

Adaptive Quadratures Over Volumes

Kurt Georg^{1,2,4} and Ralf Widmann^{3,4,5}

August 1990

Abstract

We consider the numerical approximation of volume integrals over bounded domains $D := \{x \in \mathbf{R}^3 : H(x) \leq 0\}$, where $H : \mathbf{R}^3 \rightarrow \mathbf{R}$ is a suitable decidability function. The integrands may be smooth maps or singular maps such as those arising in the volume potentials for boundary integral methods. An adaptive extrapolation method is described which is based on some simple quadrature rules. It utilizes an automatic simplicial subdivision of the domain. The method offers improvements over recently given approaches. A special version is offered for the important application of the numerical computation of volume potentials in boundary integral methods. Several examples illustrate the performance of the method.

Keywords. volume integrals, adaptive integration, extrapolation method, boundary integral method, quadrature formula, trapezoidal rule

AMSMOS. 65D30, 65R20, 65N35, 45E05

1 Introduction

Let us motivate the following discussion by considering a well known integral equation related to Poisson's Equation:

$$\begin{aligned}\Delta u &= f \text{ in } \tilde{D}, \\ u &= g \text{ on } \tilde{B},\end{aligned}$$

where $\tilde{D} \subset \mathbf{R}^3$ is a bounded open nonempty domain with sufficiently regular boundary $\tilde{B} = \partial\tilde{D}$. Following [5], we utilize the fundamental solution

$$s(x, y) := \frac{1}{4\pi} \frac{1}{\|x - y\|}$$

to obtain a particular solution (which does not in general satisfy the above boundary condition) via the Poisson integral formula

$$v(x) := \int_D s(x, y) f(y) dy, \tag{1}$$

¹Partially supported by the National Science Foundation under grant number DMS-8947761

²Partially supported by the Deutsche Forschungsgemeinschaft, SFB 256

³Supported by a postdoctoral stipend of the Deutsche Forschungsgemeinschaft

⁴Department of Mathematics, Colorado State University, Ft. Collins, Colorado 80523

⁵Mathematisches Institut A, Universität Stuttgart, 7000 Stuttgart 80

where D is usually a more convenient region containing \tilde{D} . We make the substitution

$$w := u - v$$

and obtain the homogenous equation

$$\begin{aligned} \Delta w &= 0 && \text{in } \tilde{D}, \\ w &= g - v && \text{on } \tilde{B}. \end{aligned}$$

Now we consider the double layer potential ansatz for a solution w of the above equation:

$$w(x) = \int_{\tilde{B}} \frac{\partial s(x, y)}{\partial \mathbf{n}(y)} \sigma(y) \mu(dy),$$

where μ denotes the standard measure of surface area, $\mathbf{n}(y)$ for $y \in \tilde{B}$ is the unit normal vector pointing out of \tilde{D} , and σ is an unknown density function on \tilde{B} to be determined by the boundary integral method. It turns out, see, e.g., [12], that σ satisfies the following integral equation of the second kind:

$$\sigma(x) - 2 \int_{\tilde{B}} \frac{\partial s(x, y)}{\partial \mathbf{n}(y)} \sigma(y) \mu(dy) = -2g(x) + 2v(x)$$

for $x \in \tilde{B}$. In order to exploit the above equation for numerical purposes via a boundary element method, we need an efficient method for approximating the volume integrals (1). Performing this task is one of the aims of our paper.

More generally, we investigate the numerical quadrature of integrals of the form

$$\int_D f(x) dx, \tag{2}$$

where

$$D = \{x \in \mathbf{R}^3 : H(x) \leq 0\} \tag{3}$$

is a given, implicitly defined volume in \mathbf{R}^3 . The integrand is allowed to be weakly singular, and the boundary of D will usually be piecewise smooth.

A first approach, based on [11], for handling such problems was given in [2], where only an adaptive use of the trapezoidal rule was presented, and a very coarse adaptation to the boundary was employed. As suggested there, see [2, sec. 6], we now present a much more sophisticated and complex algorithm which combines several features in a hybrid structure:

1. A refined approximation of the boundary of D is used in basic integration formulae.
2. These basic integration formulae are employed in an adaptive refinement procedure.
3. Extrapolation is used whenever possible to improve accuracy and efficiency.

It turns out that a precise handling of a complex boundary structure is very time consuming. We therefore also offer a simplified version of our method for simple boundaries like cubes or tetrahedra. This will be of particular interest for the boundary integral method outlined above.

2 Piecewise-Linear Approximations of Bodies

In this section we briefly describe the Coxeter-Freudenthal triangulation in \mathbf{R}^3 , see [6, 10], and show how it can be used to approximate D . More sophisticated triangulations may be used as well; extensive discussions of triangulations of \mathbf{R}^n for general n may be found e.g., in the books [1, 9, 15]. Important aspects of the triangulation which are of concern for our discussion are: efficient storing, comparing and recovering of the simplices.

Before we describe the Coxeter-Freudenthal triangulation, let us for completeness, give the definitions of the concepts and notation we will use. A *tetrahedron* in \mathbf{R}^3 is the convex hull of four *vertices* v_0, v_1, v_2, v_3 in \mathbf{R}^3 which do not lie in a common plane. We will denote such a tetrahedron by $\sigma = [v_0, v_1, v_2, v_3]$. The convex hulls $[v_{i_0}, v_{i_1}, v_{i_2}]$, $[v_{i_0}, v_{i_1}]$ for distinct $i_0, i_1, i_2 \in \{0, 1, 2, 3\}$ are the *faces* and *edges* of σ respectively. Its *barycenter* is given by

$$b_\sigma := \frac{1}{4} \sum_{j=0}^3 v_j.$$

A *triangulation* \mathcal{T} of \mathbf{R}^3 is a collection of tetrahedra such that

- (i) $\sigma_1, \sigma_2 \in \mathcal{T}$ implies either $\sigma_1 \cap \sigma_2 = \emptyset$, or it is a common face, edge or vertex.
- (ii) $\bigcup_{\sigma \in \mathcal{T}} \sigma = \mathbf{R}^3$.
- (iii) Any compact subset of \mathbf{R}^3 intersects only finitely many $\sigma \in \mathcal{T}$.

The *diameter* $\text{diam } \sigma$ relative to any given norm is the maximum of the edge lengths. The *mesh size* of the triangulation \mathcal{T} is given by $\sup_{\sigma \in \mathcal{T}} \text{diam } \sigma$. The *nodes* of a triangulation \mathcal{T} are the vertices of the tetrahedra in \mathcal{T} .

A very important example of a triangulation is the *Coxeter-Freudenthal triangulation*. We used this triangulation in our examples, but we want to emphasize that other triangulations may be used as well.

We denote this triangulation for a unit step size by $\mathcal{K}(1)$. Its nodes are all triples of integers. A tetrahedron σ belongs to $\mathcal{K}(1)$ if for some ordering of its vertices $\sigma = [v_0, v_1, v_2, v_3]$ there exists a permutation $\pi : \{1, 2, 3\} \rightarrow \{1, 2, 3\}$ such that

$$v_i = v_{i-1} + e_{\pi(i)}, \quad i = 1, 2, 3 \tag{4}$$

holds. Here e_1, e_2, e_3 denotes the standard unit basis of \mathbf{R}^3 . The above ordering of the vertices of σ is unique and will always be tacitly assumed in the context of the Coxeter-Freudenthal triangulation. A Coxeter-Freudenthal triangulation $\mathcal{K}(\delta) := \delta\mathcal{K}(1)$ with step size $\delta > 0$ is analogously obtained.

It can be seen from

$$b_\sigma = v_0 + \frac{1}{4} \sum_{i=1}^3 (4-i)e_{\pi(i)}$$

that any tetrahedron $\sigma \in \mathcal{K}(1)$ can be compactly stored and recovered via the integer vector $4b_\sigma$.

If σ is a tetrahedron in a triangulation \mathcal{T} and has face τ , then there exists a unique tetrahedron $\tilde{\sigma} \in \mathcal{T}$, $\tilde{\sigma} \neq \sigma$ which also has τ as a face. We say that $\tilde{\sigma}$ is obtained from σ by *pivoting across* the face τ . This notion is essential for moving about in \mathbf{R}^3 in a manner which is unambiguous. For standard triangulations the pivoting rules can be efficiently implemented. In particular, for the Coxeter-Freudenthal triangulation, this can be done via pivoting by reflection: Let $\sigma = [v_0, v_1, v_2, v_3] \in \mathcal{K}(1)$ and let τ_i be the face of σ which is obtained by omitting the vertex v_i . Then the tetrahedron $\tilde{\sigma}_i$ obtained by pivoting σ across τ_i is:

$$\tilde{\sigma}_i := \begin{cases} [v_1, v_2, v_3, v_3 - v_0 + v_1] & \text{for } i = 0, \\ [v_0, v_0 - v_1 + v_2, v_2, v_3] & \text{for } i = 1, \\ [v_0, v_1, v_1 - v_2 + v_3, v_3] & \text{for } i = 2, \\ [v_2 - v_3 + v_0, v_0, v_1, v_2] & \text{for } i = 3. \end{cases}$$

Let us now briefly describe how the domain D which was implicitly defined as

$$D := \{x \in \mathbf{R}^3 : H(x) \leq 0\}$$

can be approximated with respect to a given triangulation \mathcal{T} . A more detailed discussion of the piecewise linear algorithm for approximating 3-dimensional domains can be found in [17].

Definition 1 We call a face τ of the triangulation *intersecting* if H has a negative value for at least one point in τ . In practice, this condition will be checked in the following way: we repeatedly subdivide τ into four equal parts. If δ indicates the diameter of τ , then we obtain a grid of mesh size $\delta 2^{-k}$ where k indicates the level of subdivision. Then we test the intersecting condition on all gridpoints of a subdivision with respect to a given level k . A tetrahedron is called intersecting if it contains an intersecting face.

The level k in the above Definition should be related to the desired accuracy in the integration procedure. We will come back to this point. Let us emphasize that these definitions deviate from the ones given in [2]. We do this in order to allow a more precise handling of the boundary of D .

The *piecewise linear approximation* \mathcal{D} of D with respect to \mathcal{T} is obtained by

$$\mathcal{D} := \bigcup \{\sigma : \sigma \in \mathcal{T} \text{ intersecting}\}.$$

The following algorithm describes the fundamental steps of a piecewise linear algorithm for obtaining a connected component of \mathcal{D} .

Algorithm 2 Generic Approximation of a Domain.

```

input  $\sigma \in \mathcal{T}$  intersecting;           starting tetrahedron
 $\Sigma := \{\sigma\}$ ;                       list of intersecting tetrahedra to be checked
while  $\Sigma \neq \emptyset$  do           since  $D$  is bounded, the algorithm
begin                                     will eventually stop via this line
  get  $\sigma \in \Sigma$ ;
  for all intersecting faces  $\tau$  of  $\sigma$  do
    begin
      obtain  $\tilde{\sigma}$  from  $\sigma$  by pivoting across  $\tau$ ;
      if  $\tilde{\sigma} \notin \Sigma$  then  $\Sigma := \Sigma \cup \{\tilde{\sigma}\}$ ;           check whether  $\tilde{\sigma}$  is new
    end{for};
  print  $\sigma$ ;  $\Sigma := \Sigma \setminus \{\sigma\}$ ;           output of checked intersecting tetrahedra
end{while}.

```

The above algorithm is an adaptation of an algorithm for approximating the boundary of D , see, e.g., [1, 3, 4] and most recently in [16]. In general it is trivial to obtain a starting tetrahedron: one only needs to know one point in D . The properties of standard triangulations such as the Coxeter-Freudenthal triangulation permit a compact storing, retrieval and comparing of the tetrahedra in the list Σ .

Note that the above approximation method does not make use of any smoothness assumptions on the decidability function H . However, we will introduce special quadrature formulae for the case that the tetrahedron intersects the boundary of D , and the smoothness of H increases the precision of these formulae.

It is very tempting to assume that the following is true: If H is smooth and has regular value zero, then there exists an $\delta_0 > 0$ and a $k_0 > 0$ such that \mathcal{D} covers D for all mesh sizes $\delta \leq \delta_0$ of the triangulation and all levels $k \geq k_0$ which are used for testing the intersecting property, see Definition 1. Unfortunately, this is not true. It is not difficult to find a counter example.

Hence, we have to know how much volume of the domain may be lost by our approximation. It is shown in [17] that the order of magnitude

$$O\left(\frac{1}{r} \delta^4 2^{-4k}\right). \quad (5)$$

describes the maximal amount of volume which may be lost by not recognizing one intersecting face in the test of Definition 1. Here $r > 0$ indicates the smallest radius of curvature of the boundary ∂D .

3 Basic Integration Step

We use a piecewise-linear approximation of D as discussed in the previous section to approach the problem of numerically integrating $\int_D f(x) dx$. Hence, we assume that \mathcal{D} is a family of

tetrahedra σ which essentially covers the domain D . Thus, we split the integral into

$$\int_D f(x) dx = \sum_{\sigma \in \mathcal{D}} \int_{\sigma \cap D} f(x) dx,$$

and reduce the task to numerically integrating the subproblems

$$\int_{\sigma \cap D} f(x) dx.$$

This is performed by making recursive use of a basic and simple quadrature rule. The only complication comes from the fact that σ may intersect D in various ways. In the following let v_i , $i = 0, 1, 2, 3$ denote the vertices of σ , and $\text{vol } \sigma$ its volume. We consider five different cases which are classified by the number of vertices v_i of σ lying inside D , i.e. which fulfil $H(v_i) \leq 0$. Our basic quadrature rule varies according to these cases.

Case $i = 0$. All vertices of σ are outside D . Then we use the trivial approximation

$$\int_{\sigma \cap D} f(x) dx = 0.$$

Case $i = 4$. All vertices of σ are inside D . Then we use the trapezoidal rule

$$\int_{\sigma \cap D} f(x) dx = \frac{1}{4}(f(v_0) + f(v_1) + f(v_2) + f(v_3)) \text{vol } \sigma. \quad (6)$$

Case $i = 1$. All but one vertices are outside D . By a renumbering, we can assume that v_0 is inside. In this case σ has 3 transverse edges $[v_0, v_i]$, $i = 1, 2, 3$. Every transverse edge $[v_i, v_j]$ contains a point of the boundary ∂D , i.e. a zero of H whose linear approximation is given by

$$w_{ij} = v_i + \lambda_{ij}(v_j - v_i), \quad \lambda_{ij} = \frac{H(v_i)}{H(v_i) - H(v_j)} \in [0, 1].$$

Thus the linear approximation of $\sigma \cap D$ is given by the tetrahedron $[v_0, w_{01}, w_{02}, w_{03}]$ with the volume

$$\begin{aligned} & \text{vol}[v_0, w_{01}, w_{02}, w_{03}] \\ &= \frac{1}{6} |\det(w_{01} - v_0, w_{02} - v_0, w_{03} - v_0)| \\ &= \frac{1}{6} |\det(\lambda_{01}(v_1 - v_0), \lambda_{02}(v_2 - v_0), \lambda_{03}(v_3 - v_0))| \\ &= (\lambda_{01} \lambda_{02} \lambda_{03}) \frac{1}{6} |\det(v_1 - v_0, v_2 - v_0, v_3 - v_0)| \\ &= (\lambda_{01} \lambda_{02} \lambda_{03}) \text{vol } \sigma \end{aligned}$$

Therefore we use the rule

$$\int_{\sigma \cap D} f(x) dx = f(v_0) (\lambda_{01} \lambda_{02} \lambda_{03}) \text{vol } \sigma.$$

Case $i = 3$. All but one vertices are inside D . By a renumbering, we can assume that v_0 is outside. Then we use the rule

$$\int_{\sigma \cap D} f(x) dx = \frac{1}{3}(f(v_1) + f(v_2) + f(v_3)) (1 - \lambda_{01} \lambda_{02} \lambda_{03}) \text{vol } \sigma.$$

Case $i = 2$. Two vertices are inside and two are outside D . By a renumbering, we can assume that v_0, v_1 are inside. Here $\sigma \cap D$ can be linearly approximated by the convex hull of $v_0, v_1, w_{02}, w_{03}, w_{12}, w_{13}$, which can be split into three tetrahedra

$$[v_0, v_1, w_{12}, w_{13}], [v_0, w_{02}, w_{12}, w_{13}], [v_0, w_{02}, w_{03}, w_{13}].$$

By a calculation which is analogous but slightly longer than the one given above we obtain the corresponding volumes

$$(\lambda_{12} \lambda_{13}) \text{vol } \sigma, (\lambda_{02} \lambda_{21} \lambda_{13}) \text{vol } \sigma, (\lambda_{02} \lambda_{03} \lambda_{31}) \text{vol } \sigma.$$

Thus we use the rule

$$\int_{\sigma \cap D} f(x) dx = \frac{1}{2}(f(v_0) + f(v_1)) (\lambda_{12} \lambda_{13} + \lambda_{02} \lambda_{21} \lambda_{13} + \lambda_{02} \lambda_{03} \lambda_{31}) \text{vol } \sigma.$$

Let us finally note in conclusion that the factor $\text{vol } \sigma$ appearing in each of the above formulae is in general a constant. In the case of the Coxeter-Freudenthal triangulation of stepsize δ , it is $\delta^3/6$.

4 The Composite Integration Rules

We already mentioned that we make use of the basic integral formulae in section 3 in a recursive way. This leads to an adaptive method. However, as was already pointed out in [2], it is much more efficient to use an extrapolation method for the case of smooth integrands and a tetrahedron inside D .

In order to describe such extrapolation steps, we need to introduce subdivisions of a given tetrahedron $\sigma = [v_0, v_1, v_2, v_3]$ and corresponding composite integration rules based on the integrals of section 3. We consider the two cases:

- (i) The tetrahedron σ lies inside the domain D .
- (ii) The tetrahedron σ hits the boundary of the domain D .

4.1 Interior Tetrahedron

The basic integration rule for the first case is the trapezoidal rule, see (6):

$$I_0 := \int_{\sigma \cap D} f(x) dx = \frac{1}{4}(f(v_0) + f(v_1) + f(v_2) + f(v_3)) \text{vol } \sigma. \quad (7)$$

Now we subdivide σ into 8 subtetrahedra of equal volume by halving all edges. With the notation $v_{ij} := \frac{1}{2}(v_i + v_j)$ these 8 subtetrahedra are

$$\begin{aligned} & [v_0, v_{01}, v_{02}, v_{03}], [v_{01}, v_{02}, v_{03}, v_{13}], \\ & [v_{01}, v_1, v_{12}, v_{13}], [v_{01}, v_{02}, v_{12}, v_{13}], \\ & [v_{02}, v_{12}, v_2, v_{23}], [v_{02}, v_{03}, v_{13}, v_{23}], \\ & [v_{03}, v_{13}, v_{23}, v_3], [v_{02}, v_{12}, v_{13}, v_{23}]. \end{aligned} \quad (8)$$

If we apply the above trapezoidal rule to each of these eight subtetrahedra and take the sum of the results, we obtain the following composite trapezoidal rule

$$I_1 := \frac{1}{4} \left(f(v_0) + f(v_1) + f(v_2) + f(v_3) + 4f(v_{01}) + 6f(v_{02}) \right. \\ \left. + 4f(v_{03}) + 4f(v_{12}) + 6f(v_{13}) + 4f(v_{23}) \right) \frac{1}{8} \text{vol } \sigma,$$

or, by making use of the first integral I_0 ,

$$I_1 = \frac{1}{8} I_0 + \left(4f(v_{01}) + 6f(v_{02}) + 4f(v_{03}) + 4f(v_{12}) + 6f(v_{13}) + 4f(v_{23}) \right) \frac{1}{32} \text{vol } \sigma.$$

We note that the above subdivision is the only one which is consistent with the Coxeter-Freudenthal triangulation. Let us now generalize the above composite trapezoidal rule by making use of repeated subdivisions. We introduce the subdivision of level n where n indicates the number of halvings involved and denote by V_n the set of vertices which belong to the subdivision of level n , but not to any previous subdivision:

$$\begin{aligned} V_0 &= \{v_0, v_1, v_2, v_3\} \\ V_1 &= \{v_{01}, v_{02}, v_{03}, v_{12}, v_{13}, v_{23}\} \\ &\vdots \end{aligned}$$

The general composite trapezoidal rule I_n at level n obeys the following recursion formulae

$$I_n = \frac{1}{8} I_{n-1} + \frac{1}{4 \cdot 8^n} \left(\sum_{v \in V_n} \alpha(v) f(v) \right) \text{vol } \sigma, \quad (9)$$

where I_0 is given in (7) and the coefficient $\alpha(v)$ counts the number of tetrahedra of the subdivision of level n which contain v .

Let us indicate how the above subdivisions and the calculation of the coefficients $\alpha(v)$ are implemented. To this end, we identify σ with the tetrahedron

$$\sigma_0 = \left[\left(\begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \right) \right] \quad (10)$$

of the Coxeter-Freudenthal triangulation $\mathcal{K}(1)$. This identification is performed via an obvious affine transformation which is easily implemented. Then the above subdivision of level n corresponds to the subdivision of σ_0 which is generated by the Coxeter-Freudenthal triangulation of step size 2^{-n} .

Under the above identification, it is easily seen that a vertex $v = (v(1), v(2), v(3))$ belongs to V_n if and only if $1 \geq v(1) \geq v(2) \geq v(3) \geq 0$ and $2^n v(i)$ is an odd integer for at least one $i \in \{1, 2, 3\}$. For the calculation of the coefficient $\alpha(v)$, we recall the structure of the Coxeter-Freudenthal triangulation as indicated in the recursion formulae (4). It is convenient to consider the number of simplices which contain v in position v_i , $i=0,1,2,3$, and take $\alpha(v)$ as the sum of these numbers. Obviously, the maximum number of occurrences is

$3! + 3! + 3! + 3! = 24$. But there are restrictions: let k_1 denote the number of co-ordinates of v which belong to $\{0, 1\}$, and let k_2 denote the number of co-ordinates of v which are equal, but do not belong to $\{0, 1\}$. Then accounting for these restrictions, we obtain the formula

$$\alpha(v) = \frac{24}{(k_1 + 1)! k_2!}.$$

4.2 Boundary Tetrahedron

Let us now consider the second case where the tetrahedron σ hits the boundary. Since in each subdivision of σ all basic integration formulae of section 3 may occur, the coefficients α cannot be used. The composite integrals I_n corresponding to a subdivision of level n have to be computed as a (large) sum of 8^n basic integrals over all tetrahedra of the given subdivision. Contrary to the case of an interior basic tetrahedron σ , no recursion is possible here. Hence, the subdivisions have to be generated explicitly by making use of a tree structure.

Furthermore, since the computational complexity of this case increases drastically with increasing level n , we try to handle as many situations as possible via the first case. This is done by an adaptive method described in the next section.

5 The Adaptive Extrapolation Method

The composite integrals in the last section are used in an extrapolation scheme similar to the well known Romberg method. The theoretical background for such extrapolation methods, i.e., a generalization of the celebrated Euler-Maclaurin formula for n -dimensional simplices, can be found in [7, 13, 14]. These investigations were also extended to the case of integrands with certain known singularities, see, e.g., [8].

However, let us emphasize that the composite integration methods of the last section were introduced in a much more general context where the Euler-Maclaurin formula does not necessarily hold. In particular, we handle two important special cases:

- We have to allow for intersections with the boundary since we are permitting general domains D . Hence, in general, all basic integrals of section 3 are relevant for the composite rule.
- We need to allow for weak singularities of the integrand since we focus on applications to boundary element methods. These singularities may be unknown and should automatically be detected and treated by the method.

Nevertheless, extrapolation works in most cases, and it is essential for the efficiency of the method to make use of extrapolation steps as often as possible. Therefore, we propose to do cautious extrapolation steps which are monitored by tests. These tests force the method to fall back to adaptive strategies near singular or boundary points.

Let us now briefly sketch the proposed method. We consider a current tetrahedron σ . The composite integrals I_0, I_1, \dots, I_n are calculated by either of the two methods described in the subsections 4.2 or 4.1, depending on whether σ hits the boundary or not. If we assume

a quadratic expansion of the composite integrals in terms of the stepsize, then we may use a standard Romberg extrapolation tableau:

$$\begin{array}{rcccc}
I_0 & = & T_{0,0} & & \\
I_1 & = & T_{1,0} & T_{1,1} & \\
I_2 & = & T_{2,0} & T_{2,1} & T_{2,2} \\
& & \vdots & \vdots & \vdots & \ddots \\
I_n & = & T_{n,0} & T_{n,1} & T_{n,2} & \cdots & T_{n,n}.
\end{array} \tag{11}$$

However, because of the general situation described above, the quadratic expansion will not always hold, and thus the Romberg extrapolation tableau has to be monitored by tests. The essential idea of this test is to check whether the relation

$$\frac{T_{i-1,k} - T_{n,n}}{T_{i,k} - T_{n,n}} \approx 4^{k+1} \tag{12}$$

holds. The following steps are performed:

Algorithm 3 Sketch of the adaptive integration method.

1. Initially compute the above tableau (11) for $n = 2$.
2. Check the coefficients according to (12).
 - 2.1. If the above coefficient check is positive, then check the error estimate

$$|T_{n,n-1} - T_{n,n}| \leq \epsilon.$$

- 2.1.1. If this error check is positive, accept $T_{n,n}$ as the integral over the current tetrahedron σ .
 - 2.1.2. Otherwise, increase n and go back to step 2.
- 2.2. If the coefficient check fails, then check the error estimate

$$|T_{n-1,0} - T_{n,0}| \leq \epsilon$$

for the underlying composite rules.

- 2.2.1. If this error check is positive, accept $T_{n,0}$ as the integral over the current tetrahedron σ .
 - 2.2.2. Otherwise, split σ into eight subtetrahedra, and apply the method (beginning with step 1) to each of these subtetrahedra as the current tetrahedron recursively.

Remark 4 As was mentioned earlier, the method varies between two different computations of the composite rule, depending on whether the tetrahedron hits the boundary or not. In the first case the integrals over each subtetrahedron have to be computed individually. Therefore, it is necessary to generate a tree structure in order to avoid redundant calculations.

This structure, which is also used for handling the splitting technique (step 2.2.2), allows in addition to reuse already calculated integrals in case of a splitting. However, the computation of the integral in the second case is drastically more efficient because of the recursion (9). This is true despite the fact that in the case of a splitting all calculations are discarded.

Remark 5 For the calculation of the basic integrals, a test is needed to decide whether a given tetrahedron intersects the boundary of D or not. For reasons of efficiency, this test is only performed on the vertices of this initial tetrahedron. However, after performing several subdivisions, it may turn out that the result of the initial test was not correct. Therefore, we safeguard the algorithm by checking the values of the function H at all vertices of the subtetrahedra which occur in the composite integrals. If it turns out during the refinements, that the initial tetrahedron does intersect the boundary, then the composite integrals are recalculated. This applies seldom, but it is necessary in order to obtain high accuracy. The possible error which we would get otherwise has the same order of magnitude as the one described in (5).

Remark 6 The dimension of the Romberg tableau should be limited for efficiency (and memory) reasons. We suggest a small number like 3 or 4. Hence, the increasing of n mentioned in step 2.1.2 is not always possible and must otherwise be replaced by a splitting.

Remark 7 From the estimate (5) it is seen that it is important to ask for a minimal level of subdivisions before accepting an approximate integral in Algorithm 3.

Remark 8 We note that the tolerance ϵ used in Algorithm 3 is of a local nature. This is unavoidable since we permit weak singularities. The reader should view this tolerance as an artificially prescribed machine tolerance or as a parameter for monitoring a stopping criterion.

Remark 9 It is important to decide what has to be done about possible singular values of the integrand f in (2). In order to avoid excessive and unnecessary computations, we suggest chopping the function f and replacing it by

$$\begin{cases} f(y) & \text{for } |f(y)| \leq \beta, \\ \beta & \text{for } f(y) > \beta, \\ -\beta & \text{for } f(y) < -\beta, \end{cases}$$

for some large constant $\beta > 0$. As was discussed in [2], it suffices to choose

$$\beta \approx \frac{1}{\sqrt{\epsilon}}$$

for the case of a typical weak singularity y_0 of the form $f(y) \sim \|y - y_0\|^{-1}$.

6 Numerical Examples

Let us consider some numerical examples in order to illustrate some features of our integration method. The tables below show the performance of the method for different tolerances “Tol” and list the approximate value for the integral under “Integral”. Furthermore, the “Error” is either calculated (for the case that the exact integral is known) or is estimated (against the best known approximation). Under “Time”, we list the run time of the integration. The programs are written in Turbo C and were run on a PC with the 80386/387 processors.

The first example is very simple: the domain D consists of one simplex σ_0 as defined in (10). We check the integration method on two different integrands, namely,

$$f(x) = e^{\|x\|_1}, \quad (13)$$

$$f(x) = \frac{1}{\|x\|_1}. \quad (14)$$

Note that the first integrand is smooth, and the second has a singularity at a vertex of σ_0 . The exact values of the integrals are $(e - 1)/6$ and $(3/4) \ln 3 - \ln 2$, respectively.

It can be seen in Table 1 that the presence of a singularity increases the computational effort very much, but that the integral can still be approximated to any given precision. The table displays only three different values for the first integrand: this is due to the fact that the algorithm produces just one Romberg tableau (no splitting) in this case, and the accuracy of the Romberg tableau increases by more than a factor of 10 for each additional level.

TABLE 1. One single tetrahedron.

Tol	Integral (13)	Error	Time	Integral (14)	Error	Time
10^{-3}	0.8455308710882	4.8e-06	0.00	0.1308369524448	2.5e-05	0.16
10^{-4}	0.8455356853786	8.3e-11	0.04	0.1308070282350	5.0e-06	0.22
10^{-5}	0.8455356853786	8.3e-11	0.04	0.1308126913759	6.6e-07	0.47
10^{-6}	0.8455356853786	8.3e-11	0.04	0.1308116852846	3.5e-07	0.80
10^{-7}	0.8455356853786	8.3e-11	0.04	0.1308119685160	6.7e-08	1.40
10^{-8}	0.8455356852957	2.5e-13	0.22	0.1308120326447	3.3e-09	2.61
10^{-9}	0.8455356852957	2.5e-13	0.22	0.1308120353218	6.2e-10	3.90
10^{-10}	0.8455356852957	2.5e-13	0.22	0.1308120358915	5.0e-11	6.97
10^{-11}	0.8455356852957	2.5e-13	0.22	0.1308120359363	4.8e-12	12.50
10^{-12}	0.8455356852957	2.5e-13	0.22	0.1308120359404	7.3e-13	26.22

The domain D in the second example consists of three intersecting ellipsoids, given by the equation

$$\min \left\{ 4(x_1 + 0.5)^2 + 16x_2^2, \quad (1 - x_1 + \sqrt{3}x_2)^2 + 4(\sqrt{3}x_1 + x_2)^2, \right. \\ \left. (1 - x_1 - \sqrt{3}x_2)^2 + 4(\sqrt{3}x_1 - x_2)^2 \right\} + 16x_3^2 - 1.44 \leq 0.$$

We use a triangulation of this domain generated with a stepsize $\delta = 0.15$ which consists of

Figure 1: Three intersecting ellipsoids.

1822 tetrahedra. In Figure 1, where this domain is shown, a different mesh has been used for illustrational purposes. A singular integrand is chosen:

$$f(x) = \frac{1}{4\pi\|x\|_2}. \quad (15)$$

Here, the result on the finest level is used for an estimation of the error. The minimal level “m.l.” introduced in Remark 4 at the end of the last section is chosen according to the given tolerance “Tol” as indicated in equation (5).

TABLE 2. Three intersecting ellipsoids.

Tol	m.l.	Integral	Error	Time
10^{-3}	2	0.1190282622071	1.1e-03	89.75
10^{-4}	2	0.1178143688147	9.2e-05	93.59
10^{-5}	2	0.1177682563301	1.4e-04	97.33
10^{-6}	2	0.1178506368376	5.6e-05	201.96
10^{-7}	2	0.1179107826426	4.2e-06	436.49
10^{-8}	3	0.1179080621627	1.5e-06	1594.54
10^{-9}	3	0.1179066142789	_____	4793.19

The increase in time vis a vis the first example is not only due to the fact that now 1822 tetrahedra are treated, but even more to the fact that an extensive tree structure has to be used to handle the boundary of the domain.

This becomes more obvious from our last example in which we compare the performance of the algorithm on the unit ball $D = \{x \in \mathbf{R}^3 : \|x\|_2 \leq 1\}$ against the unit cube $D = \{x \in \mathbf{R}^3 : \|x\|_\infty \leq 1\}$. We use triangulations of these domains generated with a stepsize $\delta = 0.5$ which consist of 324 and 384 tetrahedra, respectively. In both cases, the singular integrand (15) is again used. In the first case, the error estimate is calculated against the exact integral 0.5, and in the second case against the numerical integral obtained with $\text{Tol} = 10^{-14}$.

TABLE 3. Unit ball.

Tol	m.l.	Integral	Error	Time
10^{-3}	2	0.4998001348054	2.0e-04	18.95
10^{-4}	2	0.4992646639814	7.4e-04	20.93
10^{-5}	2	0.4999944823488	5.5e-06	60.42
10^{-6}	2	0.5000267420260	2.7e-05	197.67
10^{-7}	3	0.5000178643370	1.8e-05	634.55
10^{-8}	4	0.5000084937637	8.5e-06	1973.53
10^{-9}	4	0.5000045127159	4.5e-06	5718.18

TABLE 4. Unit cube.

Tol	m.l.	Integral	Error	Time
10^{-3}	2	0.7578572877327	2.6e-04	6.59
10^{-4}	2	0.7573218169086	2.8e-04	8.93
10^{-5}	2	0.7573625160284	2.4e-04	16.04
10^{-6}	2	0.7575608729428	4.1e-05	39.38
10^{-7}	2	0.7575752120969	2.7e-05	138.11
10^{-8}	2	0.7576015123360	6.4e-07	462.08
10^{-9}	2	0.7576020161343	1.4e-07	1709.85

The notable difference in performance is explained by the fact that the unit cube is triangulated exactly, i.e., its boundary does not intersect the interior of a tetrahedron which is used for its triangulation. This implies that every integration can be performed via the composite trapezoidal rule described in Section 4.1. Since this rule has a quadratic expansion, the adaptive extrapolation steps described in Section 5 are more efficient (in fact, they are only slowed down by the one singular point).

However, the method still has to perform the tests described in Remark 5 for the unit cube, which are superfluous in the case of exact triangulations like the one of the cube. In fact, also some additional overhead could be dropped.

It turns out that the latter case is important for some of the applications we have in mind: For a typical volume potential like (1), which is used for handling inhomogeneous PDEs via boundary integral methods, the domain D can easily be chosen in such a way that it can be exactly triangulated. For this reason, we have generated a simplified version which assumes an exact triangulation of the domain. With this simplified algorithm we obtain the following result for the above unit cube:

TABLE 5. Unit cube, simplified algorithm.

Tol	Integral	Error	Time
10^{-3}	0.7578572877327	2.6e-04	4.56
10^{-4}	0.7573218169086	2.8e-04	6.15
10^{-5}	0.7573625160284	2.4e-04	10.90
10^{-6}	0.7575608729428	4.1e-05	26.75
10^{-7}	0.7575752120969	2.7e-05	94.63
10^{-8}	0.7576015123360	6.4e-07	313.90
10^{-9}	0.7576020161343	1.4e-07	1160.69

We note that for this simple example, it is possible to make use of the symmetries of the unit cube C , since these symmetries are respected by the integrand (15). Hence,

$$\int_C \frac{dx}{4\pi\|x\|_2} = 48 \int_{\sigma_0} \frac{dx}{4\pi\|x\|_2},$$

where the standard tetrahedron σ_0 is defined in (10). Table 6 displays the performance of the simplified algorithm for the latter integral.

The small discrepancies of Tables 5,6 in the error column are due to the fact that the Coxeter-Freudenthal triangulation on which our method is based, does not respect the symmetries of the unit cube. These symmetries are respected, however, by the Union Jack triangulation which is described in [15].

TABLE 6. One single tetrahedron, simplified algorithm.

Tol	Integral	Error	Time
10^{-3}	0.7603511929760	2.7e-03	0.16
10^{-4}	0.7586889108763	1.1e-03	0.21
10^{-5}	0.7579298897986	3.3e-04	0.38
10^{-6}	0.7576699117538	6.8e-05	0.72
10^{-7}	0.7575610429846	4.1e-05	1.92
10^{-8}	0.7576106123859	8.5e-06	7.42
10^{-9}	0.7576073433956	5.2e-06	23.13
10^{-10}	0.7576038828591	1.7e-06	87.30
10^{-11}	0.7576027448781	5.9e-07	295.88
10^{-12}	0.7576021960099	4.2e-08	859.42
10^{-13}	0.7576021553132	3.9e-10	1127.56

Acknowledgements. The authors wish to thank Eugene L. Allgower for many helpful discussions.

References

- [1] E. L. ALLGOWER AND K. GEORG, *Numerical Continuation Methods: An Introduction*, vol. 13 of Series in Computational Mathematics, Springer Verlag, Berlin, Heidelberg, New York, 1990. pp 388.
- [2] E. L. ALLGOWER, K. GEORG, AND R. WIDMANN, *Volume integrals for boundary element methods*, J. Comput. Appl. Math., 38 (1991), pp. 17–29.
- [3] E. L. ALLGOWER AND S. GNUTZMANN, *An algorithm for piecewise linear approximation of implicitly defined two-dimensional surfaces*, SIAM J. Numer. Anal., 24 (1987), pp. 452–469.
- [4] E. L. ALLGOWER AND P. H. SCHMIDT, *An algorithm for piecewise-linear approximation of an implicitly defined manifold*, SIAM J. Numer. Anal., 22 (1985), pp. 322–346.
- [5] K. E. ATKINSON, *A survey of boundary integral equation methods for the numerical solution of Laplace's equation in three dimensions*, in Numerical Solution of Integral Equations, M. Goldberg, ed., New York, 1990, Plenum Press, pp. 1–34.
- [6] H. S. M. COXETER, *Discrete groups generated by reflections*, Ann. of Math., 6 (1934), pp. 13–29.
- [7] E. DE DONCKER, *New Euler-Maclaurin expansions and their application to quadrature over the s -dimensional simplex*, Math. Comp., 33 (1979), pp. 1003–1018.
- [8] E. DE DONCKER-KAPENGA, *Asymptotic expansions and their applications in numerical integration*, in Numerical Integration: Recent Developments, Software and Applications, P. Keast and G. Fairweather, eds., vol. 203 of NATO ASI Series C: Mathematical and Physical Sciences, Boston, 1987, Reidel Publ. Comp.
- [9] B. C. EAVES, *A Course in Triangulations for Solving Equations with Deformations*, vol. 234 of Lecture Notes in Economics and Mathematical Systems, Springer Verlag, Berlin, Heidelberg, New York, 1984.
- [10] H. FREUDENTHAL, *Simplizialzerlegungen von beschränkter Flachheit*, Ann. of Math., 43 (1942), pp. 580–582.
- [11] K. GEORG, *Approximation of integrals for boundary element methods*, SIAM J. Sci. Stat. Comput., 12 (1991), pp. 443–453.
- [12] R. B. GUENTHER AND J. LEE, *Partial Differential Equations of Mathematical Physics and Integral Equations*, Prentice Hall, Englewood Cliffs, 1987.
- [13] J. N. LYNES, *Quadrature over a simplex: Part 1. A representation for the integrand function*, SIAM J. Numer. Anal., 15 (1978), pp. 122–133.
- [14] —, *Quadrature over a simplex: Part 2. A representation for the error functional*, SIAM J. Numer. Anal., 15 (1978), pp. 870–887.

- [15] M. J. TODD, *The Computation of Fixed Points and Applications*, vol. 124 of Lecture Notes in Economics and Mathematical Systems, Springer Verlag, Berlin, Heidelberg, New York, 1976.
- [16] R. WIDMANN, *An efficient algorithm for the triangulation of surfaces in \mathbf{R}^3* . Preprint, Colorado State University, October 1990.
- [17] —, *Efficient triangulation of 3-dimensional domains*. Preprint, Colorado State University, November 1990.