A Simple Cartesian Grid Finite Element Method Using Cut Cells

by

Meriem Ben-Salah

MS (Leibniz University of Hanover) 2008

A report submitted in partial satisfaction of the

Requirements for the degree of

Masters of Science, Plan II

in

Mechanical Engineering

at the

University of California at Berkeley

Committee in Charge:

Professor Panayiotis Papadopoulos, Chairman

Professor Phillip Colella

Fall 2011

Abstract

The cost and quality of unstructured Finite Element meshes have been and still are a stumbling block of the Finite Element Analysis. While the effect and cost of the unstructured mesh quality on the numerical convergence are improved, research is being conducted on developing new methodologies using the simplest mesh possible, a rectangular grid. In this master's report, a novel method for solving two-dimensional partial differential equations, using the Finite Element method on a uniform linear rectangular grid that is non-fitted to the domain boundary is presented. An algorithm for obtaining maximal accuracy from regular grids with minimal boundary intervention is presented. A new approach for applying Dirichlet and Neumann boundary conditions using the partial cells nodal information is developed in one-dimension and in two-dimensions. The partial cells method preserves the convergence order obtained by Finite Element solutions using unstructured meshes for the Dirichlet type of boundary conditions. For some set of problems, the method performs even better than the classical Finite Element method due to the quality of the rectangular grid elements. An adequate trade-off between accuracy and computational performance is experienced. A slight loss of sparsity and a numerically tolerable increase in the condition number of the Finite Element system arise. The accuracy of the partial cells Finite Element method is shown to be negligibly sensitive to the location of the physical domain with respect to the rectangular grid.

Keywords: Finite Element, Cartesian Grid, Cut Cell

Acknowledgments

Research supported by Microsoft (Award #024263) and Intel (Award #024894) funding and by matching funding by U.C. Discovery (Award #DIG07-10227).

Contents

List of Figures

List of Tables

1 Introduction

The cost and quality of unstructured Finite Element meshes have been and still are a stumbling block of the Finite Element Analysis. While the effect and cost of the unstructured mesh quality on the numerical convergence are improved, research is being conducted on developing new methodologies using the simplest mesh possible, a rectangular grid. The latter set of methods are appealing because of their meshing and mathematical simplicity. Moreover, mesh adaptivity, the use of multigrid solvers and coupling to physical submodeling are easily adoptable. When modeling fluid dynamics in irregular geometries, Colella ([1], [2]) uses an embedded boundary volume-of-fluid method, where the intersection of each grid cell with the free boundary, a cut-cell, is specified. The dependent variables are approximated at the centers of the rectangular grid volumes. Johnson ([3], [4], [5]) presents the Boeing TRANAIR rectangular grid finite element method, where the boundary surfaces are prescribed via a network of panels. Recursion is necessary for accurate panel locations. Venkatasubban determines with the Bombardier-Learjet EULAIR method ([6], [7]) the cut-cell region of a rectangular grid by solely the knowledge of integration points on the boundary. Recursion is not necessary in this case. The NASA TIGER solver developed by Melton et. al. ([8], [9], [10], [11]) is a Finite Volume Method that uses non-body-fitted cartesian grids and a border of irregularly shaped cells at the intersection between the grid and the geometry. Young et. al. [12] handle boundary regions through adaptive refining in the proximity of the domain boundary.

In this master's report, a novel method for solving two-dimensional partial differential equations using the Finite Element method on a uniform linear rectangular grid that is non-fitted to the domain boundary is presented. An algorithm for obtaining maximal accuracy from regular grids with minimal boundary intervention is presented. The rectangular grid elements containing arbitrarily shaped domain boundaries are christened partial elements. A new approach for applying Dirichlet and Neumann boundary conditions using the nodal information of the partial cells is developed. The method preserves the convergence order obtained by Finite Element solutions using unstructured meshes for the Dirichlet type of boundary conditions. For some set of problems, the method performs even better than the classical Finite

Element method due to the quality of the rectangular grid elements. An adequate trade-off between accuracy and computational performance is experienced. A slight loss of sparsity and a numerically tolerable increase in the condition number of the Finite Element system arise. The accuracy of the partial cells Finite Element method is shown to be negligibly sensitive to the location of the physical domain with respect to the rectangular grid.

The partial cells method supplements the Finite Element method with the use of rectangular grid without the need for adaptive or conforming meshing along the boundary. Partial elements that are traversed by the domain boundary are distinguished from regular elements that are fully contained in the domain, full elements. Full elements are treated in the classical manner. The treatment of partial elements leads to the fulfillment of boundary conditions. Nodes of the partial grid that lie outside the domain, which are called hanging nodes, do not contribute to the interpolation of the partial differential equation. Involving the degrees of freedom associated with all nodes of the partial element, linear collocation equations are calculated based on the location of the portion of the boundary contained in the partial element. Thereafter, the equations are assigned to the hanging degrees of freedom and assembled in the global system.

First, the partial cells method in one-dimension is presented and convergence in the H^0 and H^1 error norms is shown in Section 2. The effect of the size of the partial domain on the convergence of the method is studied. In Section 2.3, methods that were attempted and compared for preliminary evaluations of the partial cells method are presented. In Section 3, the method is presented in a two-dimensional setting. The convergence behavior for several boundary value problems is studied. The effect of randomness of the location of the domain boundary with respect to the rectangular grid is studied. In addition, the numerical performance of the partial cells method is compared to classical Finite element methods using unstructured meshes. In Section 4, the report is concluded with some remarks and future work.

2 Motivation from One-Dimension

In this section, the proposed method is motivated from a simple one-dimensional problem, which illustrates most of the salient features and the derived response to various approaches for dealing with irregular boundaries.

2.1 Method

Consider a scalar second-order differential equation on a one-dimensional domain $(0, 1)$ that is to be solved by a Galerkin method-based finite element approximation. Let a spatially uniform discretization of the domain with size $h = 1/n$ be constructed, such that the left boundary point ($x = 0$) is occupied by the first finite element node, while the right boundary point ($x = 1$) does not coincide with a node. Thus, the last element is only partially occupying the physical domain of the problem, as in Figure 1. To solve the differential equation by the finite element method using this regular mesh, one may proceed to:

1. Compute the element arrays for all elements including the last element.

2. After assembly, eliminate the equation associated with the degree of freedom of the node that lies outside the physical domain.

3. Substitute the eliminated equation by an interpolation equation to impose the boundary condition at $x = 1$.

Regardless of the type of domain element used for the solution, there are several possible interpolation choices for the partial element. Assuming that the physical domain covers a region of size αh ($0 < \alpha < 1$) inside the rightmost subdomain (x_n, x_{n+1}) in Figure 1, these include:

a. Linear interpolation: Imposing a Dirichlet boundary condition, $u|_{x=1} = \overline{u}$, one readily finds the interpolation equation

$$\overline{u} \ = \ N_n(\alpha h)u_n + N_{n+1}(\alpha h)u_{n+1} \ , \tag{1}$$

where $N_n(t) = -\frac{(t-h)}{h}$ and $N_{n+1}(t) = \frac{t}{h}$. Alternatively, to impose the associated Neumann boundary condition $\frac{\mathrm{d}u}{\mathrm{d}x}|_{x=1} = \overline{q}$, one finds that

$$\overline{q} \;=\; \frac{\mathrm{d}N_n}{\mathrm{d}x}(\alpha h)u_n + \frac{\mathrm{d}N_{n+1}}{\mathrm{d}x}(\alpha h)u_{n+1} \; . \tag{2}$$

Either equation (1) or (2) would be used in lieu of the eliminated equation associated with the degree of freedom of node $n+1$.

b. Quadratic Interpolation: It is easy to impose the Dirichlet or Neumann boundary condition using a quadratic interpolation defined by the degrees of freedom of nodes $n-1$, n and $n+1$. For the Dirichlet boundary condition case, this leads to

$$\overline{u} \;=\; N_{n-1}(\alpha h)u_{n-1} + N_n(\alpha h)u_n + N_{n+1}(\alpha h)u_{n+1} \; , \tag{3}$$

where $N_{n-1}(t) = \frac{(t-h)(t-2h)}{2h^2}$, $N_{n-1}(t) = -\frac{t(t-2h)}{h^2}$, and $N_{n+1}(t) = \frac{t(t-h)}{2h^2}$. We may similarly impose the associated Neumann boundary conditions as

$$\overline{q} \;=\; \frac{\mathrm{d}N_{n-1}}{\mathrm{d}x}(\alpha h)u_{n-1} + \frac{\mathrm{d}N_n}{\mathrm{d}x}(\alpha h)u_n + \frac{\mathrm{d}N_{n+1}}{\mathrm{d}x}(\alpha h)u_{n+1} \; . \tag{4}$$

Figure 2 shows the two kinds of interpolation. Higher-order interpolations are also possible, but will not be discussed here.

2.2 Convergence analysis

The preceding methods are analyzed using a simple test problem to appreciate the rates of convergence attained by the interpolation of the boundary condition at $x = 1$. The test problem is specified as follows:

$$\frac{\mathrm{d}^2u}{\mathrm{d}x^2} \;=\; -x \quad \text{on} \quad 0 < x < 1, \tag{5}$$

$$u|_{x=0} \;=\; 0. \tag{6}$$

The Dirichlet boundary condition is set to

$$\overline{u} \;=\; 0.5 \quad \text{at } x = 1 \; , \tag{7}$$

while, alternatively, the corresponding Neumann boundary condition is

$$\overline{q} \;=\; 0.5 \quad \text{at } x = 1 \; . \tag{8}$$

In addition, the parameter α is set to $\alpha = 0.5$, which means that the physical boundary bisects the last subdomain (x_n, x_{n+1}).

2.2.1 Dirichlet-Dirichlet Model Problem

In the Dirichlet-Dirichlet case, where the boundary condition (7) is enforced, the interpolation of the dependent variable is taken to be quadratic over the last two elements. Figure 3 shows the rates of convergence in the H^0 and H^1 norm, respectively. The rates of convergence are 2 and 1, hence are identical to those expected of conventional elements without any special treatment related to partial cells, see, e.g., [13].

2.2.2 Dirichlet-Neumann Model Problem

The theoretical rates are also observed in the case of the Dirichlet-Neumann problem, which makes use of the boundary condition (8). Here, the interpolation of the flux results from taking the derivative of the (quadratic) interpolation of the dependent variable over the last two elements of the domain. The relevant results for the H^0 and H^1 norm are shown in Figure 4.

2.2.3 Partial Element Effect on Convergence

For the preceding one-dimensional problem, the finite element method exhibits the same order of convergence independently of the type of boundary conditions imposed. It is important to investigate whether these rates persist independently of the size of the partial element containing the boundary of the domain and that the recovered results are, hopefully, insensitive to the specific placement of the actual boundary point in the last element of the mesh. Therefore, the previous canonical problem is resolved by varying the location of the boundary of the physical domain with respect to the finite element mesh.

By varying the size of the (uniform) mesh, three different partial coverings of the last domain element by the physical domain are defined: they cover the leftmost 30% (0.3), 50% (0.5) of 80% (0.8) of the last element's domain. The convergence rates in the H^0 and H^1 norms for the Dirichlet-Dirichlet boundary problem are shown in Figure 5. For all three cases, the same rates apply, which shows that there is no sensitivity to the size of the partial element.

The rates of convergence for the Dirichlet-Neumann problem exhibit the same insensitivity to the size of the partial element, provided that a linear interpolation of the flux is effected, as described earlier in this section. Figure 6 shows the corresponding H^0 and H^1 rates.

2.3 Alternative Methodologies

The proposed approach is one of several that one may consider in attempting to use partially covered elements with Dirichlet or Neumann boundary conditions. Other options include:

(a) Constant flux interpolation

In this case, the Neumann boundary condition is enforced by computing the derivative of the dependent variable in the cut element itself. While this localizes (hence simplifies) the enforcement of the boundary condition, it compromises the convergence rate, as shown in Figure 7. This is due to the fact that the lower order of accuracy in the interpolation of the flux translates to a commensurately lower order of accuracy in the overall solution. An exception occurs when the physical boundary bisects the element, in which case the theoretical rate of convergence is recovered due to superconvergence. However, such bisection cannot be guaranteed and does not generalize to higher dimensions, hence is viewed as inconsequential.

(b) Underintegration:

In analogy to the active set method, one may consider treating partial elements by selectively suppressing the contributions to global arrays from Gauss points that lie outside the physical domain. This method also leads to solutions of lower-order of accuracy, since underintegration leads to unstable elements in the vicinity of the exterior boundary that, at best, transmit low-order accurate information to the interior of the domain.

(c) Scaled Stiffness

Scaling the stiffness of partial elements by the ratio of the length of the physical domain enclosed in the cut cell to the length of the entire element circumvents the singularity of the underintegration, but still leads to lower than the theoretical convergence.

3 Two-dimensional development

3.1 Method

The preceding one-dimensional analysis is used as a starting point for the development of an accurate and efficient two-dimensional treatment of partial elements. In what follows, a two-dimensional development is presented in the context of four-node quadrilateral elements. A parallel analysis on triangles or higher-order elements is also possible, although such analysis is not included here. The canonical problem will be a second-order elliptic partial differential equation of the form

$$\nabla^2 u + f \;=\; 0 \quad \text{in } \Omega \tag{9}$$

subject to boundary conditions

$$\begin{aligned} u \;&= \overline{u} \quad \text{on } \Gamma_d & (10)\\ -\nabla u \cdot \mathbf{n} \;&= \overline{q} \quad \text{on } \Gamma_n \;, & (11) \end{aligned}$$

where $\overline{u}$ and $\overline{q}$ are the imposed boundary conditions on the Dirichlet and Neumann parts of the boundary Γ_d and Γ_n, respectively. Also, the boundary $\partial\Omega$ of the domain Ω is assumed to possess a unique outward normal $\mathbf{n}$ at any point, and also to satisfy the conditions $\overline{\Gamma_d \cup \Gamma_n} = \partial\Omega$ and $\Gamma_d \cap \Gamma_n = \emptyset$.

The main prescriptors of the proposed method are as follows:

1. All meshes are taken to be structured, hence all four-node elements are actually square elements of side-length h. This is a highly desirable feature from the point of view of mesh generation. A simple edge-defined bounding box is generated to broadly circumscribe the physical domain. Without any loss of generality, it may be assumed that the element edges align with the coordinates x and y of a rectangular Cartesian system.

2. The elements of the mesh grid are sorted into three distinct categories: (a) fully included in the physical domain ("full elements"), (b) partially included in the physical domain ("partial elements") and (c) totally outside the physical domain ("void elements"), see Figure 9. Generally, an element is classified as

partial if it contains at least one interior and at least one exterior node to the physical domain. A bounding box is used to expedite the classification of the elements. The proposed algorithm requires that there exists at least one node that is interior to the physical domain, which is, of course, a very mildly limiting assumption. Special cases arise when the physical boundary passes through nodes, as in Figure 10. These cases are resolved, as described later in this section.

3. Partial elements are subject to boundary conditions, which, unlike conventional elements apply along lines that intersect the element domain. In the ensuing development, it is essential to determine which nodes of the partial elements lie inside, outside or on the boundary of the physical domain. Degrees of freedom of nodes that lie outside of the domain ("hanging nodes") do no contribute the normal domain-based residual and stiffness terms as do the rest of the nodes. Their contribution to the overall solution is through enforcement of the physical boundary conditions, as discussed later.

4. Without consideration of boundary conditions and degrees of freedom associated with hanging nodes, element arrays are computed for full and partial elements alike, but only rows corresponding to non-hanging nodes are assembled into the global arrays. Also, upon assembling, degrees-of-freedom associated exclusively with void elements are outright deleted.

5. In analogy to the one-dimensional case, all equations associated with degrees-of-freedom of the hanging nodes are substituted by appropriate boundary condition interpolation equations. Formulating such equations requires finding the intersection of the physical domain with partial element domains. Initially, all intersections are determined between the physical domain and the boundaries of partial elements. It is optimal to have as many intersection points as hanging nodes, since each intersection generates a boundary condition that can be assigned to a neighboring hanging node. Such a pairing of a hanging node to an intersection point eliminates the potential singularity in the global stiffness associated with a degree(s) of freedom of the hanging node. To simplify

the structure of the pairing algorithm, all hanging node-to-intersection point assignments are handled locally (i.e., at the level of the partial element).

6. The pairing of hanging nodes to intersection points is resolved by a minimum-distance search (namely, a hanging node with multiple candidate intersection pairings is always assigned to its nearest intersection point). Once a pairing is established, Dirichlet boundary conditions may be imposed by a unidirectional interpolation of the degree of freedom u, which is effected using the value u_i at the interior node i and the value $\overline{u}$ at the intersection point, such that the value u_h of this degree of freedom at the hanging node is determined from

$$\overline{u} \;=\; \begin{cases} \dfrac{\overline{x}-x_h}{h}u_i + \dfrac{\overline{x}-x_i}{h}u_h & \text{for } x\text{-directional interpolation} \\ \dfrac{\overline{y}-y_h}{h}u_i + \dfrac{\overline{y}-y_i}{h}u_h & \text{for } y\text{-directional interpolation} \end{cases} \quad , \qquad (12)$$

where $(\overline{x}, \overline{y})$ are the coordinates of the intersection point paired with the hanging node. When there are fewer intersection points than hanging nodes, additional interpolation points need to be generated on the boundary of the physical domain contained in a partial element. A simple algorithm for determining the location of such additional interpolation points involves averaging over a pair of intersection points in one direction, and mapping the coordinate to boundary point in the physical domain boundary. Some examples of such intersection cases are depicted in Figure 8.

Neumann boundary conditions may be imposed as

$$-\,\overline{q}\Big|_{\Gamma_n} \;=\; \begin{cases} \begin{pmatrix} \dfrac{1}{h}(u_i+u_h) \\ 0 \end{pmatrix} \cdot \begin{pmatrix} n_x \\ n_y \end{pmatrix} & \text{for } x\text{-directional interpolation} \\ \begin{pmatrix} 0 \\ \dfrac{1}{h}(u_i+u_h) \end{pmatrix} \cdot \begin{pmatrix} n_x \\ n_y \end{pmatrix} & \text{for } y\text{-directional interpolation} \end{cases} \qquad (13),$$

where (n_x, n_y) are the coordinates of the unit normal $\mathbf{n}$.

7. An alternative, bi-directional interpolation, may be also pursued as follows: Denote by S_1 and S_2 the intersection points of the domain boundary with a partial element and let these points have coordinates $(\overline{x}_1, \overline{y}_1)$ and $(\overline{x}_2, \overline{y}_2)$,

respectively. One, two or three sampling points with coordinates (x_k, y_k), $k = 1, \ldots$, are subsequently located on the portion of the domain boundary which traverses a partial element, depending on whether there are one, two or three hanging nodes, see Figure 11.

For the case of Dirichlet boundary conditions, the bi-directional interpolation yields the equation

$$\overline{u}_k \; = \; \sum_{I=1}^{4} N_I(\overline{x}_k, \overline{y}_k) u_I \quad , \quad k = 1, \ldots \; , \tag{14}$$

where N_I are the conventional element interpolation functions of a four-node square element.

$$N_1(\overline{x}, \overline{y}) \; = \; \frac{(y_2 - \overline{y})(x_2 - \overline{x})}{(y_2 - y_1)(x_2 - x_2)}, \tag{15}$$

$$N_2(\overline{x}, \overline{y}) \; = \; \frac{(y_2 - \overline{y})(\overline{x} - x_1)}{(y_2 - y_1)(x_2 - x_2)}, \tag{16}$$

$$N_3(\overline{x}, \overline{y}) \; = \; \frac{(\overline{y} - y_1)(x_2 - \overline{x})}{(y_2 - y_1)(x_2 - x_2)}, \tag{17}$$

$$N_4(\overline{x}, \overline{y}) \; = \; \frac{(\overline{y} - y_1)(\overline{x} - x_1)}{(y_2 - y_1)(x_2 - x_2)}. \tag{18}$$

For the case of Neumann boundary conditions, the corresponding equations become

$$-\,\overline{q} \; = \; \sum_{I=1}^{4} \left[\nabla N_I(\overline{x}_k, \overline{y}_k) \cdot \mathbf{n} \right] u_I \quad , \quad k = 1, \ldots \; . \tag{19}$$

Unless stated otherwise, the forthcoming numerical simulations employ the preceding bi-directional interpolation for partial elements.

8. Since some hanging nodes are shared by several partial elements, the boundary condition interpolation equations generated per partial element are assembled into the global arrays in the usual finite element manner.

9. For the case of a partial element as in Figure 12, the node on the physical boundary is viewed as a hanging node which is assigned any imposed Dirichlet boundary conditions. However, when imposing Neumann boundary conditions, it is essential to perform an interpolation of the boundary condition over the

entire partial element, as previously described. This is because the physical and element boundaries coincide only at a point, and imposing the Neumann boundary directly at that point entails a significant error.

3.2 Dirichlet Model Problems

Circular Domain

Consider the solution of the two-dimensional canonical problem (9) for constant f on a circular domain of radius R and subject to Dirichlet boundary conditions (10) on the whole boundary. The analytical solution to the problem reads

$$u \;=\; u(r) \;=\; \overline{u} + \frac{f}{4}R^2 - \frac{f}{4}r^2 \;, \tag{20}$$

where r is the radial distance from the center of the circle.

As concluded from Figure 13, the rate of convergence in the H^0 error norm is quadratic in the element size h, while the corresponding H^1 norm is linear in h, as would be the case with conventional linearly complete elements for this problem.

It is instructive to compare the convergence rates for this problem when using uni-directional vs. bi-directional interpolation for the partial elements. Figure 14 shows that in the H^0-error norm, the bi-directional interpolation exhibits an appreciable gain in accuracy over the uni-directional case, whereas both methods exhibit the same convergence rate in the H^1-error norm.

Annular Domain

The homogeneous counterpart of (9) is solved on a annular domain of outer radius R_1 and inner radius R_2. Each of the boundaries is subject to homogeneous Dirichlet boundary conditions, $u = \overline{u}_1$ on the outer boundary $u = \overline{u}_2$ on the inner boundary.

The analytical solution to this problem is given by

$$u \;=\; u(r) \;=\; \frac{\overline{u}_1 - \overline{u}_2}{\ln(R_1) - \ln(R_2)}\ln(r) + \overline{u}_1 - c_1\ln(R_1) \;. \tag{21}$$

Again, the theoretical rates of convergence are attained, as seen from Figure 15.

3.3 Dirichlet-Neumann Problem for Annular Domain

The homogeneous version of the partial differential equation (9) is solved on an annular domain, such that the outer boundary is subject to Dirichlet boundary conditions, $u(R_1) = \overline{u}$, while the inner boundary is subject to Neumann boundary conditions,

$$\frac{\partial u}{\partial n}(R_2) \;=\; \bigtriangledown u \cdot \boldsymbol{n}(R_2) \;=\; -\overline{q}\ . \tag{22}$$

For this problem, the analytical solution reads

$$u \;=\; u(r) \;=\; (-R_2\overline{q})(\ln(r) - \ln(R_1)) + \overline{u}\ . \tag{23}$$

The convergence rate in the H^0-norm is approximately of order 1.5 in h, while the corresponding rate in the H^1-norm is linear, as seen from Figure 16. In contrasting the results to those of the Dirichlet-Dirichlet, it is reasonable to attribute the loss of half-an-order in the H^0-norm to the linear interpolation for the Neumann boundary conditions. However, the H^1 rate is, predictably, unaffected by this linear approximation.

3.4 Effect of randomness

In this section, the effect of symmetry in the placement of partial cells across the finite element domain is investigated. The goal is to qualitatively understand if the convergence rates are affected by the location of finite elements relative to the underlying regular grid. This is particularly relevant if elements are placed in such a manner that use is made of superconvergence properties, as was the case in one-dimension (see Section 2.3). To eliminate the possibility of symmetry-enhanced behavior, the physical domain is positioned at an offset from the center of the rectangular grid, thus breaking any symmetries.

Dirichlet on Circular Domain

Let the two-dimensional finite element bounding box occupy the domain $[0, 2]\times$

$[0, 2]$, and also let the circular domain have radius $R = 1/2$. The finite element solution is computed for various placements of the center of the circle, which is taken to move horizontally starting from the center coordinates $(2/3, 2/3)$ and then vertically starting again from horizontally starting from the center coordinates $(2/3, 2/3)$. In both cases, the position of the center is incremented by 0.1 units. In addition, the finite element grid size remains constant throughout the analysis and equal to $h = 0.015625$.

The variation of the H^0 and H^1 errors is plotted in Figure 17 and 18 plotted against the position of the center of the circle. In both cases, it is seen that the sensitivity of the error to the exact placement of the center is very small.

Dirichlet-Dirichlet and Dirichlet-Neumann on Annular Domain

The previous analysis is repeated here for the annular domain with outer radius $R_1 = 0.5$ and inner radius $R_2 = 0.25$. The H^0 and H^1 errors display the same minimal sensitivity to the placement of the center as in the circular case, see Figures 19 and 20.

The same observation applies to the Dirichlet-Neumann case, for which the dependence of errors on center placement is depicted in Figures 21 and 22.

3.5 Structured vs unstructured grids

Given that the principal motivation of this work is circumventing the need for unstructured meshing, it is important to investigate the quality of the finite element solutions as compared to those obtained by on unstructured grids, i.e. grids that conform to the boundary. To this end, a circular domain is discretized using 12×4^n quadrilateral elements for $n = 0, 1, \ldots, 7$. As implied by the size of the meshes, refinement was effected by dividing each element of the original 12-element mesh of Figure 23 into 4 elements and then repeating the procedure $n = 6$ additional times. The nodes situated on the exterior boundary are always forced to lie exactly on the circle. As the original mesh was successively refined, the mesh quality toward the center of the circle deteriorates. Therefore, in order to make a fair comparison to the partial cells solutions, the unstructured mesh was smoothed with each refinement by employing the neighbor averaging technique using as many cycles as needed for the position of all nodes to converge. This led to a good quality meshing throughout the domain, although some non-uniformity persists, as shown in Figure 23.

Figure 24 reveals that the partial cells algorithm on a structured grid leads to the same convergence rate as the conventional finite element solution on an unstructured mesh. However, the absolute error of the partial cell solution is smaller than that of the unstructured grid solution. This result may be interpreted in connection with the classical error bound,

$$\|\boldsymbol{u} - \boldsymbol{u}^h\|_E \leq ch^{q+1} \ , \tag{24}$$

as meaning that although the polynomial order of completeness is the same ($q = 1$), the constant c in the structured grid solution is driven by the square shape. In addition, the preceding formula implies an averaged mesh-size h which, in the case of the structured grid is essentially constant.

The convergence behavior of the finite element solution using the initial mesh compared to the use of an improved mesh by the neighbor averaging technique reveals no big of a difference at coarser meshes, as shown in Figure 25. A tiny difference becomes apparent if the mesh size is very small. Convergence rate in the corresponding error norm are consistent to previous findings.

The properties of the algebraic system obtained by the partial cells method are compared to those of a system obtained by the classical Finite Element method using an unstructured mesh. The Dirichlet boundary value problem on a circular domain presented in Section 3.2 is solved using a rectangular grid and the partial cells method and also using a good quality unstructured mesh and the classical Finite Element method. As listed in tables 1 and 2, due to the equations associated with the hanging degrees of freedom, the condition number of the algebraic system obtained by the partial cells method is larger than the condition number of an algebraic system of comparable size obtained by the classical Finite Element method. Moreover, the number of zero elements in the system matrix decreases with the use of the partial cells method.

While no testing has been conducted on more complex domains than the circle, it is reasonable to assume that the algorithm will behave in the same manner for any smooth simply- or multiply-connected domain, since locally the boundary of any such domain may be viewed as being diffeomorphic to a circular arc.

4 Conclusions

In this master's project, a novel algorithm for solving partial differential equations in one-dimension and in two-dimensions on curved domains discretized by regular grids that are non-fitted to the domain boundary is presented. The algorithm is shown to work for circular and annular domains, and can therefore be applied for general shaped continuous domain boundaries that are topologically mappable to a circle. A simple algorithm for handling Dirichlet and Neumann boundary conditions is shown to conserve and for some model cases to outperform the convergence order of the classical Finite Element method using unstructured grids. The algorithm sensitivity in terms of the relative position of the physical domain with respect to the rectangular grid is demonstrated to be negligible. However, system sparsity and condition number are shown to deteriorate slightly with the new approach.

The future work includes employing nonlinear interpolation for imposing Neumann boundary conditions in two-dimensions, i.e. sharing the Neumann boundary conditions over four neighboring elements and more instead of a single partial element. The simplicity of the algorithm, convergence order, sparsity and condition number of the associated systems are to be investigated. The algorithm is shown to solve two-dimensional boundary value problems. Therefore, it is valuable to extend the algorithm to three-dimensional settings and investigate its cost. Looking ahead, the partial cells method offers a good alternative for solving partial differential equations with mixed boundary conditions on complex shaped domains minimizing the meshing cost and maximizing accuracy.

References

[1] A. Almgren, J. Bell, P. Colella, and L. Howell. An adaptive pro jection method for the incompressible navier-stokes equations. *In Proc. IMACS 14th World Conference*, 1994.

[2] Philip Colella. Volume-of-fluid methods for partial differential equations in irregular domains. *National Center for Theoretical Sciences Taiwan Summer Research Program on Applied Mathematics:Numerical PDEs*, 2011.

[3] Forrester T Johnson. The tranair rectangular grid approach to solving the nonlinear full-potential equation about complex configurations. *Sadhana*, 16, 1991.

[4] John E. Bussoletti, Forrester T. Johnson, Satish S. Samant, David P. Young, and Richard H. Burkhart. Em-tranair: steps toward solution of general 3d maxwell's equations. In *Proceedings of the 10th international conference on computing methods in applied sciences and engineering on Computing methods in applied sciences and engineering*, pages 49–72, Commack, NY, USA, 1991. Nova Science Publishers, Inc.

[5] P. E. Rubbert, J. E. Bussoletti, F. T. Johnson, K. W. Sidwell, W. S. Row, S. S. Samant, G. Sengupta, W. H. Weatherill, R. H. Burkhart, B. L. Everson, D. P. Young, and A. C. Woo. Advances and trends. *Amer. Sot. Mech. Eng.*, 1986.

[6] Chittur S. Venkatasubban. Eulair: a novel finite-element-based cartesian grid euler flow solver that does not require cut-cell geometric information at solid boundaries. volume 458, 2002.

[7] Chittur S. Venkatasubban. A new nite element formulation for ale (arbitrary lagrangian eulerian) compressible uid mechanics. *Int. J. Engng Sci.*, 33, 1995.

[8] Marsha Berger, Michael Aftomis, and John Melton. Accuracy, adaptive methods and complex geometry. *Proceedings First AFOSR Conference on Dynamic Motion CFD*, 1996.

[9] M. Aftosmis, J. Melton, and M. Berger. Robust and e?cient cartesian mesh generation for component-based geometry. *35th AIAA Aerospace Sciences Meeting*, 1997.

[10] M. Aftosmis, J. Melton, and M. Berger. Adaptation and surface modeling for cartesian mesh methods. *12th AIAA Computational Fluid Dynamics Conference*, 1995.

[11] John Melton. Automated three-dimensional cartesian grid generation and euler flow solutions for arbitrary geometries. *PhD thesis U.C. Davis*, 1996.

[12] David P. Young, Robin G. Melvin, Michael B. Bieterman, Forrester T. Johnson, Satish S. Samant, and John E. Bussoletti. A locally refined rectangular grid finite element method: Application to computational fluid dynamics and computational physics. *Journal of Computational Physics*, 92, 1991.

[13] P.G. Ciarlet. *The Finite Element Method for Elliptic Problems.* North-Holland, Amsterdam, 1980.

A Tables

System Size	# of zeros	condition #
177	29840	1.129614477696420e+02
737	536736	4.399926060010600e+02
3009	9027392	1.724143589686158e+03

Table 1: Properties of algebraic system obtained by the classical Finite Element method

System Size	# of zeros	condition #
176	29652	7.320227038448576e+03
721	513924	1.341935714038396e+04
2960	8736140	8.697849117182761e+04

Table 2: Properties of algebraic system obtained by the partial cells method

B Figures

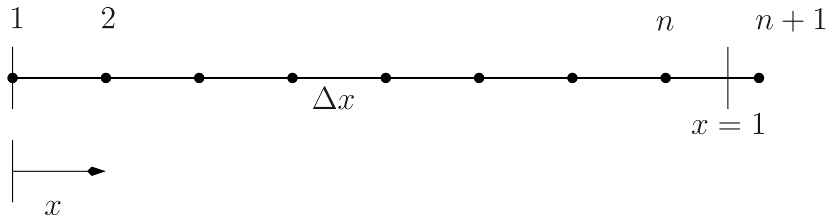

Figure 1: One dimensional domain in a grid mesh

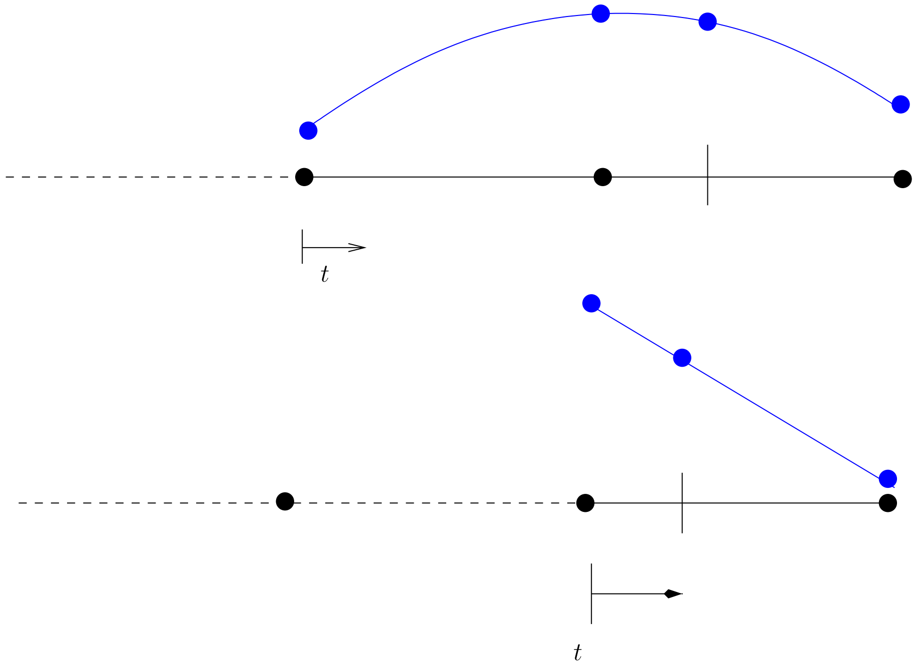

Figure 2: Top: linear Interpolation over partial element. Bottom: quadratic interpolation over full element adjacent to partial element

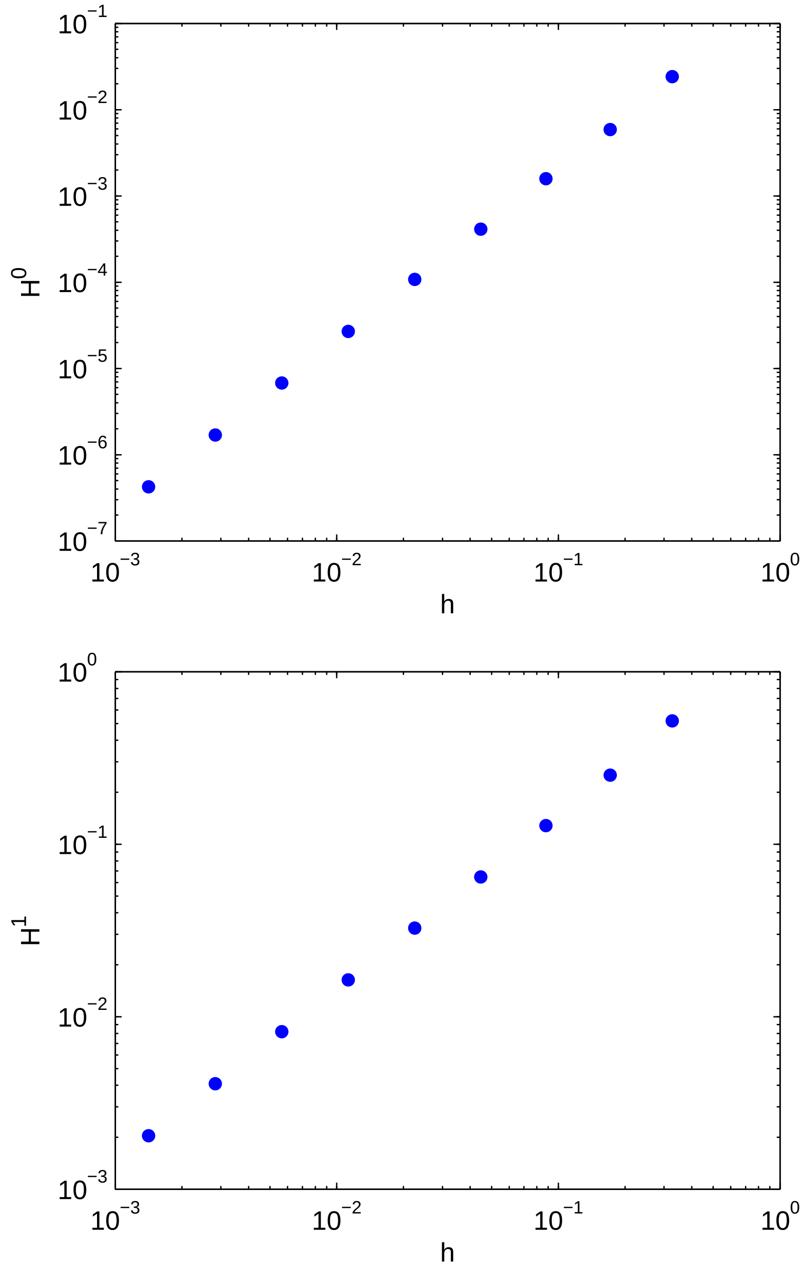

Figure 3: H^0- and H^1-convergence rate of one-dimensional Dirichlet-Dirichlet-problem solved with finite elements and partial cells

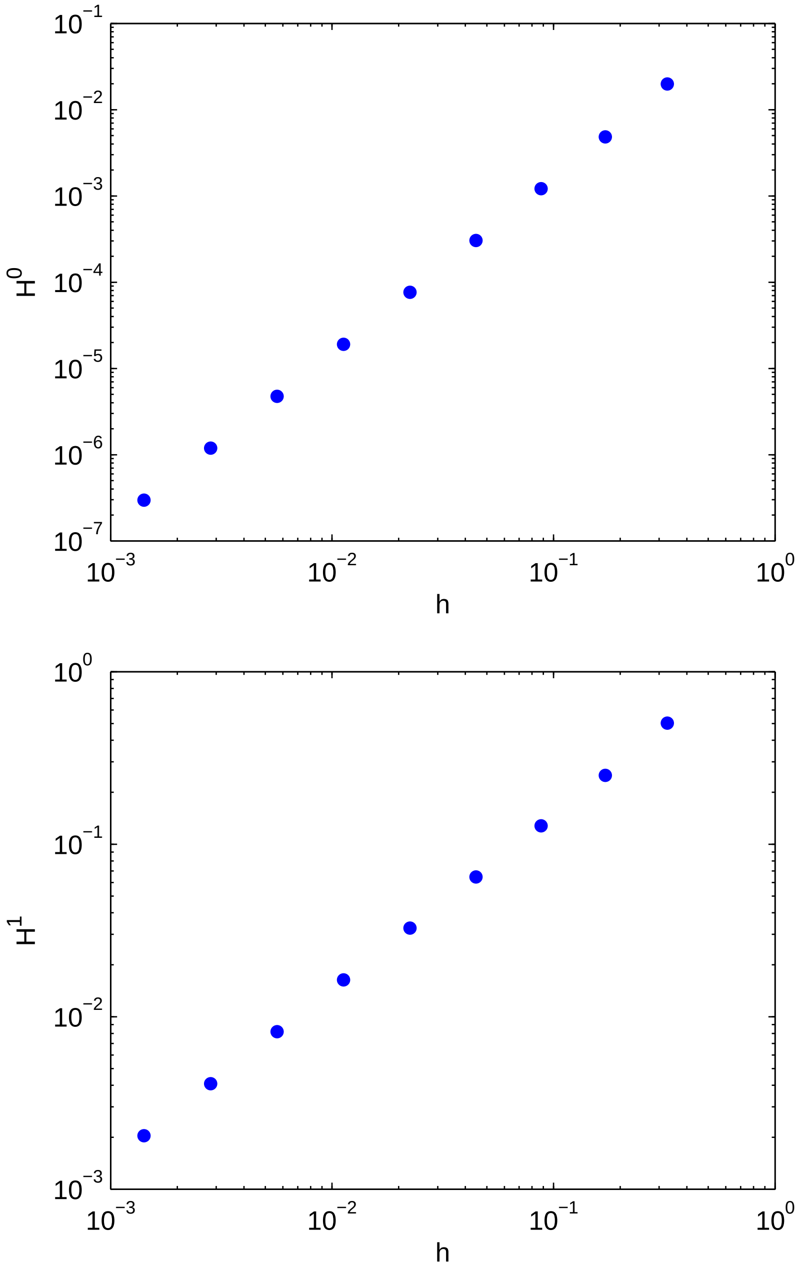

Figure 4: : H^0- and H^1-convergence rate of one-dimensional Dirichlet-Neumann-problem solved with finite elements and partial cells

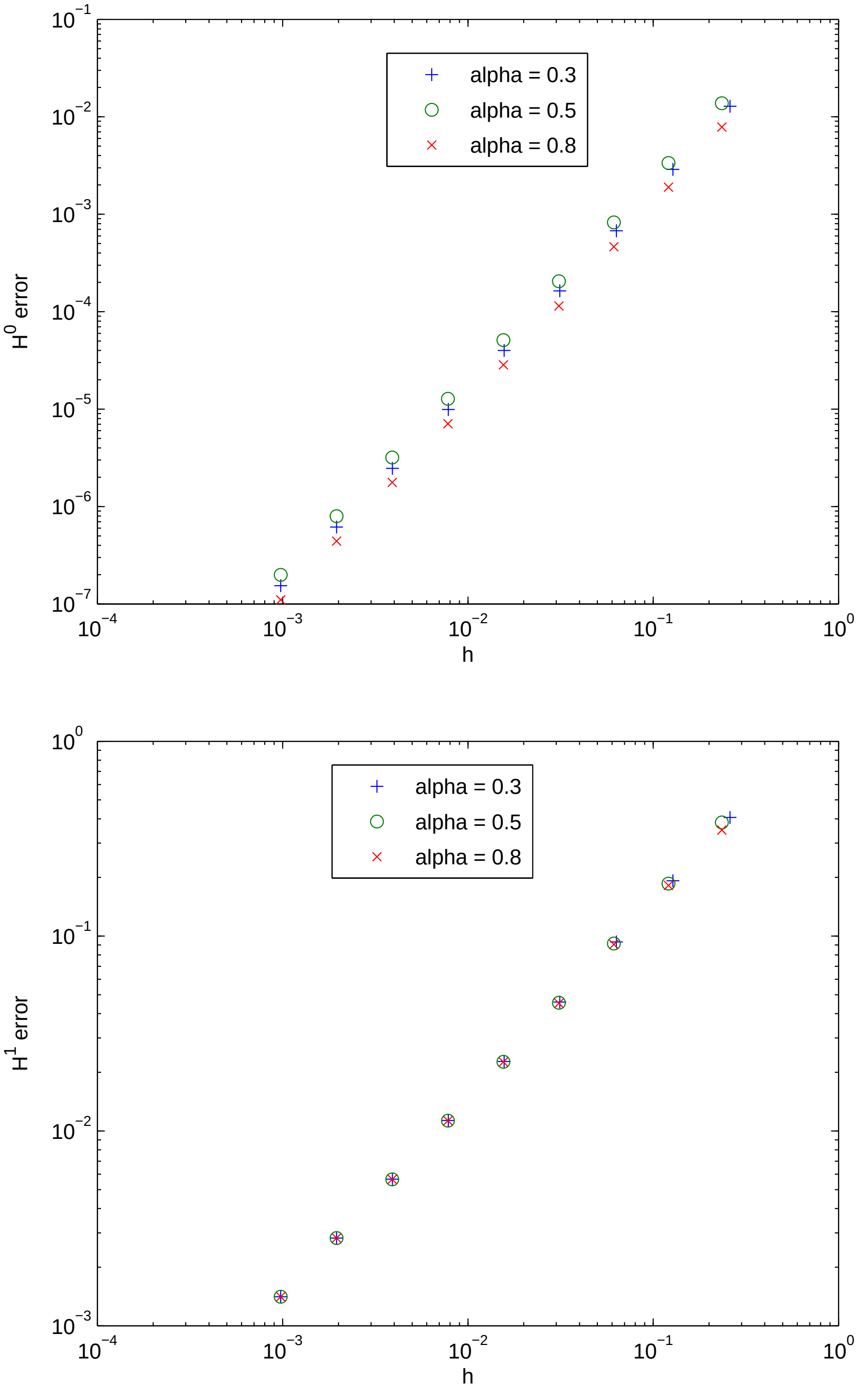

Figure 5: Comparison of H^0- and H^1-convergence rates of a one-dimensional Dirichlet-Dirichlet-problem solved with finite elements and partial cells for different partial cells ratios $\alpha = 0.3, 0.5, 08$

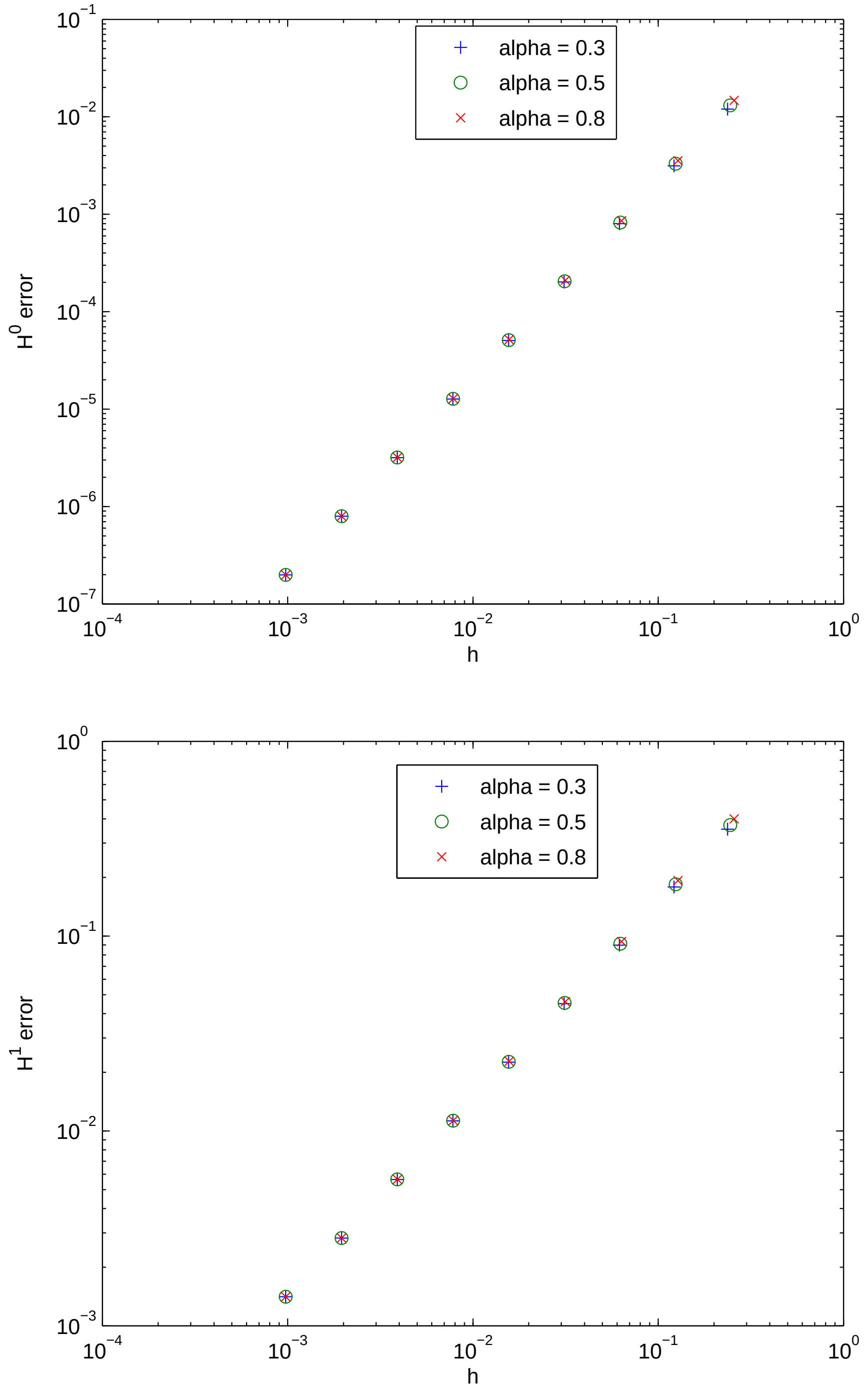

Figure 6: Comparison of H^0- and H^1-convergence rates of a one-dimensional Dirichlet-Neumann-problem solved with finite elements and partial cells for different partial cell ratios $\alpha = 0.3, 0.5, 08$

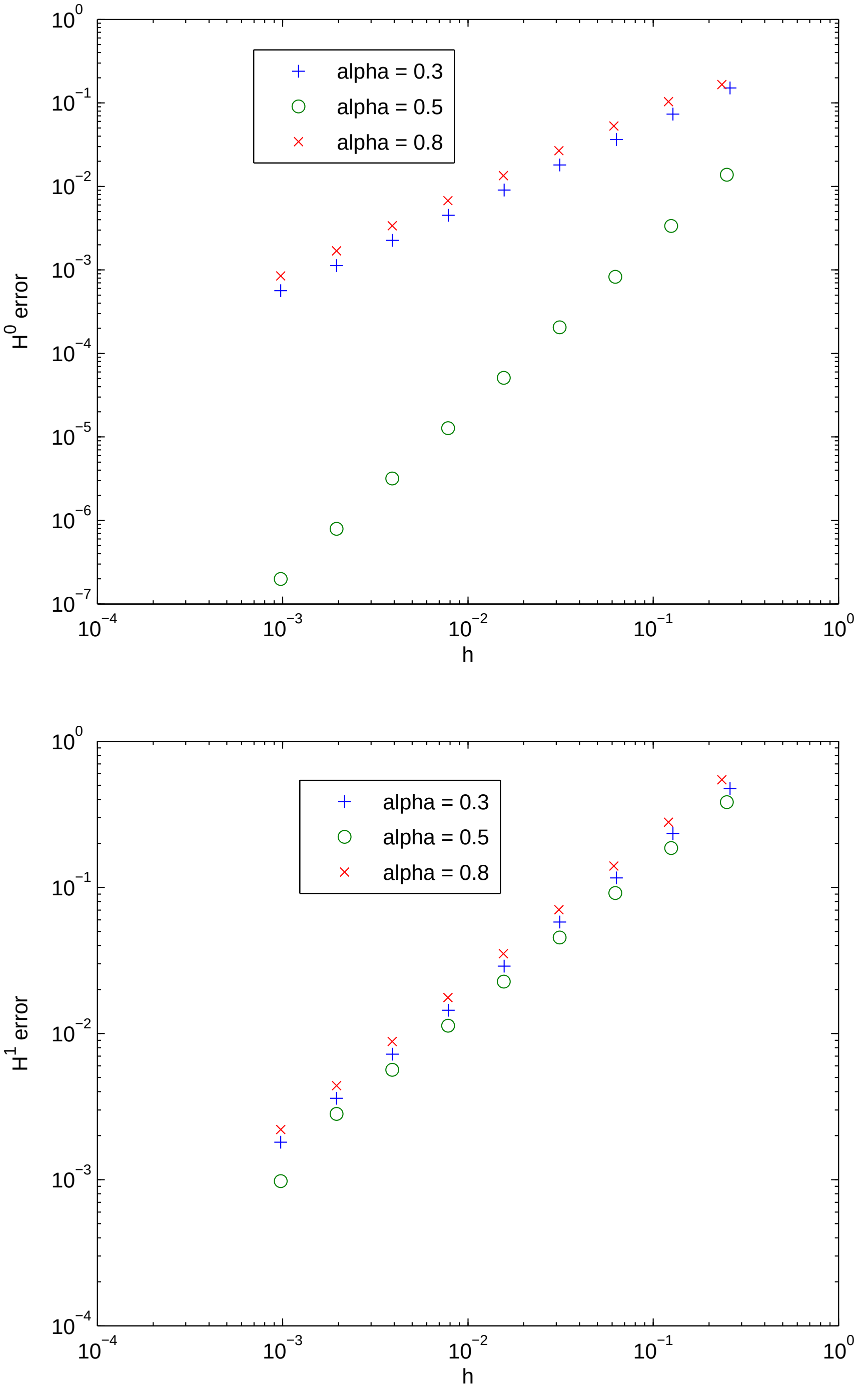

Figure 7: Comparison of H^0- and H^1-convergences of one-dimensional Dirichlet-Neumann-problem solved with finite Elements and partial Cells using constant interpolation of Neumann boundary condition for different partial cell ratios $\alpha = 0.3, 0.5, 0.8$

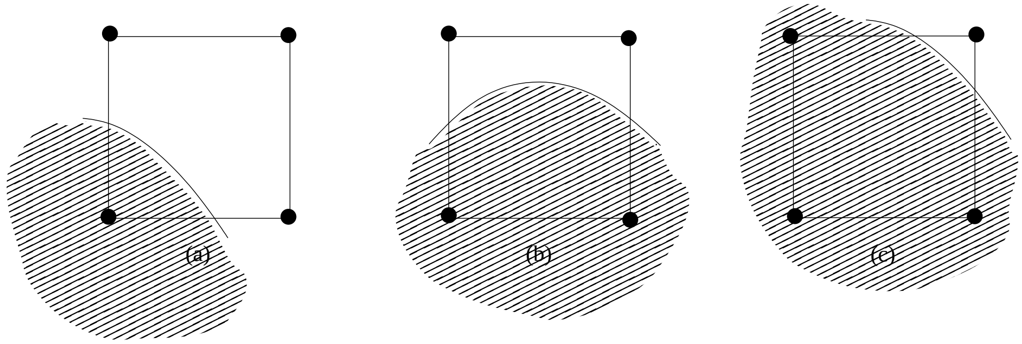

Figure 8: Partial Element Cases: (a) 3 hanging nodes and 2 intersection points (b) 2 hanging nodes and 2 intersection points (c) 1 hanging node and 2 intersection points

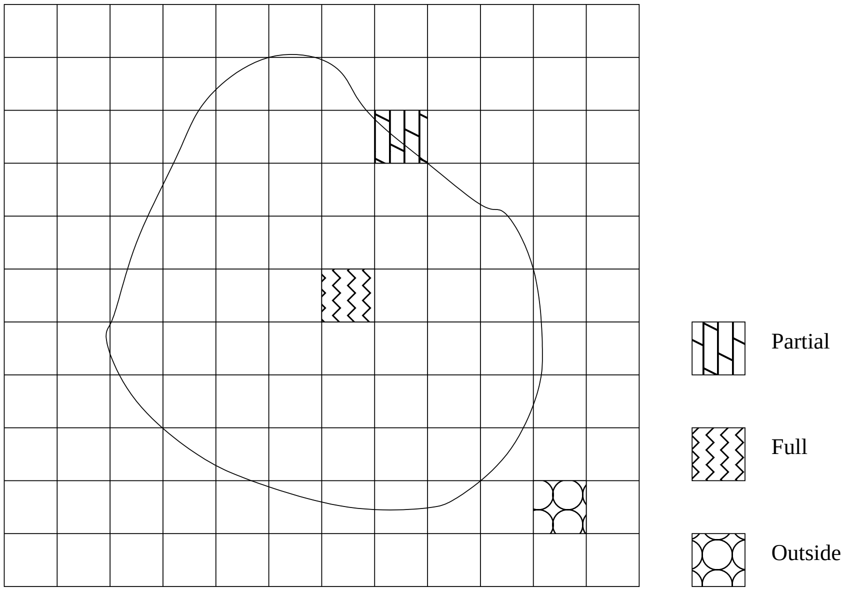

Figure 9: 3 Types of elements: partial, full, void

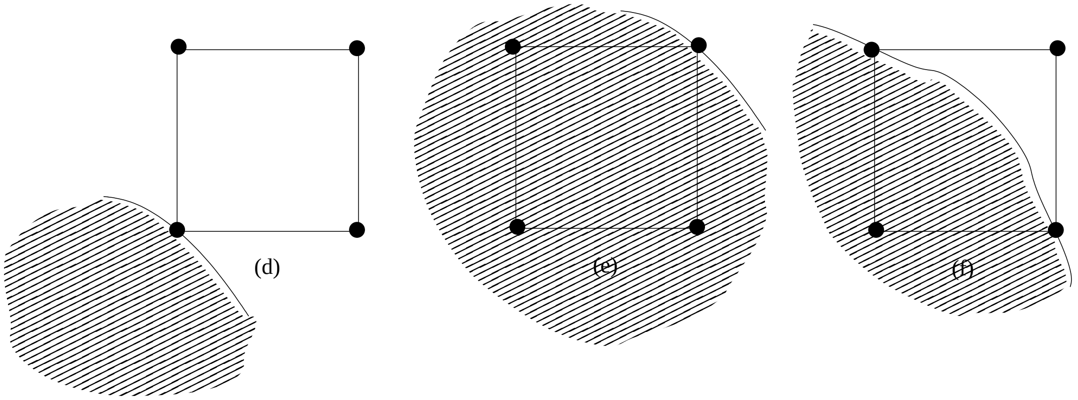

Figure 10: Special cases of elements : (d) a void element, (e) an interior element, (f) a partial element

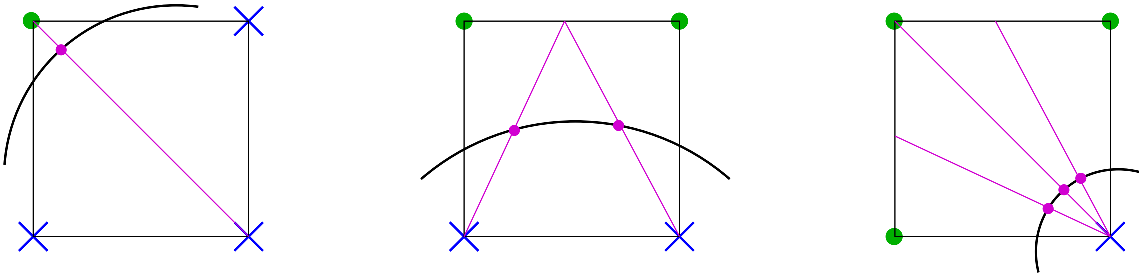

Figure 11: Calculation of intersection points for the cases: (a) one hanging node (b) two hanging nodes (c) three hanging nodes

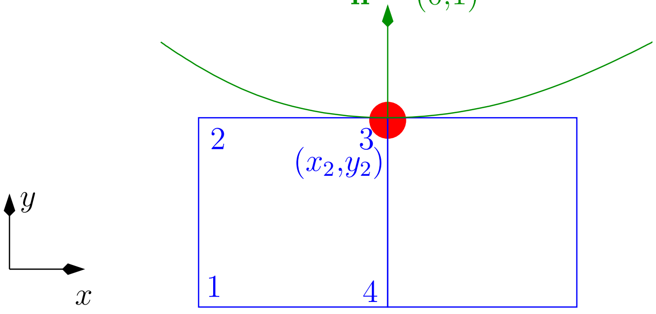

Figure 12: Bad case for interpolating a Neumann boundary condition

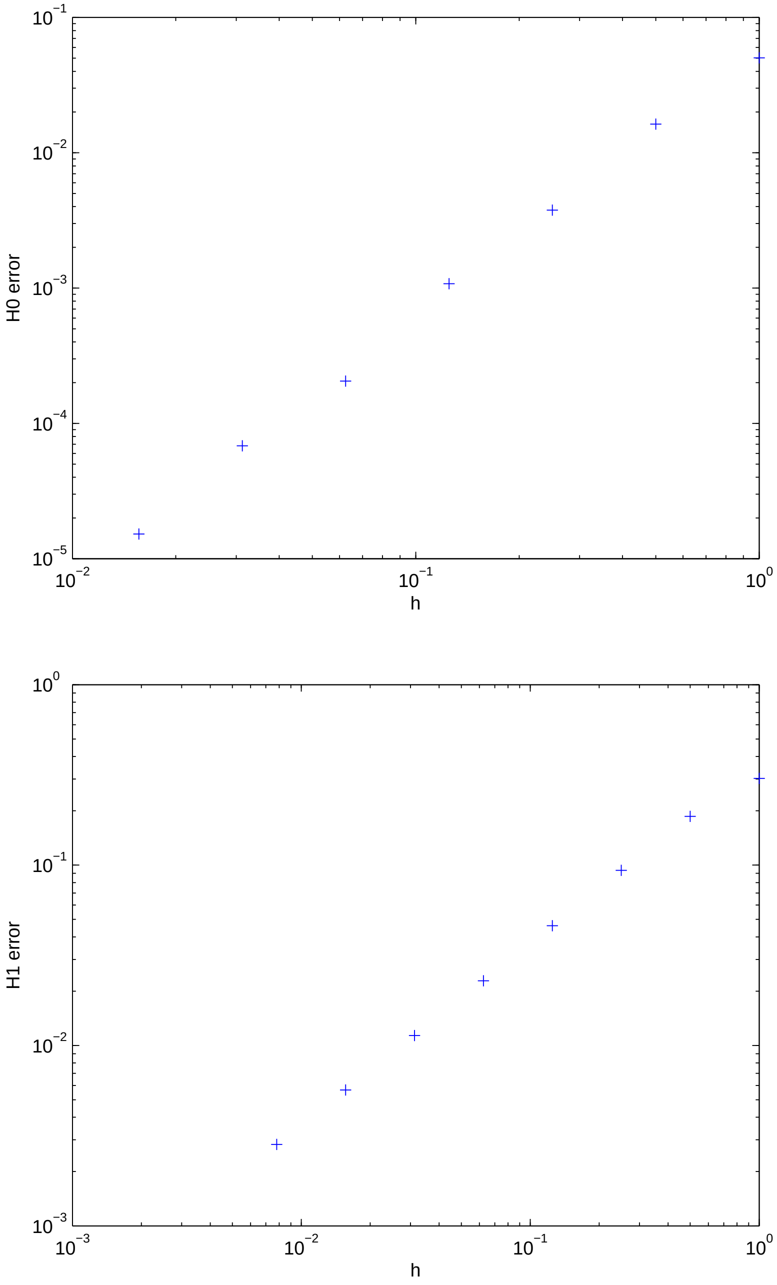

Figure 13: Convergence rate in the H^0- and H^1-error norm of a Dirichlet Problem

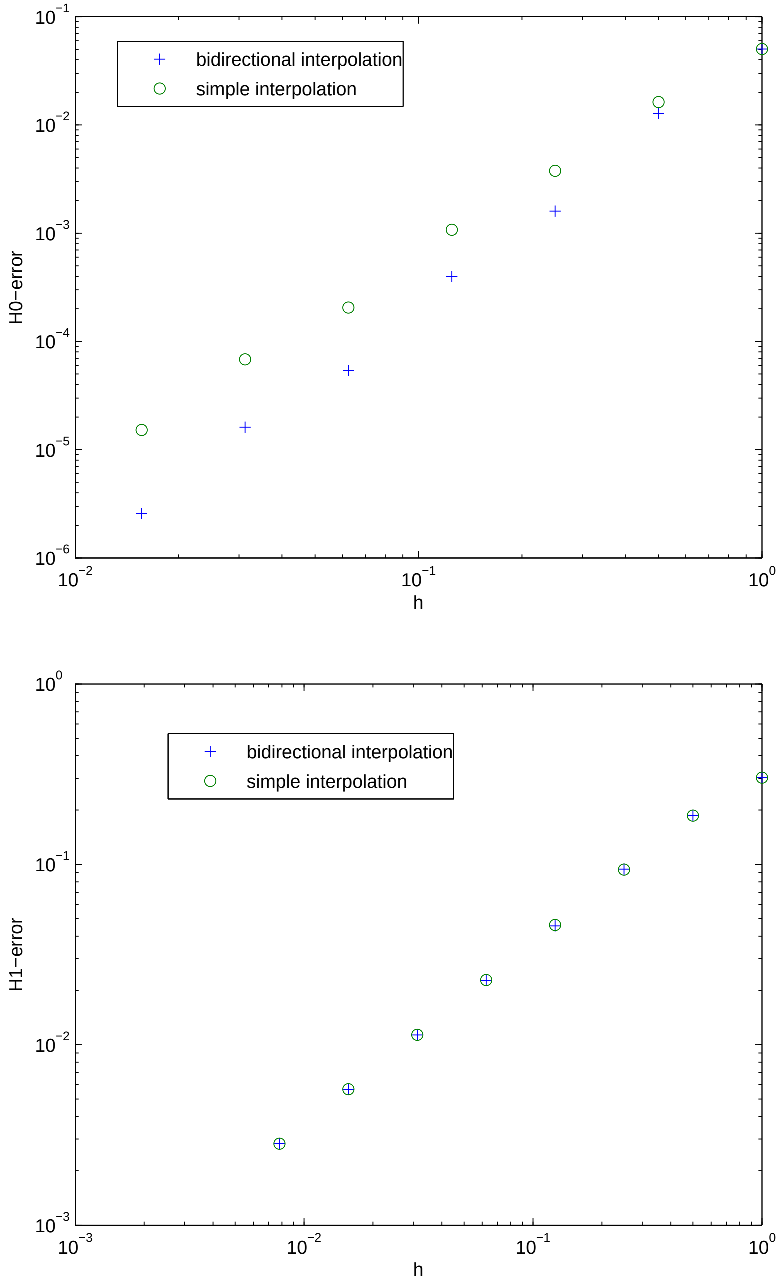

Figure 14: Convergence rate comparison of bi-directional and unidirectional interpolation in the H^0- and H^1-error norm

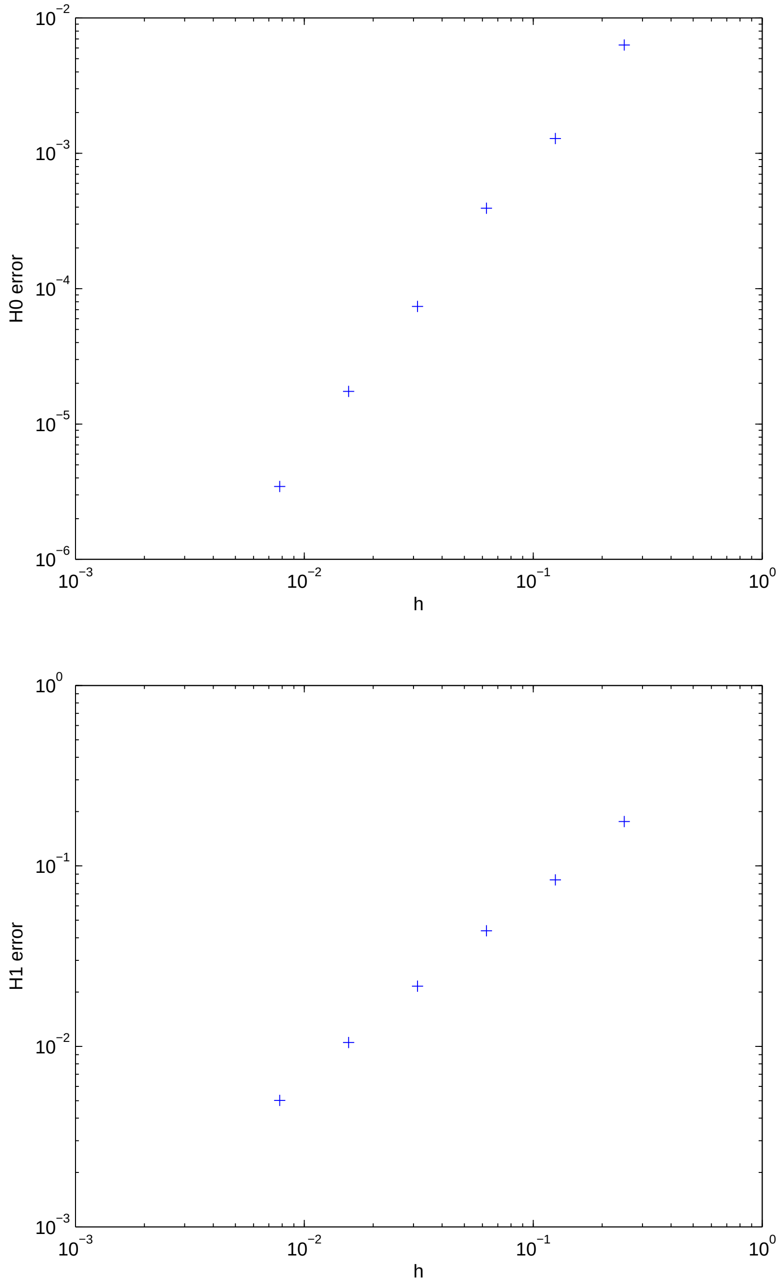

Figure 15: Convergence rate in the H^0- and H^1-error norm of an annular domain with Dirichlet-Dirichlet boundary conditions

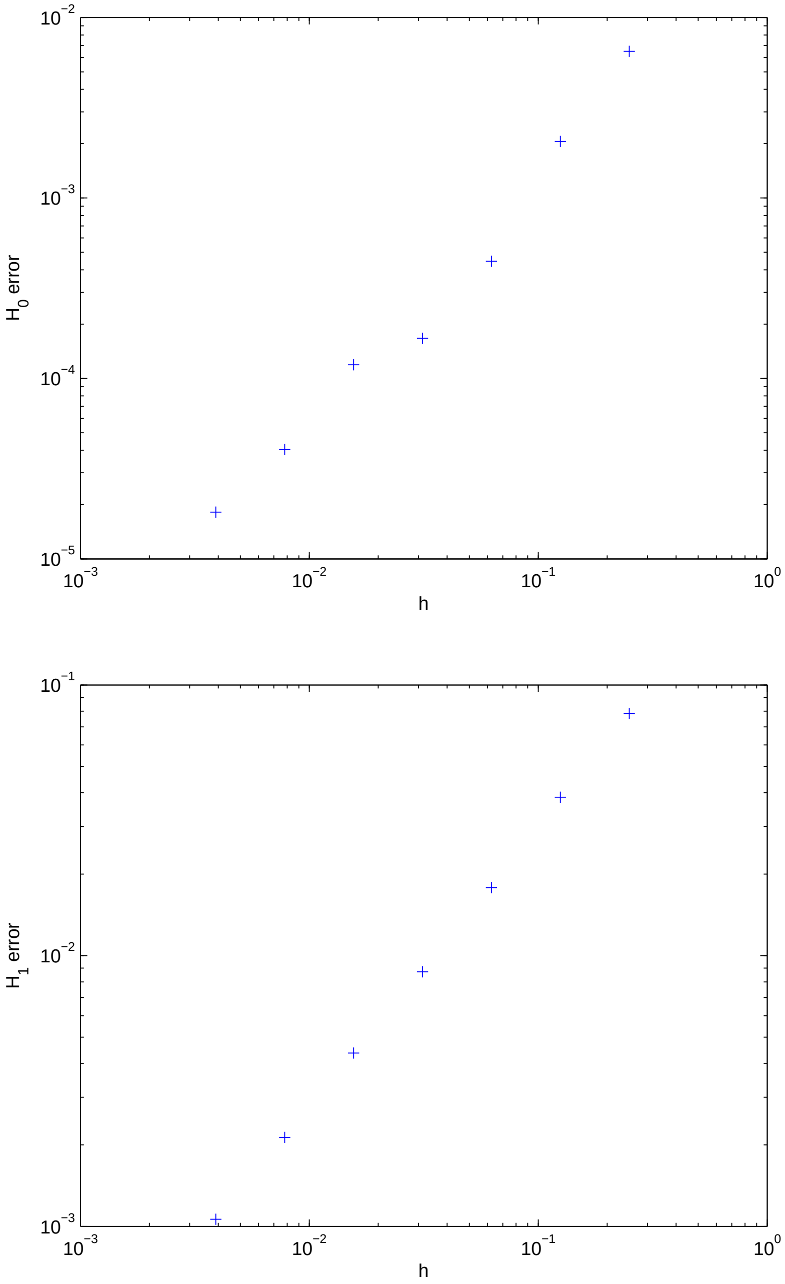

Figure 16: Convergence rate in the H^0- and H^1error norm

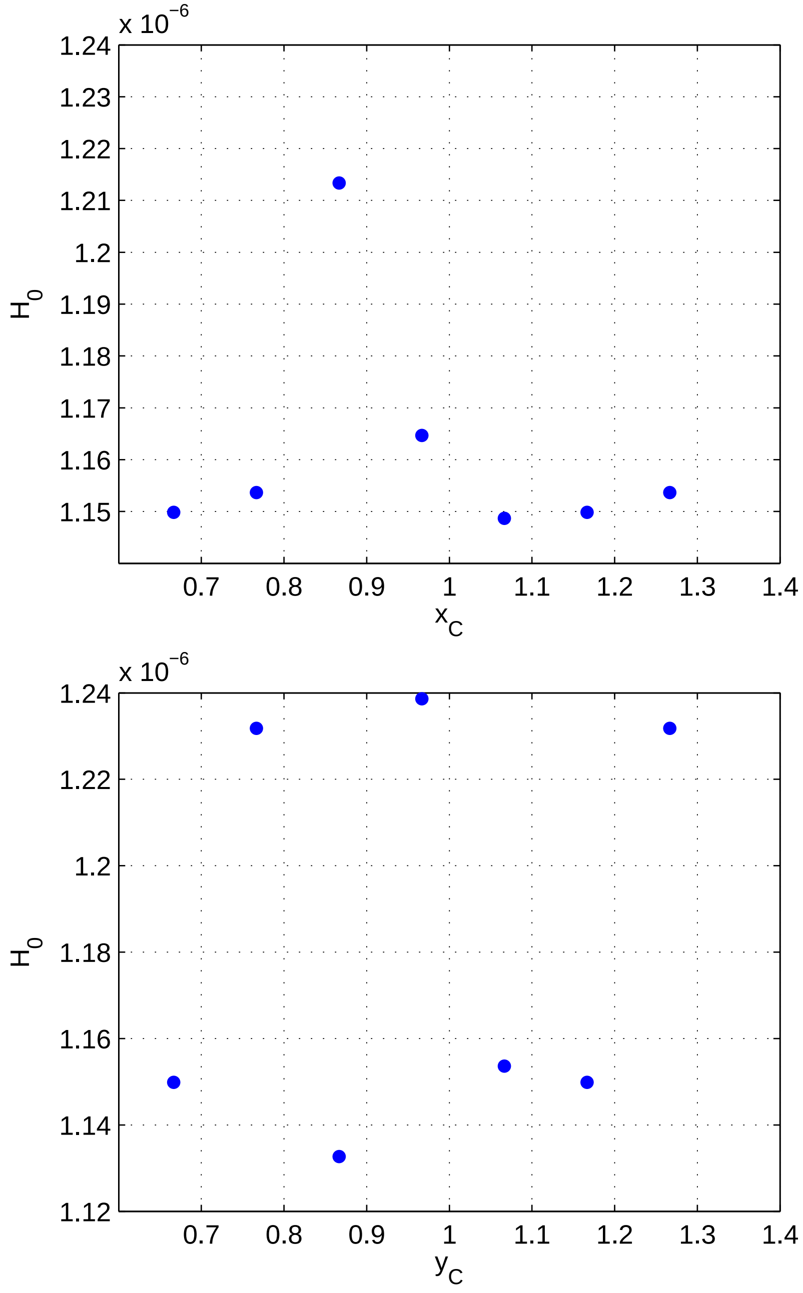

Figure 17: H^0-convergence rate of shifting the Dirichlet circle center along x-direction and y-direction

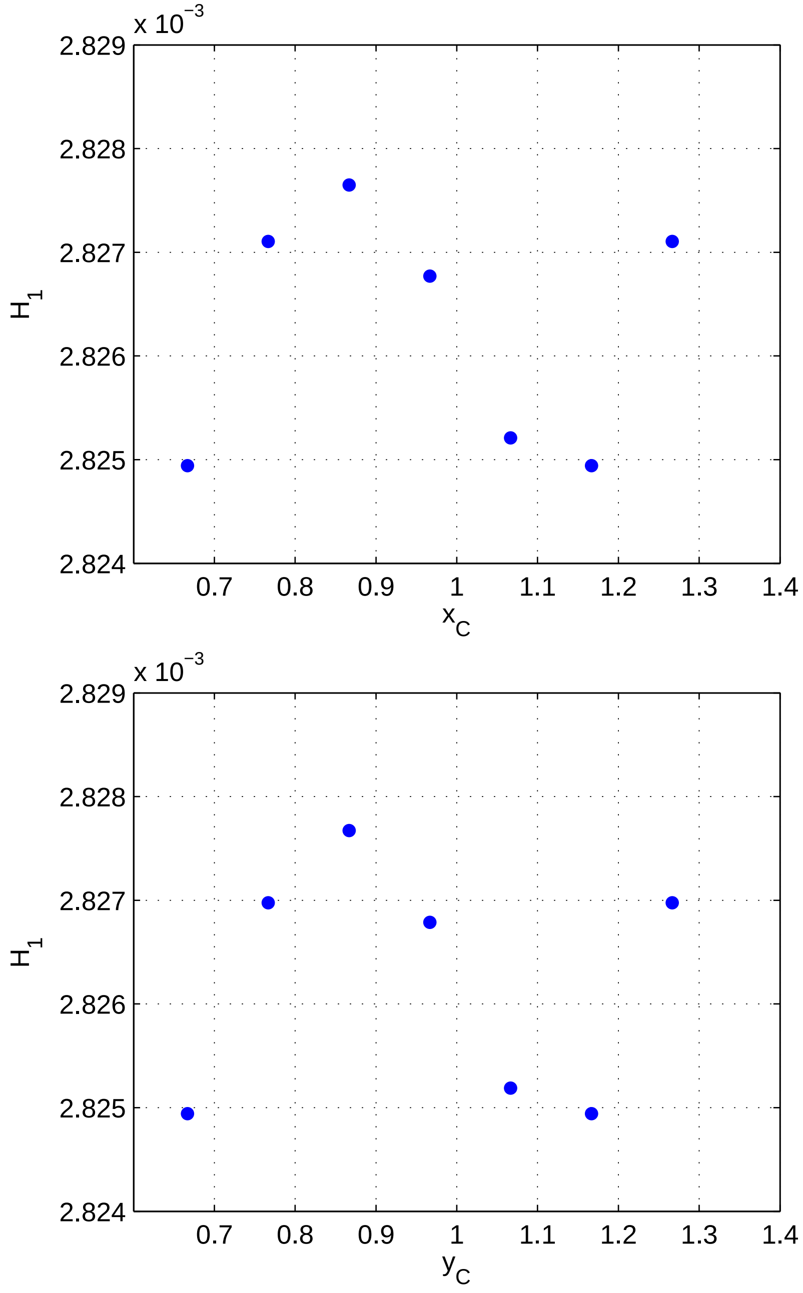

Figure 18: H^1-convergence rate of shifting the Dirichlet circle center along x-direction and y-direction

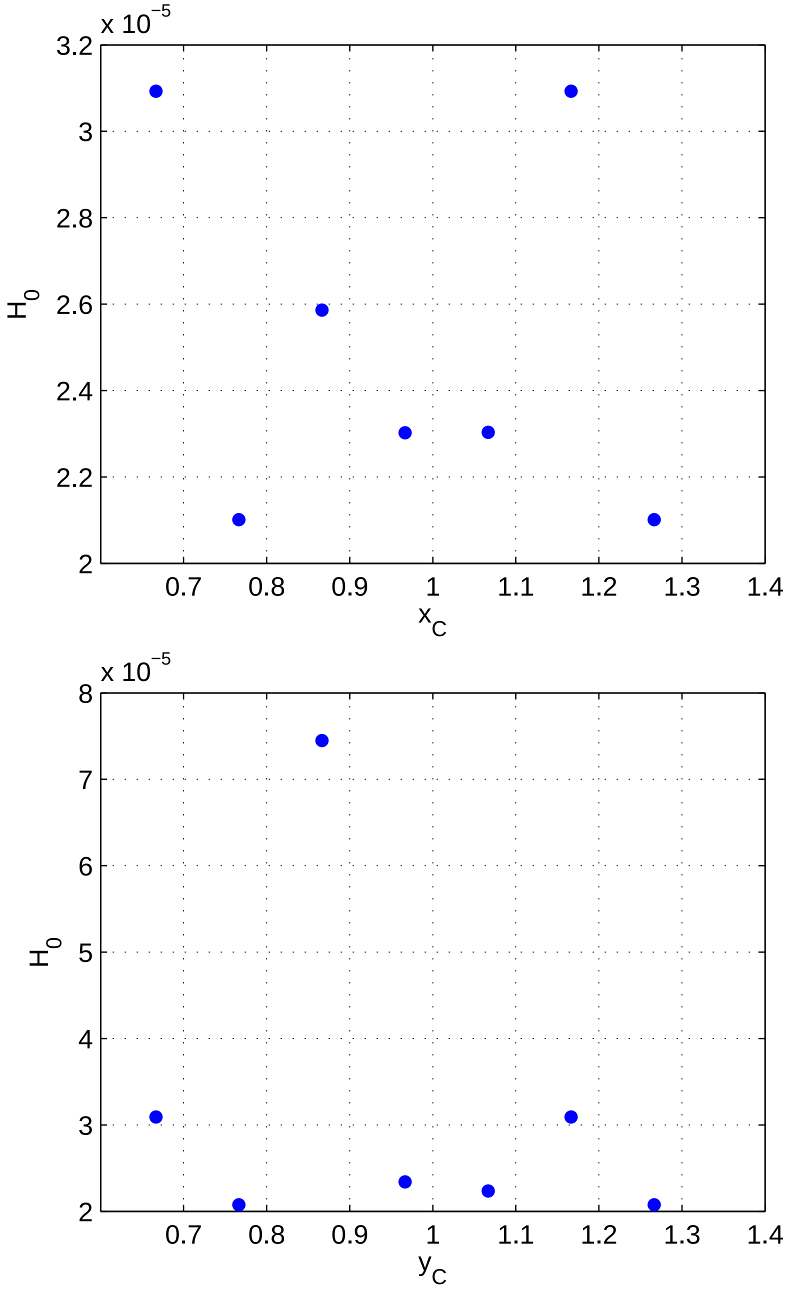

Figure 19: H^0-convergence rate of shifting the Dirichlet annulus center along x-direction and y-direction

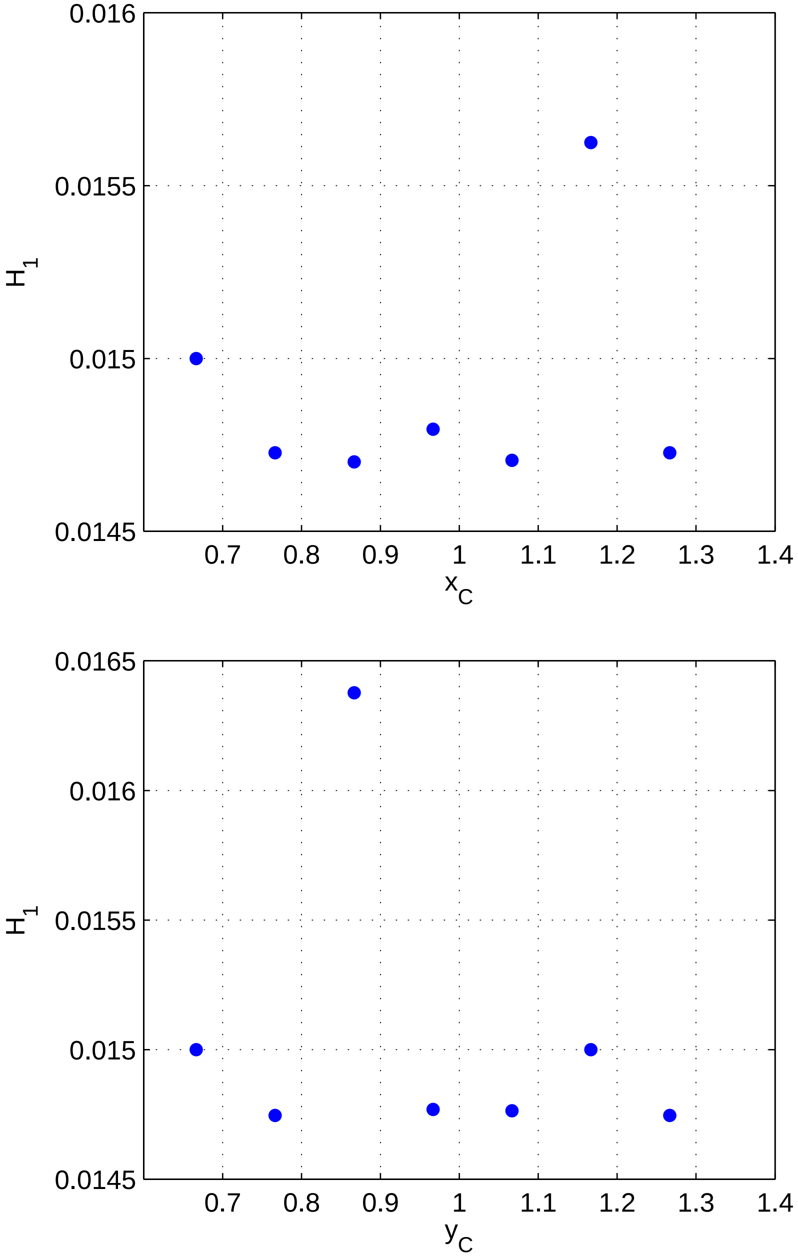

Figure 20: H^1-convergence rate of shifting the Dirichlet annulus center along x-direction and y-direction:

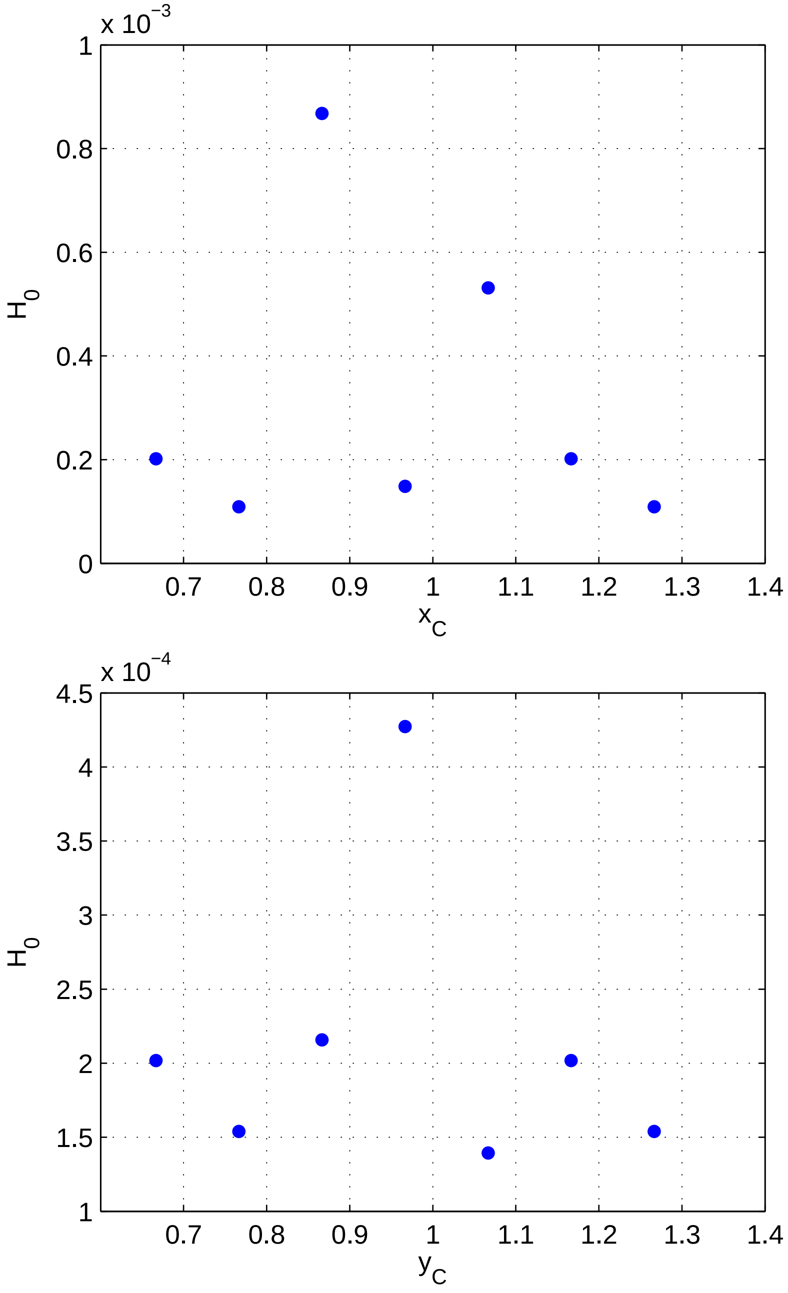

Figure 21: H^0-convergence rate of shifting the Neumann-Dirichlet annulus center along x-direction and y-direction

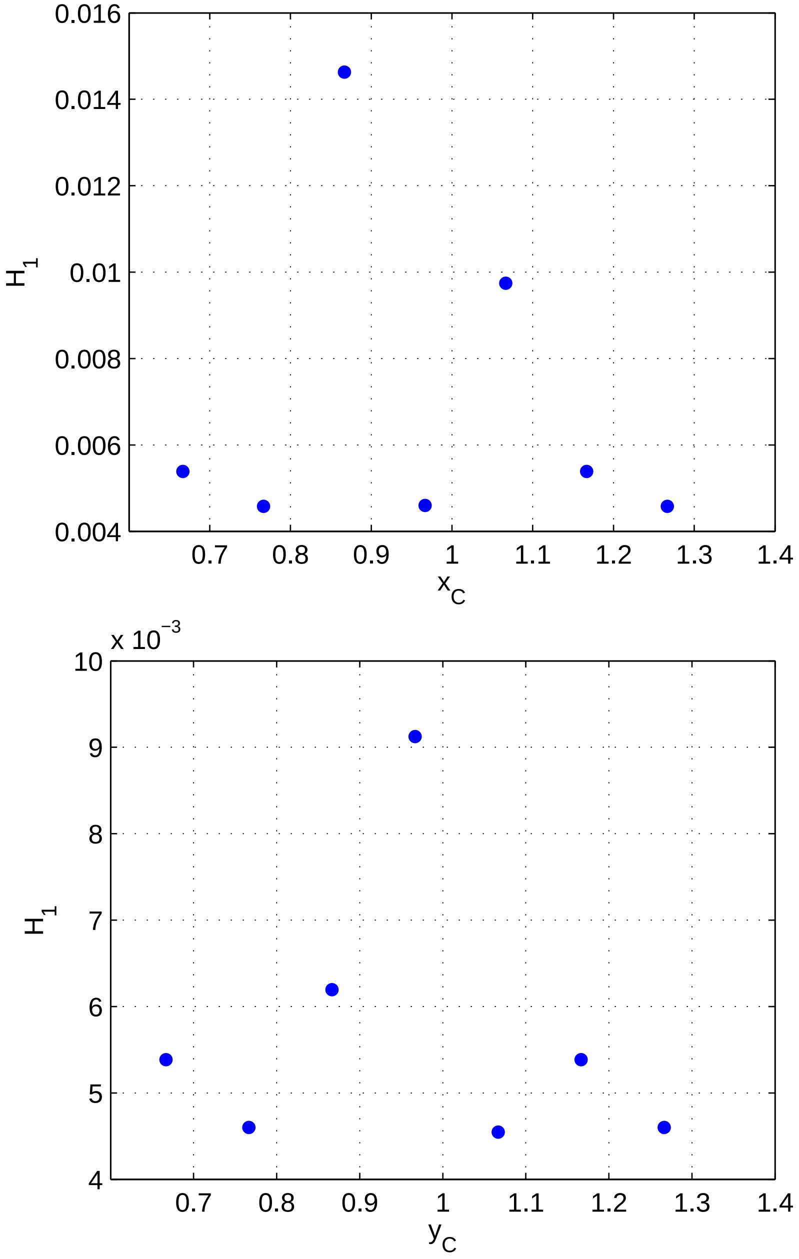

Figure 22: H^1-convergence rate of shifting the Neumann-Dirichlet annulus center along x-direction and y-direction

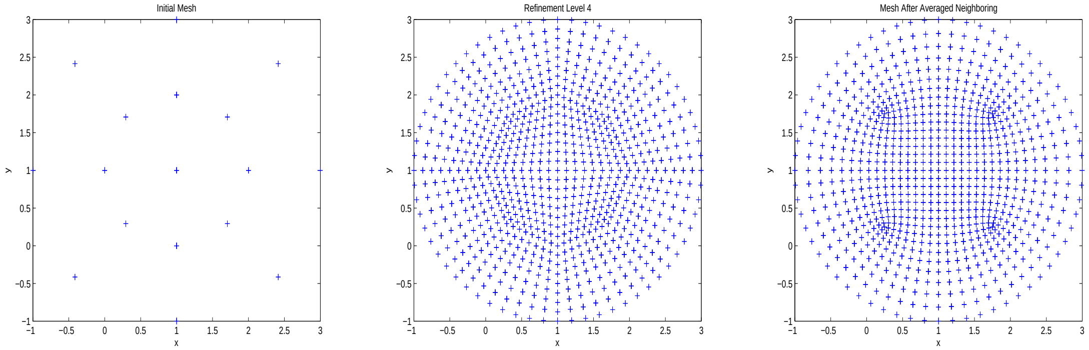

Figure 23: Initial Mesh, finer Mesh, neighbor Averaged Mesh

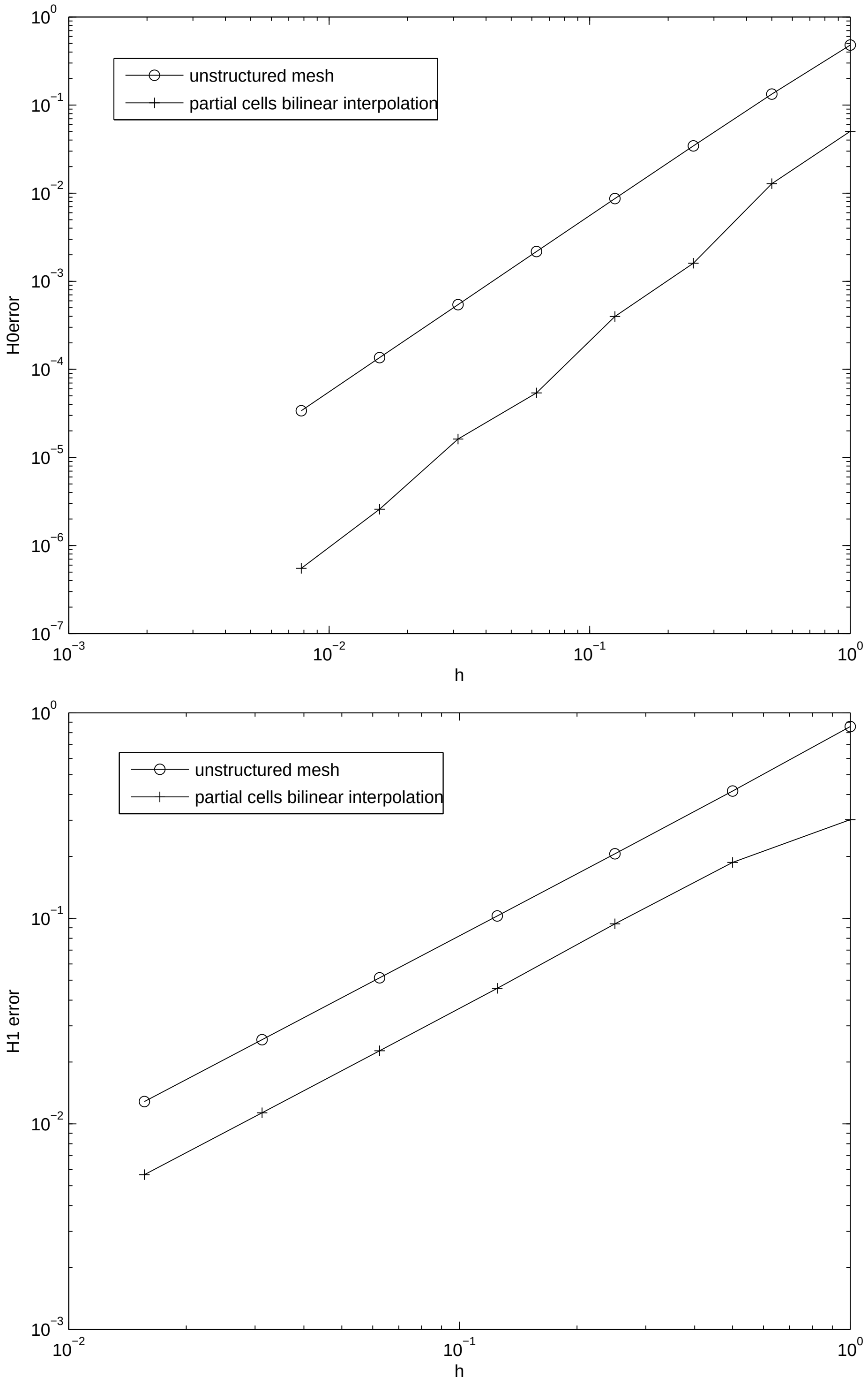

Figure 24: Convergence rate comparison of structured vs. unstructured mesh in the H^0- and H^1-error norm

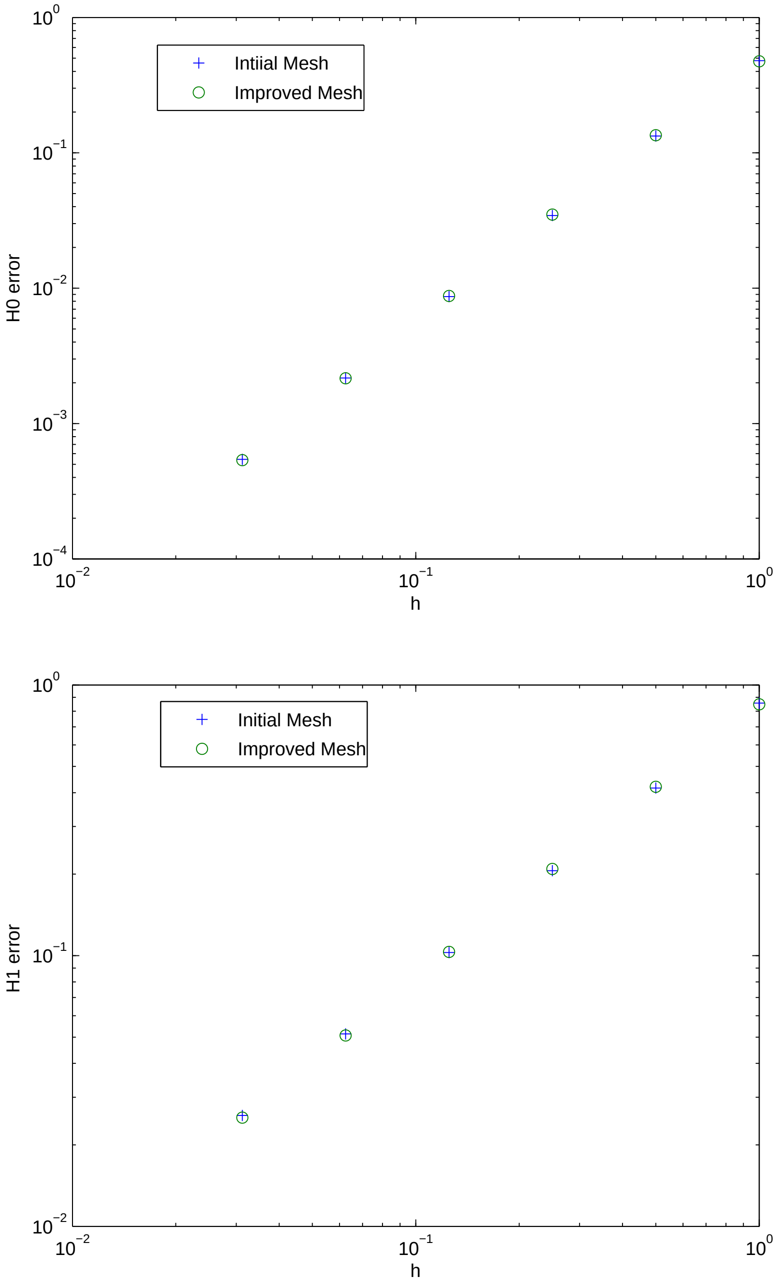

Figure 25: Convergence rate comparison of initial vs. improved mesh in the H^0- and H^1-error norm

C Algorithm

```
begin
    Generate a bounding box surrounding the physical domain;
    Generate a uniform regular grid bounded by the bounding box;
    for elements do
        if full element then
            set flag to 1;
        else if partial element then
            set flag to 2;
        else
            void element, set flag to 3;
        if $flag = 1 \| flag = 2$ then
            Calculate finite element arrays for whole element
        if $flag = 2$ then
            Identify hanging nodes H (and degrees of freedom) of element, and
            intersection points N of element edges with physical boundary;
            Determine nature of boundary condition (Dirichlet or Neumann)
            for physical boundary traversing the element;
            if Neumann boundary condition then
                if $H = 1$ & hanging node is intersection point then
                    set flag to 1;
                    return
            if bi-directional interpolation then
                Create new intersection points NI on physical boundary
                traversing the element and set N=NI;
            if $H > N$ then
                Generate additional interpolation points on the physical
                boundary traversing the element;
            Determine pairing of hanging nodes to intersection/sampling
            points;
    Assemble element arrays;
    for elements do
        if $flag = 2$ then
            for hanging node-to-intersection/sampling points do
                Set all coefficients corresponding to hanging node degrees of
                freedom to zero in global equations;
            for hanging node-to-intersection/sampling points do
                Assemble into global system all equations generated from
                enforcement of boundary conditions for hanging node degrees of
                freedom;
    Solve the algebraic system;
```

Algorithm 1: Algorithm for cut-cells in structured finite elements