AMDiS - Adaptive multidimensional simulations: adaptive finite elements for complex domains

CHRISTINA STÖCKER, SIMON VEY, AXEL VOIGT
Crystal Growth Group
Research Center caesar
Ludwig-Erhard-Allee 2, 53175 Bonn
GERMANY

Abstract: Complicated meshes used in numerical simulations on complex geometries often limit the applicability of multilevel techniques for fast solvers. We propose a way how this limitation can be circumvented if domain adaptation is used. We represent the geometry implicitly by its signed distance function and solve the problem on a larger domain taking account of the complex shape by adaptive refinement and subelement assembling routines. The software concepts of *AMDiS* [1] allow to deal with this situation in a nearly standard way. An example is shown which allows a detailed convergence analysis by comparing the numerical solution with an analytical solution.

Key-Words: mesh generation, distance function, composite finite elements.

1 Introduction

Creating a mesh is always the first step in a wide range of applications concerned with scientific computing and computer graphics. In engineering as well as medical applications usually quite complex three-dimensional geometries need to be meshed. Fully automatic mesh generators for such applications leading to high quality meshes for the numerical problem are still rare. The codes are usually quite complex and nearly inaccessible by the user which suppresses the possibility to combine the mesh generation with the numerical solution and visualization. The advantage of this interaction necessary for multilevel treatments lies in the ability to adaptively describe the complexity of the geometry as well as the solution. We therefore follow a different way which circumvents the meshing of a complex domain. An essential decision in this way is how to represent the geometry. We use a *signed distance function* $d(x, y, z, t)$, which is negative inside the region, to represent our domain. Our numerical grid is now always generated from a regular tetrahedral grid in a larger box, which is adaptively refined according to $d(x, y, z, t)$, see Figure 3 for an example. The signed distance function can either be given analytically, be computed for implicitly given boundaries by equations $f(x, y, z, t) = 0$ or be provided in a discrete form by values on the grid, which is common in *level set* applications [2], where PDEs efficiently model geometries with moving boundaries.

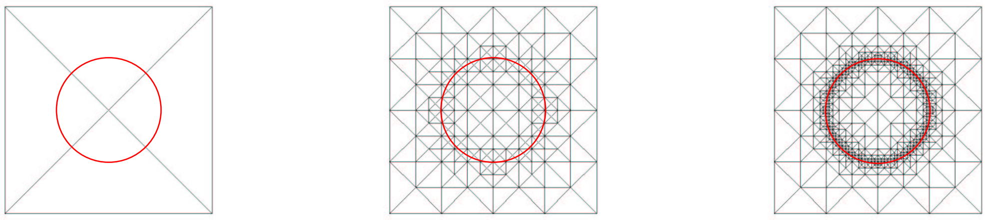
Fig.1: Different levels of refinement of the underlying grid and boundary of the domain represented by $\mathcal{M}_0$.

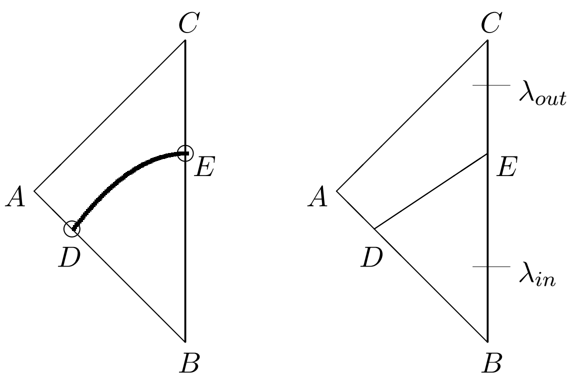

Fig.2: Element T, boundary $\mathcal{M}_0$, and definition of λ.

2 Boundary representation and integration

Let the boundary of the domain at time t be given by the zero level set of the signed distance function $\mathcal{M}_0 = \{(x, y, z) \in \Omega \mid d(x, y, z, t) = 0\}$. This boundary does not coincide with the boundary of the computational domain, which is a box embedding $\mathcal{M}_0$, nor can it be represented by nodes of the underlying grid. Figure 1 shows an illustrative two-dimensional example of a circle represented by $d(x, y, t) = (x^2 + y^2)^{1/2} - 1$ in a computational domain $[-2, 2] \times [-2, 2]$ for different refinements.

The geometry is adaptively resolved with increasing refinements, but at no level can be represented by the nodes of the grid. Instead we are confronted with the situation of $d(x, y, t) = 0$ within an element T, as pointed out in Figure 2. If parameters vary across $\mathcal{M}_0$ integrals of the form $\int_T \lambda\phi$, with λ a discontinuous function and ϕ a smooth function, have to be evaluated. The method used is similar as in [3] and is explained in Figure 2:

$$\int_T \lambda\,\phi \approx \int_{\triangle(DBE)} \lambda_{in}\phi + \int_{\square(ADEC)} \lambda_{out}\phi$$

$$= \int_{\triangle(DBE)} \lambda_{in}\phi + \int_T \lambda_{out}\phi - \int_{\triangle(DBE)} \lambda_{out}\phi.$$

Note that this formula avoids the explicit integration over quadrilaterals and requires only integration over triangles, and can be thus performed in a nearly standard way.

If in addition boundary conditions are specified at the domain wall given by $\mathcal{M}_0$, a penalty method is applied in order to fulfill them. We approximate the introduced line integral along $\mathcal{M}_0$ within an element T by an integration along the straight line DE, see Figure 2. This involves nothing else as a standard integration of an element in one dimension and

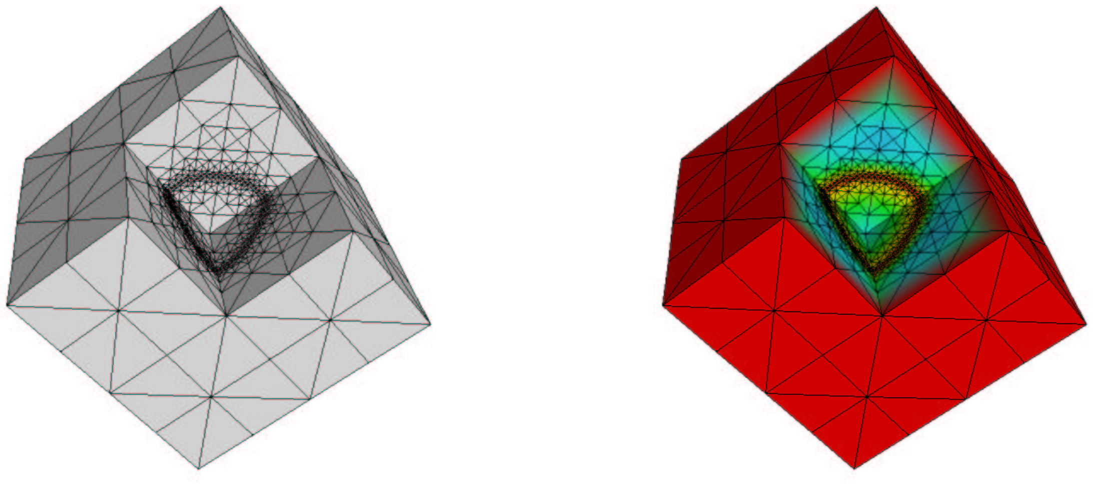

Fig.3: Adaptively refined grid and numerical solution in Ω'.

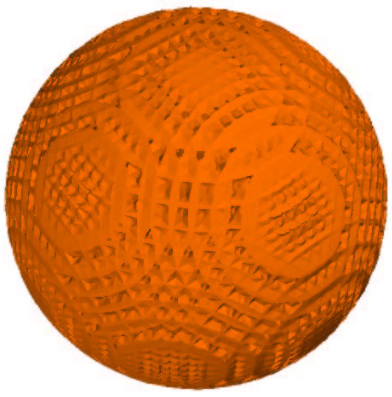

Fig.4: Isosurface $u_h = 0$ representing the Dirichlet boundary conditions.

therefore leads to no further complications. The same approach can be carried over to three dimensions.

These concepts are realized in our finite element toolbox *AMDiS* [1]. This software is primarily developed to solve realistic computations in materials science but can be used for other applications as well. The design is based on a natural hierarchy of locally refined meshes and an abstract concept of general finite element spaces which is combined with an object oriented data structure. In this way dimension independent programming is possible and complex applications can be implemented on an abstract level, keeping the numerical issues away from the user. The basic concepts are described in [4].

A more rigorous approach to circumvent resolving the boundary of complex geometries by the mesh can be found in [5], where the notion *composite finite elements* has been introduced.

3 Test example

As a first test example we consider the Laplace equation with diffusion coefficient D and homogenous Dirichlet boundary conditions

$$\begin{aligned} -\nabla \cdot D\nabla u &= 1 \text{ on } \Omega \\ u_{|\partial\Omega} &= 0 \end{aligned}$$

with Ω the unit sphere $B(0;1)$. The domain is embedded into Ω' representing a box, which can easily be meshed. As the solution on $\Omega'\backslash\Omega$ should not affect the solution on Ω we use homogenous Dirichlet boundary conditions on $\partial\Omega'$. Thus we have the problem

$$\begin{aligned} -\nabla \cdot D\nabla u &= 1 \text{ on } \Omega' \\ u_{|\partial\Omega} &= 0 \\ u_{|\partial\Omega'} &= 0. \end{aligned}$$

To realize the boundary condition $u_{|\partial\Omega} = 0$ a penalty method is used. With the penalty term the finite element formulation reads

$$\int_{\Omega'} D\nabla u_h \nabla\phi \, dx + \frac{1}{\epsilon(h)} \int_{\partial\Omega} u_h \phi \, d\sigma = \int_{\Omega'} \phi \, dx$$

where the test functions ϕ are in $V_h \subseteq H_0^1(\Omega')$ and the penalty coefficient $1/\epsilon(h)$ depends sensitively on the size h of the mesh elements (see [6]). The solution is shown in Figure 3 and the isosurface $u_h = 0$ representing the Dirichlet boundary conditions at the approximated domain of $\mathcal{M}_0$ is shown in Figure 4.

In [6] the following error estimation is given for the penalty method

$$\left| u - u_h \right|_{1,\Omega} \leq C\left(\frac{h^p}{\sqrt{\varepsilon(h)}} + \sqrt{\varepsilon(h)} \right) \left\| u \right\|_{p+1,\Omega}, \quad (1)$$

where $|.|_{1,\Omega}$ is the H^1-seminorm, p is the order of the finite element space and C a generic constant not depending on h. Therefore, we choose $\epsilon(h) \in O(h)$ to obtain optimal convergence. The error in the H^1-seminorm $E(h)$ then behaves like $O(\sqrt{h^p})$. We use global refinement to compare our solution with this estimate. During global refinement the volume of each element is bisected in one refinement step. Thus in dimension 3 the length of the element edges divides

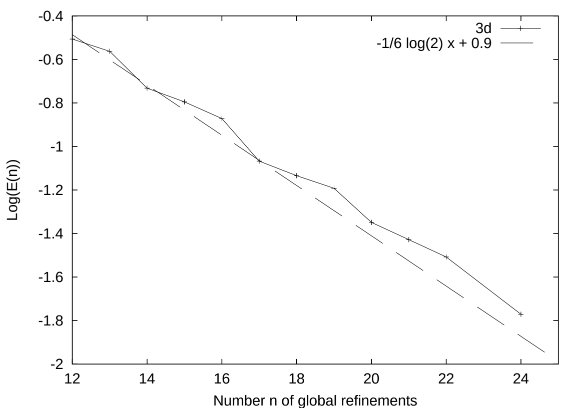

Fig.5: Convergence results showing the error of the numerical solution if compared with the analytical solution in Ω and the theoretically expected rate of convergence.

by two in 3 refinement steps. That means if h_0 is the initial size of an element edge, n the number of global refinements and h_n the size of an element edge after n global refinements, we have $h_n = \frac{h_0}{2^{n/3}}$. Thus the error should behave like

$$E(h_n) \approx \left(\frac{h_0}{2^{n/3}}\right)^{1/2}.$$

If we plot the logarithm of the error $E(h_n)$ against the number of global refinements n, we obtain a straight line with gradient $-\frac{1}{6}log(2)$. Figure 5 shows how the numerical results obey this linear behavior if linear finite elements are used.

4 Conclusion and Outlook

Even if the problem considered in Section 3 is rather simple, it should be obvious that the described methodology is neither restricted to ball shaped domain nor to the Laplace equation. Coupled systems of PDEs with moving boundaries on complex domains can be solved as well. The only ingredient needed is the *signed distance function* which describes the domain. As a final goal the simulation will be interactively visualized in virtual environments such as the *CAVE* and *Responsive Workbench*. For that purpose our finite element toolbox *AMDiS* [1] is combined with the *Julius* [7] framework for medical visualization.

References:

[1] http://www.caesar.de/cg

[2] S. Osher and J. Sethian, Fronts propagating with curvature-dependent speed: Algorithms based on Hamilton-Jacobi formulation, *J. Comput.. Phys.*, Vol.79, 1988, pp. 12–49.

[3] E. Bänsch, F. Hauer, O. Lakkis, B. Li and A. Voigt, Finite Element Method for Epitaxial Growth with Attachment-Detachment Kinetics, *J. Comput. Phys.*, Vol.194, 2004, pp. 409–434.

[4] S. Vey and A. Voigt, AMDiS - Adaptive multidimensional simulation : object oriented software concepts for scientific computing, *WSEAS Transactions on Systems*, Vol.3, 2004, pp. 1564–1569.

[5] W. Hackbusch and S. Sauter, Composite finite elements for the approximation of PDEs on domains with complicated micro-structures, *Numer. Math.*, Vol.75, 1997, pp. 447–472.

[6] P. Ciarlet, *The finite element method for elliptic problems*, North-Holland, 1978

[7] http://www.caesar.de/ssl