) () >
    < e1_7  () (v0_3 v0_5) () >
    < e1_8  () (v0_3 v0_6) () >
    < e1_9  () (v0_4 v0_7) () >
    < e1_10 () (v0_5 v0_7) () >
    < e1_11 () (v0_6 v0_7) () >
    )
    < (
      < f2_0 () (e1_0 e1_1  e1_5  e1_3 ) () >
      < f2_1 () (e1_7 e1_10 e1_11 e1_8 ) () >
      < f2_2 () (e1_0 e1_4  e1_8  e1_2 ) () >
      < f2_3 () (e1_1 e1_2  e1_7  e1_6 ) () >
      < f2_4 () (e1_5 e1_9  e1_10 e1_6 ) () >
      < f2_5 () (e1_3 e1_4  e1_11 e1_9 ) () >
      )
      < (
       < d3_0 () (f2_0 f2_1 f2_2 f2_3 f2_4 f2_5) () >
      ) nil >
    >
   >
  >
 >
>
```

**A finite element solver**

We consider the convection-diffusion boundary value problem:

$$-\,\mathbf{div}\,(a\,\mathbf{grad}\,u) + b.\,\mathbf{grad}\,u + cu = f \quad \text{in} \quad \Omega,$$

$$u\Big|_{\Gamma_1} = g_1; \quad a\frac{\partial u}{\partial n}\Big|_{\Gamma_2} = g_2.$$

$a, b, c : \Omega \to R_+, \quad \Gamma_1 \cup \Gamma_2 = \partial\Omega.$

The standard Galerkin form reads:

Find $u_h$ with $u_h|_{\Gamma_1} = g_{1h}$ such that for every $v_h|_{\Gamma_1} = 0$

$$\boldsymbol{a}(u_h, v_h) := (a\nabla u_h, \nabla v_h) + (b.\nabla u_h, v_h) + (cu_h, v_h) =$$

$$= (f, v_h) + \int_{\Gamma_2} g_2 v_h d\gamma =: \boldsymbol{l}(v_h).$$

This scheme is not suitable in the case when $a$ is small compared with $b$. Therefore we have to add some stability terms which are for piecewise linear trials the following one:

$$\boldsymbol{a}_s(u_h, v_h) := \sum_{T\in\mathcal{T}} \delta_T(b.\nabla u_h + cu_h, b.\nabla v_h)_T \quad \text{and} \quad \boldsymbol{l}_s(v_h) := \sum_{T\in\mathcal{T}} \delta_T(f, b.\nabla v_h)_T,$$

where $T$ is a triangle/tetrahedron of the mesh $\mathcal{T}$, and $\delta_T \geq 0$ is a user-chosen piecewise constant parameter. Often we set $\delta_T = k_\delta\sqrt{S_T}$ where $S_T$ is the area of the triangle $T$ or $\delta_T = k_\delta V_T^{\frac{1}{3}}$, where $V_T$ is the volume of the tetrahedron $T$.

Then the equation is

$$\boldsymbol{a}(u_h, v_h) + \boldsymbol{a}_s(u_h, v_h) = \boldsymbol{l}(v_h) + \boldsymbol{l}_s(v_h).$$

The finite element solver presents MATLAB procedures for:

- creating geometric models for 2D and 3D domains;
- mesh generation (with help of `Triangle` for 2D and **QMG** for 3D);
- creating stiffness matrix and solving the linear system;
- visualization of domains, meshes and solutions.

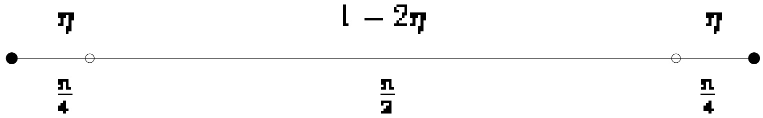
**Example A.**

We consider the following boundary value problem:

$$-\varepsilon^2 \Delta u + u = 1 \quad \text{in } [0,1]^2, \qquad u|_\Gamma = 0,$$

where $\Gamma$ is the boundary of unite square. When $\varepsilon$ is small, the solution has boundary layer on $\Gamma$.

We create Shishkin mesh for solving this problem.



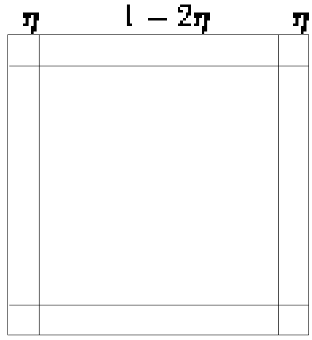

$\eta = \min\left\{\frac{1}{4},\ 2\varepsilon \ln n\right\}$, $n$ is the number of mesh points.

The unite square is divided of 3 domains:

– $D_1$: a central domain , $[1-2\eta, 1-2\eta]^2$, where we have to place 25% of all $N$ mesh points on the unite square;

– $D_2$: 4 subdomains, each is $[\eta, 1-2\eta]^2$, at the center of a square side. In this subdomain 50% nodes have to be placed;

– $D_3$: 4 square $[0,\eta]^2$ subdomains, each one has a vertex, which coincides with a square vertex, contains 25% of all nodes.



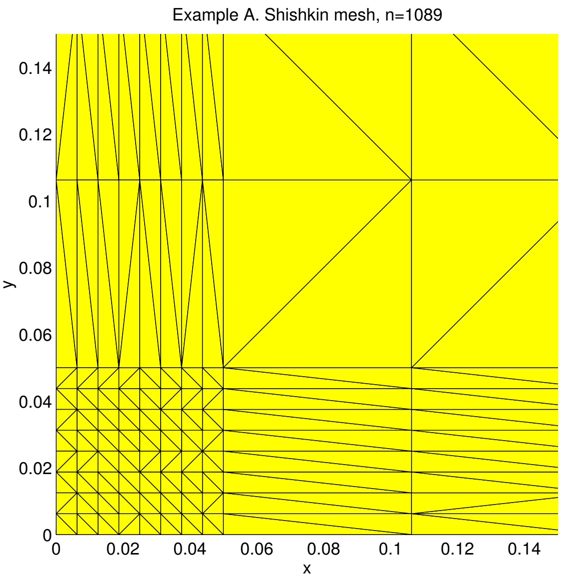

In the examples we fix the parameters $\eta = 0.05$ and the number of mesh points $N = 1089$. Then we can calculate $\varepsilon$:

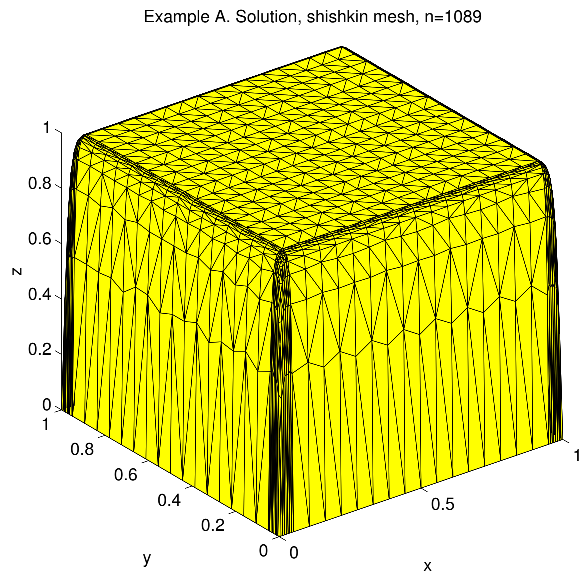
$$\eta = 2\varepsilon \ln(N^{\frac{1}{2}}) = \varepsilon \ln N, \quad \varepsilon = \frac{0.05}{\ln N} \approx 0.0071, \quad \varepsilon^2 = 5.1122 \times 10^{-5}.$$

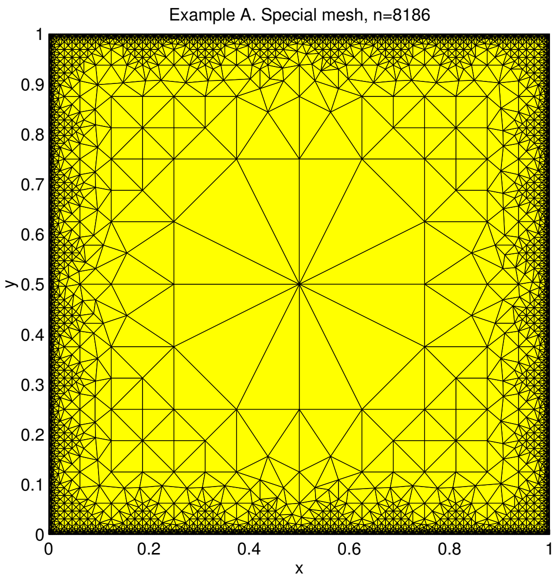
Other mesh type is created ("special mesh") in the following rules: at the first step a coarse triangulation is done, and at the refinement steps a size-function

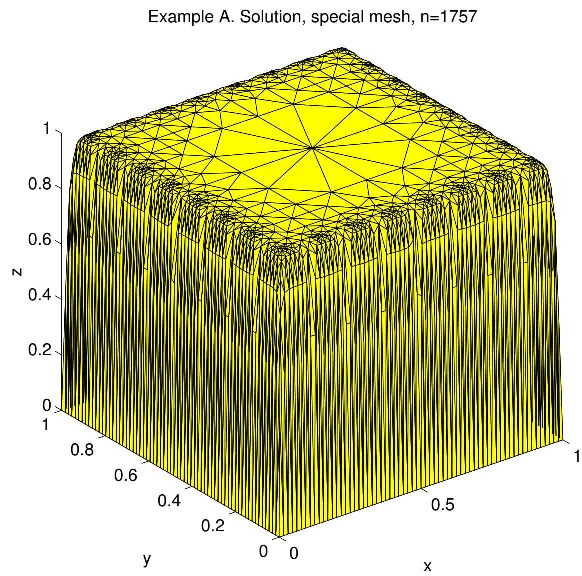
$$s_T = \begin{cases} S_T/16, & \text{if } T \cap \Gamma = P_i P_j \text{ (an edge)} \\ S_T/4, & \text{if } T \cap \Gamma = P_i \text{ (a vertex)} \\ 2S_T & \text{else} \end{cases}$$

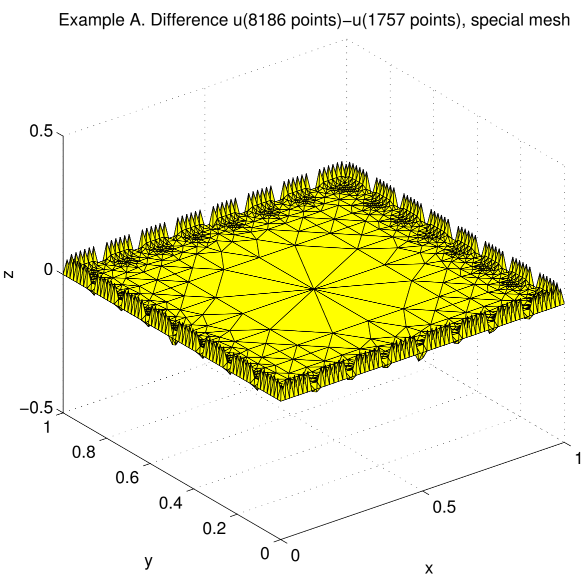
is used.

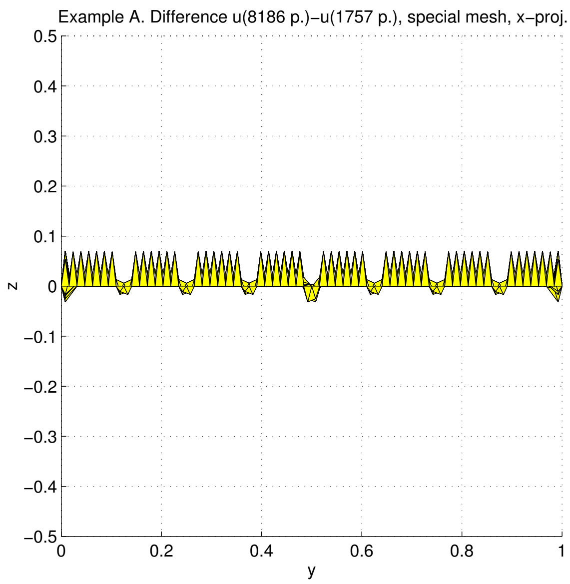

Example A. Shishkin mesh, n=1089
x
y


Example A. Solution, shishkin mesh, n=1089
x
y
z


Example A. Special mesh, n=8186
x
y


Example A. Solution, special mesh, n=1757
x
y
z


Example A. Difference u(8186 points)−u(1757 points), special mesh
x
y
z


Example A. Difference u(8186 p.)−u(1757 p.), special mesh, x−proj.
y
z

**Example 1.** Shishkin mesh 3D example

We consider the following boundary value problem:

$$-\varepsilon^2\Delta u + u = 1 \quad \text{in } [0,1]^3, \qquad u|_\Gamma = 0,$$

where $\Gamma$ is the boundary of unite 3D cube. When $\varepsilon$ is small, the solution has boundary layer on $\Gamma$.

The unite cube is divided of 4 domains:

– $D_1$: a central domain , $[1-2\eta, 1-2\eta]^3$, where we have to place 1/8 of all $N$ 3D mesh points;

– $D_2$: 6 square prism subdomains, each has a face $[\eta, 1-2\eta]^2$ at the center of a cube face and third dimension $\eta$. There we should place 1/16 nodes;

– $D_3$: 12 square prism subdomains, each has an edge $[\eta, 1-2\eta]$, at the center of a cube edge and size $\eta\times\eta\times 1-2\eta$. In this subdomain 1/32 nodes have to be placed;

– $D_4$: 8 cube $[0,\eta]^3$ subdomains, each one has a vertex, which coincides with a cube vertex contains 1/64 part of all nodes.

It follows that the distribution of the nodes should be:

$$D_1 \;\; 12.5\% \quad D_2 \;\; 37.5\% \quad D_3 \;\; 37.5\% \quad D_4 \;\; 12.5\%.$$

The distribution of nodes (in the mesh generation step) can be controlled by size function $s : R^3 \to R$:

$$s = \frac{2}{\prod_{i=1}^{3} \operatorname{sign}\left(\left|x_i - \frac{1}{2}\right| + \frac{1}{2} - \eta\right) + 3} = \begin{cases} 1/4 & \text{in } D_1 \\ 1/8 & \text{in } D_2 \\ 1/16 & \text{in } D_3 \\ 1/32 & \text{in } D_4. \end{cases}$$

We fix the parameter $\eta = 0.1$. $\quad \eta = 2\varepsilon\ln(N^{\frac{1}{3}}) = \frac{2}{3}\varepsilon\ln N, \quad \varepsilon = \dfrac{3}{20\ln N}.$

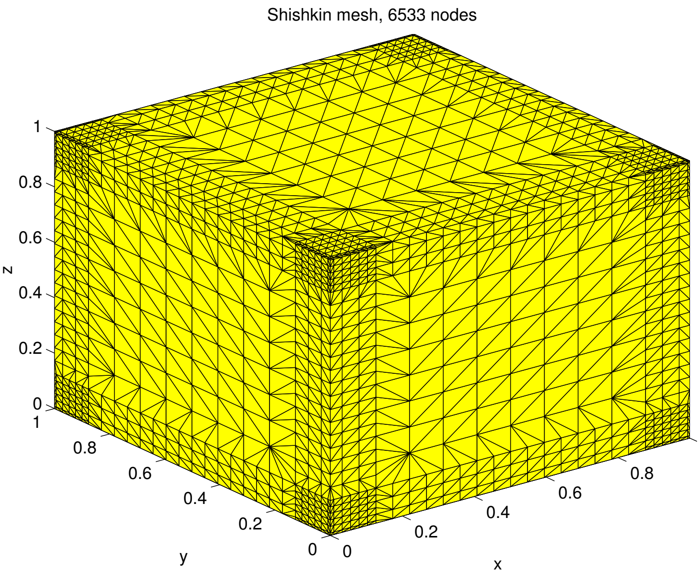
For $N = 6533$, $\varepsilon \approx 0.0171$, $\varepsilon^2 = 2.9157.10^{-4}$, we have: 30 528 tetrahedra, 3 963 internal nodes, 2 570 boundary nodes, 49 135 nonzero matrix elements, 210 min solution time.

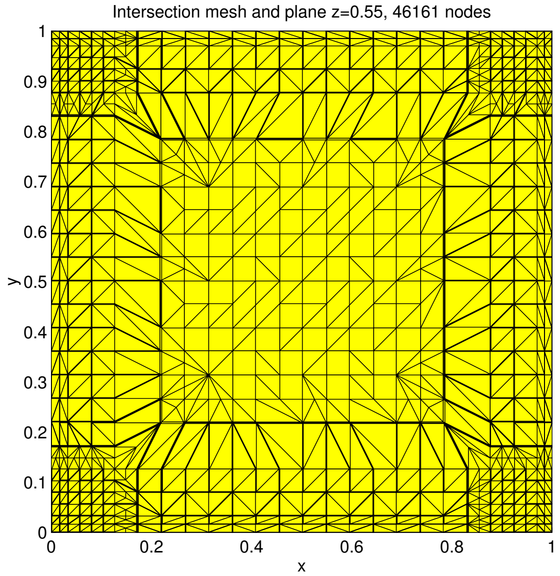
Distribution of the nodes in subdomains is: $D_1$ 8.6%, $D_2$ 32.9%, $D_3$ 43.2%, $D_4$ 15.3%.

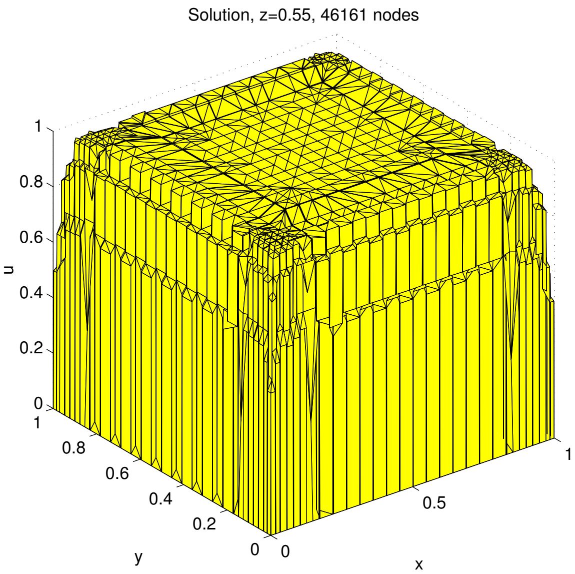
And the maximal solution value is 1.035.

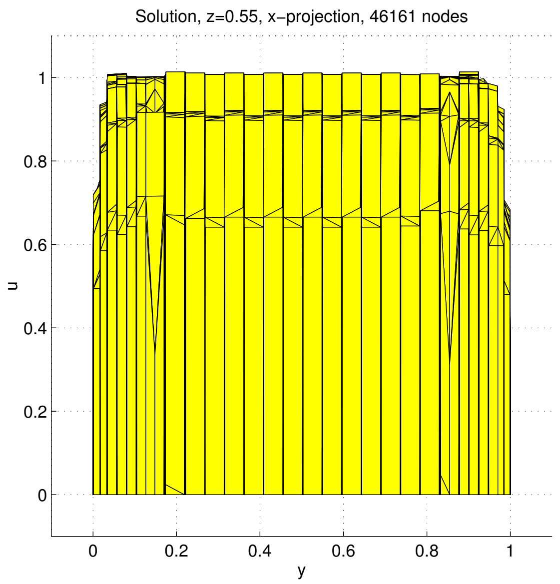
For $N = 46\,161$, $\varepsilon \approx 0.0140$, $\varepsilon^2 = 1.9507.10^{-4}$, 46 161 nodes, 244 224 tetrahedra, 35 887 internal nodes, 497 179 nonzero matrix elements, 5 hours solution time.

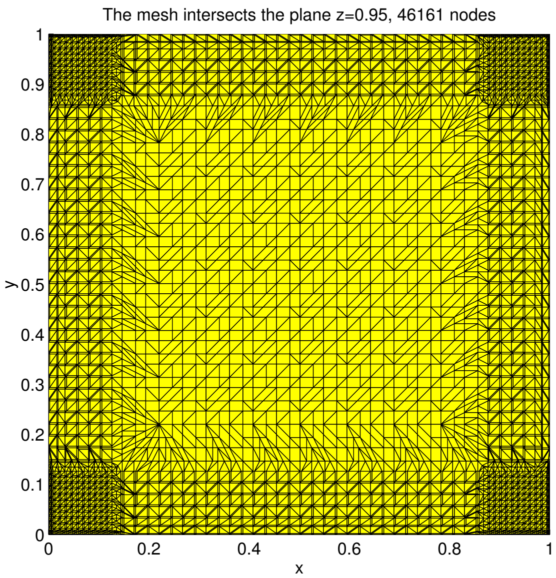
Distribution of the nodes in subdomains: $D_1$ 9.5%, $D_2$ 35.8%, $D_3$ 37.4%, $D_4$ 17.3%. and the maximal solution value is 1.024.

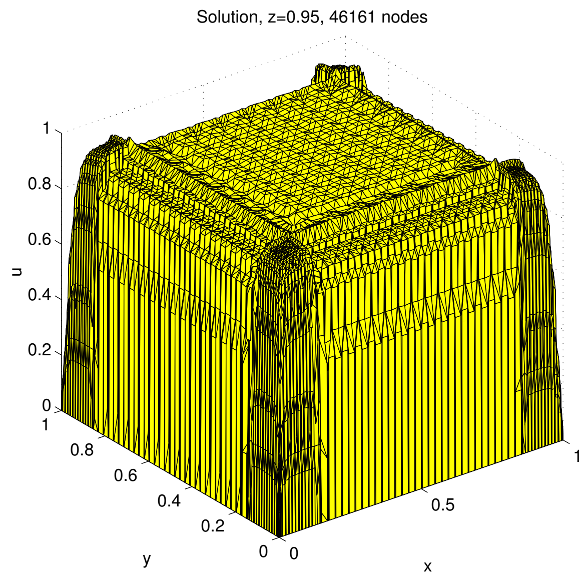

Shishkin mesh, 6533 nodes
z
y
x


Intersection mesh and plane z=0.55, 46161 nodes
y
x


Solution, z=0.55, 46161 nodes
u
y
x


Solution, z=0.55, x−projection, 46161 nodes
u
y


The mesh intersects the plane z=0.95, 46161 nodes
y
x


Solution, z=0.95, 46161 nodes
u
y
x

**Example 3.**

$$-\varepsilon\Delta u + b.\nabla u + cu = f$$

on unite square and Dirichlet boundary conditions, $u|_\Gamma = 0$.

The exact solution is $u(x,y) = xy\left(1-\exp\left(-\frac{1-x}{\varepsilon}\right)\right)\left(1-\exp\left(-\frac{1-y}{\varepsilon}\right)\right)$.

Right hand side can be calculated using the function $u$.

$$b = \left(\frac{\sqrt{2}}{2}\left(1-\frac{\sqrt{2}}{2}x\right), \frac{\sqrt{2}}{2}\left(1-\frac{\sqrt{2}}{2}y\right)\right) = \left(\frac{\sqrt{2}-x}{2}, \frac{\sqrt{2}-y}{2}\right), \quad c=1, \quad \varepsilon = 10^{-4}.$$

There are boundary layers at $x=1$ and $y=1$ boundary. The thickness of the boundary layers is $\eta = \min\{\varepsilon \ln N, \frac{1}{2}\}$.

We define a comparison function $u_0(x,y) = xy\left(1-\exp\left(-\frac{2-x}{\varepsilon}\right)\right)\left(1-\exp\left(-\frac{2-y}{\varepsilon}\right)\right)$. Then we calculate right hand side (function $f$) and boundary conditions $u_0|_\Gamma$.

A "special mesh" is created in the following rules: At the first step the unite square is divided by 16 equal squares and at the next refinement steps a size-function is used. Any triangle with exact 1 vertex at the boundary $x=1$ or $y=1$ is split by 4 but any triangle with 2 vertexes at the boundary is split by 16. Every other triangle may split only to keep mesh consistent.

There is a significant correlation between the coefficient $k_\delta$ and the errors ($\delta_T = k_\delta S_T$). The best value is 0.05.

**Errors.**

Let $u$ be the exact solution function and $\bar{u}$ be the solution obtained in the current iteration, i.e. the values $\bar{u}_i = \bar{u}(P_i)$ are known at every point $P_i$, which is a mesh node ($i = 1, \ldots, n$).

1. Absolute error ($l_2$ error): $e_{abs} = \sqrt{\sum_{i=1}^{n}(u(P_i)-\bar{u}_i)^2}$.

2. Relative error: $e_{rel} = \dfrac{e_{abs}}{\sqrt{\sum_{i=1}^{n} u(P_i)^2}} = \sqrt{\dfrac{\sum_{i=1}^{n}(u(P_i)-\bar{u}_i)^2}{\sum_{i=1}^{n} u(P_i)^2}}$.

3. Integral error ($L_2$ error):

$$e_{int} = \int_\Omega (u(x,y)-\bar{u}(x,y))^2\,dxdy = \sum_{T\in\mathcal{T}}\int_T (u(x,y)-\bar{u}(x,y))^2\,dxdy \approx$$

$$\sum_{T_{ijk}\in\mathcal{T}} \frac{1}{3}\left((u(P_{ij}) - \frac{1}{2}(\bar{u}_i+\bar{u}_j))^2 + (u(P_{jk}) - \frac{1}{2}(\bar{u}_j+\bar{u}_k))^2 + (u(P_{ki}) - \frac{1}{2}(\bar{u}_k+\bar{u}_i))^2\right) S_T,$$

where $P_{ij} = \frac{1}{2}(P_i+P_j)$, $S_T$ is the area of $T$.

4. Maximal error ($l_\infty$ error): $e_{max} = \max_{i=1,\ldots,n} |u(P_i)-\bar{u}_i|$.

Tables contain:
node numbers, $e_{abs}$ $e_{rel}$ $e_{int}$ $e_{max}$.
Regular mesh

| | | | | |
|---|---|---|---|---|
| 9 | 1.737e+01 | 6.947e+01 | 5.174e+01 | 1.737e+01 |
| 145 | 3.588e+01 | 1.043e+01 | 2.549e+00 | 1.657e+01 |
| 1089 | 2.110e+01 | 2.074e+00 | 1.503e−01 | 5.105e+00 |
| 4225 | 1.033e+01 | 4.957e−01 | 1.150e−02 | 1.995e+00 |
| 16641 | 4.183e+00 | 9.921e−02 | 1.048e−03 | 6.665e−01 |

Shishkin mesh $\eta = 3.5 \times 10^{-4}$

| | | | | |
|---|---|---|---|---|
| 1089 | 7.934e+01 | 5.391e+00 | 8.320e−01 | 9.077e+00 |

Special mesh

| | | | | |
|---|---|---|---|---|
| 25 | 2.720e+01 | 3.108e+01 | 2.019e+01 | 1.771e+01 |
| 63 | 3.308e+01 | 1.171e+01 | 2.825e+00 | 1.869e+01 |
| 171 | 2.110e+01 | 3.098e+00 | 4.002e−01 | 8.660e+00 |
| 470 | 8.668e+00 | 7.314e−01 | 1.430e−02 | 1.975e+00 |
| 1088 | 3.907e+00 | 2.009e−01 | 1.610e−03 | 6.257e−01 |
| 2573 | 3.786e+00 | 1.230e−01 | 3.950e−04 | 3.603e−01 |
| 6616 | 3.113e+00 | 6.590e−02 | 1.697e−04 | 2.579e−01 |

The next tables present the comparison function errors.
Regular mesh

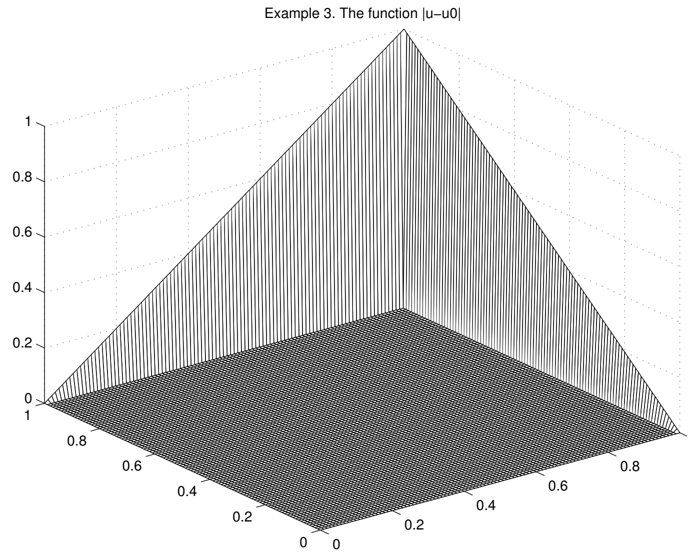
| | | | | |
|---|---|---|---|---|
| 9 | 4.579e−03 | 3.664e−03 | 1.306e−03 | 4.579e−03 |
| 145 | 3.911e−03 | 9.427e−04 | 6.723e−07 | 8.088e−04 |
| 1089 | 2.839e−03 | 2.541e−04 | 2.216e−08 | 3.567e−04 |
| 4225 | 1.252e−03 | 5.732e−05 | 1.341e−09 | 8.267e−05 |
| 16641 | 4.626e−04 | 1.072e−05 | 8.074e−11 | 1.623e−05 |

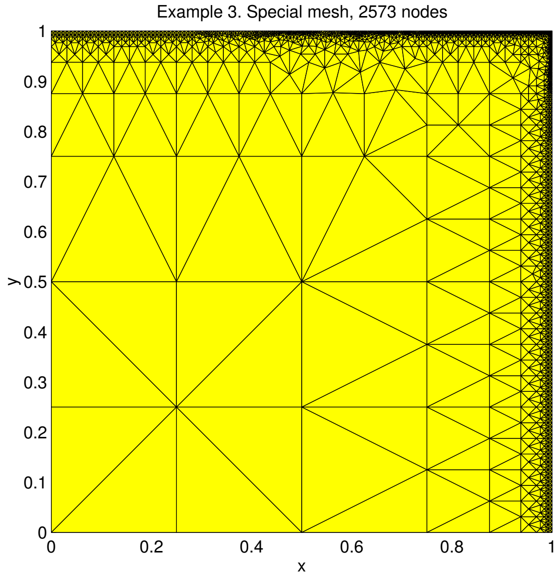
Shishkin mesh $\eta = 3.5 \times 10^{-4}$

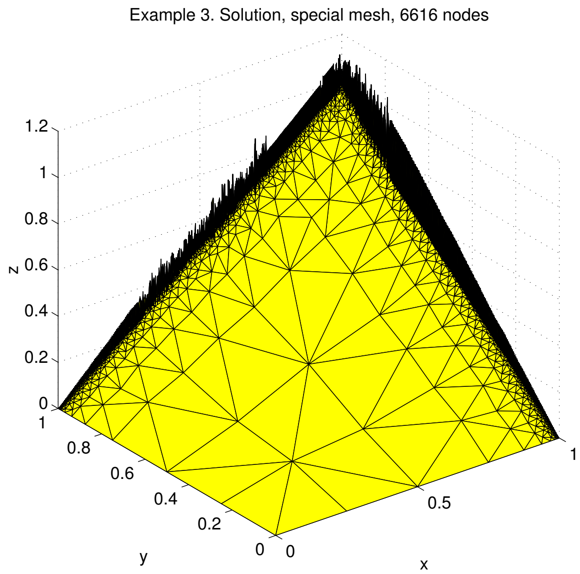
| | | | | |
|---|---|---|---|---|
| 1089 | 2.105e−02 | 9.642e−04 | 8.272e−07 | 3.762e−03 |

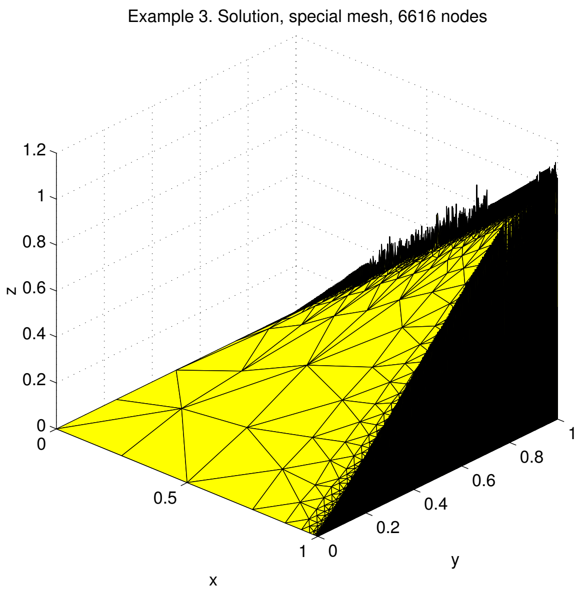
Special mesh

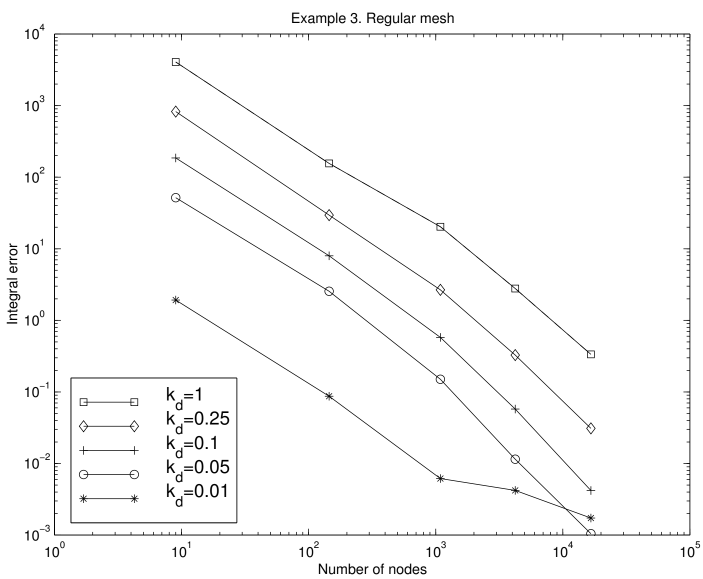
| | | | | |
|---|---|---|---|---|
| 25 | 2.026e−03 | 1.080e−03 | 8.149e−05 | 1.483e−03 |
| 63 | 5.544e−02 | 1.437e−02 | 7.475e−05 | 2.655e−02 |
| 171 | 7.160e−02 | 8.925e−03 | 6.852e−05 | 2.203e−02 |
| 470 | 9.520e−02 | 6.993e−03 | 6.455e−05 | 2.126e−02 |
| 1088 | 1.108e−01 | 5.034e−03 | 6.481e−05 | 2.131e−02 |
| 2573 | 1.173e−01 | 3.339e−03 | 5.819e−05 | 1.827e−02 |
| 6616 | 1.570e−01 | 2.893e−03 | 5.819e−05 | 1.827e−02 |

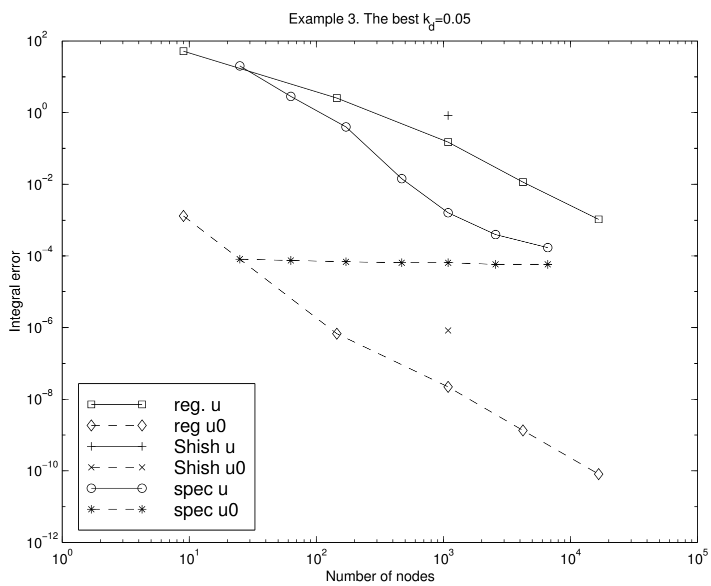

Example 3. The function |u−u0|


Example 3. Special mesh, 2573 nodes
x
y


Example 3. Solution, special mesh, 6616 nodes
x
y
z


Example 3. Solution, special mesh, 6616 nodes
x
y
z


Example 3. Regular mesh
Integral error
Number of nodes
$k_d$=1
$k_d$=0.25
$k_d$=0.1
$k_d$=0.05
$k_d$=0.01


Example 3. The best $k_d$=0.05
Integral error
Number of nodes
reg. u
reg u0
Shish u
Shish u0
spec u
spec u0

**Example 4.** Complicated domain problems in 3D.

$$-\,\mathbf{div}\,(\mu\,\mathbf{grad}\,\phi) = 0 \quad \text{on} \quad \Omega \subset R^3,$$

with Dirichlet boundary conditions

$$\phi|_{\Gamma\cap\{z=z_0\}} = z_0,$$

i.e. the boundary condition is a linear function in the direction $z$ and a constant in the directions $x$ and $y$.

The domain $\Omega$ consists of two parts $\Omega_0 \subset \Omega$, $\Omega_0 << \Omega$ and $\Omega\setminus\Omega_0$.

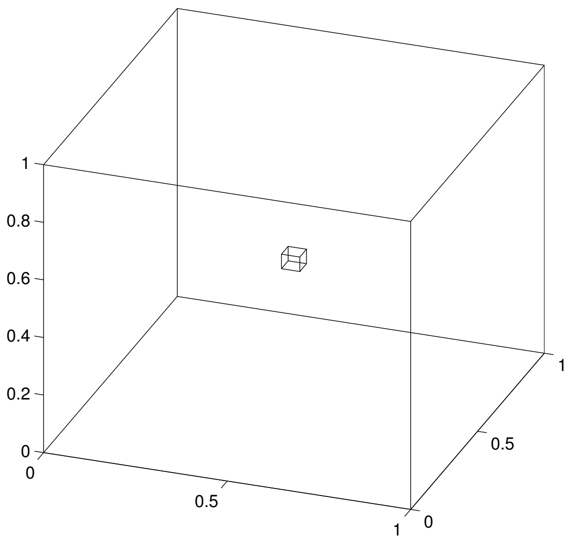
$$\mu = \begin{cases} 2.5 & \text{in} \quad \Omega_0 \\ 1 & \text{in} \quad \Omega\setminus\Omega_0 \end{cases}$$

Case 1. $\Omega = [0,1]^3$, $\Omega_0 = (0.5, 0.5) + [0, 0.05]^3$

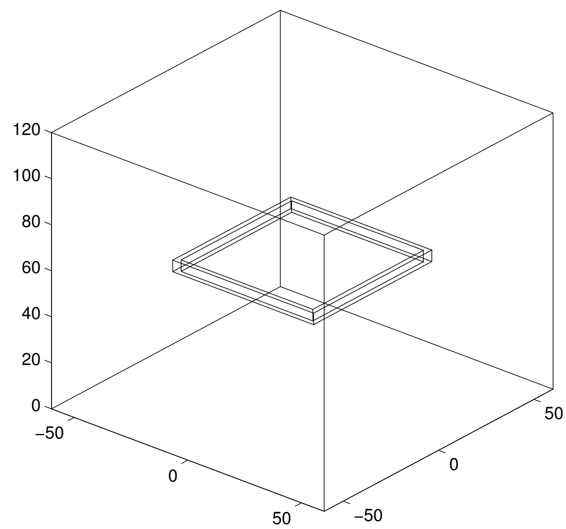
Case 2. $\Omega = [-60.60]^2 \times [0, 120]$, $\Omega_0 = ([-31, 31]^2\setminus[-29, 29]^2) \times [57.5, 62.5]$.

Case 3. Let $\Sigma$ be the unite 16-gone. Then $\Omega = 120\Sigma \times [-60, 60]$,
$\Omega_0 = (62\Sigma)\setminus(58\Sigma) \times [-2.5, 2.5]$.

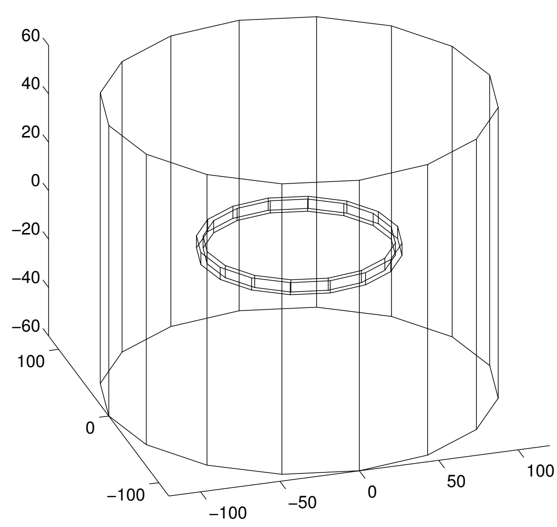
Case 1. Cube domain with a small cube



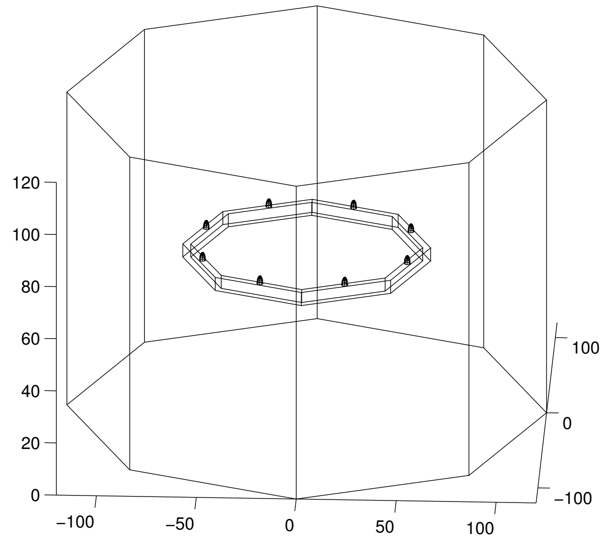

Case 2. Cube domain with a canal



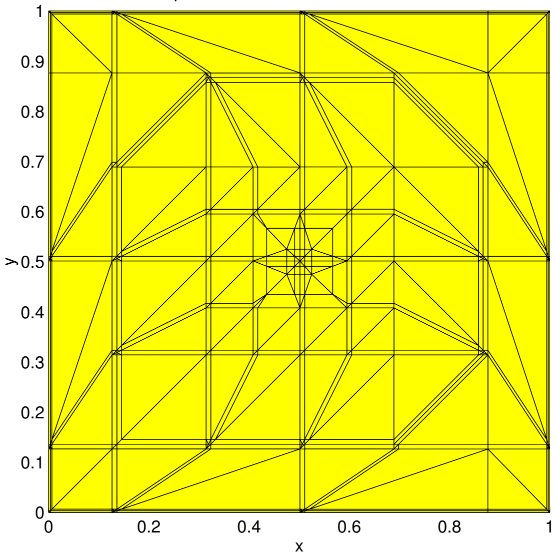
Case 4. 16−gon prism with a canal



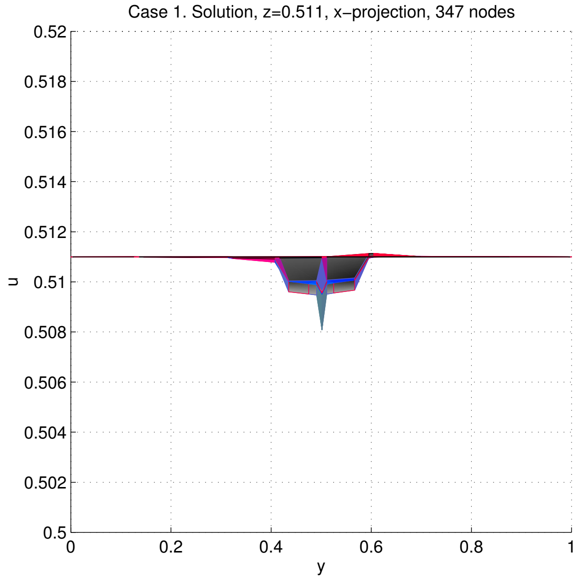

Case 4. 8−gon prism with a canal and 8 bumps



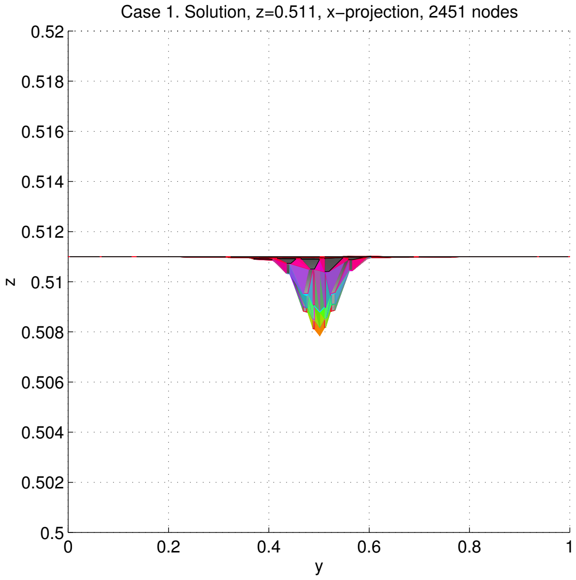

Case 1. The plane z=0.511 intersects the mesh, 347 nodes



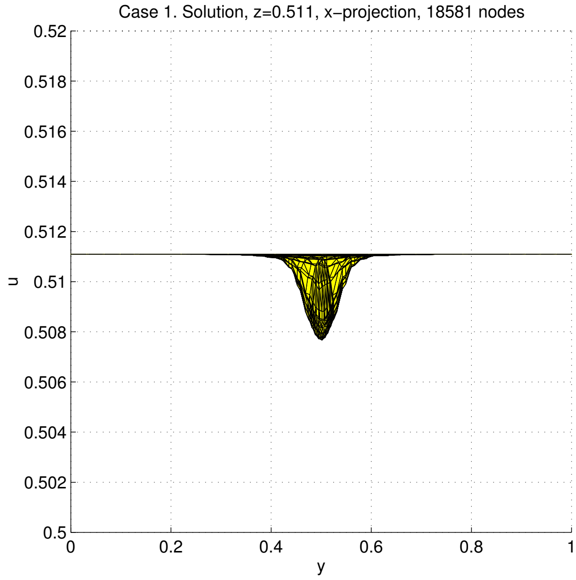

Case 1. Solution, z=0.511, x−projection, 347 nodes



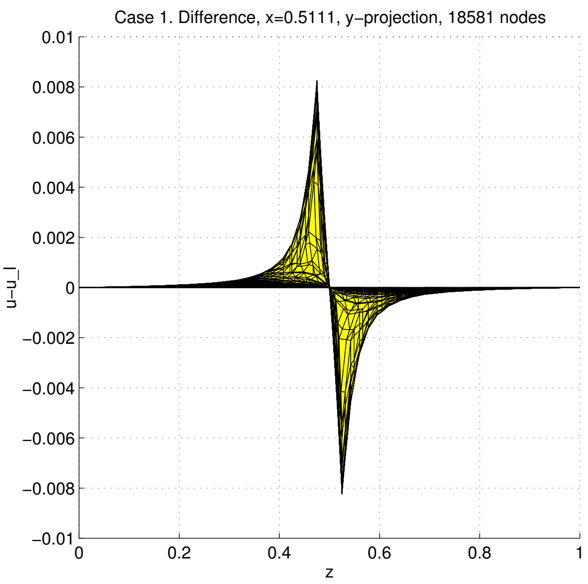

Case 1. Solution, z=0.511, x−projection, 2451 nodes

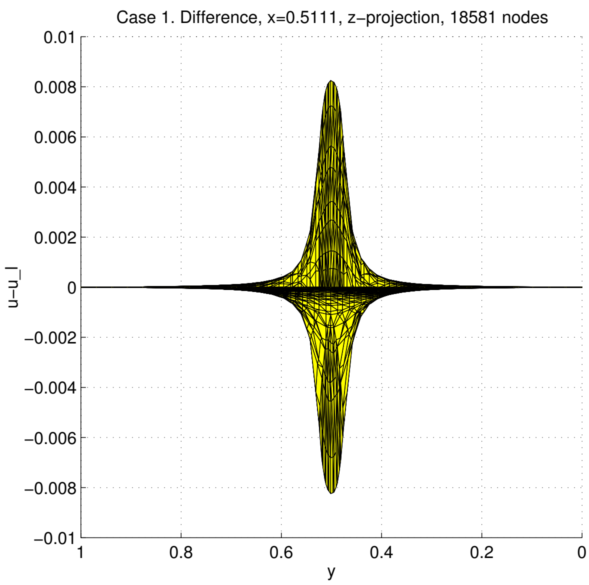


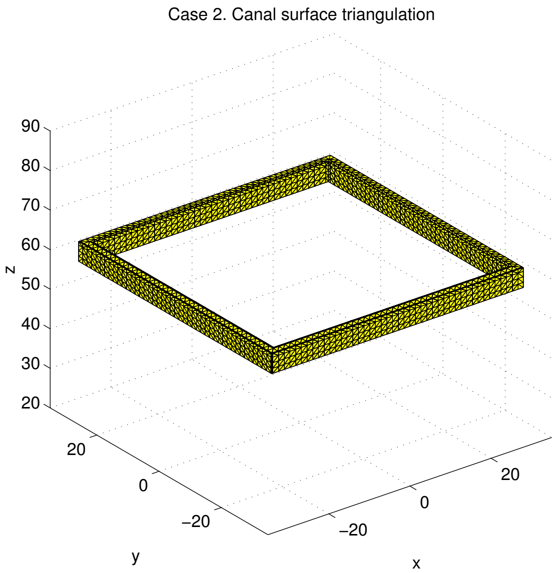

Case 1. Solution, z=0.511, x−projection, 18581 nodes

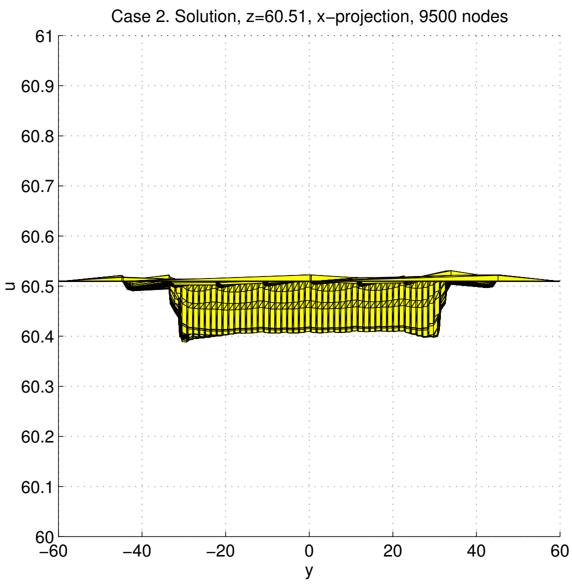


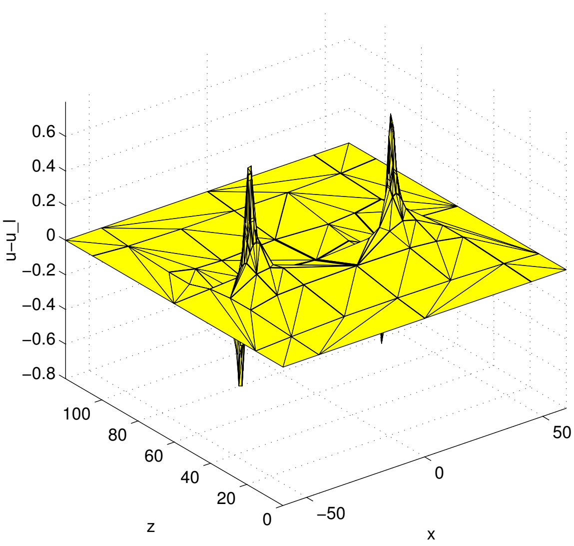
In order to present intersection of the solution $u$ and a plane $x$ =const., we calculate and visualize the "difference" $u - u_l = u(x, y, z) - z$.



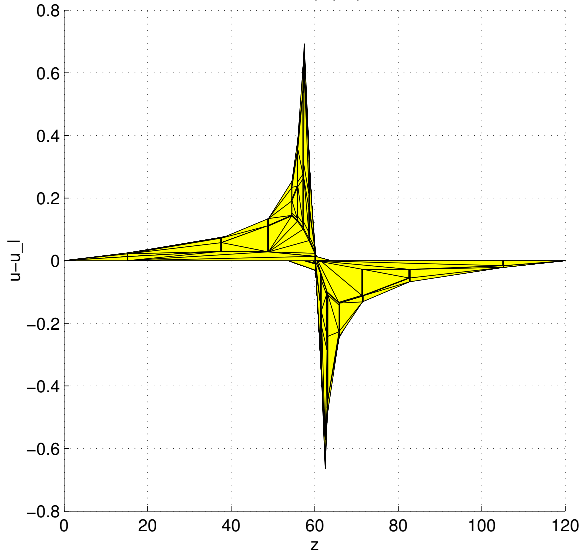



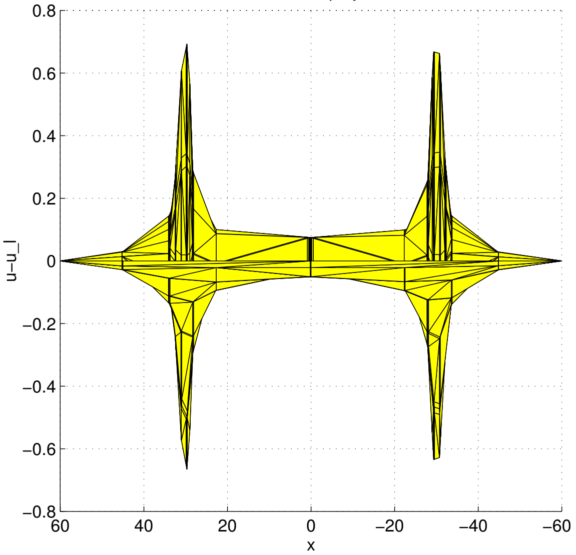





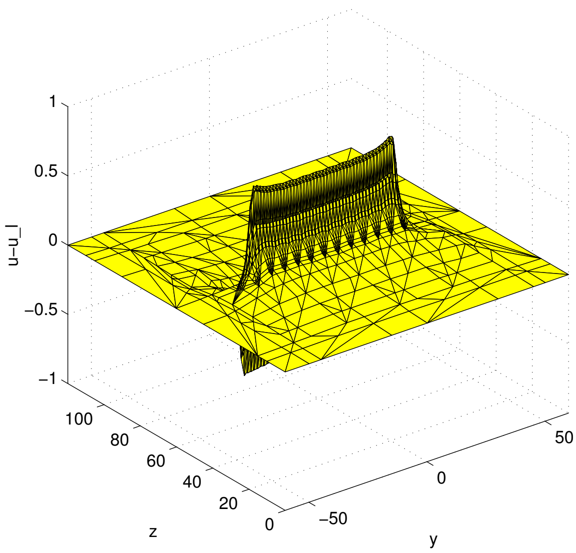

Case 2. Difference, x=0, 9500 nodes



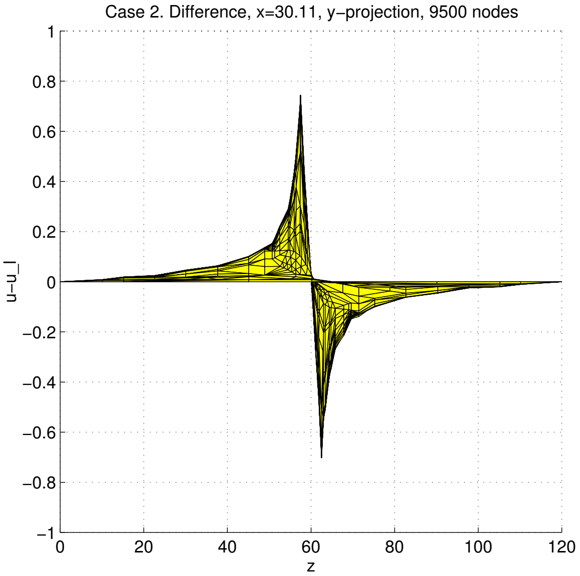

Case 2. Difference, x=0, y−projection, 9500 nodes



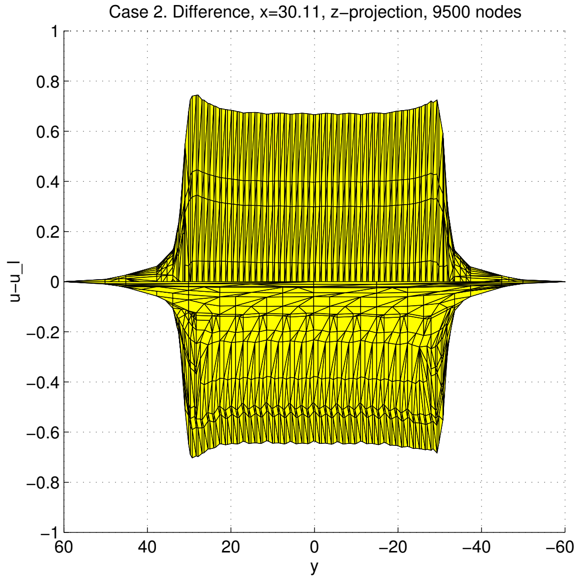

Case 2. Difference, x=0, z−projection, 9500 nodes




Case 2. Difference, x=30.11, 9500 nodes




Case 2. Difference, x=30.11, y−projection, 9500 nodes




Case 2. Difference, x=30.11, z−projection, 9500 nodes