

Volume Integrals for Boundary Element Methods

*Eugene L. Allgower*¹, *Kurt Georg*^{1,2} and *Ralf Widmann*³

Department of Mathematics
Colorado State University
Fort Collins, Colorado 80523

June 1990 / March 1991

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 element methods. An adaptive integration method is described. It utilizes an automatic simplicial subdivision of the domain. The integration step is based on ideas similar to those recently given by Georg [12] for surface integrals. Several examples illustrate the performance of the method.

Key words. Volume integrals, quadrature formula, boundary element method, trapezoidal rule, adaptive refinement.

AMS(MOS) subject classification. 65D30, 65R20, 65N35, 45E05.

1 Introduction

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

$$(1.1) \quad \begin{aligned} \Delta u &= f \quad \text{in } \tilde{D}, \\ u &= g \quad \text{on } \tilde{B}, \end{aligned}$$

where $\tilde{D} \subset \mathbf{R}^3$ is a bounded open nonempty domain with sufficiently regular boundary $\tilde{B} = \partial\tilde{D}$. Following [6], 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

$$(1.2) \quad v(x) := \int_D s(x, y) f(y) dy,$$

¹ Partially supported by the National Science Foundation via grant # DMS-8947761

² Partially supported by the Deutsche Forschungsgemeinschaft, SFB 256

³ Supported by a postdoctoral stipend of the Deutsche Forschungsgemeinschaft

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

$$w := u - v$$

and obtain the homogenous equation

$$(1.3) \quad \begin{aligned} \Delta w &= 0 \quad \text{in } \tilde{D}, \\ w &= g - v \quad \text{on } \tilde{B}. \end{aligned}$$

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

$$(1.4) \quad w(x) = \int_{\tilde{B}} \frac{\partial s(x, y)}{\partial \mathbf{n}(y)} \sigma(y) \mu(dy),$$

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., [14], that σ satisfies the following integral equation of the second kind:

$$(1.5) \quad \sigma(x) - 2 \int_{\tilde{B}} \frac{\partial s(x, y)}{\partial \mathbf{n}(y)} \sigma(y) \mu(dy) = -2g(x) + 2v(x)$$

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 (1.2) at least for boundary points $x \in \tilde{B}$. Performing this task is one of the aims of our paper.

2 Automatic Triangulation of D

In this section we briefly describe the Coxeter-Freudenthal triangulation, see [7, 10], which we use to approximate D . More sophisticated triangulations may be used as well, extensive discussions of triangulations of \mathbf{R}^n may be found e.g., in the books [2, 9, 21]. Important aspects of the triangulation which is used concern efficient storing, comparing and recovering of the tetrahedra of a triangulation.

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

$$(2.1) \quad b_\sigma := \frac{1}{4} \sum_{j=0}^3 v_j.$$

(2.2) Definition. A *triangulation* \mathcal{T} of \mathbf{R}^3 is a collection of tetrahedra such that

- (1) $\sigma_1, \sigma_2 \in \mathcal{T}$ implies either $\sigma_1 \cap \sigma_2 = \emptyset$, or it is a common face, edge or vertex.
- (2) $\bigcup_{\sigma \in \mathcal{T}} \sigma = \mathbf{R}^3$.
- (3) 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 *set of nodes* \mathcal{T}^0 of a triangulation \mathcal{T} are the vertices of the tetrahedra in \mathcal{T} . Analogously, \mathcal{T}^1 denotes the *set of edges*, \mathcal{T}^2 denotes the *set of faces*, and $\mathcal{T}^3 := \mathcal{T}$.

(2.3) The Coxeter-Freudenthal Triangulation. We denote this triangulation for a unit mesh size by $\mathcal{K}(1)$. Its nodes are all triples of integers, i.e., $\mathcal{K}^0(1) := \mathbf{Z}^3$, where \mathbf{Z} denotes the set 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

$$(2.4) \quad v_i = v_{i-1} + e_{\pi(i)}, \quad i = 1, 2, 3$$

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.

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 $4b_\sigma$. A Coxeter-Freudenthal triangulation $\mathcal{K}(\delta) := \delta\mathcal{K}(1)$ with mesh size $\delta > 0$ is analogously obtained.

(2.5) Pivoting. 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:

$$(2.6) \quad \tilde{\sigma}_i := \begin{cases} [v_1, v_2, v_3, v_3 - v_0 + v_1] & \text{for } i = 0, \\ [v_0, v_0 - v_1 + v_2, v_2, v_3] & \text{for } i = 1, \\ [v_0, v_1, v_1 - v_2 + v_3, v_3] & \text{for } i = 2, \\ [v_2 - v_3 + v_0, v_0, v_1, v_2] & \text{for } i = 3. \end{cases}$$

3 Piecewise Linear Approximation of the Domain D

Let $H : \mathbf{R}^3 \rightarrow \mathbf{R}$ be a function which describes a bounded domain:

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

and let \mathcal{T} be a triangulation of \mathbf{R}^3 .

(3.1) Definition. We call a face $\tau \in \mathcal{T}^2$ *intersecting* if H has a non-positive value for at least one vertex of τ . Analogously, a tetrahedron is called intersecting if it contains an intersecting face. It is easy to see that an intersecting tetrahedron has either three or four intersecting faces. The *piecewise linear approximation* $D_{\mathcal{T}}$ of D with respect to \mathcal{T} is obtained by

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

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

(3.2) Generic Approximation of a Domain.

comment:

```

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., [2–4] and [22]. A more detailed discussion of piecewise linear algorithms for the triangulation of 3-dimensional domains can be found in [23]. 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 . Smoothness however, plays a role for the important question concerning how well $D_{\mathcal{T}}$ approximates the domain D .

In general, $D_{\mathcal{T}}$ covers D , and hence in order to obtain a better approximation it is reasonable to chop those tetrahedra σ which lie transverse to the boundary by a suitable approximation of a tangent plane. More precisely, we make the following

(3.3) Remark. We call a tetrahedron σ *transverse* if H is nonpositive on at least one vertex and positive on at least one vertex. There is a unique affine function $H_\sigma : \mathbf{R}^3 \rightarrow \mathbf{R}$ which coincides with H on the vertices of σ . Note that for intersecting tetrahedra σ we have that $\sigma \cap H_\sigma^{-1}(-\infty, 0] \neq \sigma$ if and only if σ is transverse. In this case, the intersection $\sigma \cap H_\sigma^{-1}(-\infty, 0]$ is a polyhedron. We obtain an improved approximation of D by setting

$$\tilde{D}_\mathcal{T} := \{ \sigma \cap H_\sigma^{-1}(-\infty, 0] : \sigma \in D_\mathcal{T} \}.$$

It is possible to prove a proposition analogous to (3.4) in [1] which essentially states that under suitable smoothness assumptions on H and under suitable assumptions on \mathcal{T} (which are satisfied for standard triangulations), $\tilde{D}_\mathcal{T}$ is a quadratic approximation of D with respect to the mesh size of \mathcal{T} . Here it is assumed that the domain D is compact and connected.

4 The Integration Method

The aim of this paper is to numerically approximate volume integrals of the form

$$\int_D f(x) dx.$$

As a first step we consider a piecewise linear approximation $D_\mathcal{T}$ of the volume D and obtain

$$(4.1) \quad \int_D f(x) dx = \sum_{\sigma \in D_\mathcal{T}} \int_\sigma \chi_D f(x) dx,$$

where χ_D denotes the characteristic function of D . The individual integrals on the right side are approximated by a repeated use of the trapezoidal rule

$$(4.2) \quad \int_\sigma g(x) dx \approx \frac{1}{4} \sum_{i=0}^3 g(v_i) \text{vol}(\sigma),$$

where the v_i denote the vertices of σ and $\text{vol}(\sigma)$ its volume. We use this rule in an adaptive refinement procedure: First we calculate a coarse approximation ‘`int1`’ via (4.2). Then we subdivide the tetrahedron σ into eight tetrahedra by cutting all edges in half, and apply the trapezoidal rule to each of these eight tetrahedra. By summing these values up, we obtain a refined approximation ‘`int2`’ of the given integral. The two approximations are compared. If they differ by less than a given tolerance TOL, then the algorithm stops. Otherwise, each of the eight tetrahedra is again treated similarly. A recursive call of this rule enables the algorithm to locally adjust the refinement according to the given tolerance. The above approximation (4.1)-(4.2) is suggested as a first simple approach. A number of possible improvements for future study are mentioned in the concluding remarks below.

Let us illustrate this technique by citing a few lines of the C-program which was used to compute the examples below. A `vertex` is a structure consisting of the co-ordinates of the vertex and the value of the integrand.

```
typedef struct {coord_vector coord;
               double          int_val; } vertex;
```

The main procedure ‘`volume_integration`’ is called by the user who provides the desired tolerance ‘`tol`’ and the path of the data file which contains the barycenters of the piecewise linear approximation $D_{\mathcal{T}}$ of the domain D . This list which may be obtained by an algorithm of type (3.2), e.g. by the algorithm published in [23], is read by the procedure ‘`read_center`’. The actual tetrahedra are retrieved from these barycenters via the procedure ‘`start`’. The function ‘`integration`’ operates recursively on a given tetrahedron with vertices ‘`v0`’, ‘`v1`’, ‘`v2`’, ‘`v3`’. The variable ‘`int_glob`’ sums up all the values of the approximate integrals over the list of tetrahedra, and the variable ‘`int_loc`’ is used to store the value of the approximate integral over one tetrahedron of this list. The latter value is obtained by one recursive call of ‘`integration`’.

```
void volume_integration (char path[], double tol)

{int i;

  num_ctr = read_center (path);
  for (i = 1 ; i <= num_ctr ; i++)
    {start (i);
     level = 0;
     int_loc = 0.0;
     integration (v0, v1, v2, v3, level);
     int_glob += int_loc;
    }
}
```

The central part of the algorithm consists of the function ‘`integration`’ which recursively calls itself to generate the desired adaptation effect. The variable ‘`level`’ indicates the current level of the subdivision. The program ‘`mid_vertex`’ computes the midpoint of two nodes and the value of the integrand on this midpoint. A coarse approximation ‘`int1`’ of the integral over the current tetrahedron is calculated according to the trapezoidal rule ‘`trapez`’, and it is compared to a finer approximation ‘`int2`’ obtained by one subdivision of the current tetrahedron. The two values are compared. If the difference falls within the tolerance ‘`tol`’, then the recursive adaptation is stopped and ‘`int2`’ is added to the approximate value ‘`int_loc`’ of the initial tetrahedron. Otherwise, the adaptive refinement proceeds.

```
void integration (vertex *v0, vertex *v1, vertex *v2,
                 vertex *v3, char level)

{vertex *v01, *v02, *v03, *v12, *v13, *v23;
 double int1, int2;

  mid_vertex (v01, v0, v1);
  mid_vertex (v02, v0, v2);
```

```

mid_vertex (v03, v0, v3);
mid_vertex (v12, v1, v2);
mid_vertex (v13, v1, v3);
mid_vertex (v23, v2, v3);

int1 = trapez (v0, v1, v2, v3, level);
int2 = trapez (v0, v01, v02, v03, level+1)
      + trapez (v01, v1, v12, v13, level+1)
      + trapez (v02, v12, v2, v23, level+1)
      + trapez (v03, v13, v23, v3, level+1)
      + trapez (v01, v02, v03, v13, level+1)
      + trapez (v01, v02, v12, v13, level+1)
      + trapez (v02, v03, v13, v23, level+1)
      + trapez (v02, v12, v13, v23, level+1);

if (fabs (int1-int2) < tol)
    int_loc +=int2;
else
    {integration (v0, v01, v02, v03, level+1);
    integration (v01, v1, v12, v13, level+1);
    integration (v02, v12, v2, v23, level+1);
    integration (v03, v13, v23, v3, level+1);
    integration (v01, v02, v03, v13, level+1);
    integration (v01, v02, v12, v13, level+1);
    integration (v02, v03, v13, v23, level+1);
    integration (v02, v12, v13, v23, level+1);
    }
}

```

(4.3) The Numerical Handling of the Singular Values. In the above method, it is important to decide what has to be done about the singular values of the integrand g in (4.2). Let us consider replacing the singular value of g by a big number β . More precisely, we substitute for g the function

$$\tilde{g}(y) := \begin{cases} g(y) & \text{for } |g(y)| \leq \beta, \\ \beta & \text{for } g(y) > \beta, \\ -\beta & \text{for } g(y) < -\beta. \end{cases}$$

If the singular point y is a vertex point of the refinement, then it is easy to see that, due to the tolerance test, the method refines towards this point until approximately

$$\beta\alpha \approx \text{TOL},$$

where α is the volume of the tetrahedron τ of the refinement having vertex x . On the other hand, a typical singularity which we want to handle is of the form (1.2), i.e.,

$$g(y) \sim \frac{1}{\|x - y\|}.$$

It is easy to see by using polar coordinates that

$$\int_{\tau} g(x) dx = O(\delta^2)$$

in this case, where δ measures the diameter of τ . Hence the method has been designed to neglect this integral piece if

$$\delta^2 \approx \text{TOL}.$$

By making the coarse approximation

$$\alpha \approx \delta^3,$$

it now becomes clear that it is sufficient to set

$$\beta \approx \frac{1}{\sqrt{\text{TOL}}}$$

in order to achieve the desired accuracy through the adaptive refinement. This technique will be used in the numerical examples. Numerical experiments confirm the above discussion: if β is increased further, then only the number of refinements near the singularity increases without essentially improving the accuracy of the global integral. For example in table (5.2) below, increasing β to $1/\text{TOL}$ only splits the 64 low level tetrahedra and effects the global integral by less than TOL.

5 Numerical Examples

In order to show the performance on the integration method, we first compute two examples on a very simple body namely on only one tetrahedron

$$\sigma = \left[\begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \right].$$

The meshsize of this tetrahedron is one. Usually the meshsize of an approximation will be much smaller, so that less refinement steps will be necessary.

We consider two integrals. The first is performed over a smooth function:

$$\int_{\sigma} e^{\|x\|_1} dx,$$

the second over a function with a singularity at the fourth vertex v of σ :

$$\int_{\sigma} \frac{1}{\|v - x\|_1} dx.$$

This function is chopped at $\sqrt{\text{TOL}}$ according to (4.3). Below are two tables indicating the performance of the integration method for varying tolerances TOL. The calculations were performed on a PC with double precision (approximately 14 decimal digits). Note that the first integral can be calculated exactly, the value being $\frac{1}{6}(e-1)^3 \approx 0.8455356853$. Thus the given error indicates the correct discretization error.

(5.1) Table.

TOL	1.e-3	1.e-4	1.e-5	1.e-6	1.e-7	1.e-8	1.e-9
Num. Int.	0.8521492	0.8472100	0.8464383	0.8459475	0.8456407	0.8455956	0.8455614
Error	0.0066135	0.0016743	0.0009026	0.0004118	0.0001050	0.0000599	0.0000257
Level	Number of tetrahedra contributing to the integral at this level						
1	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0
3	512	8	0	0	0	0	0
4	0	4032	2432	0	0	0	0
5	0	0	13312	32736	672	0	0
6	0	0	0	256	256768	167376	0
7	0	0	0	0	0	758144	2095832
8	0	0	0	0	0	0	10560

(5.2) Table.

TOL	1.e-3	1.e-4	1.e-5	1.e-6	1.e-7	1.e-8	1.e-9
Num. Int.	0.1337525	0.1320129	0.1312104	0.1309721	0.1308805	0.1308375	0.1308224
Level	Number of tetrahedra contributing to the integral at this level						
1	0	0	0	0	0	0	0
2	32	0	0	0	0	0	0
3	248	432	32	0	0	0	0
4	56	608	3560	1664	0	0	0
5	64	248	2160	18784	26240	1760	0
6	0	64	608	5216	50464	231872	118800
7	0	0	248	1248	13632	125056	1107072
8	0	0	64	248	3504	34880	306800
9	0	0	0	56	608	7400	82832
10	0	0	0	64	248	2160	17632
11	0	0	0	0	64	608	5216
12	0	0	0	0	0	248	1248
13	0	0	0	0	0	64	248
14	0	0	0	0	0	0	56
15	0	0	0	0	0	0	64

We see that the computational complexity in both cases is similar: The method does not take any advantage of the smoothness of the integrand in the first case. It is very suggestive to try an extrapolation method for the case that the integrand is smooth. For each integer $n > 0$, we consider an equidistant subdivision of σ into n^3 tetrahedra $\{\sigma_i\}_{i=1,\dots,n^3}$ by drawing $n - 1$ parallel planes to each face of σ , such that each edge is subdivided into n equidistant intervals. We now subdivide the integral $\int_{\sigma} g(x) dx$ correspondingly:

$$(5.3) \quad \int_{\sigma} g(x) dx = \sum_{i=1}^{n^3} \int_{\sigma_i} g(x) dx$$

and approximate each summand via the trapezoidal rule (4.2). Thus we obtain a composite trapezoidal rule for approximating $\int_{\sigma} g(x) dx$, which we call $I(n)$. Similarly to the famous Euler-Maclaurin sum formula, it is possible to prove the following

(5.4) Asymptotic Expansion Theorem. Let $k > 0$ be an integer. Under the assumption that the integrand g is sufficiently smooth, there exist constants $\{c_\kappa\}_{\kappa=1,\dots,k} \subset \mathbf{R}$ which are independent of the degree of subdivision n such that

$$\int_\sigma g(x) dx - I(n) = \sum_{\kappa=1}^k \frac{c_\kappa}{n^{2\kappa}} + O\left(\frac{1}{n^{2k+2}}\right).$$

Similar expansion theorems have been given in [18], see also [20, sec. 3.17]. The above theorem justifies an extrapolation method analogous to Romberg's method. We use this method to recompute the first example. The table below gives the numerical results for increasing degree of subdivision n . It is written in the typical Romberg fashion.

(5.5) Table.

n	$I(n)$			
0	1.29970312e-00			
2	9.53204562e-01	8.37705043e-01		
4	8.72082458e-01	8.45041757e-01	8.45530871e-01	
8	8.52149173e-01	8.45504744e-01	8.45535610e-01	8.45535685e-01

The corresponding errors are given in the next

(5.6) Table.

n	$I(n)$				
0	4.54167433e-01				
2	1.07668877e-01	-7.83064223e-03			
4	2.65467728e-02	-4.93928458e-04	-4.81420723e-06		
8	6.61348748e-03	-3.09409725e-05	-7.51401347e-08	8.31525382e-11	

Comparing tables (5.1) and (5.6) we note that the extrapolation method leads already to a higher accuracy in the third line, where only $4^3 + 2^3 + 0^3 = 73$ tetrahedra are used as opposed to 2,106,392 in the adaptive method. This difference of performance is so drastic that we strongly propose to use extrapolation whenever possible: even inside a recursive adaptation the bulk of tetrahedra at the different levels will not contain singularities.

Our next example involves a body D consisting of two 3-spheres which intersect as shown in the following Figure.

(5.7) Figure. Two intersecting 3-spheres

Note that D is obtained via

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

where

$$H(x_1, x_2, x_3) = \min \left\{ \left(x_1 - \frac{3}{4} \right)^2, \left(x_1 + \frac{3}{4} \right)^2 \right\} + x_2^2 + x_3^2 - 1.$$

We approximate this body according to algorithm (3.2) and obtain a list $D_{\mathcal{T}}$ of 2618 tetrahedra. The integral

$$\int_D \frac{1}{4\pi \|x\|_2} dx$$

is calculated according to the method indicated in (4.1). The results for various tolerances are reported in the following

(5.8) Table.

TOL	1.e-2	1.e-3	1.e-4	1.e-5	1.e-6	1.e-7	1.e-8
Num. Int.	0.7179339	0.7206386	0.7207396	0.7180701	0.7211982	0.7220026	0.7220424
Level	Number of tetrahedra contributing to the integral at this level						
1	20944	20928	19992	7800	2088	152	24
2	0	80	7568	96560	99840	79456	34368
3	0	384	192	68448	355768	495208	742624
4	0	0	1344	2112	417984	1267120	1793008
5	0	0	1536	1344	2880	2651616	4481504
6	0	0	0	1536	1344	5088	13905280
7	0	0	0	0	1344	2112	14880
8	0	0	0	0	1536	1344	3552
9	0	0	0	0	0	1344	2112
10	0	0	0	0	0	1536	1344
11	0	0	0	0	0	0	1536

As can be seen from table (5.2), one isolated singularity does not influence the computational complexity of the adaptive method very much. However, this changes for integrals such as the one discussed here, since our approach generates an artificial two-dimensional manifold of singularities via the characteristic function χ_D . A better approach is indicated in remark (6.4).

6 Concluding Remarks

(6.1) Efficiency. The procedure in section 4 was written in a concise way in order to make the logical structure easier to understand. However, this was done at the cost of computational efficiency since for many points the integrand is calculated twice. To avoid this, a much more complex program must be written. The present paper is merely intended to underline the usefulness of our approach which is based on piecewise linear approximations. For simple bodies such as cubes, sophisticated and efficient adaptive integration routines have already been developed, see, e.g., Friedman & Wright [11]. It would be desirable to adapt such or similar techniques to the case which is considered here.

(6.2) Extrapolation Method. The numerical examples plainly indicate that the computational complexity for integrals over general bodies in \mathbf{R}^3 is rather high, and that extrapolation techniques can reduce this complexity drastically, see table (5.6). Hence, such techniques should be implemented whenever possible (i.e., for smooth integrands over tetrahedra).

The procedure in section 4 is not suitable for use with an extrapolation method. A modification which avoids repeated evaluations of the integrand will be written for this important case and appear elsewhere. Furthermore, since the computational efficiency of this method mainly depends on the number of subdivisions which are used, the Bulirsch numbers $n = 1, 2, 3, 4, 6, 8, 12 \dots$ should be used, see [19, eqn. (3.4.5)(b)]. It is also pointed out there that rational interpolation is more efficient than polynomial interpolation in this context.

(6.3) Mixing the Integration Methods. For the typical application (1.2) which we have in mind, it is known a priori whether the integrand is singular over a given tetrahedron or not. Note that numerical singularity also occurs if the singular point is very near to, but not inside the tetrahedron. It is therefore advisable to make an a priori choice of the integration method: an adaptive refinement procedure as indicated in section 4 for the case that the integrand is singular; and an extrapolation method for the case that the integrand is sufficiently smooth. Since the tetrahedra in the list $D_{\mathcal{T}}$ are usually small, very few refinement steps should provide for a sufficient accuracy in both cases. Because of the considerable computational complexity in 3 dimensions, it should also be investigated whether extrapolation steps are possible even inside a recursive adaptation step. It would be desirable to obtain an automatic switching between these two options which could be monitored by the performance of the method.

(6.4) Handling of the Boundary. As the numerical example in table (5.8) shows, it is inefficient to handle the boundary via the characteristic function of the body as proposed in (4.1). This creates a two-dimensional manifold of artificial singularities for the integrand, and forces the algorithm to perform a considerable amount of adaptive refinements. Convergence is slow, since a simple argument shows that the discretization error caused by the characteristic function is inversely proportional to the third root of the number of tetrahedra. A better method would be to use a more subtle approximation at the boundary as indicated in (3.3). This requires a procedure which calculates the volume of $\sigma \cap H_{\sigma}^{-1}(-\infty, 0]$ for every transverse tetrahedron σ . Furthermore, it is necessary to investigate how recursive adaptation and extrapolation techniques can be modified to include these better approximations at the boundary. Results will be reported elsewhere.

(6.5) Improved Integration Methods. We are grateful to a referee for suggesting for future study the incorporation of more efficient integration schemes given by Lyness [15–17]. An error analysis can be carried out based upon results in [18]. A comparison of efficiency with the scheme in [5] would be worthwhile. For singular integrals, the change of variables suggested in [8] may also be considered. However, we point out that the method given here does not require the a priori knowledge of the exact position or type of the singularity. Several of the improvements mentioned here have been carried out in [13].

Bibliography

- [1] Allgower, E. L. & Georg, K. (1989): Estimates for piecewise linear approximations of implicitly defined manifolds. *Appl. Math. Lett.* **1.5**, 1–7.
- [2] Allgower, E.L. & Georg, K. (1990): *Numerical continuation methods: An introduction*. Springer Verlag, Berlin, Heidelberg, New York.
- [3] Allgower, E. L. & Gnutzmann, S. (1987): An algorithm for piecewise linear approximation of implicitly defined two-dimensional surfaces. *SIAM J. Numer. Anal.* **24**, 452–469.
- [4] Allgower, E. L. & Schmidt, Ph. H. (1986): Computing volumes of polyhedra. *Math. Comp.* **46**, 171–174.

- [5] Atkinson, K. E. (1985): The numerical evaluation of particular solutions for Poisson's equation. *IMA Journal of Numerical Analysis* **5**, 319–338.
- [6] Atkinson, K. E. (1990): 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 editor, Plenum Press, New York.
- [7] Coxeter, H. S. M. (1934): Discrete groups generated by reflections. *Ann. of Math.* **6**, 13–29.
- [8] Duffy, M. (1982): Quadrature over a pyramid or cube of integrands with a singularity at the vertex. *SIAM J. Numer. Anal.* **19**, 1260–1262.
- [9] Eaves, B. C. (1984): *A course in triangulations for solving equations with deformations*. Lecture Notes in Economics and Mathematical Systems **234**, Springer Verlag, Berlin, Heidelberg, New York.
- [10] Freudenthal, H. (1942): Simplicialzerlegungen von beschränkter Flachheit. *Ann. of Math.* **43**, 580–582.
- [11] Friedman, J. H. & Wright, M. H. (1981): A nested partitioning procedure for numerical multiple integration. *ACM Transactions on Mathematical Software* **7**, 76–92.
- [12] Georg, K. (1990): Approximation of integrals for boundary element methods. To appear in: *SIAM J. of Sci. Stat. Comput.* **12**.
- [13] Georg, K. & Widmann, R. (1990): Adaptive quadratures over volumes. Preprint, Colorado State University.
- [14] Guenther, R. B. & Lee, J. (1987): *Partial differential equations of mathematical physics and integral equations*. Prentice Hall, Englewood Cliffs.
- [15] Lyness, J. N. (1976): Applications of extrapolation techniques to multidimensional quadrature of some integrand functions with a singularity. *J. Comp. Phys.* **20**, 346–364.
- [16] Lyness, J. N. (1978): Quadrature over a simplex: Part 1. A representation for the integrand function. *SIAM J. Numer. Anal.* **15**, 122–133.
- [17] Lyness, J. N. (1978): Quadrature over a simplex: Part 2. A representation for the error functional. *SIAM J. Numer. Anal.* **15**, 870–887.
- [18] Lyness, J. N. & McHugh, B. J. J. (1963): Integration over multidimensional hypercubes. 1: A progressive procedure. *Computer J.* **6**, 264–270.
- [19] Stoer, J. & Bulirsch, R. (1980): *Introduction to numerical analysis*. Springer Verlag, Berlin, Heidelberg, New York.
- [20] Stroud, A. H. (1971): *Approximate calculation of multiple integrals*. Prentice Hall, Englewood Cliffs, New Jersey.
- [21] Todd, M. J. (1976): *The computation of fixed points and applications*. Lecture Notes in Economics and Mathematical Systems **124**. Springer Verlag, Berlin, Heidelberg, New York.
- [22] Widmann, R. (1990): *An efficient algorithm for the triangulation of surfaces in \mathbf{R}^3* . Preprint, Colorado State University.
- [23] Widmann, R. (1990): *Efficient triangulation of 3-dimensional domains*. Preprint, Colorado State University.