Numerical Aspects of Parabolic Free Boundary Problems

—

Adaptive Finite Element Methods

A. Schmidt and K.G. Siebert

Institut für Angewandte Mathematik
Universität Freiburg

Lecture notes — Course MA 3

Summer School, Jyväskylä, 1996

Abstract

In this course we deal with the numerical approximation of various parabolic free boundary problems: motion of interfaces under curvature, phase change problems, phase transition problems. We introduce finite element methods and discuss their convergence properties as well as their efficient implementation. Numerical aspects and implementation details of adaptive finite element methods in two and three dimensions will be presented, as well as grid adaptation techniques based on error indicators and estimators.

Contents

1 An introduction to a posteriori error estimation for elliptic problems

Although the subject of this course are *parabolic* free boundary problems, we give an introduction about adaptive finite element techniques for *elliptic* problems. The reason is, that most of the principles are much clearer and easier to describe and understand in the context of elliptic problems, while they are applicable to parabolic problems in a similar way.

We consider the model problem: Find a solution u of

$$\begin{aligned} -\Delta u &= f && \text{in } \Omega, \\ u &= 0 && \text{on } \partial\Omega \end{aligned} \tag{1.1}$$

where Ω is a bounded domain in $\mathbf{R}^d$ with a polyhedral boundary and $f \in L_2(\Omega)$ is some given right hand side.
Let $H^1(\Omega)$ be the Sobolev space of all functions with weak derivatives of first order and let $\mathring{H}^1(\Omega)$ be the subspace of all those functions in $H^1(\Omega)$ that vanish on the boundary of Ω. Then the weak formulation of (1.1) is stated as:

$$u \in \mathring{H}^1(\Omega): \qquad \int_\Omega \nabla u \nabla \varphi \;=\; \int_\Omega f\,\varphi \qquad \forall \varphi \in \mathring{H}^1(\Omega). \tag{1.2}$$

Now, let $V_h \subset \mathring{H}^1(\Omega)$ be a finite dimensional subspace. Then we have the discrete problem:

$$u_h \in V_h: \qquad \int_\Omega \nabla u_h \nabla \varphi_h \;=\; \int_\Omega f\,\varphi_h \qquad \forall \varphi_h \in V_h. \tag{1.3}$$

1.1 A posteriori error estimation in the energy norm

If V_h is for example the finite element space consisting of piecewise polynomials of degree $p \geq 1$ on a given triangulation $\mathcal{T}$ (with zero boundary values) we have the following *a priori* estimate

$$|u-u_h|_{H^1(\Omega)} \;:=\; \left(\int_\Omega |\nabla(u-u_h)|^2\right)^{1/2} \;\leq\; c\left(\sum_{T\in\mathcal{T}} h_T^{2p}|u|_{H^{p+1}(\Omega)}\right)^{1/2} \tag{1.4}$$

where h_T is the diameter of a simplex T (see [13] e.g.).
The aim of *a posteriori* error estimation is to establish an estimation of the form

$$|u-u_h|_{H^1(\Omega)} \;\leq\; c\left(\sum_{T\in\mathcal{T}} \eta_T(u_h,f)^2\right)^{1/2} \tag{1.5}$$

where the value of η_T does only depend on the discrete solution u_h on a simplex T and its adjacent neighbours and given data f on the simplex T. Thus, η_T is a computable value and we can control the adaptive procedure by these values (see Section 3).
The most important tool for such an a posteriori error estimation is an interpolation estimate for functions $v \in \mathring{H}^1(\Omega)$. Since for $d \geq 2$ we do not have the embedding of $H^1(\Omega)$ into $C^0(\bar{\Omega})$, we can not make use of the usual Lagrange interpolant. But we can use the Clément interpolant which avoids the pointwise evaluation of an H^1 function [14].

It is defined in the following way: Let V_h contain the space of piecewise linear functions on the underlying triangulation. Then for $v \in H^1(\Omega)$ we can construct the piecewise linear interpolant in the following way: For each node a of the triangulation let v_a be the L_2 projection of v to the space of all piecewise linear function on all simplices surrounding the node a and define for the interpolant $R_h v$ the nodal value at a by $(R_h v)(a) := v_a(a)$. Defining the nodal values of $R_h v$ for all nodes of the triangulation by this procedure, gives a unique piecewise linear function on the whole triangulation. Setting $(R_h v)(a) := 0$ for all nodes a that belong to the boundary of Ω defines an interpolant $R_h : \mathring{H}^1(\Omega) \to V_h$ (i.e. the interpolant has zero boundary values also) and we get the following estimates:

$$\|v - R_h v\|_{L_2(T)} \leq c h_T |v|_{H^1(M_T)}, \tag{1.6}$$

$$|R_h v|_{H^1(T)} \leq c |v|_{H^1(M_T)} \tag{1.7}$$

where $M(T)$ is the patch of all simplices T' that have a non empty intersection with T.
Now we will derive the a posteriori estimate: Let $H^{-1}(\Omega)$ be the dual space of $\mathring{H}^1(\Omega)$, i.e. $H^{-1}(\Omega) = (\mathring{H}^1(\Omega))^*$. For $f \in L_2(\Omega)$ define $F \in H^{-1}(\Omega)$ by

$$\langle F, \varphi \rangle_{H^{-1}(\Omega) \times \mathring{H}^1(\Omega)} := \int_\Omega f \varphi \qquad \text{for all } \varphi \in \mathring{H}^1(\Omega)$$

where $\langle ., . \rangle_{H^{-1}(\Omega) \times \mathring{H}^1(\Omega)}$ is the dual pairing on $H^{-1}(\Omega) \times \mathring{H}^1(\Omega)$.
We can look at $-\Delta$ as an operator

$$-\Delta : \mathring{H}^1(\Omega) \to H^{-1}(\Omega)$$

by defining $-\Delta v \in H^{-1}(\Omega)$ for a function $v \in \mathring{H}^1(\Omega)$ in the following way:

$$\langle -\Delta v, \varphi \rangle_{H^{-1}(\Omega) \times \mathring{H}^1(\Omega)} := \int_\Omega \nabla v \nabla \varphi \qquad \text{for all } \varphi \in \mathring{H}^1(\Omega). \tag{1.8}$$

It is clear that $-\Delta$ is a linear continuous operator. Moreover $-\Delta$ is invertible since (1.2) is uniquely solvable for a given right hand side $F \in H^{-1}(\Omega)$ and it is an isometric isomorphism, i.e.

$$\| -\Delta v \|_{H^{-1}(\Omega)} = |v|_{H^1(\Omega)} \tag{1.9}$$

because

$$\sup_{\varphi \in \mathring{H}^1(\Omega) \setminus \{0\}} \frac{\langle -\Delta v, \varphi \rangle_{H^{-1}(\Omega) \times \mathring{H}^1(\Omega)}}{|\varphi|_{H^1(\Omega)}} = \sup_{\varphi \in \mathring{H}^1(\Omega) \setminus \{0\}} \frac{\int_\Omega \nabla v \nabla \varphi}{|\varphi|_{H^1(\Omega)}}$$

$$\begin{cases} \leq & |v|_{H^1(\Omega)} \qquad \text{by Cauchy's inequality,} \\ \geq & |v|_{H^1(\Omega)} \qquad \text{taking } \varphi = v. \end{cases}$$

Remark: We can use such an abstract framework in more general situations also: Let V be an Hilbert space and $a(.,.) : V \times V \to \mathbf{R}$ a continuous V–coercive bilinear form, i.e.

$$a(v, \varphi) \leq c^* \|v\|_V \|\varphi\|_V \quad \text{and} \quad c_* \|v\|_V^2 \leq a(v, v) \quad \forall v, \varphi \in V,$$

Defining $A : V \to V^*$ by

$$\langle Av, \varphi \rangle_{V^* \times V} := a(v, \varphi) \qquad \forall v, \varphi \in V,$$

we conclude

$$c_* \|v\|_V \;\leq\; \|Av\|_{V^*} \;\leq\; c^* \|v\|_V,$$

and the following analysis will also carry over to this situation.

Returning back to our model problem we rewrite (1.2) as:

$$u \in \mathring{H}^1(\Omega): \qquad -\Delta u \;=\; F \qquad \text{in } H^{-1}(\Omega).$$

By this equation and by (1.9) we have for the error $e := u - u_h$

$$|e|_{H^1(\Omega)} \;=\; |u-u_h|_{H^1(\Omega)} \;=\; \|-\Delta(u-u_h)\|_{H^{-1}(\Omega)} \;=\; \|F + \Delta u_h\|_{H^{-1}(\Omega)}$$

Thus, we have an expression for the error in terms of u_h and data f. The problem is that we can not evaluate this expression because the norm on $H^{-1}(\Omega)$ involves the evaluation of a supremum over all $\varphi \in \mathring{H}^1(\Omega)\backslash\{0\}$. As a consequence we have to estimate this supremum. For that we need the orthogonality of the error, i.e.

$$\begin{aligned} 0 \;&=\; \int_\Omega \nabla(u-u_h)\nabla\varphi_h \;=\; \langle -\Delta(u-u_h),\, \varphi_h\rangle_{H^{-1}(\Omega)\times\mathring{H}^1(\Omega)} \\ &=\; \langle F + \Delta u_h,\, \varphi_h\rangle_{H^{-1}(\Omega)\times\mathring{H}^1(\Omega)} \end{aligned}$$

for all $\varphi_h \in V_h$. Now, denote by $[\partial_\nu u_h]$ the jumps of the normal derivatives of the discrete solution u_h across a $(d-1)$–simplex. We obtain by the orthogonality of the error, integration by parts, a scaled trace theorem, and the interpolation estimate (1.6)

$$\begin{aligned} |e|_{H^1(\Omega)} \;&=\; \|F + \Delta u_h\|_{H^{-1}(\Omega)} \\ &=\; \sup_{\substack{\varphi\in\mathring{H}^1(\Omega)\\ |\varphi|_{H^1(\Omega)}=1}} \langle F + \Delta u_h,\, \varphi\rangle_{H^{-1}(\Omega)\times\mathring{H}^1(\Omega)} \\ &=\; \sup_{\substack{\varphi\in\mathring{H}^1(\Omega)\\ |\varphi|_{H^1(\Omega)}=1}} \langle F + \Delta u_h,\, \varphi - R_h\varphi\rangle_{H^{-1}(\Omega)\times\mathring{H}^1(\Omega)} \\ &=\; \sup_{\substack{\varphi\in\mathring{H}^1(\Omega)\\ |\varphi|_{H^1(\Omega)}=1}} \sum_{T\in\mathcal{T}} \int_T f\,(\varphi - R_h\varphi) - \int_T \nabla u_h \nabla(\varphi - R_h\varphi) \\ &=\; \sup_{\substack{\varphi\in\mathring{H}^1(\Omega)\\ |\varphi|_{H^1(\Omega)}=1}} \sum_{T\in\mathcal{T}} \int_T (f + \Delta u_h)\,(\varphi - R_h\varphi) - \frac{1}{2}\int_{\partial T\backslash\partial\Omega} [\partial_\nu u_h](\varphi - R_h\varphi) \\ &\leq\; c \sup_{\substack{\varphi\in\mathring{H}^1(\Omega)\\ |\varphi|_{H^1(\Omega)}=1}} \sum_{T\in\mathcal{T}} \Big(h_T \|f + \Delta u_h\|_{L_2(T)} + \frac{1}{2} h_T^{1/2} \|[\partial_\nu u_h]\|_{L_2(\partial T\backslash\partial\Omega)} \Big) |\varphi|_{H^1(M_T)} \\ &\leq\; c\Big(\sum_{T\in\mathcal{T}} \underbrace{h_T^2 \|f + \Delta u_h\|^2_{L_2(T)} + \frac{1}{2} h_T \|[\partial_\nu u_h]\|^2_{L_2(\partial T\backslash\partial\Omega)}}_{=:\eta_T(u_h,f)^2} \Big)^{1/2} \end{aligned}$$

where we used the fact that the overlap of different patches M_T is bounded by a constant. This establishes the a posteriori error estimate (1.5).

The above estimate makes sure that the error estimator $\eta := \Big(\sum_{T\in\mathcal{T}} \eta_T(u_h,f)^2\Big)^{1/2}$ is *reliable*. But we also have to answer the question whether the estimator is *efficient* also, i.e. can we estimate the estimator by the error itself. This is very important especially for higher order elements, because we only used the approximation property of the piecewise linear functions. Let f_h be an approximation of the right hand side f belonging to some finite dimensional space (for example the piecewise L_2 projection on each element, or some other interpolant of the right hand side). Then we can prove

$$\eta_T(u_h,f_h) \;\le\; c\Big(|u-u_h|_{H^1(M(T))} + h_T\|f-f_h\|_{L_2(M(T))}\Big) \tag{1.10}$$

where $M(T)$ now denotes the patch of all those simplices T' sharing a complete $(d-1)$–simplex with T. The last term $h_T\|f-f_h\|_{L_2(M(T))}$ is of higher order if f is smooth. This term reflects that we first have to approximate given data sufficiently, i.e. $\|f-f_h\|_{L_2(M(T))}$ is small, and then we get an efficient error estimator which we can not expect for a poor approximation of given data. The proof of this estimate is very technical (one has to construct suitable cut–off functions to localize the element residual $f+\Delta u_h$ and the singular residual $[\partial_\nu u_h]$ and estimate them separately) and is omitted here (see [61] for example).
Remark: Usually, $\eta_T(u_h,f_h)$ is used as error estimator, since it is often not possible to compute the L_2–norm of an arbitrary function exactly. By the triangle inequality it is clear that as well

$$\begin{aligned} \eta_T(u_h,f_h) &\le \eta_T(u_h,f) + h_T\|f-f_h\|_{L_2(T)} \qquad \text{as} \\ \eta_T(u_h,f) &\le \eta_T(u_h,f_h) + h_T\|f-f_h\|_{L_2(T)} \end{aligned}$$

holds.
Since we usually can not compute the right hand side $\int_\Omega f\varphi_h$ of our discrete problem (1.3) exactly, the orthogonality of the error is disturbed. Applying an analysis which includes this defect will result in the a posteriori error estimation

$$|u-u_h|_{H^1(\Omega)} \;\le\; c\Big(\sum_{T\in\mathcal{T}} \eta_T(u_h,f_h)^2\Big)^{1/2} + c\|F-F_h\|_{V_h^*}$$

where we have replaced the right hand side of (1.3) by a computable value $\langle F_h,\,\varphi\rangle_{V_h^*\times V_h} := \int_\Omega f_h\varphi_h$.

The above analysis is not restricted to this simple model problem but can also be used for nonlinear problems (see [60]):
Let $F : \mathring{H}^1(\Omega) \to H^{-1}(\Omega)$ be an operator (maybe nonlinear) and let $u \in \mathring{H}^1(\Omega)$ be a regular solution of

$$F(u) \;=\; 0 \qquad \text{in } H^{-1}(\Omega),$$

i.e. the Frechet–derivative of $DF(u)$ of F at u is invertible and bounded. Assume that DF and DF^{-1} are locally Lipschitz continuous. Now, let u_h be a discrete solution which is"near" u, i.e. $|u-u_h|_{H^1(\Omega)}$ is small enough. Then we get the following estimates:

$$c\,|u-u_h|_{H^1(\Omega)} \;\le\; \|F(u_h)\|_{\mathring{H}^1(\Omega)} \;\le\; C\,|u-u_h|_{H^1(\Omega)}$$

where the constants c, C depend on the norms of $\|DF(u)\|$ and $\|(DF(u))^{-1}\|$ and the Lipschitz constants of DF and DF^{-1}. Again the error is represented in terms of given data and the discrete solution. Now using similar techniques to those used in the model problem will also establish efficient and reliable a posteriori error estimators for nonlinear problems.

1.2 A posteriori error estimation in the L_2 norm

It is often of interest to estimate the error not in the energy norm $|.|_{H^1(\Omega)}$ but in the L_2 norm $\|.\|_{L_2(\Omega)}$. For this we use the so called *Aubin–Nitsche* trick (which is also used for the a priori error estimation [50] or [13]):
Let $w_{u-u_h} \in \mathring{H}^1(\Omega)$ be the solution of the *dual problem*

$$w_{u-u_h} \in \mathring{H}^1(\Omega): \qquad \int_\Omega \nabla\varphi \nabla w_{u-u_h} \;=\; \int_\Omega (u-u_h)\,\varphi \qquad \forall\varphi \in \mathring{H}^1(\Omega). \tag{1.11}$$

Remark: Let $a(.,.) : \mathring{H}^1(\Omega) \times \mathring{H}^1(\Omega) \to \mathbf{R}$ be a non symmetric, bilinear and $\mathring{H}^1(\Omega)$ coercive bilinear form. Then the original problem is stated as

$$u \in \mathring{H}^1(\Omega): \qquad a(u,\varphi) = F(\varphi) \qquad \forall\varphi \in \mathring{H}^1(\Omega)$$

whereas the *dual* problem is stated as

$$w_{u-u_h} \in \mathring{H}^1(\Omega): \qquad a(\varphi, w_{u-u_h}) = \int_\Omega (u-u_h)\,\varphi \qquad \forall\varphi \in \mathring{H}^1(\Omega).$$

To establish an L_2 a posteriori estimate we have to assume that the solution of the dual problem (1.11) is H^2–regular. Assuming Ω is convex one can prove that for the solution w_{u-u_h} of (1.11) we have $w_{u-u_h} \in H^2(\Omega)$ and

$$|w_{u-u_h}|_{H^2(\Omega)} \;\le\; \|-\Delta w_{u-u_h}\|_{L_2(\Omega)} \;=\; \|u-u_h\|_{L_2(\Omega)} \tag{1.12}$$

since $u-u_h \in L_2(\Omega)$ (see [34] e.g.).
Defining for $g \in H^{-2}(\Omega) := (H^2(\Omega) \cap \mathring{H}^1(\Omega))^*$

$$|g|_{H^{-2}(\Omega)} \;:=\; \sup_{\substack{\varphi \in H^2(\Omega)\cap \mathring{H}^1(\Omega) \\ |\varphi|_{H^2(\Omega)}=1}} \langle g,\, \varphi\rangle_{H^{-2}(\Omega)\times(H^2(\Omega)\cap\mathring{H}^1(\Omega))},$$

using the fact that $w_{u-u_h} \in H^2(\Omega)$, and setting $\varphi = u-u_h$ in (1.11) we conclude

$$\begin{aligned}
\|u-u_h\|^2_{L_2(\Omega)} &= \int_\Omega \nabla(u-u_h)\nabla w_{u-u_h} \\
&= \langle -\Delta(u-u_h),\, \underbrace{w_{u-u_h}}_{\in H^2(\Omega)}\rangle_{H^{-1}(\Omega)\times\mathring{H}^1(\Omega)} \\
&= \langle F+\Delta u_h,\, w_{u-u_h}\rangle_{H^{-2}(\Omega)\times(H^2(\Omega)\cap\mathring{H}^1(\Omega))} \\
&\le |F+\Delta u_h|_{H^{-2}(\Omega)} |w_{u-u_h}|_{H^2(\Omega)} \\
&\le |F+\Delta u_h|_{H^{-2}(\Omega)} \|u-u_h\|_{L_2(\Omega)}.
\end{aligned}$$

On the other hand using the higher regularity of the test function φ and integration by parts we have

$$|F+\Delta u_h|_{H^{-2}(\Omega)} \;=\; \sup_{\substack{\varphi \in H^2(\Omega)\cap \mathring{H}^1(\Omega) \\ |\varphi|_{H^2(\Omega)}=1}} \langle F+\Delta u_h,\, \varphi\rangle_{H^{-2}(\Omega)\times(H^2(\Omega)\cap\mathring{H}^1(\Omega))}$$

$$
\begin{aligned}
&= \sup_{\substack{\varphi \in H^2(\Omega)\cap \mathring{H}^1(\Omega) \\ |\varphi|_{H^2(\Omega)}=1}} \int_\Omega \nabla(u-u_h)\nabla\varphi \\
&= \sup_{\substack{\varphi \in H^2(\Omega)\cap \mathring{H}^1(\Omega) \\ |\varphi|_{H^2(\Omega)}=1}} \int_\Omega (u-u_h)(-\Delta\varphi) \\
&\leq \|u-u_h\|_{L_2(\Omega)}.
\end{aligned}
$$

Combining these two estimates we achieve

$$
\|u-u_h\|_{L_2(\Omega)} \;=\; |F+\Delta u_h|_{H^{-2}(\Omega)}. \tag{1.13}
$$

In order to establish the L_2 estimate, we have to estimate now the term $|F+\Delta u_h|_{H^{-2}(\Omega)}$. This is done in the same manner as in the case of the energy norm. In contrast to that estimate we can use the fact that the test function φ belongs to $H^2(\Omega)$. Thus, for the interpolation of φ we can make use of the usual Lagrange interpolant ($H^2(\Omega)$ is embedded in $C^0(\bar{\Omega})$, $d=2,3$!) and we gain a higher power of h_T in front of the residuals since we can rely on second derivatives of φ. As a result we have

$$
\|u-u_h\|_{L_2(\Omega)} \;\leq\; c\left(\sum_{T\in\mathcal{T}} \tilde{\eta}_T(u_h,f)^2\right)^{1/2} \tag{1.14}
$$

where $\tilde{\eta}_T$ is defined to be

$$
\tilde{\eta}_T(u_h,f)^2 \;:=\; h_T^4\|f+\Delta u_h\|^2_{L_2(T)} + \frac{1}{2}h_T^3\|[\partial_\nu u_h]\|^2_{L_2(\partial T\setminus\partial\Omega)}.
$$

Again, using the finite dimensional approximation f_h of f we can prove the efficiency

$$
\tilde{\eta}_T(u_h,f_h) \;\leq\; c\Big(\|u-u_h\|_{L_2(M(T))} + h_T^2\|f-f_h\|_{L_2(M(T))}\Big) \tag{1.15}
$$

where we also gain one additional power of h_T in front of the term $\|f-f_h\|_{L_2(M(T))}$.
This analysis also carries over to nonlinear problems under suitable assumptions on the existence of the dual problem and the regularity of its solution. Under such assumptions we can prove

$$
c\,\|u-u_h\|_{L_2(\Omega)} \;\leq\; \|F(u_h)\|_{H^2(\Omega)\cap\mathring{H}^1(\Omega)} \;\leq\; C\,\|u-u_h\|_{L_2(\Omega)}
$$

where now c and C depend on the coercivity of the dual problem (which is associated to the norms of DF and DF^{-1}) and the regularity constant for the solution of the dual problem. This inequality now establishs L_2 error estimators for nonlinear problems using the same techniques as described above [5].

2 Mesh refinement and coarsening

Finite element meshes may consist of geometric elements of various types:
simplicial: triangles or tetrahedra,
quadrilateral: rectangles, cubes, or general quadrilaterals,
more general: prisms, for example,
mixed: mixture of different types.
The choice of the mesh type for an application may depend on some special approximation properties or on the need for some special FE basis functions, which require a special local geometry. We will restrict ourselves here to the description of simplicial meshes, for several reasons:

- A simplex is one of the most simple geometric types.
- Complex domains may be approximated by a set of simplices quite easily.
- Simplicial meshes allow local refinement (see Figure 2.1) without the need of nonconforming meshes (hanging nodes), parametric elements, or mixture of element types (which is the case for quadrilateral meshes, for example, see Figure 2.2).
- Polynomials of a given degree are easily represented on a simplex using local (barycentric) coordinates. (On quadrilateral elements, the 'standard' type of ansatz spaces is a tensor product of onedimensional polynomials.)

Refinement algorithms for non-simplicial meshes can be found in the literature.

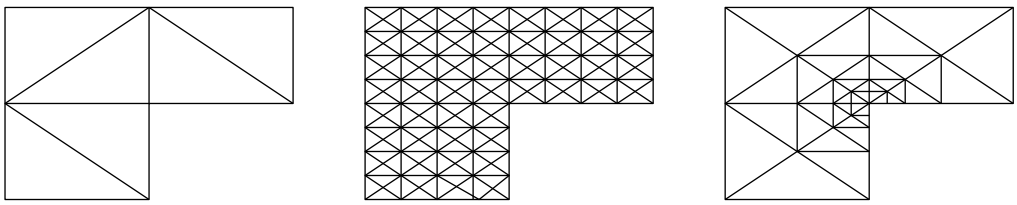

Figure 2.1: Global and local refinement of a triangular mesh.

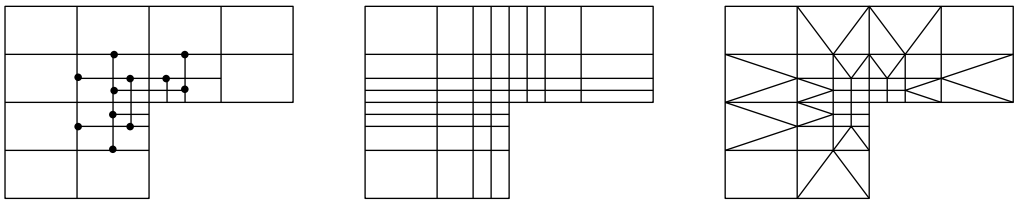

Figure 2.2: Local refinements of a rectangular mesh: with hanging nodes, conforming closure using bisected rectangles, and conforming closure using triangles. Using a conforming closure with rectangles, a local refinement has always global effects up to the boundary.

We will consider the following situation:
An initial (coarse) triangulation of the domain is given. We call it 'macro triangulation'. It may be generated by hand or by some mesh generation algorithm.

Some (or all) of the simplices are marked for refinement, depending on some error estimator or indicator. After several refinements, some other simplices may be marked for coarsening. Marking criteria and marking strategies are subject of Section 3.

2.1 Refinement algorithms for simplicial meshes

For simplicial elements, several refinement algorithms are widely used. One example is regular refinement ("red refinement"), which divides every triangle into four similar triangles, see Figure 2.3. The corresponding refinement algorithm in three dimensions cuts every tetrahedron into eight tetrahedra, and only a small number of similarity classes occur during successive refinements, see [7]. Unfortunately, hanging nodes arise during local regular refinement. To remove them and create a conforming mesh, in two dimensions some triangles have to be bisected ("green closure"). In three dimensions, several types of irregular refinement are needed for the green closure. This creates more similarity classes, even in two dimensions. Additionally, these bisected elements have to be removed before a further refinement of the mesh, in order to keep the triangulation regular.

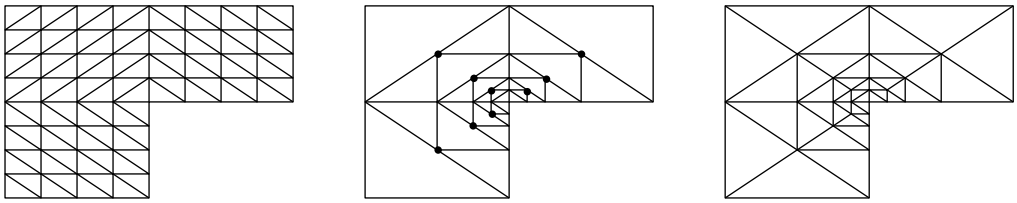

Figure 2.3: Global and local regular refinement of triangles and conforming closure by bisection.

Another possibility is to use bisection of simplices only. For every element (triangle or tetrahedron) one of its edges is marked as the refinement edge, and the element is refined into two elements by cutting this edge at its midpoint. There are several possibilities to choose such a refinement edge for a simplex, one example is to use the longest edge. Mitchell [48] compared different approaches. We will describe an algorithm where the choice of refinement edges on the macro triangulation prescribes the refinement edges for all simplices that are created during mesh refinement (the "newest vertex" bisection in Mitchell's notation). This make sure that shape regularity of the triangulations is conserved.
The refinement by bisection can be implemented using recursive or non–recursive algorithms. For tetrahedra, the first description of such refinements was done in the non–recursive way by Bänsch [4]. It needs the intermediate handling of hanging nodes during the refinement process. Two recursive algorithms, which do not create such hanging nodes and are therefore easier to implement, are published by Kossaczký [45] and Maubach [47], which result in exactly the same tetrahedral meshes as the non–recursive algorithm.
Other refinement techniques for simplicial meshes, such as Delaunay techniques, are possible and described in the literature. We do not present details here.
In the following, we will describe the recursive refinement by bisection in detail, using the notation of Kossaczký. An implementation was done for example in [58].

The refinement algorithm is based on a recursive bisectioning of elements. For every element of the mesh, one of its edges is marked as its *refinement edge*. Elements are refined by bisecting this edge. To keep the mesh conforming, bisection of an edge is only allowed when this edge

is the refinement edge for all elements which share this edge. Bisection of an edge and thus of all elements around the edge is the atomic refinement operation, and no other refinement operations are allowed. See Figures 2.4 and 2.5 for the two and three dimensional situations.

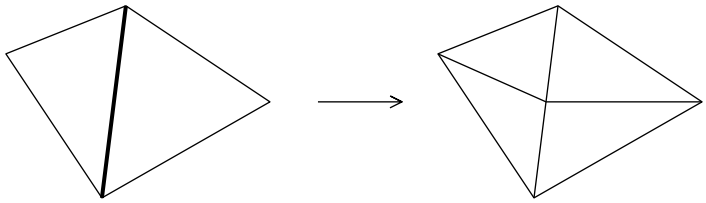

Figure 2.4: Atomic refinement operation in two dimensions. The common edge is the refinement edge for both triangles.

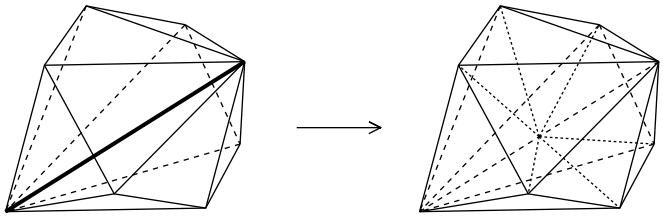

Figure 2.5: Atomic refinement operation in three dimensions. The common edge is the refinement edge for all tetrahedra around it.

If an element has to be refined, we first get all elements at this edge. In two dimensions this is just the neighbour opposite this edge or there is no other element at this edge in the case that the refinement edge belongs to the boundary. In three dimensions we have to loop around the edge and collect all neighbours at this edge. If for all collected neighbours this edge is the refinement edge also, we can refine the whole patch at same time by inserting one new vertex in the midpoint of the common refinement edge and bisecting every element of the patch. The resulting triangulation then is a conforming one.
If one of the collected neighbours has not the same refinement edge we first refine this neighbour recursively. Thus, we can formulate the refinement of an element in the following way

Algorithm 2.1 *Recursive refinement of one simplex*

```
subroutine recursive_refine(element)
{
   do
   {
      for all neighbours at refinement edge
         if neighbour has no compatible refinement edge
            recursive_refine(neighbour);
   } until all neighbours have a compatible refinement edge;

   bisect all elements at the refinement edge;
}
```

In two dimensions we used the so called newest vertex bisection and in three dimensions the algorithm described in [45]. For both variants it is proved, that for macro triangulation fulfilling

certain criteria the recursion stops. Both algorithms are for special macro triangulations the recursive variants of the non recursive algorithms described in [4]. The beauty of the recursive approach is that we do not have to handle hanging nodes and not one to one adjacencies, since we can refine the whole refinement patch at same time.
In Figure 2.6 we show a twodimensional situation where recursion is needed. For all triangles, the longest edge is the refinement edge. Let us assume that triangles A and B are marked for refinement. Triangle A can be refined at once, as its refinement edge is a boundary edge. For refinement of triangle B, we have to recursively refine triangles C and D. Again, triangle D can be directly refined, so recursion stops there. This is shown in the second part of the figure. Back in triangle C, this can now be refined together with its neighbour. After this, also triangle B can be refined together with its neighbour.

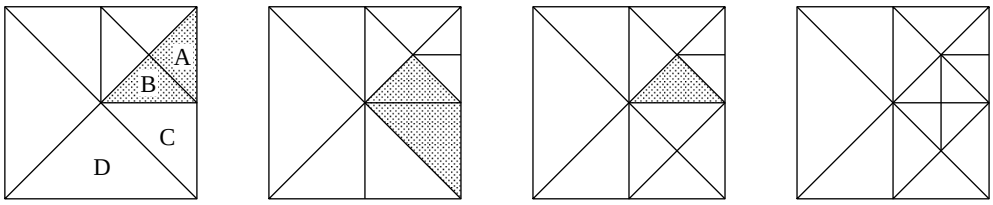

Figure 2.6: Recursive refinement in two dimensions. Triangles A and B are initially marked for refinement.

Now, the overall refinement algorithm can be formulated as follows:

Algorithm 2.2 *Refinement of the mesh*

```
subroutine refine_mesh()
{
   for all elements
      while element is marked for refinement
        recursive_refine(element);
}
```

We will use the convention, that all vertices of an element are given fixed *local indices*. Valid indices are 0, 1, and 2 for vertices of a triangle, and 0, 1, 2, and 3 for vertices of a tetrahedron. Now, the refinement edge for an element can be fixed to be the edge between the vertices with local indices 0 and 1.

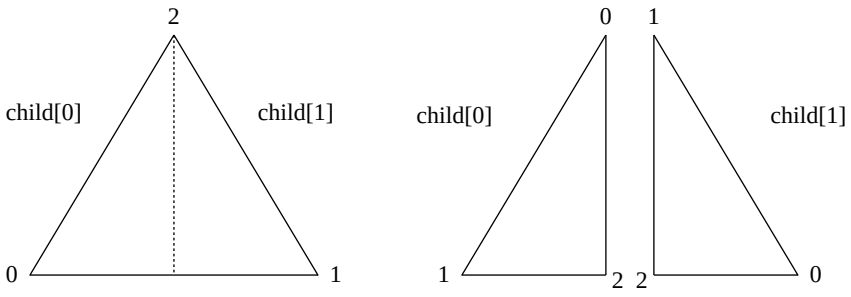

Figure 2.7: Numbering of nodes on parent and children triangles

During refinement, the new vertex numbers for the newly created child simplices are prescribed by the refinement algorithm. This is done in such a way, that only a small number of similarity

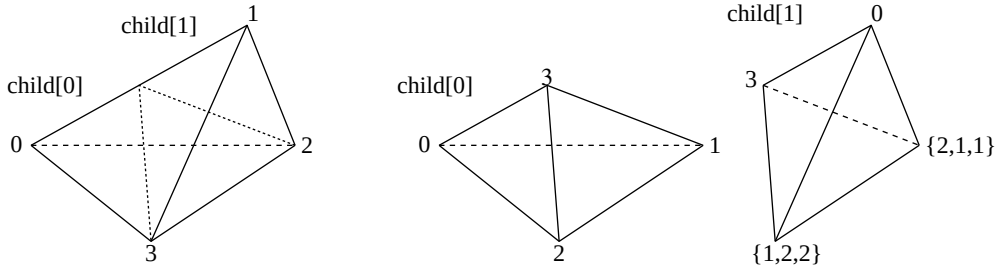

Figure 2.8: Numbering of nodes on parent and children tetrahedra

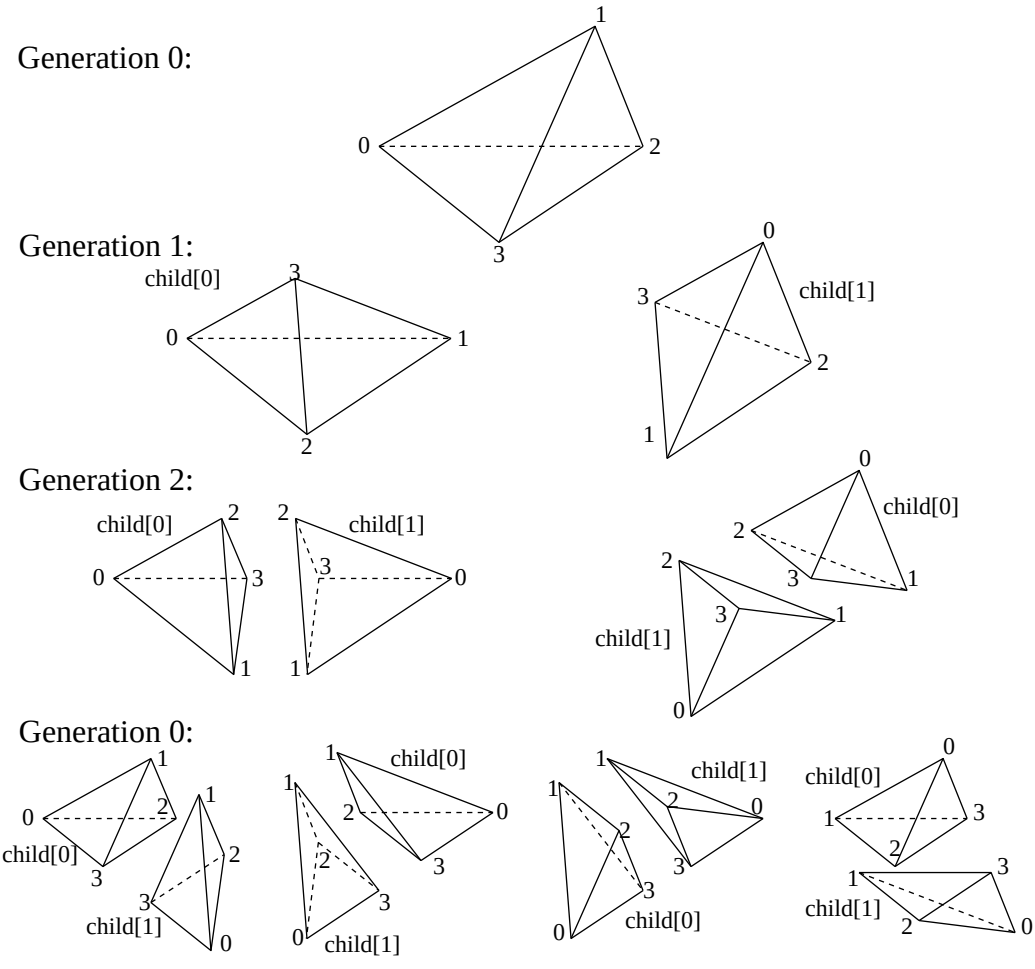

Figure 2.9: Successive refinements of a generation 0 tetrahedron

classes occur during successive refinement of one macro element. For both children elements, the index of the newly generated vertex at the midpoint of this edge has the highest local index (2 resp. 3 for triangles and tetrahedra). These numbers are shown in Figure 2.7 for 2d and in 2.8 for 3d. In 2d this numbering is the same for all refinement levels. In 3d, one has to make some special arrangements: the numbering of the second child's vertices does depend on the *generation* of the elements. There exist three different generations 0, 1, and 2, and the generation of a child element is always ((parent's generation + 1) modulo 3). In Figure 2.8 we used the following convention: for the index set $\{1, 2, 2\}$ on `child[1]` of a tetrahedron of generation `0` we use the index `1` and for a tetrahedron of generation `1` and `2` the index `2`. Figure 2.9 shows successive refinments of a generation 0 tetrahedron, producing tetrahedra of generations 1, 2, and 0 again.
Using the above refinement algorithm, the refinements of a mesh are totally determined by the local vertex numbering of the macro triangulation, plus a prescribed generation for every macro element in three dimensions.
The numbering for tetrahedra was introduced by Kossaczký. In case of the "standard" triangulation of a (unit) square and cube into two triangles resp. six tetrahedra (see Figure 2.10),

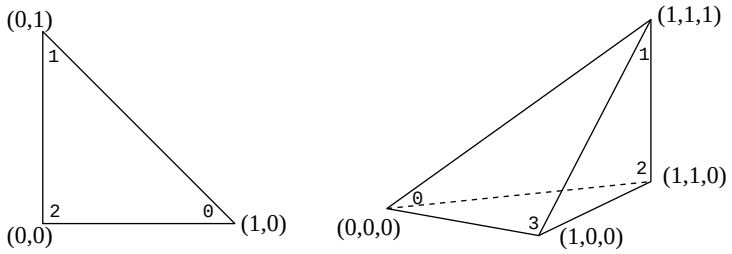

Figure 2.10: Standard elements in two and three dimensions

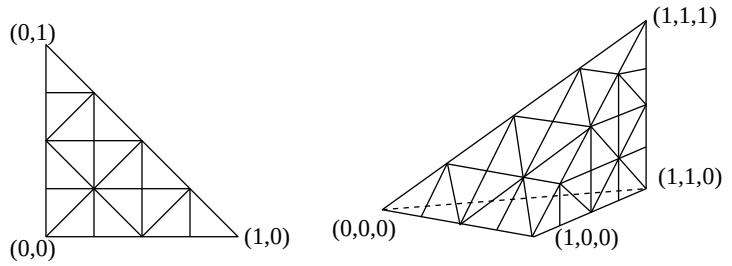

Figure 2.11: Refined standard elements in two and three dimensions

these numberings and the definition of the refinement edge during refinement of the elements guarantee that always the longest edge will be the refinement edge and will be bisected, see Figure 2.11. For the general case is proved:

Theorem 2.1 *(Kossaczký [45], Mitchell [48])*

1. *The recursion stops if the macro triangulation fulfils certain criteria.*
2. *We obtain shape regularity for all elements at all levels.*

In two dimensions, a triangulation where recursion does not stop is shown in Figure 2.12. The selected refinement edges of the triangles are shown by dashed lines. One can easily see, that

there are no patches for the atomic refinement operation. This triangulation can only be refined if other choices of refinement edges are made, or by a non-recursive algorithm.

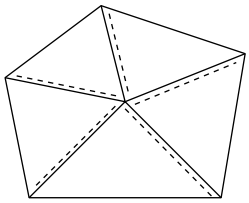

Figure 2.12: A macro triangulation where recursion does not stop

For using the refinement algorithm in a finite element package, we also need a numbering for edges, neighbours and faces. Edges and faces are needed for the implementation of higher order elements, for example, and neighbour information is used in the refinement algorithm itself and for error estimator calculation, for example.
In 2d the `i`-th edge/neighbour is the edge/neighbour opposite the `i`-th vertex; in 3d the `i`-th face/neighbour is the face/neighbour opposite the `i`-th vertex; edges in 3d are numbered in the following way:

edge 0: between vertex `0` and `1`, **edge 3:** between vertex `1` and `1`,
edge 1: between vertex `0` and `2`, **edge 4:** between vertex `1` and `3`,
edge 2: between vertex `0` and `3`, **edge 5:** between vertex `2` and `3`.

Figure 2.13 shows the numbering of the edges of child tetrahedra after refinement. The markers describe, which edge's degrees of freedom are changed during refinement, when higher order elements are used. For a more detailed description of handling higher order elements, see [58].

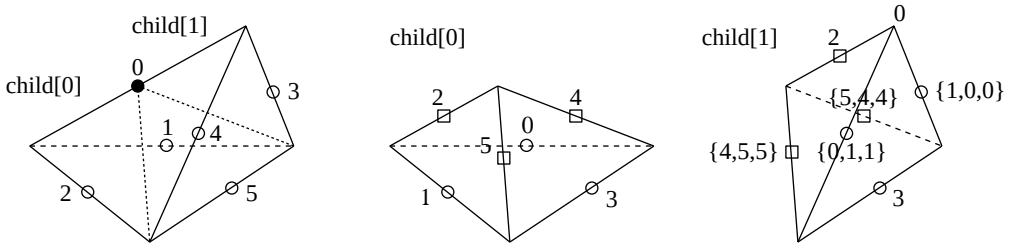

Figure 2.13: Edge numbers during refinement and degrees of freedom that are no longer needed •, passed on from parent to child elements ∘, and newly generated □

2.2 Prolongation of data during refinement

During refinement, finite element functions will have to be adjusted to the new mesh situation. Using hierarchically structured meshes, the finite element space of the coarse mesh is a subset of the space of the refined mesh (at least for typical polynomial ansatz spaces and refinement by bisection — there exist some finite elements where spaces are not nested, and the conforming closure needed by local regular refinements may lead to non–nested spaces, too). Thus, data can be represented identically on the refined mesh. During local refinement procedures, this

prolongation of information from the old mesh to the new one is usually done directly together with the mesh changes.
After the geometrical part of the refinement is done on a patch around a refinement edge, we can prolongate data handled by the degrees of freedom from parents to child on the whole patch. We will describe the prolongation in detail for the case of piecewise linear finite elements; for higher order elements, everything is similar, but more degrees of freedom are involved.
For linear element, when degrees of freedom are located at vertices only, everything takes place on the bisected edge alone. Only one new vertex is created, the midpoint of the refinement edge. To determine the value of a function f_h at this new vertex, we can interpolate the function at this point. On the edge, f_h is a polynomial of degree 1, so the value at the midpoint is just the mean of the values at the edge endpoints:

$$f_h(\text{midpoint}) = \frac{1}{2}(f_h(\text{vertex } 0) + f_h(\text{vertex } 1)).$$

Using the nodal basis functions $\phi_i(v_j) = \delta_{i,j}$, then the coefficient f_n of the new basis function ϕ_n is just

$$f_n = \frac{1}{2}(f_0 + f_1).$$

2.3 Coarsening algorithms

The coarsening algorithm is more or less the inverse of the refinement algorithm. The basic idea is to collect all those elements that were created during the refinement at same time, i.e. the parents of these elements build a compatible refinement patch. If all the elements are marked for coarsening, information is passed on the parents and the whole patch is coarsened at the same time.
If one of the elements is not marked for coarsening, we are not allowed to coarsen the patch. All element markers are reset. If one of the collected elements is not a leaf element but we are allowed to coarsen it more than one time, we first try to coarsen this element and then try to coarsen the newly collected patch.
This is the main difference between refinement and coarsening: Every element that is marked for refinement will be refined and this refinement may enforce a refinement of other elements that are not marked for refinement. An element that is marked for coarsening can only be coarsened if all elements of the coarsening patch may be coarsened together with this element. An element that is not marked for coarsening must not be coarsened, compare Section 3.2.
Thus, we can formulate the coarsening algorithm as follows:

Algorithm 2.3 *Local coarsening around one edge*

```
subroutine coarsen(element)
{
  get the parents of all elements at the coarsening edge

  for all parents
  {
    if the coarsening edge of the parent is not compatible
    {
      reset coarsening marks of all children of this patch;
      return false;
    }
  }

  for all parents
  {
    if the parent is refined more than once,
    and its children can be coarsened more than once
      return true;
  }

  coarsen all parents at the coarsening edge;
  return false;
}
```

The following routine coarsens as many elements as possible, even more than once if allowed:

Algorithm 2.4 *Coarsening of the mesh*

```
subroutine coarsen_mesh()
{
  do
  {
    do_coarsen_once_more = false;
    for all elements
      if element is marked for coarsening
        do_coarsen_once_more |= coarsen(element);
  } until do_coarsen_once_more is false
}
```

2.4 Restriction of data during coarsening

Also during coarsening, finite element functions will have to be adjusted to the new mesh situation. As now no longer the new finite element space is a superset of the old one, we loose some information. The marking strategies based on error estimators or indicators choose parts of the mesh to be coarsened, where the amount of lost information is not too big.
Nevertheless, finite element functions have to be restricted (transfered) from the fine to the coarse mesh. For linear finite elements, the easiest way to get around is just to ignore the value at the old vertex that will be removed, and interpolate the function in all remaining vertices.

In one special situation, information can be transfered identically from the old to the new mesh. If the values of a linear functional F applied to all basis functions are of interest, we can transform these values during coarsening, without making any error: If ϕ_0^{fine}, ϕ_1^{fine}, and ϕ_n^{fine} denote the basis functions corresponding to the endpoints and the midpoint of the edge inside the coarsened patch, then the new basis functions corresponding to the endpoints of the edge are

$$\phi_0^{\text{coarse}} = \phi_0^{\text{fine}} + \frac{1}{2}\phi_n^{\text{fine}}, \qquad \phi_1^{\text{coarse}} = \phi_1^{\text{fine}} + \frac{1}{2}\phi_n^{\text{fine}}.$$

This can easily be seen by interpolation of the coarse basis functions. Now, if for some linear functional F the values $\langle F, \phi_0^{\text{fine}}\rangle$, $\langle F, \phi_1^{\text{fine}}\rangle$, and $\langle F, \phi_n^{\text{fine}}\rangle$ are available, the values of F applied to the new basis functions are

$$\langle F, \phi_0^{\text{coarse}}\rangle = \langle F, \phi_0^{\text{fine}}\rangle + \frac{1}{2}\langle F, \phi_n^{\text{fine}}\rangle, \qquad \langle F, \phi_1^{\text{coarse}}\rangle = \langle F, \phi_1^{\text{fine}}\rangle + \frac{1}{2}\langle F, \phi_n^{\text{fine}}\rangle.$$

As one can easily see, the transformation matrix which transforms the old vector of functional values to the new one is just the transpose of the transformation matrix which was used for prolongation during refinement. This is the same for higher order elements.

One application of this procedure is time discretization, where scalar products with the solution u^{m-1} from the last time step appear on the right hand side of the discrete problem.

2.5 Storage methods for hierarchical meshes

There are basically two kinds of storing a finite element grid. One possibility is to store only the elements of the triangulation in a vector or a linked list. All information about elements is located at the elements. In this situation there is no direct information of a hierarchical structure for multigrid methods, e.g. Such information has to be generated and stored separately. During mesh refinement, new elements are added (at the end) to the vector or list of elements. During mesh coarsening, elements are removed. In case of an element vector, 'holes' may appear in the vector that contain no longer a valid element. One has to take care of them, or remove them by compressing the vector.

The other kind of storing the mesh is to keep the whole sequence of grids starting on the macro triangulation up to the actual one. Storing information about the whole hierarchical structure will need additional amount of computer memory, but on the other hand we can save computer memory by storing such information not explicitly on each element which can be produced by the hierarchical structure.
The simplicial grid is generated by refinement of a given macro triangulation. Refined parts of the grid can be derefined, but we can not coarsen elements of the macro triangulation. The refinement and coarsening routines construct a sequence of nested grids with a hierarchical structure. Every refined simplex is refined into two children. Elements that may be coarsened were created by refining the parent into these two elements and are now just coarsened back into this parent (compare Sections 2.1, 2.3).
Using this structure of the refinement/coarsening routines, every element of the macro triangulation is the root of a binary tree: every interior node of that tree has two pointers to the two children; the leaf elements are part of the actual triangulation, which is used to define the finite element space. The whole triangulation is a list (or vector) of given macro elements together with the associated binary trees.

Operations on elements can be performed by traversing the mesh, using standard tree traversing algorithms.
Some information is stored on the (leaf) elements explicitly, other information is located at the macro elements and is transfered to the leaf elements while traversing through the binary tree. All information that should be available for mesh elements is stored explicitly for elements of the macro triangulation. Thus, all information is present on the macro level and is transfered to the other tree elements by transforming requested data from one element to its children. These can be done by simple calculations using the hierarchic structure induced by the refinement algorithm.
An example of information which does not have to be stored for each element are the coordinates of the element's vertices (in the case of non-parametric elements and polyhedral boundary). Going from parent to child only the coordinates of one vertex changes and the new ones are simply the mean value of the coordinates of two vertices at the so called refinement edge of the parent. The other vertex coordinates stay the same.
Another example of such information is information about adjacent elements. Using adjacency information of the macro elements we can compute requested information for all elements of the mesh.
An implementation of the hierarchical mesh storage is done in [58].

3 Adaptive strategies

During this section, we will present several strategies for the local refinement and coarsening of finite element meshes and for adjustment of the time step size.

3.1 Mesh refinement strategies

Let us assume that a triangulation $\mathcal{T}_h$ of Ω, a finite element solution $u_h \in V_h$ to an elliptic problem, and an a posteriori error estimate

$$\|u - u_h\| \;\leq\; \eta(u_h) \;:=\; \left(\sum_{T \in \mathcal{T}_h} \eta_T(u_h)^2 \right)^{1/2}$$

on this mesh are given. If ε is a given allowed tolerance for the error, and $\eta(u_h) > \varepsilon$, the problem arises

- where to refine the mesh in order to reduce the error,
- while at the same time the number of unknowns should not become too large.

A global refinement of the mesh would lead to the best reduction of the error, but the amount of new unknowns might be much larger than needed to reduce the error below the given tolerance. Using local refinement, we hope to do much better.
The design of an "optimal" mesh, where the number of unknowns is as small as possible to keep the error below the tolerance, is an open problem and will probably be much too costly. Especially in the case of linear problems, the design of an optimal mesh will be much more expensive than the solution of the original problem, since the mesh optimization is a highly nonlinear problem. Some heuristic arguments have to be used in the algorithm. The aim is to produce a result that is "not too far" from an optimal mesh, but with a relatively small amount of additional work to generate it.
Several adaptive strategies are proposed in the literature, that give criteria which mesh elements should be marked for refinement. All strategies are based on the idea of an equidistribution of the local error to all mesh elements. Babuška and Rheinboldt [1] motivate that a mesh is almost optimal when the local errors are approximately equal for all elements. So, elements where the error estimate is large will be marked for refinement, while elements with a small estimated error are left unchanged.
The general outline of the adaptive algorithm is as follows. Starting from an initial triangulation $\mathcal{T}_0$, we produce a sequence of triangulations $\mathcal{T}_k$, $k = 1, 2, \ldots$, until the estimated error is below the given tolerance:

Algorithm 3.1 *General adaptive refinement strategy*

```
Start with T_0 and error tolerance ε
k := 0
do forever
   solve the discrete problem on T_k
   compute local error estimates η_T, T ∈ T_k
   if η ≤ ε then
      stop
   mark elements for refinement, according to a marking strategy
   refine mesh T_k, producing T_{k+1}
   k := k + 1
enddo
```

Since a discrete problem has to be solved in every iteration of this algorithm, the number of iterations should be as small as possible. Thus, the marking strategy should select not too few mesh elements for refinement in each cycle. On the other hand, not much more elements should be selected than is needed to reduce the error below the given tolerance.
In the sequel, we describe several marking strategies that are commonly used in adaptive finite element methods.

Maximum strategy: The simplest strategy is a maximum strategy. A threshold $\gamma \in (0,1)$ is given, and all elements $T \in \mathcal{T}_k$ with

$$\eta_T \;>\; \gamma \max_{T' \in \mathcal{T}_k} \eta_{T'} \tag{3.1}$$

are marked for refinement. A small γ leads to more refinement and non–optimal meshes, while a large γ leads to more cycles until the error tolerance is reached, but produces a mesh with less unknowns. Typically, a threshold value $\gamma = 0.5$ is used [61, 63].

Algorithm 3.2 *Maximum strategy*

```
Start with parameter γ ∈ (0,1)
η_max := max(η_T, T ∈ T_k)
for all T in T_k do
   if η_T > γ η_max then mark T for refinement
enddo
```

Extrapolation strategy: Suppose that the local error estimates have an asymptotic behaviour

$$\eta_T \;=\; c\, h_T^{\lambda} \qquad \text{as } h \to 0$$

for some $\lambda > 0$. If an element T with estimate η_T was generated by refining an element T^{old} in a previous mesh with corresponding estimate η_T^{old}, then the above behaviour suggests that the estimate at one of the childs after refining T will be approximately

$$\eta_T^{\text{new}} \;=\; \frac{\eta_T^2}{\eta_T^{\text{old}}}.$$

Now, the idea is that no elements should be refined in the current iteration, where the estimated error is smaller than the largest local estimate that is expected after the next refinement. This leads to the following algorithm:

Algorithm 3.3 *Extrapolation strategy [1]*

```
cut := max(η_T^new, T ∈ T_k)
for all T in T_k do
   if η_T > cut then mark T for refinement
enddo
```

If η_T^{old} is unknown and thus η_T^{new} cannot be computed, some other marking strategy has to be used.

Equidistribution strategy: Let N_k be the number of mesh elements in $\mathcal{T}_k$. If we assume that the error is equidistributed over all elements, i. e. $\eta_T = \eta_{T'}$ for all $T, T' \in \mathcal{T}_k$, then

$$\eta \;=\; \left(\sum_{T \in \mathcal{T}_h} \eta_T^2 \right)^{1/2} \;=\; \sqrt{N_k}\, \eta_T \;\stackrel{!}{=}\; \varepsilon \qquad \text{and} \qquad \eta_T \;=\; \frac{\varepsilon}{\sqrt{N_k}}\,.$$

We can try to reach this equidistribution by refining all elements, where it is disturbed because the estimated error is larger than $\varepsilon/\sqrt{N_k}$. To make the procedure more robust, a parameter $\theta \in (0,1)$, $\theta \approx 1$, is included in the method.

Algorithm 3.4 *Equidistribution strategy [24]*

```
Start with parameter θ ∈ (0,1), θ ≈ 1
for all T in T_k do
   if η_T > θε/√N_k then mark T for refinement
enddo
```

If the error η is already near ε, then the choice $\theta = 1$ leads to the selection of only very few elements for refinement, which results in more iterations of the adative process. Thus, θ should be chosen smaller than 1, for example $\theta = 0.9$.

Guaranteed error reduction strategy: Usually, it is not clear whether the adaptive refinement strategy Algorithm 3.1 using a marking strategy (other than global refinement) will converge and stop, or how fast the convergence is. Dörfler [17] describes a strategy with a guaranteed relative error reduction for the Poisson equation.
We need the assumptions, that

- given data of the problem (like the right hand side) is sufficiently resolved by the current mesh $\mathcal{T}_k$,
- all edges of marked mesh elements are at least bisected by the refinement procedure (using regular refinement or two/three iterated bisections of triangles/tetrahedra, for example).

The idea is to refine a subset of the triangulation that produces a considerable amount of the total error η. Given a parameter $\theta_* \in (0,1)$, the procedure is:

$$\text{Mark a set } \mathcal{A} \subseteq \mathcal{T}_k \text{ such that } \quad \sum_{T \in \mathcal{A}} \eta_T^2 \geq (1-\theta_*)^2 \eta^2\,.$$

It follows from the assumptions that the error will be reduced by at least a factor $\kappa < 1$ depending of θ_* and data of the problem. Selection of the set $\mathcal{A}$ can be done in the following way. The threshold γ is reduced in small steps of size $\nu \in (0,1)$, $\nu \approx 0$, until the maximum strategy marks a set which is large enough. This inner iteration does not cost much time, as no computations are done in it.

Algorithm 3.5 *Guaranteed error reduction strategy [17]*

```
Start with given parameters θ* ∈ (0,1), ν ∈ (0,1)
η_max := max(η_T, T ∈ T_k)
sum := 0
γ := 1
while sum < (1 − θ*)² η² do
   γ := γ − ν
   for all T in T_k do
      if T is not marked
         if η_T > γ η_max
            mark T for refinement
            sum := sum + η_T²
         endif
      endif
   enddo
endwhile
```

Using the above algorithm, Dörfler [16] describes a robust adaptive strategy also for the *nonlinear* Poisson equation $-\Delta u = f(u)$. It is based on a posteriori error estimates and a posteriori saturation criteria for the approximation of the nonlinearity.

Other refinement strategies: Jarausch [39] describes a strategy which generates quasi–optimal meshes. It is based on an optimization procedure involving the increase of a cost function during refinement and the profit while minimizing an energy functional.
For special applications, additional information may be generated by the error estimator and used by the adaptive strategy. This includes (anisotropic) directional refinement of elements [44, 59], or the decision of local h– or p–enrichment of the finite element space [15].

3.2 Coarsening strategies

Up to now we presented only refinement strategies. For linear elliptic problems, no more is needed to generate a quasi–optimal mesh with nearly equidistributed local errors.
In timedependent problems, the regions where large local errors are produced can move in time. In stationary nonlinear problems, a bad resolution of the solution on coarse meshes may lead to some local refinement where it is not needed for the final solution, and the mesh could be coarsened again. Both situations result in the need to coarsen the mesh at some places in order to keep the number of unknowns small.
Coarsening of the mesh can produce additional errors in a timedependent process. Assuming that these are bounded by an a posteriori estimate $\eta_{c,T}$, we can take this into account during the marking procedure.

Some of the refinement strategies described above can also be used to mark mesh elements for coarsening. Actually, elements will only be coarsened if all neighbour elements which are affected by the coarsening process are marked for coarsening, too. This makes sure that only elements where the error is small enough are coarsened, and motivates the coarsening algorithm in Section 2.3.

Equidistribution strategy: Equidistribution of the tolerated error ε leads to

$$\eta_T \;\approx\; \frac{\varepsilon}{\sqrt{N_k}} \qquad \text{for all } T \in \mathcal{T}.$$

If the local error at an element is considerably smaller than this mean value, we may coarsen the element without producing an error that is too large. If we are able to estimate the error after coarsening, for example by assuming an asymptotic behavior like

$$\eta_T \;\approx\; c\, h_T^{\lambda}, \quad \lambda > 0\,,$$

we can calculate a threshold $\theta_c \in (0, \theta)$ such that the local error after coarsening is still below $\theta\,\varepsilon/\sqrt{N_k}$ if it was smaller than $\theta_c\,\varepsilon/\sqrt{N_k}$ before. If the error after coarsening gets larger than this value, the elements would directly be refined again in the next iteration.

Algorithm 3.6 *Equidistribution refinement/coarsening strategy*

```
Start with parameters θ ∈ (0,1), θ ≈ 1, and θ_c ∈ (0,θ)

for all T in T_k do
   if η_T > θ ε/√N_k then mark T for refinement
   if η_T + η_{c,T} < θ_c ε/√N_k then mark T for coarsening
enddo
```

When local h– and p–enrichment and coarsening of the finite element space is used, then the threshold θ_c depends on the local degree of finite elements. Thus, local thresholds $\theta_{c,T}$ have to be used.

Guaranteed error reduction strategy: Similar to the refinement in Algorithm 3.5, Dörfler [18] describes a marking strategy for coarsening. Again, the idea is to coarsen a subset of the triangulation such that the additional error after coarsening is not larger than a fixed amount of the given tolerance ε. Given a parameter $\theta_c \in (0, 1)$, the procedure is:

$$\text{Mark a set } \mathcal{B} \subseteq \mathcal{T}_k \text{ such that } \quad \sum_{T \in \mathcal{B}} \eta_T^2 + \eta_{c,T}^2 \le \theta_c^2 \varepsilon^2\,.$$

The selection of the set $\mathcal{B}$ can be done similar to Algorithm 3.5. Under suitable assumptions, Dörfler proves that the adaptive algorithm with mesh refinement and coarsening leads to an error below the given tolerance [18].

Handling information loss during coarsening: Usually, some information is irreversibly destroyed during coarsening of parts of the mesh, compare Section 2.4. If the adaptive procedure iterates several times, it may occur that elements which were marked for coarsening in the beginning are not allowed to coarsen at the end. If the mesh was already coarsened, an error is produced which can not be reduced anymore.

One possibility to circumvent such problems is to delay the mesh coarsening until the final iteration of the adaptive procedure, allowing only refinements before. If the coarsening marking strategy is not too liberal (θ_c not too large), this should keep the error below the given bound. Dörfler [18] proposes to keep all information until it is clear, after solving and by estimating the error on a (virtually) coarsened mesh, that the coarsening does not lead to an error which is too large.

3.3 Adaptive procedures for timedependent problems

In timedependent problems, the mesh is adapted to the solution in every time step using a posteriori error estimators or indicators. Bänsch [3] lists several different adaptive procedures for timedependent problems:

- **Explicit strategy:** The current time step is solved once on the mesh from the previous time step, giving the solution u_h. Based on a posteriori estimates of u, the mesh is locally refined and coarsened. The problem is *not* solved again on the new mesh, and the estimate–adapt process is *not* iterated.
  This strategy is only usable when the solution is nearly stationary and does not change much in time, or when the time step size is very small.

- **Semi–implicit strategy:** The current time step is solved once on the mesh from the previous time step, giving an intermediate solution $\tilde{u}_h$. Based on a posteriori estimates of $\tilde{u}$, the mesh is locally refined and coarsened. This produces the final mesh for the current time step, where the discrete solution u_h is computed. The estimate–adapt process is *not* iterated.
  This strategy works quite well, if the time steps are not too large, such that regions of refinement move too fast.

- **Implicit strategy A:** In every time step starting from the previous time step's triangulation, a mesh is generated using local refinement and coarsening based on a posteriori estimates of a solution which is calculated on the current mesh. This solve–estimate–adapt process is iterated until the estimated error is below the given bound.
  This strategy guarantees that the estimated error is below the given bound. Together with an adaptive control of the time step size, this leads to global (in time) error bounds. If the time step size is not too large, the number of iterations of the solve–estimate–adapt process is usually very small.

- **Implicit strategy B:** In every time step starting from the macro triangulation, a mesh is generated using local refinements based on a posteriori estimates of a solution which is calculated on the current (maybe quite coarse) mesh; no mesh coarsening is needed. This solve–estimate–adapt process is iterated until the estimated error is below the given bound.
  Like implicit strategy A, this strategy guarantees error bounds. As the initial mesh for every time step is very coarse, the number of iterations of the solve–estimate–adapt process becomes quite large, and thus the algorithm might become expensive. On the other hand, a solution on a coarse grid is fast and can be used as a good initial guess for finer grids, which is usually better than using the solution from the old time step.
  Implicit strategy B can also be used with anisotropically refined triangular meshes, see

[29]. As coarsening of anisotropic meshes and changes of the anisotropy direction are still open problems, this implies that the implicit strategy A can not be used in this context.

Figure 3.1 shows one time step using the implicit strategy A. The adaptive algorithm as shown in the flow diagram ensures that the mesh refinement/coarsening is done at least once in each time step, even if the error estimate is below the limit. Nevertheless, the error might be not equally distributed between all elements; for some simplices the local error estimates might be bigger than allowed.

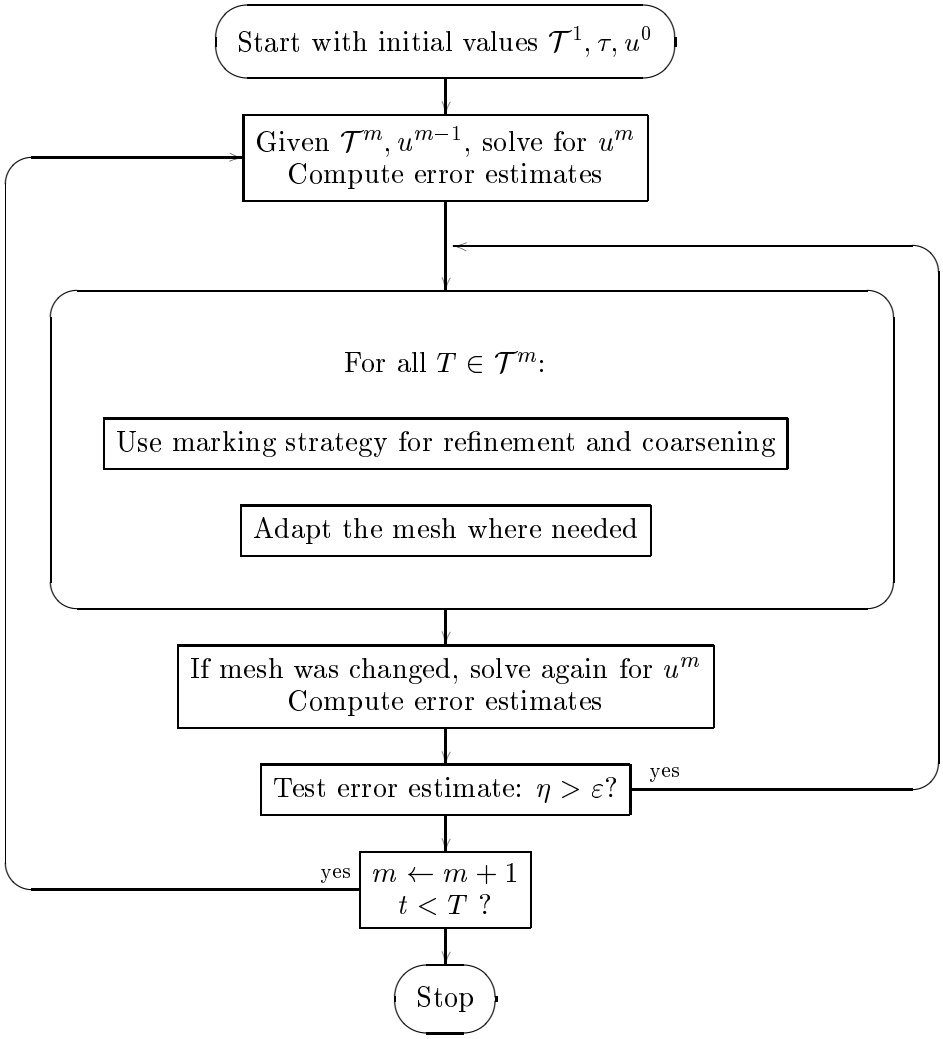

Figure 3.1: Implicit adaptive strategy A

3.4 Adaptive control of the time step size

A posteriori error estimates for parabolic problems include an estimate η_τ of the error that is produced by the time discretization with the actual time step size τ. For Euler time discretization of the heat equation $u_t - \Delta u = f$ with piecewise linear finite elements, for example, $\eta_\tau = c(\|f\|_{\Omega\times(t_m,t_{m+1})} + \|u_h^{m+1} - u_h^m\|_\Omega)$.

When a bound ε is given for the total error produced in a timestep, the widely used strategy is to allow half of this error to be produced by the spacial discretization, and the other half of the error to be produced by the time discretization (equidistribution of error in time and space). The adaptive procedure is now:

- Adjust the time step size such that $\eta_\tau \approx \varepsilon/2$,
- Adapt the mesh such that $\eta \approx \varepsilon/2$.

The adjustment of the time step size can be done via extrapolation techniques known from numerical methods for ordinary differential equations, or iteratively: The algorithm starts from the previous time step size τ_{old} or from an initial guess. A parameter $\delta_1 \in (0,1)$ (usually depending on the order of the time discretization) is used to reduce the step size until the estimate is below the given bound. If the error is smaller than the bound, the step size is enlarged by a factor $\delta_2 > 1$. In this case, the actual time step is not recalculated, only the initial step size for the next time step is changed. Two additional parameters $\theta_1 \in (0,1)$, $\theta_2 \in (0,\theta_1)$ are used to keep the algorithm robust, just like it is done in the equidistribution strategy for mesh adaption. The algorithm starts from the previous time step size τ_{old} or from an initial guess.
If $\delta_1 \approx 1$, consecutive time steps may vary only slightly, but the number of iterations for getting the new accepted time step may increase. Again, as each iteration includes the solution of a discrete problem, this value should be chosen not too large. For a 1st order time discretization scheme, a common choice is $\delta_1 \approx 0.5$, for example.

Algorithm 3.7 *Time step size control*

```
Start with parameters δ1 ∈ (0,1), δ2 > 1, θ1 ∈ (0,1), θ2 ∈ (0,θ1)

τ := τ_old
Solve time step problem and estimate the error
while η_τ > θ1 ε/2 do
   τ := τ * δ1
   Solve time step problem and estimate the error
endwhile
if η_τ ≤ θ2 ε/2 then
   τ := τ * δ2
endif
```

The above algorithm controls only the time step size, but does not show the mesh adaption. There are several possibilities to combine both controls. An inclusion of the grid adaption in every iteration of Algorithm 3.7 can result in a large number of discrete problems to solve, especially if the time step size is reduced more than once. A better procedure is first to do the step size control with the old mesh, then adapt the mesh, and after this check the time error again. In combination with implicit strategy A, this procedure is shown in Figure 3.2.
The adaptive a posteriori approach can be extended to the adaptive choice of the order of the time discretization: Bornemann [8, 9, 10] describes an adaptive variable order time discretization method, combined with implicit strategy B using the extrapolation marking strategy for the mesh adaption.

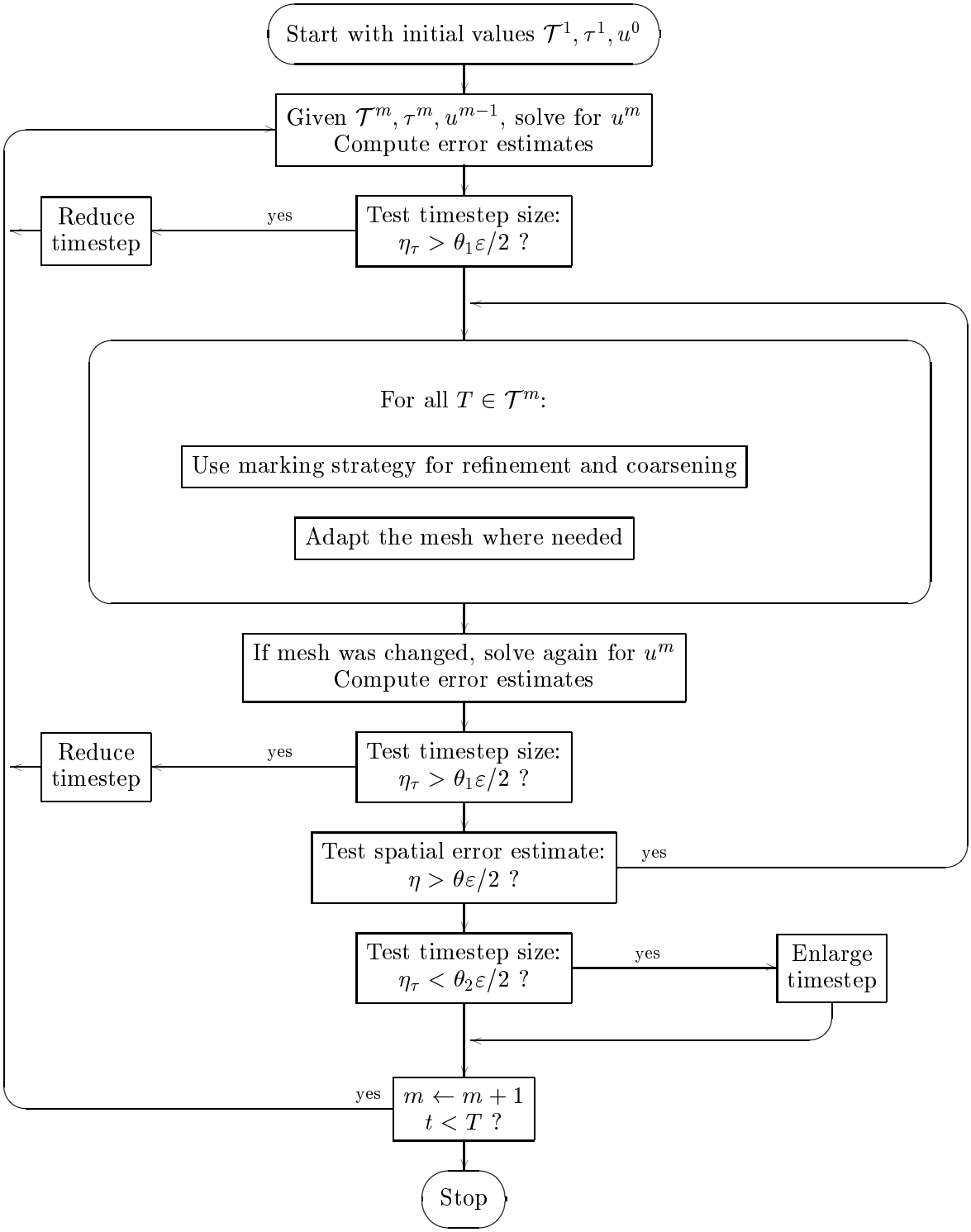

Figure 3.2: Time and space adaptive algorithm

4 The classical Stefan problem

We consider the classical two phase Stefan problem, which describes the heat diffusion and phase change in a pure material. Let $\Omega \subset \mathbf{R}^d$ denote a bounded domain, u the enthalpy and θ the temperature, and $f(t,\cdot) \in H^{-1}(\Omega)$ a given right hand side sufficiently smooth in time.

Problem 4.1 *Two phase Stefan problem*
Find $u \in L_\infty(0,T;L_\infty(\Omega)) \cap W^{1,\infty}(0,T;H^{-1}(\Omega))$ *and* $\theta \in L_\infty(0,T;H_0^1(\Omega))$ *such that*

$$\frac{d}{dt}u - \Delta\theta = f \quad \text{in } H^{-1}(\Omega),$$

with initial condition

$$u(\cdot,0) \;=\; u_0$$

and

$$\theta = \beta(u),$$

where $\beta(s) = \min(s,0) + \max(s-1,0)$, *see Figure 4.1.*

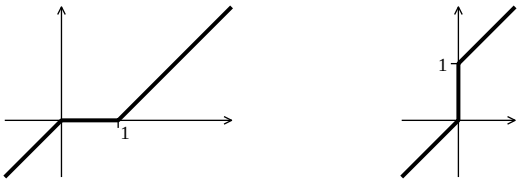

Figure 4.1: Graphs of β and β^{-1}

4.1 Elliptic variational inequalities and the Stefan problem

EVI and minimization problems: We need to cite some results about elliptic variational inequalities. For more information, see e.g. [21, 32].

Theorem 4.1 *Let* $\Omega \subset \mathbf{R}^d$ *bounded,* $V \subset H_0^1(\Omega)$, $a : V \times V \to \mathbf{R}$ *bilinear, symmetric, and elliptic,* $\Phi : \mathbf{R} \to \mathbf{R}$ *convex,* $l \in V^* \supset L_2(\Omega)$, *and*

$$J(v) := \frac{1}{2}a(v,v) - \langle l, v\rangle, \quad \phi(v) := \int_\Omega \Phi(v(x))dx.$$

Then the minimization problem

$$u \in V: \quad J(u) + \phi(u) \le J(v) + \phi(v) \quad \forall v \in V \tag{4.1}$$

has a unique solution and is equivalent to the elliptic variational inequality of second kind

$$u \in V: \quad a(u, v-u) + \phi(v) - \phi(u) \ge \langle l, v-u\rangle \quad \forall v \in V. \tag{4.2}$$

Proof (from [32]): Existence and uniqueness of a solution u for the minimization problem (4.1) follows from the strict convexity of $J + \phi$ and the fact that $J(v) + \phi(v) \to \infty$ as $\|v\| \to \infty$.

Let u be the solution of (4.1). For every $v \in V$ and $0 < s \leq 1$ we have

$$J(u) + \phi(u) \leq J(u + s(v-u)) + \phi(u + s(v-u)),$$

and thus, using the convexity of ϕ,

$$\begin{aligned} 0 &\leq J(u + s(v-u)) - J(u) + \phi(u + s(v-u)) - \phi(u) \\ &\leq J(u + s(v-u)) - J(u) + s(\phi(v) - \phi(u)). \end{aligned}$$

Dividing by s, we get in the limit $s \to 0$

$$0 \leq \langle J'(u), v-u \rangle + \phi(v) - \phi(u) \quad \forall v \in V.$$

Since $a(\cdot,\cdot)$ is symmetric, we have

$$\langle J'(v), w \rangle = a(v,w) - \langle l, w \rangle \quad \forall v, w \in V,$$

and the variational inequality (4.2) follows.

Now, let u be a solution of (4.2). For $v \in V$ we have

$$J(v) + \phi(v) - J(u) - \phi(u) = \frac{1}{2}(a(v,v) - a(u,u)) + \phi(v) - \phi(u) - \langle l, v-u \rangle.$$

Using the identity

$$a(v,v) = a(u+v-u, u+v-u) = a(u,u) + 2a(u, v-u) + a(u-v, u-v),$$

we get

$$J(v) + \phi(v) - J(u) - \phi(u) = a(u, v-u) + \phi(v) - \phi(u) - \langle l, v-u \rangle + \frac{1}{2}a(u-v, u-v).$$

Since (4.2) and $a(u-v, u-v) \geq 0$, we get that u solves (4.1). □

EVI and subdifferentials: For a function $\phi : V \to \mathbf{R}$, we define the *subdifferential* of ϕ at $u \in V$ by

$$\partial\phi(u) := \{u^* \in V^* \mid \phi(v) - \phi(u) \geq \langle u^*, v-u \rangle \text{ for all } v \in V\}.$$

Defining $A \in L(V, V^*)$ by $\langle Av, w \rangle := a(v,w)$ for all $v, w \in V$, the EVI (4.2) reads

$$\langle Au, v-u \rangle + \phi(v) - \phi(u) \;\geq\; \langle l, v-u \rangle \qquad \forall v \in V,$$

or

$$\phi(v) - \phi(u) \;\geq\; \langle l - Au, v-u \rangle \qquad \forall v \in V.$$

By definition of $\partial\phi(u)$, we have

$$l - Au \in \partial\phi(u). \tag{4.3}$$

For smooth $l - Au$, also the pointwise inclusion (almost everywhere)

$$(l - Au)(x) \in \partial\Phi(u(x)) \quad \text{for a. e. } x \in \Omega$$

holds for the solution u of (4.2), where the subdifferential of Φ is defined as

$$\partial\Phi(u(x)) := \{z \in \mathbf{R} \mid \Phi(y) - \Phi(x) \geq z(y-x) \text{ for all } y \in \mathbf{R}\}.$$

An EVI equivalent to the Stefan problem: We return to the Stefan problem

$$\frac{d}{dt}u - \Delta\theta = f \quad \text{in } H^{-1}(\Omega), \text{ with}$$

$$\theta = \beta(u) \quad \Leftrightarrow \quad u \in \beta^{-1}(\theta),$$

where $\beta(s) = \min(s, 0) + \max(s - 1, 0)$, see Figure 4.1, and suitable initial and boundary conditions. An implicit Euler discretization in time leads to

$$\frac{u^m - u^{m-1}}{\tau} - \Delta\theta^m = f^m,$$

$$u^m = \tau\Delta\theta^m + u^{m-1} + \tau f^m.$$

With $A := -\tau\Delta$ and $l^m := u^{m-1} + \tau f^m$ this reads

$$u^m = l^m - A\theta^m.$$

Using the relation $u \in \beta^{-1}(\theta)$, we get

$$l^m - A\theta^m \in \beta^{-1}(\theta^m).$$

Now, β^{-1} is just the subdifferential of the convex function

$$\Phi(y) = \frac{1}{2}y^2 + \max(y, 0), \tag{4.4}$$

which results in the inclusion

$$(l^m - A\theta^m)(x) \in \partial\Phi(\theta^m(x)) \quad \text{a.e., or}$$

$$l^m - A\theta^m \in \partial\phi(\theta^m).$$

Recalling (4.3), we see that the temperature θ^m is the solution of the minimization problem (4.1) or the elliptic variational inequality (4.2) with the special Φ from (4.4).
For solving the timedependent Stefan problem, we have to solve such an elliptic variational inequality in every time step.

4.2 A multigrid method for elliptic variational inequalities

Iterative solution methods like (linear or nonlinear) SOR are usually non-optimal in the sense, that their convergence gets slower when the discretizations get finer. This is caused by the fact that the contraction constant of the method (spectral radius of the corresponding matrix) tends to one when the mesh size tends to zero.
Using multilevel techniques, the design of some 'optimal' iterative solution methods for linear and nonlinear problems is possible. Linear multilevel solvers (multigrid solvers or multilevel preconditioners) are quite standard now, but most nonlinear problems need specialized methods. Kornhuber [41, 43] describes a finite element multigrid method for the solution of elliptic variational inequalities, where Φ is a piecewise quadratic function. We will motivate and describe this method. Similar methods were presented in [35], [36].

First, we need some notations used for the multilevel finite element method (we use the same as [43]).
Let the domain $\Omega \subset \mathbf{R}^2$ be bounded by a polygon, and let $\mathcal{T}^0$ be a given macro–triangulation of Ω. We use a sequence of nested triangulations $\mathcal{T}^1, \ldots, \mathcal{T}^j$ by successive refinements of $\mathcal{T}^0$. We want to solve the time discretized Stefan problem on the finest triangulation $\mathcal{T}^j$. Let V_k, $k = 0, \ldots, j$ be the corresponding finite element spaces of piecewise linear functions. $N_k = \{p_i,\, i = 1, \ldots, n_k\}$ denote the non-Dirichlet vertices of $\mathcal{T}^k$, and $\Lambda_k = \{\lambda_{p_i}^{(k)},\, i = 1, \ldots, n_k\}$, $k = 0, \ldots, j$, the corresponding nodal basis of V_k. By construction, we have $V_0 \subset V_1 \subset \ldots \subset V_j$ and $n_0 < n_1 < \ldots < n_j$.

Multigrid for linear elliptic problems: First of all, we give a short description of multigrid methods for the linear elliptic problem

$$u \in H_0^1(\Omega): \quad \frac{1}{2}\, a(u,u) - \langle f, u\rangle \;\leq\; \frac{1}{2}\, a(v,v) - \langle f, v\rangle \quad \forall v \in H_0^1(\Omega)$$
$$\Leftrightarrow \qquad u \in H_0^1(\Omega): \quad a(u,v) = \langle f, v\rangle \quad \forall v \in H_0^1(\Omega).$$

Finite element discretization leads to the linear system of equations

$$u_h \in V_j: \quad a(u_h, v_h) \;=\; \langle f, v_h\rangle \quad \forall v_h \in V_j.$$

One iteration of a V-cycle multigrid solver for this problem, starting with $u_h^\nu \in V_j$ and producing $u_h^{\nu+1}$, is presented in the Algorithm 4.1. The general idea of multilevel methods is to reduce high frequency components of the error by fine level smoothing and low frequency error components by coarse grid correction. The multigrid iteration below uses one pre–smoothing iteration per level and no post–smoothing. The smoothing operator $M_k(a, f)$ consists usually of one or more iterations of a standard iterative solver for the linear system, like Gauss-Seidel, Jacobi, etc., which has some smoothing properties.

Algorithm 4.1 *Linear V–cycle multigrid iteration*

`fine grid smoothing:` $v^{(j)} := M_j(a,f)(u_h^\nu)$
`residual:` $r^{(j)} = f - a(v^{(j)}, \cdot)$
$a^{(j)} := a$
`coarse grid correction:`
  `for` $k = j-1$ `to` 0 `step` -1 `do`
    `canonical restrictions:` $a^{(k)} := a^{(k+1)}|_{V_k \times V_k}$, $r^{(k)} := r^{(k+1)}|_{V_k}$
    `coarse grid smoothing:` $v^{(k)} := M_k(a^{(k)}, r^{(k)})(0)$
    `update residual:` $r^{(k)} := r^{(k)} - a^{(k)}(v^{(k)}, \cdot)$
  `enddo`
  `for` $k = 0$ `to` $j-1$ `do`
    `canonical prolongation:` $v^{(k+1)} = v^{(k+1)} + v^{(k)}$
  `enddo`
`new iterate:` $u_h^{\nu+1} := v^{(j)}$

The restrictions of the bilinear form and of the residual and the prolongation of the coarse grid correction use the formulas presented in Sections 2.2 and 2.4. The general structure of the multigrid algorithm for elliptic variational inequalities will be just the same, see Algorithm 4.3.

Multigrid for elliptic variational inequalities: The aim now is so solve the discrete optimization problem

$$u_h \in V_j : \quad J_j(u_h) + \phi_j(u_h) \;\le\; J_j(v_h) + \phi_j(v_h) \quad \forall v_h \in V_j, \tag{4.5}$$

where the nonlinear functional $\phi(v) = \int_\Omega \Phi(v(x))$ is replaced by the discrete (lumped) nonlinear functional

$$\phi_j(v_h) \;:=\; \sum_{i=1}^{n_j} \Phi(v_h(p_i)) \int_\Omega \lambda_{p_i}^{(j)}\, dx\,, \tag{4.6}$$

and J is replaced by an approximation using a lumped L_2 scalar product:

$$J_j(v_h) := \frac{1}{2}\, a(v_h, v_h) - \langle l, v_h \rangle_j, \tag{4.7}$$

$$\langle v_h, w_h \rangle_j := \sum_{i=1}^{n_j} v_h(p_i)\, w_h(p_i) \int_\Omega \lambda_{p_i}^{(j)}\, dx\,, \tag{4.8}$$

For the sake of clearness and simplicity, we describe the method for the Stefan problem with Φ from (4.4),

$$\Phi(y) = \begin{cases} \frac{1}{2}y^2 & y \le 0\,, \\ \frac{1}{2}y^2 + y & y \le 0\,. \end{cases} \tag{4.9}$$

The articles [42, 43] describe the method for some more general situations of two or more phases, where Φ is a general piecewise quadratic and convex function.
Existence and uniqueness of a solution u_h of the discrete problem (4.5) follow just as in the nondiscrete case. Convergence of the discrete solution u_h to the solution u of the continuous problem (4.1) is known [22, 32], including a priori error estimates.

The general idea of the multilevel solver is to successively minimize the energy functional in the one dimensional subspaces, which are spanned by the nodal basis functions from all levels:

$$V_i^k := \mathrm{span}\{\lambda_{p_i}^{(k)}\}, \quad i = 1, \ldots, n_k,\ k = 0, \ldots, j.$$

Starting from $u_h^\nu \in V_j$, the following algorithm describes one iteration of a 'global' relaxation procedure, producing the next iterate $u_h^{\nu+1}$. It consists of one minimization in each of the above subspaces, using damping parameters $\omega_{k,i} \in [0, 1]$.

Algorithm 4.2 *(Nonlinear multilevel relaxation)*

$w_h := u_h^\nu$
`for` $k = j, \ldots, 0$ `step` -1 `do`
  `for` $i = 1, \ldots, n_k$ `do`
    $\bar v_h \in V_i^k : \quad J_j(w_h + \bar v_h) + \phi_j(w_h + \bar v_h) \;\le\; J_j(w_h + v_h) + \phi_j(w_h + v_h) \quad \forall v_h \in V_i^k$
    $w_h := w_h + \omega_{k,i} \bar v_h, \quad \omega_{k,i} \in [0, 1]$
  `enddo`
`enddo`
$u_h^{\nu+1} := w_h$

The following theorem states the convergence of such successive minimizations even when damped minimizations are used in subspaces which are generated by coarse level basis functions:

Theorem 4.2 *For any initial iterate $u_h^0 \in V_j$ and any sequence of damping parameters with the property that no damping is used at the finest level, i. e.*

$$\omega_{j,i} = 1, \quad i = 1, \ldots, n_j \,,$$

the sequence of iterates $(u_h^\nu)_{\nu \geq 0}$ produced by Algorithm 4.2 converges to the solution u_h of the discrete problem (4.5).

The proof uses the global convergence of the leading nondamped relaxation on the finest level (which is just a nonlinear Gauss-Seidel relaxation) and the monotonicity of the local corrections: $(J_j + \phi_j)(w_h)$ does not increase during the iteration. It can be found in [41].

On the finest level $k = j$, the minimization in the one dimensional subspace V_i^j

$$\bar{v}_h \in V_i^j : \quad J_j(w_h + \bar{v}_h) + \phi_j(w_h + \bar{v}_h) \;\leq\; J_j(w_h + v_h) + \phi_j(w_h + v_h) \quad \forall v_h \in V_i^j$$

is equivalent (compare (4.3)) to the inclusion:

$$\bar{v}_h \in V_i^j : \quad l - a(w_h + \bar{v}_h, \cdot) \;\in\; \partial\phi_j(w_h + \bar{v}_h) \,.$$

With $m_\mu := \int_\Omega \lambda_{p_\mu}^{(j)}$, $a_{\mu\nu} := a(\lambda_{p_\mu}^{(j)}, \lambda_{p_\nu}^{(j)})$, $w_\mu := w_h(p_\mu)$, and $l_\mu := l(p_\mu)$, $\mu, \nu = 1, \ldots, n_j$, this corresponds to the scalar nonlinear inclusion

$$\bar{z} \in \mathbf{R} : \quad l_i \, m_i - \sum_{\mu=1,\ldots,n_j,\, \mu \neq i} a_{\mu i} \, w_\mu \;\in\; a_{ii}(w_i + \bar{z}) + m_i \, \partial\Phi(w_i + \bar{z}) \,.$$

Using $b := l_i \, m_i - \sum_{\mu \neq i} a_{\mu i} \, w_\mu$, the solution $\bar{z}$ is given explicitely by the nonlinear Gauss–Seidel step

$$w_i + \bar{z} \;=\; \begin{cases} b/(a_{ii} + m_i) & b < 0, \\ 0 & b \in [0, m_i], \\ (b - m_i)/(a_{ii} + m_i) & b > m_i \,. \end{cases} \tag{4.10}$$

For coarser levels $k < j$, the basis functions $\lambda_i^{(k)}$ are nonzero at more than one fine-level vertex, and the minimization in the one dimensional subspace V_i^k needs the solution of a system of nonlinear equations. The exact solution can no longer be calculated as easily as in the case of the finest level. So, this general nonlinear multilevel relaxation is not easy to implement and solve.
The idea for a multilevel procedure, which is easy to implement, is to minimize the functional only in a subset of the one dimensional subspace, where the solution is easy to compute.
We will see afterwards, that the final multigrid methods are equivalent to the nonlinear multilevel relaxation with damping at the coarser levels.

Depending on a current iterate u_h^ν, we define *discrete phases* N_j^- and N_j^+ by

$$N_j^-(u_h^\nu) := \{p \in N_j \mid u_h^\nu(p) < 0\} \,, \quad N_j^+(u_h^\nu) := \{p \in N_j \mid u_h^\nu(p) > 0\} \,.$$

All other nodes belong to the set of *critical nodes* $N_j^\bullet$

$$N_j^\bullet(u_h^\nu) := N_j \setminus (N_j^-(u_h^\nu) \cup N_j^+(u_h^\nu)) \,.$$

It can be shown that these discrete phases converge during the nonlinear multilevel relaxation:

Theorem 4.3 *If the discrete problem (4.5) is non–degenerate, i. e.*

$$p \in N_j^\bullet(u_h) \quad \Rightarrow \quad \langle l, \lambda_p^{(j)} \rangle - a(u_h, \lambda_p^{(j)}) \ \in \ \operatorname{int} \partial\phi_j(u_h)(\lambda_p^{(j)}) \,,$$

then the discrete phases of the iterates u_h^ν converge to the discrete phases of u_h. There esists some $\nu_0 \geq 0$, such that the discrete phases N_j^-, N_j^+, and $N_j^\bullet$ of u_h and those of u_h^ν, $\nu > \nu_0$, coincide.

The proof can be found in [43].

As long as the discrete phases of a function w_h do not change, the functional $\phi_j(w_h)$ is quadratic:

$$\begin{aligned}
\phi_j(w_h) &= \sum_{i=1}^{n_j} \Phi(w_h(p_i)) \int_\Omega \lambda_{p_i}^{(j)} \, dx \\
&= \sum_{p_i \in N_j^-} \frac{1}{2} w_h(p_i)^2 \int_\Omega \lambda_{p_i}^{(j)} \, dx + \sum_{p_i \in N_j^+} (\frac{1}{2} w_h(p_i)^2 + w_h(p_i)) \int_\Omega \lambda_{p_i}^{(j)} \, dx \\
&= \frac{1}{2} \sum_{i=1}^{n_j} w_h(p_i)^2 \int_\Omega \lambda_{p_i}^{(j)} \, dx + \sum_{p_i \in N_j^+} w_h(p_i) \int_\Omega \lambda_{p_i}^{(j)} \, dx \\
&=: \frac{1}{2} b_{w_h}(w_h, w_h) - f_{w_h}(w_h) \,.
\end{aligned}$$

This defines a bilinear form b_{w_h} and a linear functional f_{w_h}, which both depend on the current discrete phases, in general (in our special situation, only f_{w_h} depends on the phases).
The coarse grid corrections of the multilevel procedure will keep the discrete phases of the current solution unchanged and solve the problem with such simplified equations. In order to keep the discrete phases unchanged, we define lower and upper obstacle functions $\underline{\varphi}_j^\nu$ and $\bar{\varphi}_j^\nu$ by

$$\underline{\varphi}_j^\nu(p) := \begin{cases} -\infty & u_h^\nu(p) < 0, \\ 0 & u_h^\nu(p) \geq 0, \end{cases} \qquad \bar{\varphi}_j^\nu(p) := \begin{cases} 0 & u_h^\nu(p) \leq 0, \\ \infty & u_h^\nu(p) > 0, \end{cases} \qquad p \in N_j \,,$$

and the closed convex subset $\mathcal{K}_j^\nu \subset V_j$ of functions with the same discrete phases as u_h^ν is given by

$$\mathcal{K}_j^\nu = \{ v_h \in V_j \mid \underline{\varphi}_j^\nu(p) \leq v_h(p) \leq \bar{\varphi}_j^\nu(p) \text{ for all } p \in N_j \}.$$

After minimizing the energy in the fine level subspaces V_i^j, $i = 1, \ldots, n_j$, the discrete phases and obstacles are computed, and coarse level minimizations are only done in a convex subset $D_i^k \subset V_i^k \cap \mathcal{K}_j^\nu$:

$$\bar{v}_h \in D_i^k : \quad J_j(w_h + \bar{v}_h) + \phi_j(w_h + \bar{v}_h) \ \leq \ J_j(w_h + v_h) + \phi_j(w_h + v_h) \quad \forall v_h \in D_i^k \,.$$

The generation of these subsets D_i^k will be described below. This minimization can be done very efficiently and is equivalent to a truncated linear Gauss-Seidel smoother on the coarser grids. Let $\tilde{z}\lambda_{p_i}^{(k)} \in V_i^k$ be the solution of the unconstrained linearized problem, which can be computed by a linear Gauss-Seidel step similar to (4.10). For given D_i^k we define $\underline{D}_i^k, \bar{D}_i^k \in \mathbf{R}$ by $D_i^k = \{ z\lambda_{p_i}^{(k)} \mid z \in [\underline{D}_i^k, \bar{D}_i^k] \}$. Then the solution of the problem in D_i^k is given by the truncation

$$v_h \ := \ \bar{z}\lambda_{p_i}^{(k)} \quad \text{where} \quad \bar{z} \ := \ \min(\bar{D}_i^k, \max(\underline{D}_i^k, \tilde{z})) \,.$$

A computation of the set $V_i^k \cap \mathcal{K}_j^\nu$ is not easy to implement at the coarse levels, but a very efficient and simple (but non-optimal) restriction of the obstacles to the coarser finite element spaces is possible. This is used to compute the subsets D_i^k. We present the simple idea in case of the upper obstacle $\bar{\varphi}^\nu$. Like always during restrictions and prolongations in multigrid methods, such a restriction can be built out of simple atomic restrictions, which describe the local situation where one edge e of the triangulation $\mathcal{T}_k$ is bisected into two edges of level $k+1$. This defines a restriction operator $\bar{R}_e : V_{k+1} \to V_{k+1}$. The restriction $\bar{R}_{k+1}^k : V_{k+1} \to V_k$ is then defined by

$$\bar{R}_{k+1}^k := I_k \circ \bar{R}_{e_1} \circ \cdots \circ \bar{R}_{e_m} , \tag{4.11}$$

where I_k is the Lagrange interpolation operator to V_k, and $e_1, \ldots, e_m$ are the edges which were bisected during the refinement process from $\mathcal{T}_k$ to $\mathcal{T}_{k+1}$. The actual restriction $\bar{R}_{k+1}^k$ may depend on the order of these edges.

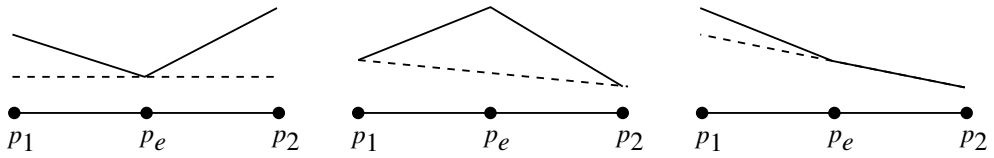

Figure 4.2: Restriction $\bar{R}_e$ of the upper obstacle: Fine grid obstacles are depicted by solid lines and coarse grid obstacles by dashed lines.

Looking at one single edge e, we have a one dimensional situation. Let p_1, p_2 denote the vertices of e, and p_e the midpoint of e. The coarse level obstacle $\bar{\varphi}_k^\nu$ is linear on this edge, and should be smaller or equal to the fine level obstacle $\bar{\varphi}_{k+1}^\nu$, which is piecewise linear on the two half edges. The idea of this restriction is simply to change the values at the vertices p_1, p_2 in a simple way such that the coarse grid function $\bar{\varphi}_k^\nu$ is bounded by the fine grid function. Figure 4.2 shows the three principal cases. The solid line denotes the function v, while the dashed line denotes $I_k \bar{R}_e v$. If $v(p_e) \leq \min(v(p_1), v(p_2))$, then $I_k R_e v$ is constant equal to $v(p_e)$. If $v(p_e) > \frac{1}{2}(v(p_1) + v(p_2))$, then $I_k \bar{R}_e v$ is just the interpolation of v. Otherwise, $I_k \bar{R}_e v$ is just the continuation of the lower half. Formulas for the restricted values at p_1, p_2 follow quite easily.
These restrictions $\bar{R}_{k+1}^k$ and $\underline{R}_{k+1}^k$ of the lower and upper obstacles are used on level k to bound the coarse grid correction and describe the sets D_i^k, $i = 1, \ldots, n_k$. These sets can not become empty, but it may happen that $D_i^k = \{0\}$ for points p_i near the interface, and thus no correction will be allowed at these points.
Let us denote the fine-grid minimization (smoothing) operation starting with u_h^ν by $M_j(u_h^\nu)$ and the coarse-grid minimization (smoothing) operation using a quadratic functional $a^{(k)}(\cdot,\cdot)$, a right hand side $r^{(k)}(\cdot)$ and upper and lower obstacles $\underline{\psi}^{(k)}$, $\bar{\psi}^{(k)}$, starting with zero, by $\bar{M}_k[a^{(k)}, r^{(k)}, \underline{\psi}^{(k)}, \bar{\psi}^{(k)}](0)$:

Subroutine $M_j(u_h^\nu)$:

$w_h := u_h^\nu$
for $i = 1$ **to** n_j **do**
$\bar{v}_h \in V_i^{(j)}$: $J_j(w_h + \bar{v}_h) + \phi_j(w_h + \bar{v}_h) \leq J_j(w_h + v_h) + \phi_j(w_h + v_h) \quad \forall v_h \in V_i^j$
$w_h := w_h + \bar{v}_h$
enddo
return w_h

Subroutine $\tilde{M}_k[a^{(k)}, r^{(k)}, \underline{\psi}^{(k)}, \bar{\psi}^{(k)}](0)$:

$w_h := 0$
$J_k(v_h) := \frac{1}{2}a^{(k)}(v_h, v_h) - \langle r^{(k)}, v_h \rangle$
for $i = 1$ **to** n_k **do**
$D_i^{(k)} := \{v_h \in V_i^k \mid \underline{\psi}^{(k)}(p_i) \le v_h(p_i) \le \bar{\psi}^{(k)}(p_i)\}$
$\bar{v}_h \in D_i^{(k)}: \quad J_k(w_h + \bar{v}_h) \le J_k(w_h + v_h) \quad \forall v_h \in D_i^{(k)}$
$w_h := w_h + \bar{v}_h$
enddo
return w_h

Using these subroutines one iteration of the multilevel algorithm reads as follows:

Algorithm 4.3 *Standard monotone multigrid method STDKH*

fine grid smoothing: $\tilde{u}_h^\nu := M_j(u_h^\nu)$
discrete phases: $N_j^-(\tilde{u}_h^\nu)$, $N_j^+(\tilde{u}_h^\nu)$, $N_j^\bullet(\tilde{u}_h^\nu)$
local linearization: $\tilde{a} := a + b_{\tilde{u}_h^\nu}$, $\tilde{l} := l + f_{\tilde{u}_h^\nu}$
coarse grid correction:
initialize:
bilinear form: $a^{(j)} = \tilde{a}$,
residual: $r^{(j)} = \tilde{l} - \tilde{a}(\tilde{u}_h^\nu, \cdot)$,
defect obstacles: $\underline{\psi}^{(j)} := \underline{\varphi}_j^\nu - \tilde{u}_h^\nu$, $\bar{\psi}^{(j)} := \bar{\varphi}_j^\nu - \tilde{u}_h^\nu$
for $k = j-1$ **to** 0 **step** -1 **do**
canonical restrictions: $a^{(k)} := a^{(k+1)}|_{V_k \times V_k}$, $r^{(k)} := r^{(k+1)}|_{V_k}$
obstacle restrictions: $\underline{\psi}^{(k)} := \underline{R}_{k+1}^k \underline{\psi}^{(k+1)}$, $\bar{\psi}^{(k)} := \bar{R}_{k+1}^k \bar{\psi}^{(k+1)}$
coarse grid smoothing: $v^{(k)} := \tilde{M}_k[a^{(k)}, r^{(k)}, \underline{\psi}^{(k)}, \bar{\psi}^{(k)}](0)$
update:
residual: $r^{(k)} := r^{(k)} - a^{(k)}(v^{(k)}, \cdot)$
obstacles: $\underline{\psi}^{(k)} := \underline{\psi}^{(k)} - v^{(k)}$, $\bar{\psi}^{(k)} := \bar{\psi}^{(k)} - v^{(k)}$
enddo
for $k = 0$ **to** $j-1$ **do**
canonical prolongation: $v^{(k+1)} = v^{(k+1)} + v^{(k)}$
enddo
new iterate: $u_h^{\nu+1} := \tilde{u}_h^\nu + v^{(j)}$

Using a multigrid methods's wording, Algorithm 4.3 as stated above is a V–cycle which uses one pre–smoothing iteration per level and no post–smoothing. Similar algorithms with more pre– and post–smoothing iterations, W–cycles, etc. are straightforward.
Kornhuber [41, 43] proves the following convergence result:

Theorem 4.4 *The standard monotone multigrid method described in Algorithm 4.3 is globally convergent.*
Assume that the discrete problem is non–degenerate, then the discrete phases of the iterates $(u_h^\nu)_{\nu \ge 0}$ *converge to the discrete phases of the discrete solution* u_h *and the error estimate*

$$\|u_h - u_h^{\nu+1}\| \ \le \ (1 - c(j+1)^{-4})\|u_h - u_h^\nu\|$$

holds, if ν *is large enough.*

Convergence follows from Theorem 4.2, as the standard monotone multigrid method is a nonlinear multilevel relaxation. The proof of the multigrid convergence rate is based on the convergence of discrete phases, Theorem 4.3, and some general multigrid convergence results.

The proven convergence rate of the multilevel method is not independent of the mesh size, but $(1 - c(j+1)^{-4})$ depends only logarithmically on $h \approx 2^{-j}$.

If the set of critical nodes $N_j^\bullet(\tilde{u}_h^\nu)$ is not empty, then the restrictions (4.11) of the upper and lower obstacles forbid any coarse grid corrections in subspaces $V_i^{(k)}$ where $N_j^\bullet \cap \text{int supp } \lambda_{p_i}^{(k)} \neq \emptyset$. This leads to inefficient multilevel convergence. For this reason, Kornhuber additionally presents a modification of his algorithm, which is equivalent to a modification ('truncation') of those coarse level basis functions that are non–zero at critical nodes.

Algorithm 4.4 *Truncated monotone multigrid method TRCKH*

Modifications of the standard monotone multigrid method, Algorithm 4.3:

modified restrictions of the bilinear form and of the residual:
treat all entries from the actual critical nodes $N_j^\bullet(\tilde{u}_h^\nu)$ as zero
modified restrictions of the upper (and lower) defect obstacle:
treat all entries from the actual critical nodes $N_j^\bullet(\tilde{u}_h^\nu)$ as ∞ $(-\infty)$
modified prolongations of the corrections:
prolongate zero to all critical nodes

Convergence of this modified algorithm is proved, too, but (for technical reasons) the proven convergence rate is worse than for the standard multigrid method. Numerical results show that mesh–independent convergence is observed which reaches the convergence rate of multigrid methods for the linear elliptic problem [42]. Thus, this truncated monotone multigrid method seems to be an 'optimal' solver.
However, the actual efficiency depends a lot on an efficient implementation of every part of the method. For small– and medium–scale problems with not too small error criterion, the SOR solver described below is usually faster because of its simple structure, but for large–scale problems the multigrid method with its $O(|N_j|)$ work bound gets superior.

4.3 Nonlinear SOR solvers for the Stefan problem

A nonlinear Gauss–Seidel solver is already included in the nonlinear multilevel relaxation Algorithm 4.2. With damping parameters $\omega_{k,i} = 0$, $k < j$, only relaxations at the finest level are allowed. This results just in a nonlinear Gauss–Seidel solver. Convergence follows directly from Theorem 4.2.
Even for fine level damping with a fixed $\omega_{j,i} = \omega \in (0, 1]$, convergence follows easily.
In practice, the SOR solver (with $\omega \in (0, 2)$) converges quite well and is very easy to implement. A good choice for ω is the optimal value for the (linear) discretized heat equation, see [51].
Elliott [22] proved convergence of an SOR solver with arbitrary $\omega_{j,i} = \omega \in (0, 2)$. It uses a modification of the above algorithm: If the overrelaxation step leads to a local change of phase, then the exact minimizer ($\omega_{j,i} = 1$) is used at this point. Thus, overrelaxation is used only where no phase changes occur.

4.4 Adaptive method

The mesh and time step adaption process is based on the following a posteriori error estimate. Here, the triangulation of timestep n is $\mathcal{M}^n$, simplices are denoted by S, and the discrete solutions by U^n resp. Θ^n.
For all $t^m \in [0,T]$,

$$
\begin{aligned}
&\|u-\hat{U}\|_{L_\infty(0,t^m;H^{-1}(\Omega))} + \|\beta(u)-\hat{\Theta}\|_{L_2(0,t^m;L_2(\Omega))} \\
&\leq \|u_0 - U^0\|_{H^{-1}(\Omega)} \\
&\quad + C_1 \sum_{n=1}^{m} \tau^n \left(\frac{d+1}{4} \sum_{S\in\mathcal{M}^n} h_S \|[\nabla\Theta^n]\|^2_{L_2(\partial S)} \right)^{1/2} \\
&\quad + C_2 \sum_{n=1}^{m} \tau^n \left(\sum_{S\in\mathcal{M}^n} h_S^2 \|\Pi^n f^n - U_t^n\|^2_{L_2(S)} \right)^{1/2} \\
&\quad + \sum_{n=1}^{m} \tau^n \|f^n - \Pi^n f^n\|_{H^{-1}(\Omega)} \\
&\quad + C \sum_{n=1}^{m} \tau^n \left(\sum_{S\in\mathcal{M}^n} h_S^4 \|\nabla(\Pi^n f^n - U_t^n)\|^2_{L_2(S)} \right)^{1/2} \\
&\quad + \sum_{n=1}^{m} \tau^n \|\nabla(\beta(U^n) - \Pi^n\beta(U^n))\|_{L_2(\Omega)} \\
&\quad + \sum_{n=1}^{m} \|[U^{n-1}]\|_{H^{-1}(\Omega)} \\
&\quad + \sum_{n=1}^{m} \int_{I^n} \|f - f^n\|_{H^{-1}(\Omega)} \\
&\quad + \frac{1}{\sqrt{2}} \left(\sum_{n=1}^{m} \tau^n \|\tau^n U_t\|^2_{L_2(\Omega)} \right)^{1/2}.
\end{aligned}
$$

For derivation and proof, see [54]. This leads to local estimator terms (which are just the localized squares of the terms from above):

$$
\begin{aligned}
E_0^n(S) &:= \begin{cases} |S| \|\Pi^0 u_0\|^2_{L_\infty(S)} & \text{if } S\cap\Sigma^0 \neq \emptyset, \\ |S| h_s \|[\nabla\Theta^0]\|^2_{L_2(\partial S)} & \text{if } S\cap\Sigma^0 = \emptyset, \end{cases} \\
E_1^n(S) &:= C_1^2 \frac{d+1}{4} h_S \|[\nabla\Theta^n]\|^2_{L_2(\partial S)}, \\
E_2^n(S) &:= C_2^2 h_S^2 \|\Pi^n f^n - U_t^n\|^2_{L_2(S)}, \\
E_3^n(S) &:= d_\Omega^2 \|f^n - \Pi^n f^n\|^2_{L_2(S)}, \\
E_4^n(S) &:= C^2 h_S^4 \|\nabla(\Pi^n f^n - U_t^n)\|^2_{L_2(S)}, \\
E_5^n(S) &:= \|\nabla(\beta(U^n) - \Pi^n\beta(U^n))\|^2_{L_2(S)}, \\
E_6^n(S) &:= \left(\frac{d_\Omega}{\tau^n}\right)^2 \|[U^{n-1}]\|^2_{L_2(S)}, \\
E_7^n(S) &:= d_\Omega^2 \|f - f^n\|^2_{L_\infty(I^n,L_2(S))},
\end{aligned}
$$

$$E_8^n(S) \;:=\; \frac{1}{2}\|\tau^n U_t\|^2_{L_2(S)}\,.$$

The local and global estimators for mesh adaption, coarsening and time step adaption are

$$\begin{aligned}
\eta_S &:= (E_1^n(S) + E_2^n(S) + E_3^n(S) + E_4^n(S) + E_5^n(S))^{1/2}\,,\\
\eta_{c,S} &:= E_6^n(S)^{1/2},\\
\eta &:= \left(\sum_{S\in\mathcal{M}^n} \eta_S^2 + \eta_{c,S}^2\right)^{1/2},\\
\eta_\tau &:= \left(\sum_{S\in\mathcal{M}^n} TE_7^n(S) + E_8^n(S)\right)^{1/2}.
\end{aligned}$$

With a given tolerance ε for the total error, we try to equidistribute the error in time also, and in every timestep the mesh size and time step size are adjusted such that

$$\eta \;\le\; \frac{\varepsilon}{\theta_{\text{space}}T} \quad\text{and}\quad \eta_\tau \;\le\; \frac{\varepsilon}{\theta_{\text{time}}\sqrt{T}}.$$

Together with an adaption of the initial mesh, such that

$$\left(\sum_{S\in\mathcal{M}^0} E_0(S)\right)^{1/2} \;\le\; \frac{\varepsilon}{\theta_{\text{initial}}},$$

and parameters $\theta_{\text{initial}} + \theta_{\text{space}} + \theta_{\text{time}} \le 1$, this assures that the total error, summed up over all time steps, is below the given bound ε.
The final adaptive algorithm is shown in the flow diagram Figure 4.3.

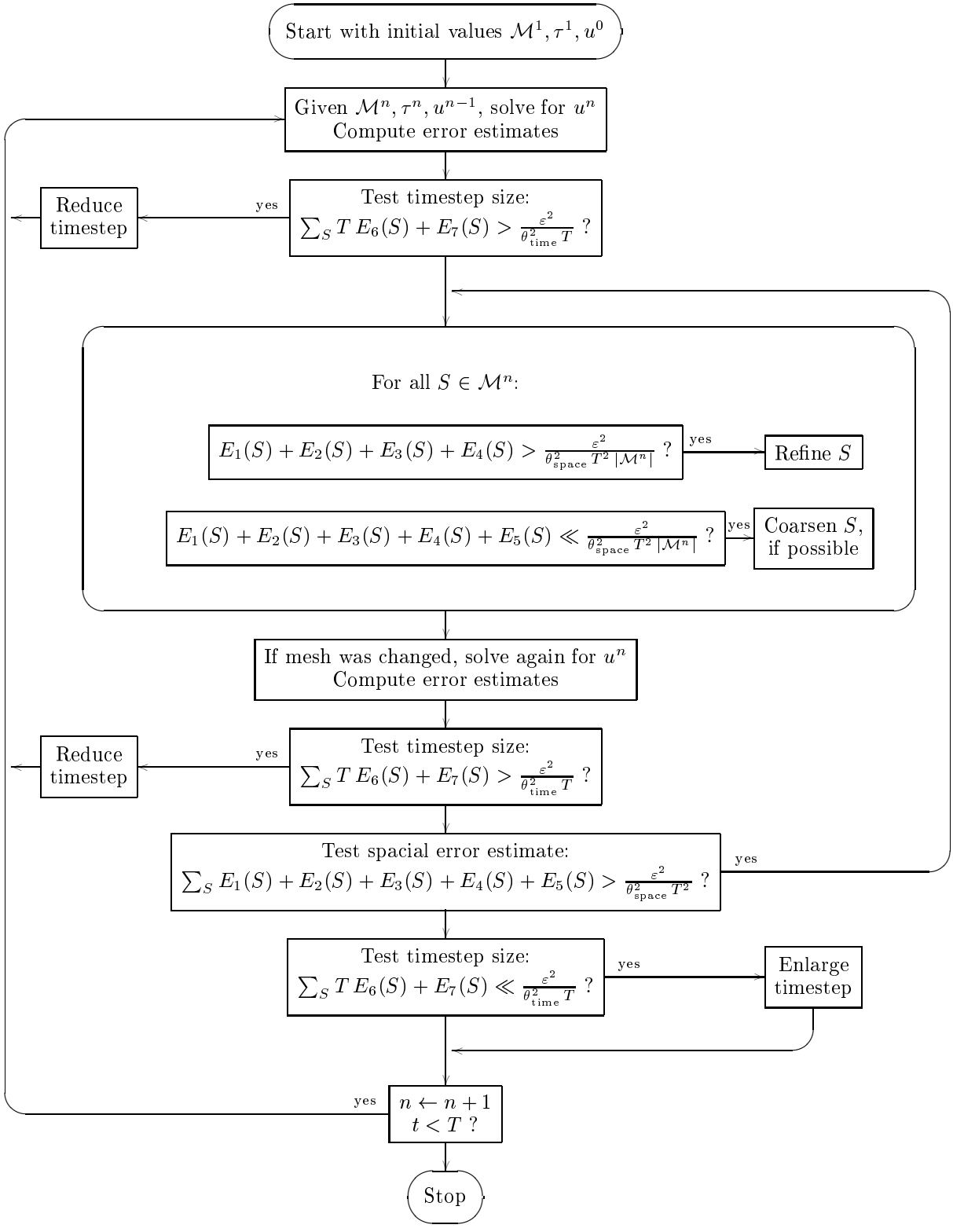

Figure 4.3: Flow diagram of the adaptive algorithm for the Stefan problem

5 The Stefan problem with surface tension and kinetic undercooling

The following section is taken from [57]. Let $\Omega \subset \mathbf{R}^n$ denote a bounded domain containing a pure substance. For $t \geq 0$ let $\Omega_s(t) \subset \Omega$ and $\Omega_l(t) \subset \Omega$ with $\Omega_s(t) \cap \Omega_l(t) = \emptyset$, $\bar{\Omega} = \bar{\Omega}_s(t) \cup \bar{\Omega}_l(t)$ be the parts of Ω containing the solid and liquid phases of the substance at time t. The moving free boundary between solid and liquid phases will be denoted by $\Sigma(t) := \bar{\Omega}_s(t) \cap \bar{\Omega}_l(t)$.
The physical constants for the different phases are heat conductivity k_s, k_l, specific heat c_s, c_l and density ρ_s, ρ_l. In each of both phases the heat equation for the (absolute) temperature Θ holds:

$$\frac{\partial \Theta}{\partial t} = D_s \, \Delta \Theta \quad \text{in } \Omega_s \,, \qquad \frac{\partial \Theta}{\partial t} = D_l \, \Delta \Theta \quad \text{in } \Omega_l \,, \tag{5.1}$$

where the diffusion constants D_s, D_l are defined by the physical constants as $D_i := k_i/(c_i \rho_i)$, $i = s, l$. At the free boundary Σ, its velocity and the temperature fulfil an equation known as "Stefan condition":

$$L \, V_\Sigma \;=\; D_s c_s \frac{\partial \Theta|_{\Omega_s}}{\partial \nu_s} + D_l c_l \frac{\partial \Theta|_{\Omega_l}}{\partial \nu_l} \;=:\; -\left[D_i \, c_i \, \frac{\partial \Theta}{\partial \nu_\Sigma} \right]_\Sigma \qquad \text{on } \Sigma(t) \,, \tag{5.2}$$

where ν_s and ν_l denote the outer normal vectors to the domains Ω_s and Ω_l, and we take $\nu_\Sigma := \nu_s = -\nu_l$ to be the normal of the free boundary Σ. V_Σ is the velocity of the free boundary Σ in direction of ν_Σ, and L is the latent heat per unit volume in the solid phase. Finally, $[\cdot]_\Sigma$ denotes the jump at the free boundary. The left hand side of (5.2) describes the rate at which heat is generated by solidification at the free boundary. The right hand side describes the heat transport into the solid and liquid phase.
Additionally, a thermodynamical condition holds at the free boundary:

$$\Theta = \Theta_M \left(1 - \frac{\gamma}{L} \, C_\Sigma - \frac{\beta}{L} \, V_\Sigma \right) \qquad \text{on } \Sigma. \tag{5.3}$$

Here, Θ_M is the melting temperature of the substance, and $\gamma = \gamma(\nu_\Sigma)$ the surface tension between solid and liquid phase (usually depending on the direction of ν_Σ). C_Σ is the mean curvature of the free boundary Σ (sum of the principal curvatures). The sign of C_Σ is taken in the way that the mean curvature for a convex solid phase Ω_s is positive. The coefficient $\beta = \beta(\nu_\Sigma)$ depends on ν_Σ in the general case, too. Equation (5.3) is known as the Gibbs–Thomson law. If γ or β depend on the direction of the normal ν_Σ then the coefficients are called "anisotropic", else "isotropic". The term $\gamma C_\Sigma / L$ describes the influence of surface tension, which stabilizes the motion and makes dendritic growth possible. With $\beta = 0$, equation (5.3) describes a situation in local thermical equilibrium (see [46] for details), while the non equilibrium situation with a moving interface is modeled with $\beta > 0$ (compare [33]). In the sequel, we will only consider the case $\beta > 0$.
In three dimensions, it may be better to use an anisotropic surface tension to model the anisotropy. This results in some anisotropic curvature to replace the mean curvature C_Σ. Use of this anisotropic curvature leads to weak formulations, compare (7.7) and (9.6).
With a given initial temperature distribution Θ_0 and an interface Σ_0, we pose the following boundary and initial value conditions:

$$\Theta(x,t) \;=\; \Theta_0(x,t) \qquad\qquad \text{on } \partial\Omega, \; t \geq 0 \,,$$

$$\begin{aligned} \Theta(x,0) &= \Theta_0(x,0) && \text{in } \Omega\,, \\ \Sigma(0) &= \Sigma_0\,. \end{aligned}$$

Using a dimensionless temperature $\theta := (c_s + c_l)\,(\Theta - \Theta_M)/2L$, the system of equations (5.1)–(5.3) transforms to

$$\begin{aligned} \frac{\partial \theta}{\partial t} &= D_{s,l}\,\Delta\theta && \text{in } \Omega_{s,l},\ t > 0\,, \\ V_\Sigma &= \frac{-2}{c_s + c_l}\left[D_i c_i \frac{\partial \theta}{\partial \nu_\Sigma}\right]_\Sigma && \text{on } \Sigma\,, \\ \theta &= -\varepsilon_C\, C_\Sigma - \varepsilon_V\, V_\Sigma && \text{on } \Sigma\,. \end{aligned}$$

Here we used $\varepsilon_C(\nu_\Sigma) = \gamma(\nu_\Sigma)\,(c_s + c_l)\,\Theta_M/2L^2$ and $\varepsilon_V(\nu_\Sigma) = \beta(\nu_\Sigma)\,(c_s + c_l)\,\Theta_M/2L^2$. The initial and boundary values transform in a similar way. With $\theta_0(x,t) := (c_s + c_l)\,(\Theta_0(x,t) - \Theta_M)/2L$ we get

$$\begin{aligned} \theta(x,t) &= \theta_0(x,t) && \text{on } \partial\Omega,\ t \geq 0\,, \\ \theta(x,0) &= \theta_0(x,0) && \text{in } \Omega\,, \\ \Sigma(0) &= \Sigma_0\,. \end{aligned} \tag{5.4}$$

In the sequel we want to work with the simplified case, where the physical constants in the solid and liquid phase are equal: $k_s = k_l$, $c_s = c_l$, $\rho_s = \rho_l$, and $D_s = D_l =: D_0$. This assumption leads not only to a pure academic example; there exist real material, like *Succinonitil*, which obey to the same physical constants in the solid and liquid phase (see [31, Table II]). On the other hand, most numerical method can be adapted to the case of different constants in both phases.
Altogether, we now get the following equations:

Problem 5.1 *Find a temperature θ and a moving free boundary Σ solving the equations*

$$\frac{\partial \theta}{\partial t} - D_0\,\Delta\theta = 0 \qquad \text{in } \Omega_s \cup \Omega_l,\ t > 0\,, \tag{5.5}$$

$$D_0\left[\frac{\partial \theta}{\partial \nu_\Sigma}\right]_\Sigma + V_\Sigma = 0 \qquad \text{on } \Sigma,\ t > 0\,, \tag{5.6}$$

$$\theta + \varepsilon_C\, C_\Sigma + \varepsilon_V\, V_\Sigma = 0 \qquad \text{on } \Sigma,\ t > 0\,, \tag{5.7}$$

together with the boundary and initial value conditions (5.4).

The Gibbs–Thomson law (5.7) is a mean curvature flow equation with driving force for the interface $\Sigma(t)$:

$$\varepsilon_V\, V_\Sigma = -\varepsilon_C\, C_\Sigma - \theta \quad \text{on } \Sigma(t),\ t > 0\,, \qquad \Sigma(0) = \Sigma_0\,.$$

For this reason we present in the next sections several formulations of the mean curvature flow together with corresponding discretizations and show their coupling to the heat equation (5.5) with Stefan condition (5.6).

6 Mean Curvature Flow for Hypersurfaces in $\mathbf{R}^n$

6.1 Some definitions and notations for hypersurfaces

A hypersurface Σ of class C^2 is a subset of $\mathbf{R}^n$ which is locally the graph of a function u of class C^2

$$\Sigma \cap V = \{\ (x', x_n) = (x_1, x_2, ..., x_{n-1}, x_n) \in V' \times \mathbf{R} \ \Big|\ x_n = u(x')\ \}.$$

Σ can also be defined through an immersion F from a compact $(n-1)$–dimensional manifold $\mathcal{S}$ without boundary into $\mathbf{R}^n$:

$$\Sigma = \{\ F(s) \ \Big|\ s \in \mathcal{S}\ \}.$$

Finally, Σ can also be defined as the zero–level set of a function w of class C^2

$$\Sigma = \{\ x \in \mathbf{R}^n \ \Big|\ w(x) = 0\ \},$$

where w satisfies: $\forall x \in \Sigma$, $\nabla w(x) \neq 0$. Locally we can always assume that e.g.

$$\frac{\partial w}{\partial x_n}(x) \neq 0.$$

Hence, by means of the implicit function theorem, the link between the two definitions of Σ in the neighbourhood $V(x_0)$ of x_0 in Σ is obvious. Indeed, there exists a function u_{x_0} defined in a neighbourhood of x_0' such that

$$x \in \Sigma \cap V(x_0) \ \Rightarrow\ x_n = u_{x_0}(x')$$

and u_{x_0} satisfies

$$\forall i \in \{1, ..., n-1\}\ :\ \frac{\partial}{\partial x_i}(u_{x_0})(x') = -\frac{\partial w}{\partial x_i}(x) \Big/ \frac{\partial w}{\partial x_n}(x).$$

Moreover, by rotating the axes at x_0, we can always assume that

$$\nabla_{x'} u_{x_0}(x_0') = 0\,,\quad \text{i.e. } \forall i \in \{1, ..., n-1\}\ :\ \frac{\partial}{\partial x_i}(u_{x_0})(x_0') = 0.$$

The unit normal vector $N(x)$ to Σ at x is defined by

$$N(x) = \frac{\nabla w}{|\nabla w|}(x).$$

This normal vector is pointing into the direction of increasing values of w. If $w(x) = -x_n + u_{x_0}(x')$ then

$$\forall i \in \{1, ..., n-1\}\ :\ N_i(x_0) = 0\ ;\ \ N_n(x_0) = -1.$$

The principal curvatures of Σ at x_0 are the $n-1$ eigenvalues κ_i, $1 \leq i \leq n-1$, of the symmetric matrix $D^2 u_{x_0}(x_0')$. Since the Laplace operator (applied to u_{x_0} at x_0') is invariant under orthogonal transformations, we have

$$D^2 u_{x_0}(x_0') \ =\ \mathrm{diag}(\kappa_1, ..., \kappa_{n-1})$$

which implies

$$\Delta_{x'} u_{x_0}(x'_0) \;=\; \kappa_1 + ... + \kappa_{n-1}.$$

The mean curvature of Σ at x_0 is the quantity $H(x_0)$ equal to

$$H(x_0) := \frac{1}{n-1} \cdot (\kappa_1 + ... + \kappa_{n-1}).$$

At some places, the sum of the principal curvatures will be denoted as *mean curvature* κ or C_Σ:

$$C_\Sigma(x_0) \;:=\; \kappa(x_0) \;:=\; (n-1)H(x_0) \;=\; \kappa_1 + ... + \kappa_{n-1}.$$

We define the principal coordinate system at x_0, assuming that the $x_1, ..., x_{n-1}$ axes are generated by the eigenvectors of the Hessian matrix $D^2 u_{x_0}(x'_0)$ and that the x_n axis is generated by the normal $N(x_0)$.

Lemma 6.1 *In a principal coordinate system at x_0, we have*

(i) $\forall i, j \in \{1, ..., n-1\} \;:\; \frac{\partial N_i}{\partial x_j}(x_0) = -\kappa_i \cdot \delta_{ij}$.

(ii) $\forall j \in \{1, ..., n\} \;:\; \frac{\partial N_n}{\partial x_j}(x_0) = 0$.

Proof: (i) Since w can be defined as : $-x_n + u_{x_0}(x')$, in a neighbourhood of x_0

$$\frac{\partial^2}{\partial x_i \partial x_j}(u_{x_0})(x'_0) \;=\; -\frac{\partial}{\partial x_j}\left(\frac{\partial w}{\partial x_i}/\frac{\partial w}{\partial x_n}\right)(x_0) \;=\; -\frac{\partial N_i}{\partial x_j}(x_0) \;=\; \kappa_i \cdot \delta_{ij}$$

because $D^2 u_{x_0}(x'_0)$ is the diagonal matrix $\mathrm{diag}(\kappa_1, ..., \kappa_{n-1})$.
(ii) This is an immediate consequence of the computation of

$$\frac{\partial}{\partial x_j}\left(\frac{\partial w}{\partial x_n}/|\nabla w|\right)(x_0).$$

Suppose that g is defined in a neighbourhood of Σ. We define the tangential gradient $\underline{\nabla} g(x_0)$ of g at x_0 on Σ as the projection of the gradient of g at x_0 onto the tangent space of Σ at x_0

$$\underline{\nabla} g(x_0) = \nabla g(x_0) - (N(x_0) \cdot \nabla g(x_0))\, N(x_0).$$

Notice that $\underline{\nabla} g(x_0)$ only depends of the values of g on Σ, in a neighbourhood of x_0. From Lemma 6.1, we derive that the tangential divergence of N is

$$\underline{\nabla} \cdot N = -(n-1)H(x_0).$$

We then define the Laplace–Beltrami operator on Σ of g at x_0 as

$$\underline{\Delta}_\Sigma g(x_0) = \underline{\nabla} \cdot \underline{\nabla} g(x_0).$$

6.2 Formulation of the MCF involving the Laplace–Beltrami operator

The formulation of the mean curvature flow problem for hypersurfaces is taken from Huisken's work [37].

Definition 6.1 *A family $(\Sigma(t))_{t\geq 0}$ of hypersurfaces evolves, from a hypersurface Σ_0, according to the mean curvature flow if*

$$\begin{aligned} \frac{\partial F}{\partial t}(s,t) &= (n-1)\,H(F(s,t))\,N(F(s,t)) \quad \forall s \in \mathcal{S}, \quad \forall t > 0, \\ F(s,0) &= F_0(s) \quad \forall s \in \mathcal{S}, \end{aligned} \tag{6.1}$$

where H denotes the mean curvature of $\Sigma(t)$, N the 'inner' normal to $\Sigma(t)$ and F_0 defines Σ_0 equal to $\Sigma(\cdot,0)$.

Lemma 6.2 *We have, for every s in $\mathcal{S}$*

$$\Delta_\Sigma F(s) := (\Delta_\Sigma \mathrm{id})(F(s)) = (n-1)\,H(F(s))N(F(s)),$$

where Δ_Σ denotes the Laplace–Beltrami operator on Σ. Hence the mean curvature flow defined in (6.1) may be written as

$$\frac{\partial F}{\partial t}(s,t) = \Delta_{\Sigma(t)} F(s,t) \qquad \forall s \in \mathcal{S}, \quad \forall t > 0.$$

Huisken has proved the following existence result

Theorem 6.1 *Suppose that Σ_0 is uniformly convex (that is the principal curvatures are positive everywhere). Then (6.1) has a smooth solution on $[0,T)$. Moreover, the hypersurfaces $\Sigma(\cdot,t)$ converge spherelike to a point when t increases to T.*

Motion with a driving force: The equation

$$\frac{\partial F}{\partial t}(s,t) = \Delta_{\Sigma(t)} F(s,t) + g(F(s,t),t)N(F(s,t)) \qquad \forall s \in \mathcal{S}, \quad \forall t > 0$$

describes the mean curvature motion of a manifold with an additional driving force $g : \mathbf{R}^n \times \mathbf{R}^+ \to \mathbf{R}$ which acts in normal direction.

6.3 The level set formulation of MCF and the formulation for graphs

We suppose that Σ is defined as the graph of a function u or as the zero level set of a function w. We want to describe the mean curvature flow of a family $(\Sigma(t))_{t\geq 0}$ in terms of u or w.

Theorem 6.2 *Assume that all the level sets of w evolve according to mean curvature flow in the sense of (6.1). Then $w = w(x,t)$ satisfies*

$$\frac{\partial w}{\partial t} - |\nabla w| \nabla \cdot \frac{\nabla w}{|\nabla w|} = 0 \quad \forall x \in \mathbf{R}^n, \quad \forall t > 0. \tag{6.2}$$

Conversely, if w is the solution of (6.2), the family of zero level sets of $w(\cdot,t)$ evolve according to the mean curvature flow, at least when w is smooth and $|\nabla w|$ does not vanish.

Proof: Since $w(F(.,t))$ is identically equal to 0, we have

$$\frac{\partial w}{\partial t}(F(\cdot,t),t) + \nabla w(F(\cdot,t),t)\ \frac{\partial F}{\partial t}(\cdot,t) = 0.$$

Because every level set of w evolves according to mean curvature flow, we have

$$\frac{\partial F}{\partial t} = (n-1)H(F)N(F) = -\frac{\nabla w}{|\nabla w|}\ \nabla \cdot \frac{\nabla w}{|\nabla w|},$$

which immediately implies (6.2). The converse part is proved in an analogous way. □

Remarks:

1) Equation (6.2) may be written as

$$\frac{\partial w}{\partial t} = \sum_{i,j} \left(\delta_{ij} - \frac{\frac{\partial w}{\partial x_i}\frac{\partial w}{\partial x_j}}{|\nabla w|^2} \right) \frac{\partial^2 w}{\partial x_i \partial x_j}.$$

2) When $\Sigma(t)$ are presented as graphs of functions $u(\cdot,t)$, Equation (6.2) for w leads to the following equation for u:

$$\frac{\partial u}{\partial t} - \sqrt{1+|\nabla u|^2}\quad \nabla \cdot \frac{\nabla u}{\sqrt{1+|\nabla u|^2}} = 0.$$

Motion with a driving force: The equation

$$\frac{\partial w}{\partial t} - |\nabla w| \nabla \cdot \frac{\nabla w}{|\nabla w|} \ = \ |\nabla w|\, g \quad \forall x \in \mathbf{R}^n, \quad \forall t > 0.$$

describes the mean curvature motion of all level sets of w with an additional driving force $g : \mathbf{R}^n \times \mathbf{R}^+ \to \mathbf{R}$ which acts in normal direction.

6.4 Viscosity solutions for the level set formulation

Generalized motion by mean curvature: The contents of this paragraph is taken from the work of Evans and Spruck [26]. As Equation (6.2) degenerates when $|\nabla w| = 0$, we now turn to the concept of viscosity solutions $w = w(x,t)$ for the differential equation

$$w_t - |\nabla w|\ \nabla \cdot \frac{\nabla w}{|\nabla w|} = 0, \tag{6.3}$$

with initial values

$$w(\cdot,0) = w_0. \tag{6.4}$$

Definition 6.2 *A function $w \in C^0(\mathbf{R}^n \times [0,\infty)) \cap L^\infty(\mathbf{R}^n \times [0,\infty))$ is called a weak subsolution of 6.3 provided for each $\psi \in C^\infty(\mathbf{R}^{n+1})$ the following conclusion holds true:*
If $w - \psi$ has a local maximum at a point $(x_0,t_0) \in \mathbf{R}^n \times (0,\infty)$ then at (x_0,t_0)

$$\psi_t - \left(\delta_{ij} - \frac{\psi_{x_i}\psi_{x_j}}{|\nabla\psi|^2} \right) \psi_{x_i x_j} \le 0, \ \textit{if } \nabla\psi(x_0,t_0) \neq 0, \tag{6.5}$$

and

$$\psi_t - (\delta_{ij} - \eta_i\eta_j)\psi_{x_i x_j} \le 0, \ \textit{if } \nabla\psi(x_0,t_0) = 0 \tag{6.6}$$

for some $\eta \in \mathbf{R}^n$ with $|\eta| \le 1$.

Definition 6.3 *A function $w \in C^0(\mathbf{R}^n \times [0,\infty)) \cap L^\infty(\mathbf{R}^n \times [0,\infty))$ is called a weak supersolution of (6.3) provided for each $\psi \in C^\infty(\mathbf{R}^{n+1})$ the following conclusion holds true:*
If $w - \psi$ has a local minimum at a point $(x_0, t_0) \in \mathbf{R}^n \times (0,\infty)$ then at (x_0, t_0)

$$\psi_t - \left(\delta_{ij} - \frac{\psi_{x_i}\psi_{x_j}}{|\nabla\psi|^2}\right)\psi_{x_i x_j} \geq 0, \ \text{if } \nabla\psi(x_0,t_0) \neq 0, \tag{6.7}$$

and

$$\psi_t - (\delta_{ij} - \eta_i\eta_j)\psi_{x_i x_j} \geq 0, \ \text{if } \nabla\psi(x_0,t_0) = 0 \tag{6.8}$$

for some $\eta \in \mathbf{R}^n$ with $|\eta| \leq 1$.

Definition 6.4 *A function $w \in C^0(\mathbf{R}^n \times [0,\infty)) \cap L^\infty(\mathbf{R}^n \times [0,\infty))$ is called a weak solution of (6.3) if w is a weak subsolution and a weak supersolution of (6.3).*

Since we want to treat the mean curvature flow problem for a given initial hypersurface, we have to say what we mean by the generalized motion by mean curvature also in the case where the surface does not continue to be a classical surface. We shall see that under natural assumptions there exists a unique weak solution of (6.3).

Definition 6.5 *Let Σ_0 be a compact set in $\mathbf{R}^n$ and let $w_0 \in C^0(\mathbf{R}^n)$ a real-valued function which is constant outside some large ball such that*

$$\Sigma_0 = \{x \in \mathbf{R}^n | w_0(x) = 0\}.$$

If w is a weak solution of (6.3), (6.4), we define the set

$$\Sigma_t = \{x \in \mathbf{R}^n | w(x,t) = 0\}$$

and call Σ_t, $t \geq 0$ the generalized motion of Σ_0 by mean curvature.

One can show, that Σ_t does not depend on the particular choice of the initial function w_0 which represents the initial surface.
One can check that this concept of the generalized motion by mean curvature coincides with the classical motion if and so long as the latter exists.

Some properties of weak solutions:

Theorem 6.3 *Assume that w is a weak solution of (6.3) and $\Psi \in C^0(\mathbf{R}^n)$ is real-valued. Then $v = \Psi(w)$ is a weak solution of (6.3).*

Theorem 6.4 *Let $(w_k)_{k\in\mathbf{N}}$ be a sequence of uniformly bounded weak solutions of (6.3) which converges uniformly on compact subsets of $\mathbf{R}^n \times [0,\infty)$ to w. Then w is a weak solution of (6.3). The same holds true for weak subsolutions and weak supersolutions of (6.3).*

Existence of weak solutions: The existence of a weak solution of problem (6.3) is proved by a regularization procedure. The idea is to regularize the singular problem for $\epsilon \in (0,1)$ by the partial differential equation

$$w_t^\varepsilon - \left(\delta_{ij} - \frac{w_{x_i}^\varepsilon u_{x_j}^\varepsilon}{\varepsilon^2 + |\nabla w^\varepsilon|^2}\right) w_{x_i x_j}^\varepsilon = 0 \quad \text{in } \mathbf{R}^n \times (0,\infty) \quad \text{with} \tag{6.9}$$

$$w^\varepsilon(\cdot,0) = w_0 \quad \text{in } \mathbf{R}^n. \tag{6.10}$$

Let us assume that w^ε is a smooth solution of the regularized problem (6.9). We write $\overline{x} = (x, x_{n+1})$ and define

$$v^\varepsilon(\overline{x}) = w^\varepsilon(x) - \varepsilon x_{n+1} \quad \text{for } \overline{x} \in \mathbf{R}^{n+1}. \tag{6.11}$$

Then

$$|\nabla v^\varepsilon|^2 = \varepsilon^2 + |\nabla w^\varepsilon|^2$$

and v^ε is a solution of

$$v_t^\varepsilon - \left(\delta_{ij} - \frac{v_{x_i}^\varepsilon v_{x_j}^\varepsilon}{|\nabla v^\varepsilon|^2}\right) v_{x_i x_j}^\varepsilon = 0 \quad \text{in } \mathbf{R}^{n+1} \times (0,\infty) \quad \text{with} \tag{6.12}$$

$$v^\varepsilon(\cdot,0) = v_0^\varepsilon \tag{6.13}$$

where $v_0^\varepsilon(\overline{x}) = w_0(x) - \varepsilon x_{n+1}$. But this means that each level set of v^ε moves according to its mean curvature. This is in particular the case for the zero level sets

$$\Sigma_t^\varepsilon = \{\overline{x} \in \mathbf{R}^{n+1} | v^\varepsilon(\overline{x}, t) = 0\}$$

and each level set is a graph:

$$\Sigma_t^\varepsilon = \{(x, x_{n+1}) \in \mathbf{R}^{n+1} | x_{n+1} = \frac{1}{\varepsilon} w^\varepsilon(x)\}.$$

These ideas are used to prove the following theorem.

Theorem 6.5 *Let w_0 be smooth and constant outside some large ball in $\mathbf{R}^n$. For each $\varepsilon \in (0,1)$ there exists a unique smooth bounded solution v^ε of (6.12), (6.13) and*

$$\sup_{\varepsilon \in (0,1)} \|v^\varepsilon\|_{C^1(\mathbf{R}^n \times (0,\infty))} \leq c \|w_0\|_{C^{1,1}(\mathbf{R}^n)}. \tag{6.14}$$

This result is then used to prove the existence and uniqueness of a weak solution of the mean curvature flow problem (6.3),

Theorem 6.6 *Assume $w_0 \in C^0(\mathbf{R}^n)$ is real valued and constant outside some large ball in $\mathbf{R}^n$. Then there exists a unique weak solution w of (6.3), (6.4) which is constant outside some large ball in $\mathbf{R}^n \times [0,\infty)$.*

The uniqueness of weak solutions is established by a comparision argument.

Theorem 6.7 *Assume that $\underline{w}$ is a weak subsolution and $\overline{w}$ is a weak supersolution of (6.3) satisfying $\underline{w} \leq \overline{w}$ on $\mathbf{R}^n \times \{0\}$. if $\underline{w}$ and $\overline{w}$ are constant on $\{(x,t) | \ |x| + t \geq R\}$ for some postive R, then $\underline{w} \leq \overline{w}$ on $\mathbf{R}^n \times [0,\infty)$.*

6.5 Mean curvature flow via the Allen-Cahn equation

The distance function: In this subsection we assume that Σ is a smooth surface. One should think of it as the boundary of some domain in $\mathbf{R}^n$. The distance function of Σ is given by

$$dist(x) = \inf_{y \in \Sigma} |x - y|, \qquad x \in \mathbf{R}^n$$

From [25] we know:

Theorem 6.8 *Let $dist(\cdot, t)$ be the distance function to the surfaces evolving according to mean curvature flow until the extinction time T. Then dist satisfies*

(i) dist is lower semicontinuous on $\mathbf{R}^n \times [0, T)$, i. e.

$$\forall x \in \mathbf{R}^n, \quad \forall t \in [0, T) : \quad \lim_{y \to x, s \to t} dist(y, s) \geq dist(x, t),$$

(ii) d is continuous from below in time on $\mathbf{R}^n \times [0, T)$:

$$\forall x \in \mathbf{R}^n, \quad \forall t \in [0, T) : \quad \lim_{y \to x, s \to t-0} dist(y, s) = dist(x, t),$$

(iii) On $\mathbf{R}^n \times (0, T) \cap \{dist > 0\}$, dist satisfies $\partial dist/\partial t - \Delta dist \geq 0$ in the sense of viscosity solutions, that is for every smooth function ψ in $C^\infty(\mathbf{R}^n \times (0, T))$, such that $dist - \psi$ has a minimum at (x_0, t_0) in $\mathbf{R}^n \times (0, T)$ with $dist(x_0, t_0) > 0$, then $\partial\psi/\partial t - \Delta\psi(x_0, t_0) \geq 0$.

In the following we shall need an oriented distance function, which is usually normalized in such a way that it is positive in the 'interior' I and negative in the 'exterior' O.

$$d(x) = \begin{cases} dist(x) & , \quad x \in I \\ 0 & , \quad x \in \Sigma \\ -dist(x) & , \quad x \in O \end{cases}$$

Lemma 6.3 *For every x in a neighbourhood of Σ the decomposition*

$$x = a(x) + d(x)N(x)$$

holds with $a(x) \in \Sigma$. d is as smooth as the surface Σ is. The normal to Σ at $a(x)$ in the direction of increasing d is given by

$$N(x) = \nabla d(x).$$

The trace of the Hessian matrix of d is the mean curvature at $x \in \Sigma$:

$$(n - 1)H(x) = \Delta d(x).$$

For a **proof** see the book of Gilbarg, Trudinger [30, 14.6 Appendix, resp. Section 2.1].
Let us see how the distance function in smooth mean curvature flow behaves. We assume that $d(x,t)$ is the oriented distance function of a surface $\Sigma(t)$ which is moved by its mean curvature. We assume that d is a smooth function of both variables x and t.
Then we have

$$d_t - |\nabla d| \nabla \cdot \frac{\nabla d}{|\nabla d|} = 0.$$

and since $|\nabla d(\cdot, t)| = 1$, we get

$$d_t - \Delta d \;=\; 0 \quad \text{on } \Sigma(t)$$

Note that the level sets of d do not move according to their mean curvature except the zero level set $\Sigma(t)$. The following Lemma is obtained by an easy calculation.

Lemma 6.4 *If w is a smooth solution of equation (6.2) with $|\nabla w| \neq 0$, then the oriented distance function $d(x,t)$ with respect to $\Sigma(t) = \{x | w(x,t) = 0\}$ satisfies*

$$\begin{aligned} d_t - \Delta d &\geq 0 \quad \textit{in } I \times (0,T) \\ d_t - \Delta d &= 0 \quad \textit{on } \Sigma(t) \times (0,T) \\ d_t - \Delta d &\leq 0 \quad \textit{in } O \times (0,T) \end{aligned}$$

in a neighbourhood of $\Sigma(t)$.

The Allen–Cahn equation: In order to define a weak viscosity type solution for singular mean curvature flow, there are many possibilities to introduce artificial viscosity into the equation (6.2). As we shall see one slightly hidden way to do this is to consider the Allen-Cahn equation

$$\chi^\epsilon_t - \Delta \chi^\epsilon + \frac{1}{\epsilon^2} \psi(\chi^\epsilon) \;=\; 0 \quad \text{in } \mathbf{R}^n \times (0,T) \tag{6.15}$$

with initial condition

$$\chi^\epsilon = \chi^\epsilon_0 \quad \text{on } \mathbf{R}^n \times \{0\}.$$

The function ψ is the derivative of a double well potential Ψ:

$$\psi(s) = \frac{1}{2}\Psi'(s), \quad \Psi(s) = (s^2 - 1)^2, \quad s \in \mathbf{R}.$$

First we have to convince ourselves that this differential equation has something to do with mean curvature flow in the case when there are no singularities developing during the evolution. A detailed asymptotic analysis can be found in [56]. Here we shall just see that there is a connection between the Allen-Cahn equation and mean curvature flow for $\epsilon \to 0$, $\epsilon > 0$.
For the smooth case it should be enough to know the following fact, which is used later for the asymptotic analysis. The function

$$q(s) = \tanh(s) = \frac{e^{2s} - 1}{e^{2s} + 1}, \quad s \in \mathbf{R}$$

is a solution of the ordinary differential equation

$$\frac{d^2q}{ds^2}(s) = \psi(q(s)), \quad q(\pm\infty) = \pm 1.$$

The function

$$w^\varepsilon(x,t) = q(\frac{d}{\varepsilon})$$

then satisfies the equation

$$w_t^\varepsilon - \Delta w^\varepsilon + \frac{1}{\varepsilon^2}\psi(w^\varepsilon) = q'(\frac{d}{\varepsilon})\frac{1}{\varepsilon}(d_t - \Delta d).$$

Asymptotic analysis for the Allen-Cahn equation: The intention of this paragraph is to formulate the results of [25] on the asymptotics for the Allen-Cahn equation, namely that for $\varepsilon \to 0$ the solution u^ε of (6.15) becomes ± 1 in an 'interior' resp. 'exterior' region with an 'interface' between the inside and the outside being a generalized motion governed by mean curvature. The main theorem is the following.

Theorem 6.9 *Let $\Sigma(0)$ be the smooth boundary of a domain in $\mathbf{R}^n$ and let d^0 be the signed distance function for $\Sigma(0)$. Set*

$$h^\varepsilon(x) \;:=\; q\left(\frac{d^0(x)}{\varepsilon}\right), \quad x \in \mathbf{R}^n.$$

Let χ^ε be a solution to the Allen-Cahn equation (6.15) on $\mathbf{R}^n \times (0,\infty)$ with initial value h^ε. Let w be a weak solution of

$$w_t - |\nabla w|\nabla \cdot \frac{\nabla w}{|\nabla w|} = 0,$$

with initial value d^0 which is thought to be continued constantly outside some ball. Then

$$\chi^\varepsilon \to 1 \qquad (\varepsilon \to 0)$$

uniformly on all compact subsets of I and

$$\chi^\varepsilon \to -1 \qquad (\varepsilon \to 0)$$

uniformly on all compact subsets of O where

$$I = \{(x,t) \in \mathbf{R}^n \times (0,\infty) \mid w(x,t) > 0\}, \quad O = \{(x,t) \in \mathbf{R}^n \times (0,\infty) \mid w(x,t) < 0\}.$$

Double obstacle potential: Similar results can be obtained with ψ coming from other potentials, for example the double obstacle potential, which will be used in the numerical method described below:

$$\psi(s) = \frac{1}{2}\Psi'(s), \quad \Psi(s) = s^2 - 1, \quad s \in [-1,1].$$

Motion with a driving force: Mean curvature flow with a driving force $g : \mathbf{R}^n \times \mathbf{R}^+ \to \mathbf{R}$ which acts in normal direction is modelled by the equation

$$\partial_t(\chi_\varepsilon) - \Delta\chi_\varepsilon + \frac{1}{\varepsilon^2}\psi(\chi_\varepsilon) \;=\; \frac{c_0}{2\varepsilon}g \quad \text{in } \Omega \times (0,\infty),$$

with $c_0 := \int_{-1}^{1} \sqrt{\Psi(s)}ds$.

Time and space dependent regularization parameter: The same asymptotic analysis holds if the regularization constant ε is replaced by a density function $\varepsilon a(x,t)$, where $a : \mathbf{R}^n \times \mathbf{R}^+ \to \mathbf{R}^+$ is bounded. The appropriate Allen–Cahn equation reads:

$$\varepsilon\partial_t(a\chi_\varepsilon) - \varepsilon\nabla\cdot(a\nabla\chi_\varepsilon) + \frac{1}{a\varepsilon}\psi(\chi_\varepsilon) \;=\; \frac{c_0}{2}g \quad \text{in } \Omega \times (0,\infty).$$

Anisotropic motion: Anisotropic motion by mean curvature can be modelled in the Allen–Cahn equation by adding an anisotropy function. The equation

$$\varepsilon\partial_t\chi_\varepsilon - \varepsilon\Delta\chi_\varepsilon + \frac{1}{\varepsilon}\Psi'(\chi_\varepsilon) \;=\; \theta\,.$$

is the gradient flow of the energy functional

$$E_\varepsilon(\chi) \;:=\; \int \varepsilon|\nabla\chi|^2 + \frac{1}{\varepsilon}\Psi(\chi) - \theta\chi\,.$$

Replacing $|\nabla\chi|^2$ by an anisotropy $F(\nabla\chi)$, where F is convex and homogenous of degree 2, the corresponding energy is

$$E_{F,\varepsilon}(\chi) \;:=\; \int \varepsilon F(\nabla\chi) + \frac{1}{\varepsilon}\Psi(\chi) - \theta\chi$$

and the corresponding Allen–Cahn equation reads

$$\varepsilon\partial_t\chi_\varepsilon - \varepsilon\nabla F'(\nabla\chi_\varepsilon) + \frac{1}{\varepsilon}\Psi'(\chi_\varepsilon) \;=\; \theta\,.$$

Elliott and Schätzle [23] proved convergence of the solution to an anisotropic mean curvature flow

$$V = \mathrm{tr}(B(N)D^2B(N)R) - B(N)\theta,$$

where $B := \sqrt{2F}$, N is the unit normal, V the normal velocity, and R the second fundamental form of the interface. Again, in two dimensions and for smooth anisotropy, this is equivalent to

$$V = -\tilde{f}(\tilde{f} + \tilde{f}'')\kappa - \tilde{f}\theta,$$

where $\tilde{f}$ is 2π-periodic, and $\tilde{f}(\alpha) := \sqrt{2F(\cos\alpha, \sin\alpha)}$.
Belletini and Paolini [6] get similar results in the context of Finsler geomety.

Phase field equations: The Stefan problem with kinetic undercooling and surface tension, Problem 5.1, can be approximated by a coupling of the Allen–Cahn equation with a heat equation. This gives the following system of equations which is called "phase field equations" for the temperature θ_ε and phase parameter χ_ε [11]:

$$\varepsilon\partial_t(\theta_\varepsilon + \chi_\varepsilon) - \varepsilon\Delta\theta_\varepsilon \;=\; f\,,$$

$$\partial_t\chi_\varepsilon - \Delta\chi_\varepsilon + \frac{1}{\varepsilon}\Psi'(\chi_\varepsilon) \;=\; \theta_\varepsilon\,.$$

7 Finite element methods for the MCF level set formulation

Weak formulation: For the numerical computation of a function w whose level sets move according the mean curvature flow, we start from the regularized equation (6.9). We consider the problem in a bounded domain $\Omega \subset \mathbf{R}^n$ with homogenous Neumann boundary conditions on $\partial\Omega$. This is motivated by the fact, that if w_0 is constant outside a ball, then w is constant outside a ball for all times $t > 0$.

$$\begin{aligned} \frac{w_t}{\sqrt{\varepsilon^2 + |\nabla w|^2}} - \nabla \cdot \frac{\nabla w}{\sqrt{\varepsilon^2 + |\nabla w|^2}} &= 0 \qquad \text{in} \quad \Omega \times (0,\infty), \\ \frac{\partial w}{\partial \nu} &= 0 \qquad \text{on} \quad \partial\Omega \times (0,\infty), \\ w &= w_0 \qquad \text{in} \quad \Omega \times \{0\}. \end{aligned} \tag{7.1}$$

In order to derive a weak formulation, which can be used for a finite element discretization, we multiply the first equation in (7.1) with an arbitrary test function $\varphi \in H^1(\Omega)$, and integrate by parts:

$$\begin{aligned} \int_\Omega \frac{w_t\varphi + \nabla w \cdot \nabla \varphi}{\sqrt{\varepsilon^2 + |\nabla w|^2}} &= 0 \qquad \forall \varphi \in H^1(\Omega), 0 < t < \infty, \\ w &= w_0 \qquad \text{in} \quad \Omega \times \{0\}. \end{aligned} \tag{7.2}$$

Time discretization: We use a constant time step size $\tau > 0$ and the notations $w^m(x) \approx w(m\tau, x)$ and $Q_m := \sqrt{\varepsilon^2 + |\nabla w^m|^2}$, $m = 0, 1, \ldots$, and replace the time derivative in (7.2) by an implicit Euler time discretization. With $w^0 := w_0$, this leads to

$$\frac{1}{\tau}\int_\Omega \frac{(w^m - w^{m-1})\varphi}{Q_m} + \int_\Omega \frac{\nabla w^m \cdot \nabla \varphi}{Q_m} = 0 \qquad \forall \varphi \in H^1(\Omega),\ m = 1, 2, \ldots \tag{7.3}$$

In every time step, this is a nonlinear equation for w^m. In order to get linear equations, we can use a semi–implicit discretization in time, where the linear part is discretized implicitely, while the nonlinear parts are discretized explicitely. All denominators are computed using the previous solution w^{m-1}:

$$\frac{1}{\tau}\int_\Omega \frac{(w^m - w^{m-1})\,\varphi}{Q_{m-1}} + \int_\Omega \frac{\nabla w^m \cdot \nabla \varphi}{Q_{m-1}} = 0 \qquad \forall \varphi \in H^1(\Omega),\ m = 1, 2, \ldots, \tag{7.4}$$

Discretization in space: We assume that the bounded domain Ω has polygonal boundary $\partial\Omega$, and that a conforming triangulation $\mathcal{T}_h$ of Ω into simplices (triangles or tetrahedra) is given. Let $V_h \subset H^1(\Omega)$ denote the space of piecewise linear finite elements over $\mathcal{T}_h$. Now we can approximate the time–discretized problems using finite elements. Equation (7.3) leads to

$$\begin{aligned} \int_\Omega \frac{\frac{1}{\tau}(w_h^m - w_h^{m-1})\varphi_h + \nabla w_h^m \cdot \nabla \varphi_h}{Q_m} &= 0 \qquad \forall \varphi_h \in V_h,\ m = 1, 2, \ldots, \\ w_h^0 &= I_h w_0\,. \end{aligned} \tag{7.5}$$

For each time step, this is a nonlinear equation for w^m. It can be solved by a modified Newtons method, for example.

The linearized semi–implicit equation (7.4) leads to

$$\int_\Omega \frac{\frac{1}{\tau}(w_h^m - w_h^{m-1})\varphi_h + \nabla w_h^m \cdot \nabla \varphi_h}{Q_{m-1}} = 0 \qquad \forall \varphi_h \in V_h,\ m = 1, 2, \ldots, \tag{7.6}$$

$$w_h^0 = I_h w_0 \,.$$

Thus, in each time step a linear equation has to be solved.

Adaptive methods: Up to now, no error estimates are known for problems (7.5) or (7.6), neither a priori nor a posteriori. In the context of free boundary problems, we are mainly interested in the evolution of the zero level set of w, not in the evolution of all level sets. So, a first heuristic criterion for local mesh sizes is to choose a fine mesh near the zero level set, and a coarser mesh far away from it.
In spirit of the a posteriori L_2 error estimate for the heat equation, we can also use a local error indicator like

$$\eta_T(w_h^m) := h_T^{3/2} \left\| \left[\frac{\nabla w_h^m}{Q_m} \right] \right\|_{L_2(\partial T \setminus \partial\Omega)} .$$

A combination of both criteria is used for the numerical examples that will be presented. The computations are done by Fried [27, 28].

Motion with a driving force: Mean curvature motion with a driving force $g : \Omega \to \mathbf{R}$, where every level set moves in normal direction according to the geometric law

$$V = -C + g$$

leads to a right hand side g in equation (7.1):

$$\frac{w_t}{\sqrt{\varepsilon^2 + |\nabla w|^2}} - \nabla \cdot \frac{\nabla w}{\sqrt{\varepsilon^2 + |\nabla w|^2}} = g \qquad \text{in} \quad \Omega \times (0, \infty).$$

The changes in the discrete scheme are straightforward.

Anisotropic motion: The motion of level sets by mean curvature is the gradient flow for the energy

$$E(w) := \int_\Omega |\nabla w| \,.$$

If we are given a smooth function $f : S^{n-1} \to \mathbf{R}^+$ and study the gradient flow of the anisotropic energy

$$E_f(w) := \int_\Omega f\left(\frac{\nabla w}{|\nabla w|}\right) |\nabla w| \,,$$

we end up with the following weak formulation:

$$\int_\Omega \frac{w_t \varphi}{|\nabla w|} + \left(f\left(\frac{\nabla w}{|\nabla w|}\right) \frac{\nabla w}{|\nabla w|} + \nabla f\left(\frac{\nabla w}{|\nabla w|}\right)\right) \cdot \nabla \varphi = 0$$

$$\text{for all } \varphi \in H^1(\Omega),\ 0 < t < \infty, \tag{7.7}$$

$$w = w_0 \quad \text{in} \quad \Omega \times \{0\}.$$

The level sets of a smooth solution w move according to the anisotropic law

$$V \;=\; -\,C_f\,,$$

where C_f is the anisotropic mean curvature according to f. In two dimensions, f can be written as $f(\cos\alpha, \sin\alpha) = \tilde{f}(\alpha)$ where $\tilde{f} : [0, 2\pi] \to \mathbf{R}$. If $\tilde{f}$ is smooth and $\tilde{f} + \tilde{f}'' > 0$, this law of motion is equivalent to

$$V \;=\; -\,(\tilde{f} + \tilde{f}'')\,C\,.$$

Regularization and discretization of this equation can be done like above.

8 Adaptive discretization of the Allen–Cahn equation

We consider the Allen–Cahn equation (6.15) with regularization parameter

$$\varepsilon\partial_t(a\chi_\varepsilon) - \varepsilon\nabla\cdot(a\nabla\chi_\varepsilon) + \frac{1}{2a\varepsilon}\Psi'(\chi_\varepsilon) \ \ni \ \frac{c_0}{2}g \quad \text{in } \Omega\times(0,\infty), \tag{8.1}$$

$$\chi_\varepsilon(\cdot,0) \ = \ \chi_\varepsilon^0(\cdot) \ \text{ in } \Omega, \quad \chi_\varepsilon(\cdot,t) \ = \ f(\cdot,t) \ \text{ on } \partial\Omega\times(0,\infty),$$

using the double obstacle potential

$$\Psi(s) := \begin{cases} 1-s^2 & \text{if } s\in[-1,1] \\ +\infty & \text{if } s\notin[-1,1], \end{cases}$$

$$\frac{1}{2}\Psi'(s) = \begin{cases} (-\infty,1] & \text{if } s=-1 \\ -s & \text{if } s\in(-1,1) \\ [-1,\infty) & \text{if } s=1, \end{cases}$$

$c_0 := \int_{-1}^1 \sqrt{\Psi(s)}ds = \pi/2$, and $\Omega\subset\mathbf{R}^n$ a bounded domain. The function $a(x,t)$ denotes a density function which may vary in space and time, and g a driving force. The zero level set of χ_ε^0 coincides with the initial surface Σ^0.
It can be shown, that $|\chi_\varepsilon|<1$, and χ_ε attains the values -1 or $+1$ outside a narrow transition layer $\mathcal{T}_\varepsilon(t)$ of local size $O(\varepsilon a(x,t))$ in the vincinity of any regular point $x\in\Sigma(t)$. The latter is not true if the double well potential is used instead of the double obstacle potential.

Variational formulation: Using the convex set

$$K := \{\varphi\in H^1(\Omega) \mid |\varphi|\le 1 \text{ in } \Omega,\ \varphi = f \text{ on } \partial\Omega\},$$

the variational inequality equivalent to (8.1) is: Find $\chi_\varepsilon\in L_2(0,\infty,K)\cap H^1(0,\infty,L_2(\Omega))$ such that $\chi_\varepsilon(\cdot,0)=\chi_\varepsilon^0(\cdot)$ and, for a.e. $t>0$ and all $\varphi\in K$

$$\int_\Omega \varepsilon\partial_t(a\chi_\varepsilon)(\varphi-\chi_\varepsilon) + \varepsilon a\nabla\chi_\varepsilon\nabla(\varphi-\chi_\varepsilon) - \frac{1}{2a\varepsilon}\chi_\varepsilon(\varphi-\chi_\varepsilon) - \frac{\pi}{4}g(\varphi-\chi_\varepsilon) \ \ge \ 0\,.$$

Discretization: In [53], this weak variational formulation is discretized using piecewise linear finite elements on a triangulation of Ω and an explicit time discretization. Lumping is used for the L_2 scalar products. Convergence of this method, even after singularities, is shown in [52, 55].
We use the following matrix–vector notation: Let X denote the vector of nodal values of a piecewise linear finite element function, $M_k = (m_{jl}^k)_{j,l}$ denotes the lumped mass matrix with weight a^k, $k=-1,0,1$, and $S=(s_{jl})_{j,l}$ the stiffness matrix with weight a. Let φ_j denote a nodal basis function, and Π^0, Π^1 the piecewise constant resp. linear interpolation operators at barycenters resp. nodes of the triangulation. Then the entries of these matrices are

$$m_{jl}^k = \int_\Omega \Pi^0(a^k)\,\Pi^1(\varphi_j\varphi_l)\,, \quad s_{jl} = \int_\Omega \Pi^0(a)\,\nabla\varphi_j\nabla\varphi_l\,.$$

In time step $i+1$, when the discrete solution X^i from the previous time is known, this leads to the following discrete algorithm for computation of X^{i+1}:

$$\begin{aligned} X^{i+1/2} \ &= \ (M_1)^{-1}\left(\left(M_1 - \tau^i S + \frac{\tau^i}{\varepsilon^2}M_{-1}\right)X^i + \tau^i\frac{1}{\varepsilon}\frac{\pi}{4}M_0\Pi^1 g^{i+1}\right), \\ X^{i+1} \ &= \ P_K X^{i+1/2}, \end{aligned}$$

where P_K is the projection to the convex set K by truncation of the nodal values to the range $[-1, 1]$.
A similar formulation can be derived also in a rotationally symmetric context [53]. If a changes in time, then the two M_1 matrices which appear in the scheme above are computed with different weights.

Adaptive methods: The property $|\chi_\varepsilon| = 1$ outside $\mathcal{T}_\varepsilon(t)$ can be used numerically in solving the discrete problem only in a narrow region around the discrete interface. The mesh adaption will take this into account and generate a fine mesh only where it is needed, while it coarsens the mesh as much as possible at other places.
Our control parameter is the density function a. Depending on this function, the width of the transition area will vary. We will try to adapt this width to local properties of the solution, such as the curvature of the discrete interface.
The local mesh size is chosen proportional to the density function a, according to a priori analysis.
Up to now, no a posteriori error estimates for the Allen–Cahn equation are known. For that reason, we will exploit the known a priori estimates and add some heuristics for the local choice of the density function $a(x, t)$. A priori analysis leads to the following formula for the local mesh size depending on ε and a:

$$h(x,t) \;\approx\; c\varepsilon^\lambda a(x,t), \quad \text{with } c \in (0,1),\ \lambda > 1\,.$$

As we use an explicit time discretization, the time step size is bounded by size of the smallest mesh element for stability of the numerical scheme:

$$\tau \;\leq\; c \min_T h_T^2\,.$$

In [53] a time–independent order parameter $a(x)$ is used, which is chosen by a priori knowledge of the curvature of $\Sigma(t)$ and the formation of singularities.
We try to generate such information a posteriori out of the current discrete solution, and adapt the mesh accordingly.
The main idea is the following: For a smooth interface, there is a strip around the surface, where the distance function is smooth (each point has a unique nearest point on the interface). We will try to keep the transition area inside this strip. Locally, the width w of this strip is

$$w \;=\; \frac{1}{\max|\kappa_i|},$$

where κ_i denote the principal curvatures of the surface. Here "locally" means that no "global" effects take place. A global effect is for example, that two different parts of the interface come close together. These will be taken into account below. As the width of the transition area depends on εa, we can choose a small a where the curvature is high:

$$a(x,t) \;\approx\; \frac{c}{\max_i|\kappa_i(\tilde{x},t)|}\,,$$

where $\tilde{x}$ is the nearest point to x on $\Sigma(t)$. The situation is sketched in Figure 8.1: The thick line is the interface, the dashed line denotes the strip, and the shaded area is an acceptable transition area.

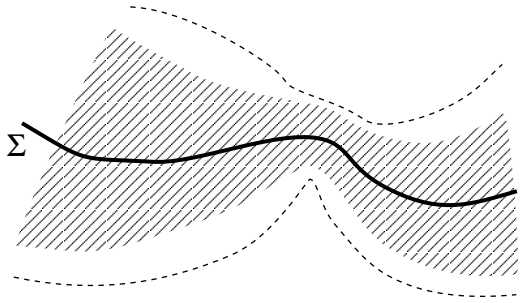

Figure 8.1: Situation around an interface without global effects; only curvature affects the transition area

Computation of the curvature: A good way to compute the curvature of the zero level set is to use the Allen–Cahn equation once more. With a vanishing right hand side, the solution evolves according to its mean curvature, and the time derivative approaches the mean curvature:

$$\partial_t \chi_\varepsilon \;\approx\; \kappa |\nabla \chi_\varepsilon| \quad \text{on } \Sigma\,.$$

Numerical experiments show, that

$$\kappa_h \;:=\; \frac{x_h^{m+1} - x_h^m}{\tau |\nabla x_h^m|}\,,$$

where x_h^{m+1} is computed by an explicit time step for the homogenous Allen–Cahn equation, is a good approximation to the curvature κ of the interface Σ^m.
This algorithm is used in mesh elements near the zero level set (in those elements where the sign of X changes) to compute the curvature of the discrete interface. Such information is distributed over the whole mesh in order to be able to calculate a local meshsize everywhere.

Global bounds for the transition area: There exist situations where curvature alone does not determine the width of the strip around the interface where the distance function is smooth. The most important situation is depicted in Figure 8.2. Two parts of the interface with relatively small curvature are near each other. Here, the width of the strip is half the distance between these parts, and the transition area should be more narrow than the strip.
The idea for determination of such situations is to test if different parts of the transition area are nearly touching each other. If this is found, the density function a is reduced in both parts, which results in a narrower transition area after a few time steps.
In order to be able to find places, where different parts of the transition area touch, we can store with every element of the triangulation its nearest point on the interface, or a direction towards that point. As criterion for such a place, we can check if two elements of the transition area are near each other, but their associated directions differ a lot (as the scalar product is negative, for example).
Once such elements have been located, the bound for the density function a has to be distributed over the regions of interest. This can be done by first setting the density at both interface parts, and then distributing it over the whole mesh as described above.

A similar situation for global restictions on the transition area comes from boundary conditions: If reflection symmetry or axial symmetry is valid for a part of the boundary, the transition area

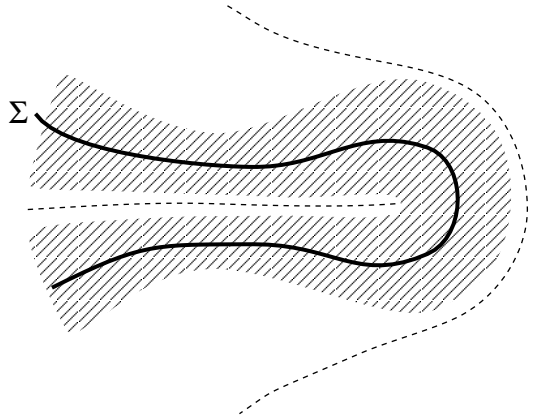

Figure 8.2: Situation around an interface with global effects

should not be allowed to extend up to this boundary, see Figure 8.3. As such symmetries are known a priori as data of the problem, this situation can be checked.

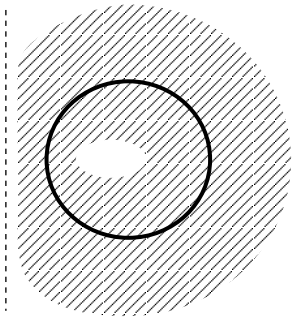

Figure 8.3: Symmetry boundary (dashed) on the left hand of a circle. The transition area is restricted.

9 A parametric finite element method for MCF

Time discretization: We start from the equation for the mean curvature flow with a driving force g, where the family of surfaces $\Sigma(t)$ is parametrized by local charts $\Phi_i(t) : \Omega_i \to \mathbf{R}^n$, $i \in I$, compare Section 6.2. The parameter domains $\Omega_i \subset \mathbf{R}^{n-1}$ are independent of t. ν_Σ denotes the unit normal of $\Sigma(t)$.

$$\frac{\partial}{\partial t}\Phi_i - \underline{\Delta}_{\Sigma(t)}\Phi_i = (g\,\nu_\Sigma)\circ\Phi_i \quad \text{in } \Omega_i \times (0, T_{\max}),\ i \in I. \tag{9.1}$$

With discrete time values $0 = t_0 < t_1 < t_2 < \ldots \le T_{\max}$ and time steps $\tau_m := t_{m+1} - t_m$, using an implicit Euler discretization of the time derivative, we derive semidiscrete equations for the charts Φ_i^m, $m = 0, 1, \ldots$, approximating $\Phi_i(t_m)$ and manifolds Σ^{m+1} which approximate $\Sigma(t_{m+1})$:

$$\begin{aligned} \Phi_i^0 &= \Phi_i(0)\,, \\ \frac{\Phi_i^{m+1} - \Phi_i^m}{\tau_m} - \underline{\Delta}_{\Sigma^m}\Phi_i^{m+1} &= g\,\nu_{\Sigma^m} \qquad \text{in } \Omega_i,\ i \in I,\ m = 1, 2, \ldots\,, \\ \Sigma^{m+1} &:= \bigcup_{i\in I} \Phi_i^{m+1}(\Omega_i)\,. \end{aligned}$$

This discretization is only implicit in the sense, that the Laplace–Beltrami operator $\underline{\Delta}_{\Sigma^m}$ is applied to the unknown parametrization Φ_i^{m+1}, but $\underline{\Delta}_{\Sigma^m}$ itself is still defined by the known surface Σ^m. We can leave now the local parametrizations of the manifolds over domains $\Omega_i \subset \mathbf{R}^{n-1}$ and use a *global* parametrization of the next manifold Σ^{m+1} over the previous manifold Σ^m. We define $\Phi^{m+1} :\ \Sigma^m \to \Sigma^{m+1}$ by $\Phi^{m+1}(\Phi_i^m(y)) := \Phi_i^{m+1}(y)$ for $y \in \Omega_i$. This definition is independent of the choice of local charts. Now we get the following semidiscrete equation:

$$\begin{aligned} \frac{\Phi^{m+1} - x}{\tau_m} - \underline{\Delta}_{\Sigma^m}\Phi^{m+1} &= g(x)\,\nu_{\Sigma^m}(x) \qquad \text{on } \Sigma^m\,, \\ \Sigma^{m+1} &:= \Phi^{m+1}(\Sigma^m). \end{aligned}$$

Thus, we have converted the *nonlinear* differential equation (9.1) into a sequence of *linear* equations.

Parametric finite element discretization: Multiplication with a test function and integration by parts leads to the following weak formulation. For $m = 0, 1, 2, \ldots$, find a parametrization $\Phi^{m+1} \in H^1(\Sigma^m)^n$ which solves

$$\int_{\Sigma^m} \frac{\Phi^{m+1} - x}{\tau_m}\,\psi + \int_{\Sigma^m} \underline{\nabla}_{\Sigma^m}\Phi^{m+1}\,\underline{\nabla}_{\Sigma^m}\psi = \int_{\Sigma^m} g\,\nu_{\Sigma^m}\,\psi \quad \text{for all } \psi \in H^1(\Sigma^m)\,. \tag{9.2}$$

For a finite element discretization of (9.2), we first need a discretization of the manifold Σ^m. We are able to construct a discretization Σ_h^0 of Σ^0 by a conforming (globally continuous) mesh of parametric n-simplices of polynomial order k (i. e. curved simplices, if $k > 1$), based on triangulations of the parameter domains Ω_i, $i \in I$. Each simplex interpolates Σ^0 at a set of Lagrange nodes corresponding to the polynomial degree k. This discretization can be constructed

by polynomial Lagrange interpolation of local charts; see [40, 49] for details. For $m > 0$, a discretization Σ_h^m of Σ^m is generated from Σ_h^{m-1}, as we will show later.
We use isoparametric finite elements over Σ_h. Let $P_{\hat{T}}$ be a finite dimensional function space on the n dimensional unit simplex $\hat{T}$ satisfying $P_k \subset P_{\hat{T}}$ (usually, we take $P_{\hat{T}} = P_k$). For every $T \subset \Sigma_h^m$ let $F_T : \hat{T} \to T$ be the (polynomial) parametrization. Using this notation, we define the isoparametric finite element space

$$W_h^m \;:=\; \{v_h \in C^0(\Sigma_h^m, \mathbf{R}) \;:\; (v_h \circ F_T) \in P_{\hat{T}} \;\text{ for all } T \in \Sigma_h^m\}.$$

In [62], the Sobolev space $H^1(\Sigma)$ is defined for a Lipschitz manifold Σ. This definition applies here, as our discretization Σ_h^m is Lipschitz continuous, and we get $W_h^m \subset H^1(\Sigma_h^m)$.
The choice $P_{\hat{T}} = P_k$, yielding same order polynomials for the discretization and the finite element functions, is natural in the sense that we can expect optimal error estimates [13].
Analogous to (9.2), we now look for a weak solution in the finite element space W_h^m on Σ_h^m. This gives an approximation Φ_h^{m+1} to the parametrization Φ^{m+1} and defines a discrete manifold Σ_h^{m+1} approximating Σ^{m+1}.
Let Σ_h^0 be a given triangulation of Σ_0. For $m = 0, 1, 2, \ldots$, the problem is to find a finite element solution $\Phi_h^{m+1} \in W_h(\Sigma_h^m)^n$ of the equation

$$\begin{aligned} \int_{\Sigma_h^m} \frac{\Phi_h^{m+1}(x) - x}{\tau^m}\,\psi_h \;+\; \int_{\Sigma_h^m} \nabla\Phi_h^{m+1}\,\nabla\psi_h \quad &= \quad \int_{\Sigma_h^m} g\,\nu\,\psi_h \quad \text{for all } \psi_h \in W_h(\Sigma_h^m)\,, \\ \Sigma_h^{m+1} \quad &:= \quad \Phi_h^{m+1}(\Sigma_h^m)\,. \end{aligned} \tag{9.3}$$

For each time step this leads to a decoupled system of n linear systems of equations for the components of the parametrization. The matrices of the n linear systems are all equal, but right hand sides differ between the components.

Anisotropic motion: Anisotropic motion by mean curvature can be implemented in several ways. One possibility is to use given anisotropy functions $\varepsilon_V, \varepsilon_C : S^{n-1} \to \mathbf{R}^+$ which depend on the direction of the surface normal and look for a family of surfaces which solve the problem

$$\begin{aligned} \varepsilon_V\left(\nu_{\Sigma(t)}(x)\right) V_{\Sigma(t)}(x) + \varepsilon_C\left(\nu_{\Sigma(t)}(x)\right) C_{\Sigma(t)}(x) \;&=\; g(x,t) \\ &\qquad \text{for all } x \in \Sigma(t),\; t \in (0, T_{\max})\,, \\ \Sigma(0) \;&=\; \Sigma_0\,. \end{aligned} \tag{9.4}$$

With the same techniques as above, this leads to the finite element problems

$$\begin{aligned} \int_{\Sigma_h^m} \frac{\varepsilon_V(\nu_{\Sigma^m})}{\varepsilon_C(\nu_{\Sigma^m})}\,\frac{\Phi_h^{m+1}(x) - x}{\tau^m}\,\psi_h \;+\; \int_{\Sigma_h^m} \nabla\Phi_h^{m+1}\,\nabla\psi_h \quad &= \quad \int_{\Sigma_h^m} \frac{1}{\varepsilon_C(\nu_{\Sigma^m})}\,g\,\nu\,\psi_h \\ &\qquad \text{for all } \psi_h \in W_h(\Sigma_h^m)\,, \\ \Sigma_h^{m+1} \quad &:= \quad \Phi_h^{m+1}(\Sigma_h^m)\,. \end{aligned} \tag{9.5}$$

Another approach is to use the gradient flow of the anisotropic surface energy

$$E_f(\Sigma) \;:=\; \int_\Sigma f(\nu_\Sigma(x))\,dx$$

with an anisotropy function $f : S^{n-1} \to \mathbf{R}^+$. The isotropic mean curvature flow is the gradient flow for $f \equiv 1$. The gradient flow for E_f leads to the weak formulation

$$\int_{\Sigma(t)} x_t \psi + \int_{\Sigma(t)} f(\nu_\Sigma)\, \nabla x \cdot \nabla \psi - \nabla f(\nu_\Sigma)\, \nabla x \cdot \nabla \psi\, \nu_\Sigma \;=\; \int_{\Sigma(t)} g\, \nu_\Sigma\, \psi \tag{9.6}$$

$$\text{for all } \psi \in H^1(\Sigma(t), \mathbf{R}^n).$$

Discretizations in time and space can be done in the same way as shown above. A coefficient $\varepsilon_V(\nu_\Sigma)$ can be added as well. In two dimensions, using $\tilde{f}(\alpha) := f(\cos\alpha, \sin\alpha)$, (9.6) is equivalent to (9.4) with $\varepsilon_V \equiv 1$ and $\varepsilon_C(\cos\alpha, \sin\alpha) = (\tilde{f} + \tilde{f}'')(\alpha)$.

Convergence results: Dziuk [20] proved convergence for the semi–discrete method (only discretization in space) in two dimensions (for curves). For the same scheme including anisotropy, convergence is proved in [19].

Adaptive methods: The discretization of the surface has to be adapted in time, to be able to resolve the evolving structures. This is done mainly by local refinement and coarsening, driven by an a posteriori error indicator. As the evolution equation is similar to a nonlinear heat equation, it seems appropriate to use error indicators which were derived in the heat equation context. In our case, this leads to a geometrical condition

$$\eta_T(\Sigma_h) \;:=\; h_T^{3/2} \|[\nu_\Sigma]\|_{L_2(\partial T)} + h_T^2 \left\| \frac{g}{\varepsilon_C} \right\|_{L_2(T)},$$

which involves the jumps of the surface normals over edges of the discrete surface and the forcing term g. Refinement and coarsening of the triangulation can now be done as usual.
Depending on the forcing term and the resulting evolution of the surface, complex structures can develop from a small part of an initially smooth surface. As the grid points of the discretization are mainly moved into the direction of the surface normal, the distances of the vertices vary from one time step to the other. The distances get smaller at locally concave parts of the solid phase and grow where the interface is convex. In the three dimensional case, there may evolve parts of the surface, where the two principal curvatures differ a lot (one positive and the other negative, for example). The grid gets strongly deformed after some time. The largest angles of some triangles may tend to 180 degrees, which would enlargen the errors in the numerical method. We have to prevent the mesh from degenerating too much and take special care of those triangles.
Unfortunately, obtuse angles which develop during deformation of the mesh cannot be prevented refinement and coarsening alone, because of the possible distortion of the mesh during the evolution. To avoid a degenerate grid, we have to use additional grid modification procedures. These are based mainly on angle–dependent refinement and displacement of the vertices on the surface. Altogether, the algorithm is an adaptive h–r–method for the discretization and movement of the curved free boundary [57].

In the following, we describe a method for displacement of the grid nodes on the (discrete) surface which leads to good results.
The coordinates of the vertices (and midpoints of the edges) are changed in a way, that makes the triangulation "better". We use a single–stepping algorithm that moves only one vertex at a

time, and loops through all nodes of the grid. Depending on the quality criterium for the grid, the direction of node movement is chosen.
For each node P of the triangulation, we define a functional $F(P)$ that measures the local mesh quality, and try to maximize F by changing the coordinates of P. To this aim, we calculate the gradient $\frac{d}{dP}F(\bar{P})$ at the old position $\bar{P}$ of the node and search the maximum of F in the direction of the gradient, by a bisectioning line search algorithm. As we want to vary the nodes only *on* the surface, not in the whole space, we have to project all points from the descent line onto the discrete surface during the maximization process. During the iterations of the node movement algorithm, we have to remember the old positions of all nodes, because we need them for the projection onto the discrete surface defined by these values.
We use a functional F, which works on the angles of the triangulation; it minimizes the difference between the N angles around a vertex P. It is based on the conformal energy (see [38] for definition and results) of a parametrization $\Phi : D_N \to \{T \subset \Gamma_h : T \cap P \neq \emptyset\}$ of the neighbourhood of P over the unit N–polygon D_N (compare Figure 9.1). We set

$$F(P) \;:=\; 2 \int\limits_{D_N} |\Phi_u \wedge \Phi_v| \;-\; \int\limits_{D_N} |\nabla \Phi|^2 \,.$$

F is maximal ($= 0$) if Φ is a conformal mapping; in this case all angles around P are equal. Straightforward computation gives

$$\begin{aligned} F(P) \;&=\; \sum_{j=1}^{N} \left(\frac{|Q_j - P|^2 + |Q_{j+1} - P|^2 - 2\,\cos(\alpha)\,\langle Q_j - P,\, Q_{j+1} - P\rangle}{2\,\sin(\alpha)} \right. \\ &\qquad\qquad \left. - |(Q_j - P) \wedge (Q_{j+1} - P)| \right) \qquad \text{and} \\ \frac{d}{dP}F(P) \;&=\; \sum_{j=1}^{N} \left(\frac{1-\cos(\alpha)}{\sin(\alpha)} \Big((P - Q_{j+1}) + (P - Q_j) \Big) \right. \\ &\qquad\qquad \left. - \left(\frac{(Q_j - P) \wedge (Q_{j+1} - P)}{|(Q_j - P) \wedge (Q_{j+1} - P)|} \wedge (Q_j - Q_{j+1}) \right) \right) , \end{aligned}$$

where Q_j, $j = 1, \ldots, N$, are the neighbouring vertices to P and $\alpha = 2\pi/N$.

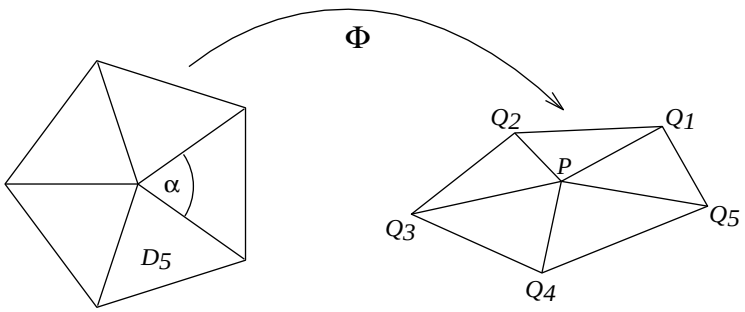

Figure 9.1: Unit 5–polygon D_5 and parametrization of a neighbourhood of P

10 A parametric FEM for the Stefan problem with surface tension

In [57] we describe a parametric finite element algorithm for the Stefan problem with surface tension, Problem 5.1. It will be an explicitely coupled algorithm of the mean curvature flow method for the motion of the interface, where the discrete temperature appears on the right hand side, and a heat equation solver, where the discrete interface and its curvature appear. The finite element method for discretization of the mean curvature flow of the free boundary $\Sigma(t)$ was already presented above in Section 9, Equation (9.5). The finite element method for the heat equation is described next.

Discretization of the heat equation: We assume in the moment, that the interface $\Sigma(t)$ is known. After an implicit Euler time discretization with time step size τ_m we combine equations (5.5)–(5.7) in a weak formulation. Assuming that θ^m is known, Equation (5.5) leads to

$$\int\limits_\Omega \frac{\theta^{m+1}-\theta^m}{\tau_m}\,\varphi + D_0 \int\limits_\Omega \nabla\theta^{m+1}\nabla\varphi \;=\; \int\limits_{\Sigma(t_{m+1})} V_\Sigma\,\varphi \qquad \text{for all } \varphi \in H_0^1(\Omega)\,,$$

as integration by parts produces the jump term of (5.6) on $\Sigma(t_{m+1})$. Using Equation (5.7), which gives a relationship between the velocity V_Σ, the curvature C_Σ and the temperature at the free boundary, we get for all $\varphi \in H_0^1(\Omega)$:

$$\int\limits_\Omega \frac{\theta^{m+1}-\theta^m}{\tau_m}\,\varphi + D_0 \int\limits_\Omega \nabla\theta^{m+1}\nabla\varphi + \int\limits_{\Sigma(t_{m+1})} \frac{1}{\varepsilon_V}\,\theta^{m+1}\,\varphi \;=\; -\int\limits_{\Sigma(t_{m+1})} \frac{\varepsilon_C}{\varepsilon_V}\,C_\Sigma\,\varphi\,.$$

The integral over $\Sigma(t_{m+1})$ on the left–hand side leads to a more stable discretization, as we use an implicit discretization for the temperature on the free boundary and $1/\varepsilon_V$ is positive.
Let $\mathcal{T}_\Omega^m$ be a triangulation of the domain Ω into n–simplices (triangles or tetrahedra), which will be used during the m–th time step, and $V_h^m \subset H_0^1(\Omega)$ the corresponding finite element space. Then the discrete problem is:
Given smooth surfaces $\Sigma(t_m)$, $m = 0, 1, \ldots$, with mean curvature $C_{\Sigma(t_m)}$, boundary values $\theta_0 \in C^0(\bar\Omega \times [0, T_{\max}])$ and discrete initial values $\theta_h^0 \in V_h^0$, find for each $m = 0, 1, 2, \ldots$, finite element temperatures $\theta_h^{m+1} \in V_h^{m+1}$ which solve

$$\int\limits_\Omega \frac{\theta_h^{m+1}-\theta_h^m}{\tau_m}\varphi_h + D_0 \int\limits_\Omega \nabla\theta_h^{m+1}\nabla\varphi_h + \int\limits_{\Sigma(t_{m+1})} \frac{1}{\varepsilon_V}\,\theta_h^{m+1}\,\varphi_h = -\int\limits_{\Sigma(t_{m+1})} \frac{\varepsilon_C}{\varepsilon_V}\,C_{\Sigma(t_{m+1})}\,\varphi_h \tag{10.1}$$

for all $\varphi_h \in V_{h,0}^{m+1}$, with boundary values $\theta_h^{m+1} = I_h^{m+1}(\theta_0(\cdot, t_{m+1}))$ on $\partial\Omega$.
The discretizations for the free boundary evolution and the heat equation will be combined below to a numerical method for the solution of Problem 5.1.

Adaptive method: Similar to the methods from [24], we obtain an a posteriori L_2 error estimate for the solution of the associate elliptic problem (Equation (10.1) without the temporal differential quotient), assuming a smooth interface $\Sigma(t)$ and regularity of u for all $t \in [0, T]$,

$$\theta(\cdot, t) \in H^{1,\infty}(\Omega) \cap H^{2,2}(\Omega \setminus \Sigma(t)),$$

with a corresponding a priori estimate (see [12] for estimates for the anisotropic parabolic problem). Let

$$\eta_T(\theta_h) := \begin{cases} \left(D_0^2 h^4 \int\limits_T |\Delta\theta_h|^2 + \frac{D_0^2}{2} h^3 \int\limits_{\partial T \setminus \partial\Omega} \left| \left[\frac{\partial \theta_h}{\partial \nu_T} \right] \right|^2 \right)^{\frac{1}{2}} & \text{if } T \cap \Sigma = \emptyset, \text{ and else} \\ \left(h^2 \int\limits_{\Sigma \cap T} \left(\frac{\varepsilon_C C_\Sigma + \theta_h}{\varepsilon_V} \right)^2 + D_0^2 h^3 \int\limits_T |\Delta\theta_h|^2 + \frac{D_0^2}{2} h^2 \int\limits_{\partial T \setminus \partial\Omega} \left| \left[\frac{\partial \theta_h}{\partial \nu_T} \right] \right|^2 \right)^{\frac{1}{2}} . \end{cases}$$

The lower h exponents near Σ account for the loss of regularity of the solution of the dual problem at the free boundary. We can prove (for the elliptic problem) the a posteriori error estimate

$$\|\theta - \theta_h\|_{L_2(\Omega)} \leq c_\eta \eta(\theta_h) := c_\eta \Big(\sum_{T \in \mathcal{T}_\Omega} \eta_T(\theta_h)^2 \Big)^{1/2} .$$

under the above regularity assumption. The values $\eta_T(\theta_h)$ are used by the adaptive method to drive the refinement and coarsening of the temperature mesh.

Computation of the curvature: The right–hand side of the heat equation (10.1) uses the value of the mean curvature C_Σ of the free boundary $\Sigma(t_{m+1})$. As we approximate the free boundary Σ by a piecewise polynomial, globally Lipschitz–continuous discrete free boundary Σ_h^{m+1}, there is no straightforward definition of a curvature for Σ_h^{m+1}. In the sequel, we use the notation Σ_h for the free boundary and drop the time step index.
One possibility is to compute a discrete curvature using again the identity $C_\Sigma \, \nu_\Sigma = -\underline{\Delta}_\Sigma \mathrm{id}_\Sigma$, which is true for smooth manifolds. With a weak formulation and H^{-1}–projection we define a vector–valued curvature $\vec{C}_h \in W_h(\Sigma_h)^n$ by

$$\left(\vec{C}_h, \psi_h \right)_{L_2(\Sigma_h)} = \left(\underline{\nabla}_{\Sigma_h} \mathrm{id}_{\Sigma_h}, \underline{\nabla}_{\Sigma_h} \psi_h \right)_{L_2(\Sigma_h)} \qquad \text{for all } \psi_h \in W_h .$$

Using approximate normals ν_h at the grid vertices, the discrete scalar mean curvature is now defined by

$$C_h := I_h \left(\langle \vec{C}_h, \nu_h \rangle \right)$$

as the piecewise linear interpolate of the scalar product of the discrete curvature vector with the approximate normals.

Other approaches to a definition of a curvature of the discrete interface use a weak formulation of the second fundamental form or an area variation formula.

Adaptive algorithm for the Stefan problem: The adaptive methods for the heat equation and the interface motion are combined, defining an algorithm for the numerical solution of Problem 5.1.
In each time step, given the old values θ_h^m and Σ_h^m, we first move the interface Σ_h to a new position $\tilde{\Sigma}_h^{m+1}$ by solving Equation (9.3) with temperature θ_h^m on the right hand side:

$$\int\limits_{\Sigma_h^m} \frac{\varepsilon_V}{\varepsilon_C} \frac{\Phi_h^{m+1}(x) - x}{\tau^m} \psi_h + \int\limits_{\Sigma_h^m} \underline{\nabla} \Phi_h^{m+1} \, \underline{\nabla} \psi_h = - \int\limits_{\Sigma_h^m} \frac{1}{\varepsilon_C} \theta_h^m \, \nu \, \psi_h \qquad \text{for all } \psi_h \in W_h(\Sigma_h^m) .$$

The triangulation of the interface is adaptively refined, coarsened and changed according to the adaptive h–r–method from Section 9, which results in the new free boundary Σ_h^{m+1}. The curvature C_h^{m+1} of Σ_h^{m+1} is computed. Now, the new temperature grid $\mathcal{T}_h^{m+1}$ and the new temperature θ_h^{m+1} are computed by the semi–implicit adaptive strategy for the solution of Equation (10.1) using integrals over Σ_h^{m+1} and curvature C_h^{m+1} on the right hand side, solving for all $\varphi_h \in V_{h,0}^{m+1}$

$$\int_\Omega \frac{\theta_h^{m+1} - \theta_h^m}{\tau_m}\,\varphi_h + D_0 \int_\Omega \nabla\theta_h^{m+1}\nabla\varphi_h + \int_{\Sigma_h^{m+1}} \frac{1}{\varepsilon_V}\,\theta_h^{m+1}\varphi_h = -\int_{\Sigma_h^{m+1}} \frac{\varepsilon_C}{\varepsilon_V}\,C_h^{m+1}\varphi_h\,.$$

The temperature meshes $\mathcal{T}_h^{m+1}$ and the surface meshes Σ_h^{m+1} are both adaptively generated using information only from the corresponding a posteriori error estimates (and the interface motion); no direct relationship between them exists.
All integrals over the free boundary are computed by quadrature formulas. The coordinates of the quadrature points on the (curved) surface are computed easily from their position in the reference simplex using the local parametrization of the discrete interface. For integrals which involve θ_h or another function from the finite element space on the n dimensional temperature mesh, the positions of these quadrature points in the temperature mesh (simplex number and barycentric coordinates) can be found using local search operations, when the positions of the surface grid vertices in the temperature mesh are (approximately) known, for example from the last time step. Such a local search is done by running from one simplex to an adjacent one which is closer to the point, until all barycentric coordinates are nonnegative. Some global search operations are needed only once before the first time step, but the same technique can be used, and their number may be minimized using neighbourhood information about the vertices on the surface.

The initial temperature grid $\mathcal{T}_h^0$ is generated by an application of the adaptive finite element method to an elliptic problem like (10.1) with a *given* time derivative on the right hand side. Besides the generation of $\mathcal{T}_h^0$, computing the initial temperature θ_h^0 as the solution of this elliptic problem ensures that it is compatible with the discrete interface Σ_h^0, its discrete curvature $C_{\Sigma_h^0}$ and the quadrature formulas used.

In Figure 10.1, a flow diagram shows the whole adaptive algorithm, including the generation of an initial temperature grid. This algorithm is easily supplemented by a higher order time discretization (Richardson–extrapolation is implemented) and adaptive time step control (using the parabolic error estimates and methods from [24], for example). Because of technical details (more intermediate solutions have to be computed and combined), the algorithm looses a bit of its simple structure in these cases, and we do not present details here.

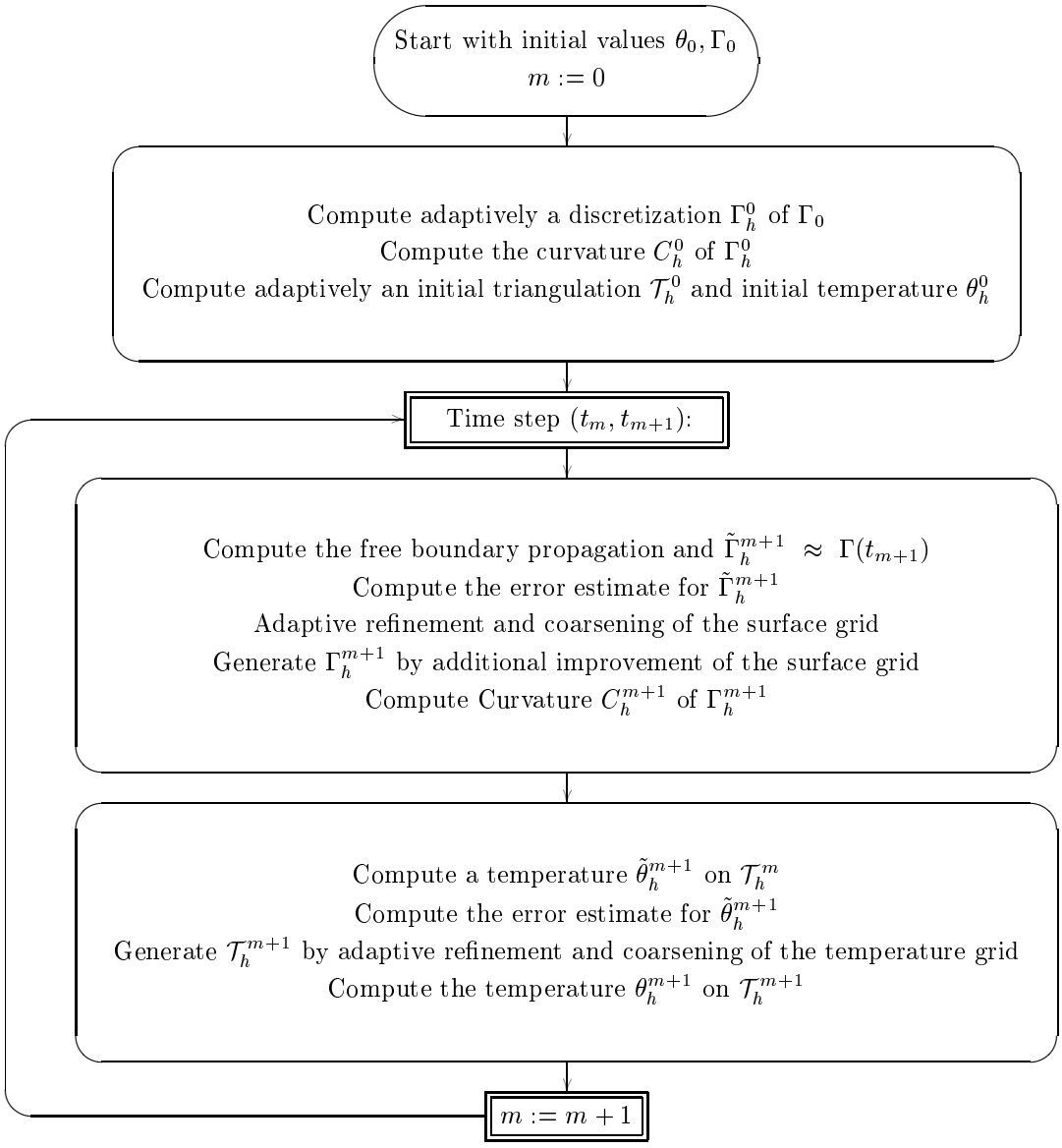

Figure 10.1: Flow diagram of the adaptive algorithm

References

[1] I. Babuška and W. Rheinboldt, *Error estimates for adaptive finite element computations*, SIAM J. Numer. Anal., 15 (1978), pp. 736–754.

[2] C. Bandle, A. Brillard, G. Dziuk, and A. Schmidt, *Course on mean curvature flow.* Lecture notes, Freiburg, 1994.

[3] E. Bänsch, *Adaptive finite element techniques for the Navier–Stokes equations and other transient problems*, in Adaptive Finite and Boundary Elements, C. A. Brebbia and M. H. Aliabadi, eds., Computational Mechanics Publications and Elsevier, 193, pp. 47–76.

[4] ——, *Local mesh refinement in 2 and 3 dimensions*, IMPACT Comput. Sci. Engrg., 3 (1991), pp. 181–191.

[5] E. Bänsch and K. G. Siebert, *A posteriori error estimation for nonlinear problems by duality techniques.* Preprint 30, Universität Freiburg, 1995.

[6] G. Bellettini and M. Paolini, *Anisotropic motion by mean curvature in the context of Finsler geometry.* Preprint, 1995.

[7] J. Bey, *Tetrehedral grid refinement.* Report 18, SFB 382 Tübingen, 1995.

[8] F. A. Borneman, *An adaptive multilevel approach to parabolic equations I*, IMPACT Comput. Sci. Engrg., 2 (1990), pp. 279–317.

[9] ——, *An adaptive multilevel approach to parabolic equations II*, IMPACT Comput. Sci. Engrg., 3 (1990), pp. 93–122.

[10] ——, *An adaptive multilevel approach to parabolic equations III*, IMPACT Comput. Sci. Engrg., 4 (1992), pp. 1–45.

[11] G. Caginalp, *An analysis of a phase–field model of a free boundary*, Arch. Rat. Mech. Anal., 92 (1986), pp. 205–245.

[12] X. Chen and F. Reitich, *Local existence and uniqueness of solutions of the stefan problem with surface tension and kinetic undercooling*, J. Math. Anal. Appl., 164 (1992), pp. 350–362.

[13] P. G. Ciarlet, *The finite element method for elliptic problems*, North-Holland, 1987.

[14] P. Clément, *Approximation by finite element functions using local regularization*, R. A. I. R. O., 9 (1975), pp. 77–84.

[15] L. Demkowicz, J. T. Oden, W. Rachowicz, and O. Hardy, *Toward a universal h–p adaptive finite element strategy, Part 1 – Part 3*, Comp. Methods Appl. Mech. Engrg., 77 (1989), pp. 79–212.

[16] W. Dörfler, *A robust adaptive strategy for the nonlinear poisson equation*, Computing, 55 (1995), pp. 289–304.

[17] ——, *A convergent adaptive algorithm for poisson's equation*, SIAM J. Numer. Anal., 33 (1996), pp. 1106–1124.

[18] ——, *A time- and spaceadaptive algorithm for the linear time-dependent Schrödinger equation*, Numer. Math., 73 (1996), pp. 419–448.

[19] G. Dziuk, *Convergence of a semi–discrete scheme for the anisotropic curve shortening flow.* In preparation.

[20] ——, *Convergence of a semi discrete scheme for the curve shortening flow*, Math. Meth. Appl. Sc., 4 (1994), pp. 589–606.

[21] I. Ekeland and R. Temam, *Analyse convexe et problèmes variationelles*, Dunod Gauthier-Villars, 1974.

[22] C. M. Elliott, *On the finite element approximation of an elliptic variational inequality arising from an implicit time discretization of the Stefan problem*, IMA J. Numer. Anal., 1 (1981), pp. 115–125.

[23] C. M. Elliott and R. Schätzle, *The limit of the anisotropic double–obstacle allen–cahn equation.* CMAIA Report 95-05, Univ. of Sussex, 1995.

[24] K. Eriksson and C. Johnson, *Adaptive finite element methods for parabolic problems I: A linear model problem*, SIAM J. Numer. Anal., 28 (1991), pp. 43–77.

[25] L. C. Evans, H. M. Soner, and P. E. Souganidis, *Phase transitions and generalized motion by mean curvature*, Comm. Pure and Appl. Math, 45 (1992), pp. 1097–1123.

[26] L. C. Evans and J. Spruck, *Motion of level sets by mean curvature I*, J. Diff. Geom., 33 (1991), pp. 635–681.

[27] M. Fried, *Computation of viscosity solutions for mean curvature flow.* In preparation.

[28] ——, *Berechnung des Krümmungsflusses von Niveauflächen.* Thesis, Freiburg, 1993.

[29] J. Fröhlich, J. Lang, and R. Roitzsch, *Selfadaptive finite element computations with smooth time controller and anisotropic refinement.* Preprint SC 96-16, ZIB Berlin, 1996.

[30] D. Gilbarg and N. S. Trudinger, *Elliptic partial differential equations of second order*, Springer, 1983.

[31] M. E. Glicksman, R. J. Schaefer, and J. D. Ayers, *Dendritic growth — a test of theory*, Metal. Trans. A, 7A (1976), pp. 1747–1759.

[32] R. Glowinski, *Numerical Methods for Nonlinear Variational Problems*, Springer, 1984.

[33] M. E. Gurtin, *Toward a nonequilibrium thermodynamics of two–phase materials*, Arch. Ration. Mech. Anal., 100 (1988), pp. 275–312.

[34] W. Hackbusch, *Theorie und Numerik elliptischer Differentialgleichungen*, Teubner, 1986.

[35] R. W. Hoppe, *A globally convergent multi-grid algorithm for moving boundary problems of two-phase Stefan type*, IMA J. Numer. Anal., 13 (1993), pp. 235–253.

[36] R. W. Hoppe and R. Kornhuber, *Adaptive multilevel methods for obstacle problems*, SIAM J. Numer. Anal., 31 (1994), pp. 301–323.

[37] G. Huisken, *Flow by mean curvature of convex surfaces into spheres*, J. Diff. Geom., 20 (1984), pp. 237–266.

[38] J. E. Hutchinson, *Computing conformal maps and minimal surfaces*, Proc. C.M.A., Canberra, 26 (1991), pp. 140–161.

[39] H. Jarausch, *On an adaptive grid refining technique for finite element approximations*, SIAM J. Sci. Stat. Comput., 7 (1986), pp. 1105–1120.

[40] K. Kalik and W. Wendland, *The approximation of closed manifolds by triangulated manifolds and the triangulation of closed manifolds*, Computing, 47 (1992), pp. 255–275.

[41] R. Kornhuber, *Monotone multigrid methods for elliptic variational inequalities I*, Numer. Math., 69 (1994), pp. 167–184.

[42] ——, *Adaptive monotone multigrid methods for some non–smooth optimization problems.* Preprint 156, WIAS Berlin, 1995.

[43] ——, *Monotone multigrid methods for elliptic variational inequalities II*, Numer. Math., 72 (1996), pp. 481–500.

[44] R. Kornhuber and R. Roitzsch, *On adaptive grid refinement in the presence of internal or boundary layers*, IMPACT Comput. Sci. Engrg., 2 (1990), pp. 40–72.

[45] I. Kossaczký, *A recursive approach to local mesh refinement in two and three dimensions*, J. Comput. Appl. Math., 55 (1994), pp. 275–288.

[46] J. S. Langer, *Instabilities and pattern formation in crystal growth*, Rev. Modern Phys., 52 (1980), pp. 1–28.

[47] J. M. Maubach, *Local bisection refinement for n-simplicial grids generated by reflection*, SIAM J. Sci. Comput., 16 (1995), pp. 210–227.

[48] W. Mitchell, *A comparison of adaptive refinement techniques for elliptic problems*, ACM Trans. Math.Softw., 15 (1989), pp. 326–347.

[49] J. C. Nedelec, *Curved finite element methods for the solution of integral singular equations on surfaces in* $\mathbf{R}^3$, Comput. Meth. Appl. Mech. Engrg., 8 (1976), pp. 61–80.

[50] J. Nitsche, *Ein Kriterium für die Quasi–Optimalität des Ritzschen Verfahrens*, Numer. Math., 11 (1968), pp. 346–348.

[51] R. H. Nochetto, M. Paolini, and C. Verdi, *An adaptive finite element method for two-phase stefan problems in two space dimensions. Part II: Implementation and numerical experiments*, SIAM J. Sci. Stat. Comput., 12 (1991), pp. 1207–1244.

[52] ——, *Double obstacle formulation with variable relaxation parameter for smooth geometric front evolutions: Asymptotic interface error estimates*, Asymptotic Anal, 10 (1995), pp. 173–198.

[53] ——, *A dynamic mesh algorithm for curvature dependent evolving interfaces*, J. Comput. Phys., 123 (1996), pp. 296–310.

[54] R. H. Nochetto, A. Schmidt, and C. Verdi, *A posteriori error estimation and adaptivity for Stefan problems.* In preparation.

[55] R. H. Nochetto and C. Verdi, *Convergence past singularities for a fully discret approximation of curvature driven interfaces.* Quaderno 9/1994, Milano, 1994.

[56] M. Paolini and C. Verdi, *Asymptotic and numerical analyses of the mean curvature flow with a space-dependent relaxation parameter*, Asymptotic Analysis, 5 (1992), pp. 553–574.

[57] A. Schmidt, *Computation of threedimensional dendrites with finite elements*, J. Comput. Phys., 125 (1996), pp. 293–312.

[58] A. Schmidt and K. G. Siebert, *ALBERT: An adaptive hierarchical finite element toolbox.* In preparation.

[59] K. G. Siebert, *A posteriori error estimator for anisotropic refinement*, Numer. Math., 73 (1996), pp. 373–398.

[60] R. Verfürth, *A posteriori error estimates for nonlinear problems: Finite element discretization of elliptic equations*, Math. Comp., 62 (1994), pp. 445–475.

[61] ——, *A posteriori error estimation and adaptive mesh-refinement techniques*, J. Comp. Appl. Math., 50 (1994), pp. 67–83.

[62] J. Wloka, *Partial differential equations*, Cambridge University Press, 1987.

[63] O. C. Zienkiewicz, D. W. Kelly, J. Gago, and I. Babuška, *Hierarchical finite element approaches, error estimates and adaptive refinement*, in The mathematics of finite elements and applications IV, J. Whiteman, ed., Academic Press, 1982, pp. 313–346.

Acknowledgements. Section 6 was taken from [2]. We thank Michael Fried for supplying Section 7.