Implementing the boundary element method for 2-D viscous sintering

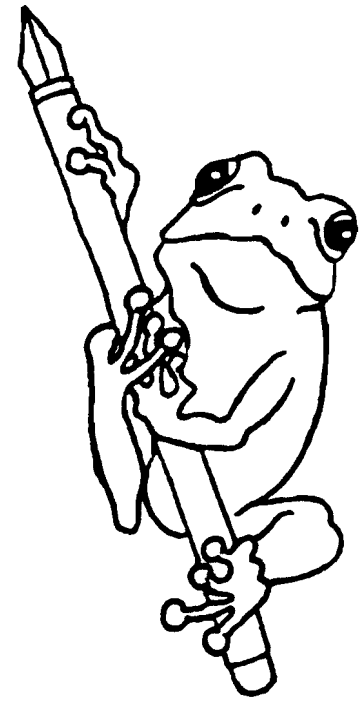
Eindhoven University of Technology
Department of Mathematics and Computing Science

RANA 91-04
April 1991

IMPLEMENTING THE BOUNDARY ELEMENT METHOD FOR 2-D VISCOUS SINTERING

by

G.A.L. van de Vorst
R.M.M. Mattheij



ISSN: 0926-4507
Reports on Applied and Numerical Analysis
Department of Mathematics and Computing Science
Eindhoven University of Technology
P.O. Box 513
5600 MB Eindhoven
The Netherlands

Implementing the Boundary Element Method for 2-D Viscous Sintering

G.A.L. van de Vorst, R.M.M. Mattheij
Department of Mathematics, University of Technology,
P.O. Box 513, 5600 MB Eindhoven, The Netherlands

Abstract

By viscous sintering it is meant processes in which a granular compact is heated to a temperature at which the viscosity of the material under consideration becomes low enough for surface tension to cause the powder particles to deform and coalesce. Here a two-dimensional model is considered. The governing (Stokes) equations describe the deformation of a viscous fluid region under the influence of the curvature of the outer boundary. A boundary element method is applied to solve the equations for an arbitrarily initial-shaped fluid region. The numerical problems that can arise in computing the curvature, in particular at places where a *cusp* is arising, are discussed. A number of numerical examples is shown for *simply connected* regions which transform themselves into circles as time increases.

1 Introduction

Sintering is the process of bringing a powder of metals, ionic crystals, or glasses to temperatures near to their melting point so that, due to the high(er) mobility, the compact densifies, thereby releasing the surface energy of the powder particles. The driving force arises from the excess free energy of the surface of the powder over that of the solid material.

There are a number of physical principles which can be held responsible for the sintering phenomena; for a review see for example Exner [2]. We are mainly interested in the case of sintering when the material transport can be modeled as a viscous Newtonian volume flow, driven solely by surface tension (viscous

sintering). This gives us a simple model of what is known as the sol-gel technique, which can for example be used to produce high-quality glasses. In this technique a glassy aerogel is heated to a temperature at which the viscosity of the glass becomes low enough for surface tension acting on the interior surface of the gel to cause the gel to collapse into a dense homogeneous material.

It is impossible to give a deterministic description of the flow within such a complex sintering geometry as an aerogel. We shall therefore investigate simple geometries; to start with in 2-D only, aiming at eventually deriving constitutive laws of the effects obtained.

A classical problem in sintering literature is the two-dimensional initial-stage unit model, the sintering of two cylinders, which has been solved analytically by Hopper [4]. The sintering of an infinite line of cylinders was simulated numerically by Ross *et al.* [9]; they were also the first to perform the simulation using a finite element method. Jagota and Dawson [5] have reported some results obtained with the finite element method for the sintering of two spheres and an infinite line of cylinders (i.e. 3-dimensional axisymmetric problems). Recently, Kuiken [6] applied a boundary element method to solve viscous sintering problems for bodies with rather smooth boundaries.

Here we present another way of implementing a boundary element method for viscous sintering problems. Our aim is to develop a code which tells us how a fluid with an arbitrarily shaped region transforms itself through time, driven only by the surface tension. The motivation to use a boundary element method rather than a finite element method is given by the fact that for computing the shape of the movement of the fluid region in time, only the velocity of the boundary curve has to be obtained. Another reason is that remeshing a boundary curve is much easier than remeshing a full 2-dimensional grid, as has to be done in a finite element method.

This paper is built up as follows: Firstly, we shall rewrite the viscous sintering problem, which is described by a set of partial differential equations (Stokes), into a set of integral equations. These integral equations are solved by the boundary element method, as proposed by Brebbia [1], in which *linear* elements are employed. Next, we shall discuss the numerical problems that arise in computing the curvature, at places on the boundary where a cusp is arising. We have solved this problem by using a specially tailored remeshing scheme. Finally, we shall give a number of numerical examples to demonstrate the usefulness of our method.

2 Problem formulation

The viscous sintering problem is modeled by a 2-dimensional viscous incompressible Newtonian fluid flow, see also Kuiken [6]. We denote the dimensionless

velocity of the fluid by $\mathbf{v}$ and the *pressure* by p. The region of flow is defined by a closed curve Γ and the interior area denoted by Ω. Furthermore we assume the region of flow is a *simply connected* surface.

A viscous incompressible Newtonian fluid can be described by the Stokes' equations, i.e. in dimensionless form (Batchelor [3]),

$$\begin{aligned} \Delta \mathbf{v} - \text{grad } p &= 0 \\ \text{div } \mathbf{v} &= 0, \end{aligned} \tag{1}$$

with stress tensor

$$\mathcal{T}_{ij} = -\delta_{ij} p + \left(\frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i} \right). \tag{2}$$

On the boundary Γ the normal component of the stress tensor is proportional to the local curvature κ (=div $\mathbf{n}$) of the boundary. This condition can also be expressed as

$$\mathcal{T}\mathbf{n} = \kappa \mathbf{n}, \tag{3}$$

where $\mathbf{n}$ is the outward unit normal vector of Γ.

The equations (1)-(3) described above do not ensure a unique solution $\mathbf{v}$. It can be seen, cf [10], that a superposition of an arbitrary rigid-body *translation* or an arbitrary rigid-body *rotation* upon any particular solution of these equations, is also a solution of equation (1) and will not alter the stress field at the boundary. Thus in total we need to add three extra conditions to ensure that the velocity field obtained is unique. In order to get rid of the translation freedom, we formulate the problem to be stationary at a (reference) point in the fluid, say $\mathbf{x}^r$. With regard to this reference point the velocity of the boundary points are computed. The most natural choice for this reference point is the centre of mass: the point where the gravity forces would grip the body, thus:

$$\mathbf{v}(\mathbf{x}^r) = 0. \tag{4}$$

Furthermore we assume the tangential component of the velocity at the boundary is zero, i.e.

$$\int_\Gamma (\mathbf{v}, \tau)\, d\Gamma = 0, \tag{5}$$

where τ is the tangential vector of the boundary Γ. Combining this with Stokes formula it follows from equation (5) that the flow in Ω is *irrotational.*

The problem defined by the equations (1)-(5) has a unique solution. If we were to solve these equations for a fixed boundary Γ we would find, in general, a non-zero flow field $\mathbf{v}$ on Γ. Thus in time (t, i.e. the dimensionless time) the boundary is moving. The displacement of the boundary can be found from the derived velocity field $\mathbf{v}$ in the following way,

$$\frac{d\mathbf{x}}{dt} = \mathbf{v}(\mathbf{x}) \qquad (\mathbf{x} \in \Gamma), \tag{6}$$

subjected to an initial boundary Γ_0 at $t = t_0$. It is worth noting that equation (6) says that the *material* points of the boundary are moving in the direction of the *characteristic* curves.

We remark that we are only interested in the movement of the region Ω. Hence only the velocity at the boundary is required, from which the shape and motion of the body can be calculated directly. Therefore we shall represent the solution in terms of boundary distributions of the single- and double-layer potentials for the Stokes' equation. The integral equation that can be derived for the Stokes' equation at a point $\mathbf{x}$, say, when the boundary is sufficiently "smooth" (see also [10]), reads in matrix notation

$$\mathcal{C}\mathbf{v}(\mathbf{x}) + \int_\Gamma \mathcal{Q}\mathbf{v}\, d\Gamma_y = \int_\Gamma \mathcal{U}\mathbf{b}\, d\Gamma_y. \tag{7}$$

Here $\mathcal{C}$,$\mathcal{Q}(\mathbf{x},\mathbf{y})$ and $\mathcal{U}(\mathbf{x},\mathbf{y})$ are 2×2 matrices with coefficients c_{ij}, q_{ij} and u_{ij} respectively:

$$c_{ij} = \begin{cases} \delta_{ij} & \mathbf{x} \in \Omega \\ \frac{1}{2}\delta_{ij} & \mathbf{x} \in \Gamma, \end{cases} \tag{8}$$

$$q_{ij} = \frac{r_i r_j}{\pi R^4} r_k n_k, \tag{9}$$

$$u_{ij} = \frac{1}{4\pi}\Big[\, \delta_{ij} \log \frac{1}{R} + \frac{r_i r_j}{R^2} \,\Big], \tag{10}$$

where $r_i = x_i - y_i$ and $R = \sqrt{r_1^2 + r_2^2}$, and the vector $\mathbf{b}$

$$\mathbf{b} = \kappa \mathbf{n}. \tag{11}$$

Many authors attribute the analysis and the integral equation (7) to Ladyzhenskaya [7] (1963), but actually it was Lorentz [8] who derived this formulation essentially, back in 1896.

In order to make the integral equation (7) uniquely solvable, we have to add the conditions (4) and (5). Since the reference point $\mathbf{x}^r$ is taken in Ω, we obtain from equation (7)

$$\int_\Gamma \mathcal{Q}^r\mathbf{v}\, d\Gamma_y = \int_\Gamma \mathcal{U}^r\mathbf{b}\, d\Gamma_y. \tag{12}$$

Here $\mathcal{Q}^r = \mathcal{Q}(\mathbf{x}^r,\mathbf{y})$ and $\mathcal{U}^r = \mathcal{U}(\mathbf{x}^r,\mathbf{y})$. From the equations (5), (7) and (12) we derive the following two integral equations which have to be solved:

$$\mathcal{C}\mathbf{v}(\mathbf{x}) + \int_\Gamma (\mathcal{Q} - \mathcal{Q}^r)\mathbf{v}\, d\Gamma_y = \int_\Gamma (\mathcal{U} - \mathcal{U}^r)\mathbf{b}\, d\Gamma_y, \tag{13}$$

$$\int_\Gamma (\mathbf{v}, \tau)\, d\Gamma = 0. \tag{14}$$

3 The Boundary Element Method

The problem is ideally suited to be solved by a boundary element method (Brebbia [1]). Therefore the boundary Γ will be discretized into a sequence of N elements and the velocity and surface tension are written in terms of their values at a sequence of nodal points. From the discretized form of equation (13) for every nodal point, together with the discretized form of equation (14), we derive a system of (2N+1) linear algebraic equations with 2N unknowns.

From this system we obtain the approximate velocity at time $t = t_k$ at the nodal boundary points. The displacement of the boundary at time $t = t_{k+1} = t_k + \Delta t$ can then be obtained by discretizing equation (6). Here we will use a simple Forward Euler discretization scheme, i.e.

$$\mathbf{x}(t_{k+1}) = \mathbf{x}(t_k) + \Delta t \mathbf{v}(\mathbf{x}(t_k)). \tag{15}$$

As mentioned in section 2, equation (15) is an approximation of the trajectories of the material boundary points. These trajectories may not intersect each other, which gives restrictions for the time step Δt. The integral equations, subjected to the derived (new) boundary, are solved again which give the velocity at $t = t_{k+1}$, etc...

After discretization of the boundary Γ into a polygon, we define the polynomial functions $\tilde{\mathbf{v}}$ and $\tilde{\mathbf{b}}$, cf (11), which apply at a typical element 'j',

$$\tilde{\mathbf{v}} = \Phi v^j \qquad \text{and} \qquad \tilde{\mathbf{b}} = \Phi b^j, \tag{16}$$

where v^j and b^j are the velocity and surface tension of the boundary nodal point $\mathbf{x}^j$. The interpolation function Φ is a $2 \times 2M$ matrix of shape functions:

$$\Phi = \begin{bmatrix} \phi_1 & 0 & \phi_2 & 0 & \dots & \phi_M & 0 \\ 0 & \phi_1 & 0 & \phi_2 & \dots & 0 & \phi_M \end{bmatrix}. \tag{17}$$

The functions ϕ_l are the standard finite-element-type polynomials, see also Brebbia [1]. The number M is equal to the degree of the polynomial approximation plus one. After substituting the approximation polynomial (16) into equation (13) with a discretized boundary, we obtain the following equation for an *arbitrary* nodal point i:

$$\mathcal{C}^i v^i + \sum_{j=1}^{N} \Big(\int_{\Gamma_j} (\mathcal{Q} - \mathcal{Q}^r)\Phi \, d\Gamma_y \Big) v^j = \sum_{j=1}^{N} \Big(\int_{\Gamma_j} (\mathcal{U} - \mathcal{U}^r)\Phi \, d\Gamma_y \Big) b^j. \tag{18}$$

For constant elements, i.e. where $\tilde{\mathbf{v}}$, $\tilde{\mathbf{b}}$ are assumed to be constant over each element, we note that $\mathcal{C}^i = 0.5\mathcal{I}$ for a "smooth" boundary. $\mathcal{C}^i$ will be for higher-order elements in general a 2×2 matrix.

Consider the following types of integrals, in matrix notation, which have to be evaluated for every element Γ_j and every nodal point $\mathbf{x}^i$,

$$\begin{aligned} \hat{H}^{ij} &= \int_{\Gamma_j} \mathcal{Q}\Phi \, d\Gamma_y, & H_r^j &= \int_{\Gamma_j} \mathcal{Q}^r \Phi \, d\Gamma_y, \\ G^{ij} &= \int_{\Gamma_j} \mathcal{U}\Phi \, d\Gamma_y, & G_r^j &= \int_{\Gamma_j} \mathcal{U}^r \Phi \, d\Gamma_y, \\ H^{ij} &= \begin{cases} \hat{H}^{ij} & i \neq j \\ \hat{H}^{ij} + C^i & i = j. \end{cases} \end{aligned} \tag{19}$$

We obtain for equation (18)

$$\sum_{j=1}^{N} \left(H^{ij} - H_r^j \right) v^j = \sum_{j=1}^{N} \left(G^{ij} - G_r^j \right) b^j. \tag{20}$$

If we now let i vary from 1 to N, and note that the right-handside vector, say $\mathbf{F}$, of equation (20) is known we derive the following system:

$$\mathcal{H}\mathbf{v} = \mathbf{F}, \tag{21}$$

where $\mathcal{H} = H^{ij}$.
The system (21) can be solved uniquely when the discretized form of the extra relation (14) is added.

In what follows we consider *linear* approximations of $\mathbf{v}$ and $\mathbf{b}$ over an element, i.e. we apply linear boundary elements only. For more details we refer to [10].

4 Computation of curvature and mesh

The driving force of the boundary movement is a tension that depends on the curvature of the boundary. Therefore it is important that a good approximation is used to determine this curvature. Especially when in a certain point of the boundary a *cusp* arises. Here the curvature becomes unbounded, thus the approximate curvature can have large errors.

The curvature at a boundary point, say $\mathbf{x}^2$ is approximated by fitting a quadratic polynomial through this point and its neighbours $\mathbf{x}^1$ and $\mathbf{x}^3$, i.e.

$$\tilde{\mathbf{x}}(s) = \mathbf{x}^1\phi_1(s) + \mathbf{x}^2\phi_2(s) + \mathbf{x}^3\phi_3(s), \tag{22}$$

where $\phi_1(s) = \frac{1}{2}s(s-1)$; $\phi_2(s) = 1 - s^2$; $\phi_2(s) = \frac{1}{2}s(s+1)$ and $-1 \leq s \leq 1$. For the approximate curvature at the nodal point $\mathbf{x}^2 = \tilde{\mathbf{x}}(0)$, we obtain:

$$\begin{aligned} \kappa(\mathbf{x}^2) &= \frac{(x_2)_s(x_1)_{ss} - (x_1)_s(x_2)_{ss}}{\left[((x_1)_s)^2 + ((x_2)_s)^2 \right]^{\frac{3}{2}}} \\ &\doteq \frac{4\left[(x_2^3 - x_2^1)(x_1^1 - 2x_1^2 + x_1^3) - (x_1^3 - x_1^1)(x_2^1 - 2x_2^2 + x_2^3) \right]}{\left[(x_1^3 - x_1^1)^2 + (x_2^3 - x_2^1)^2 \right]^{\frac{3}{2}}}. \end{aligned} \tag{23}$$

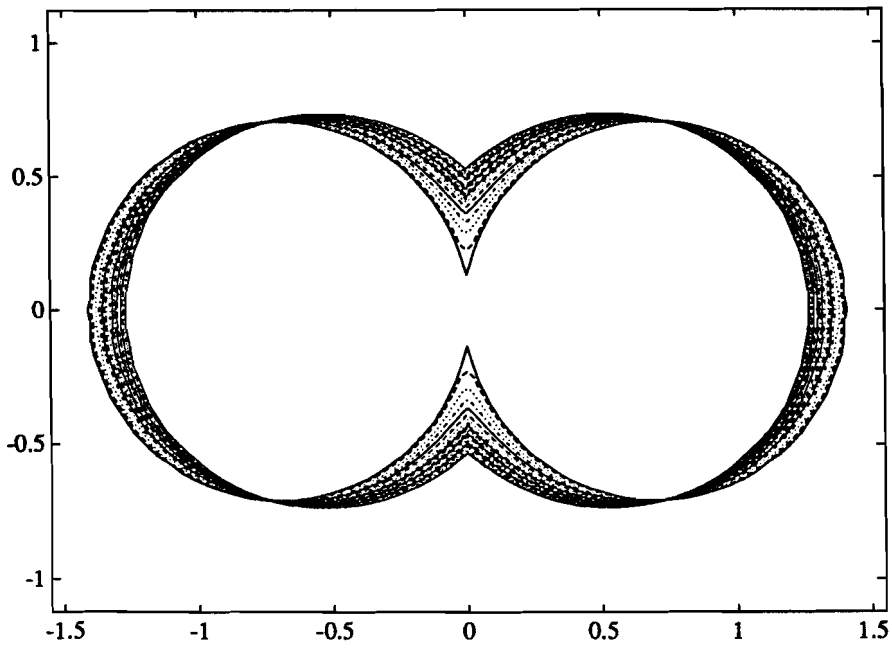


Figure 1: Two sintering cylinders with equal diameters. The same mesh is used throughout the simulation. The boundary curves refer to values of time t=0.0(0.1)2.0.

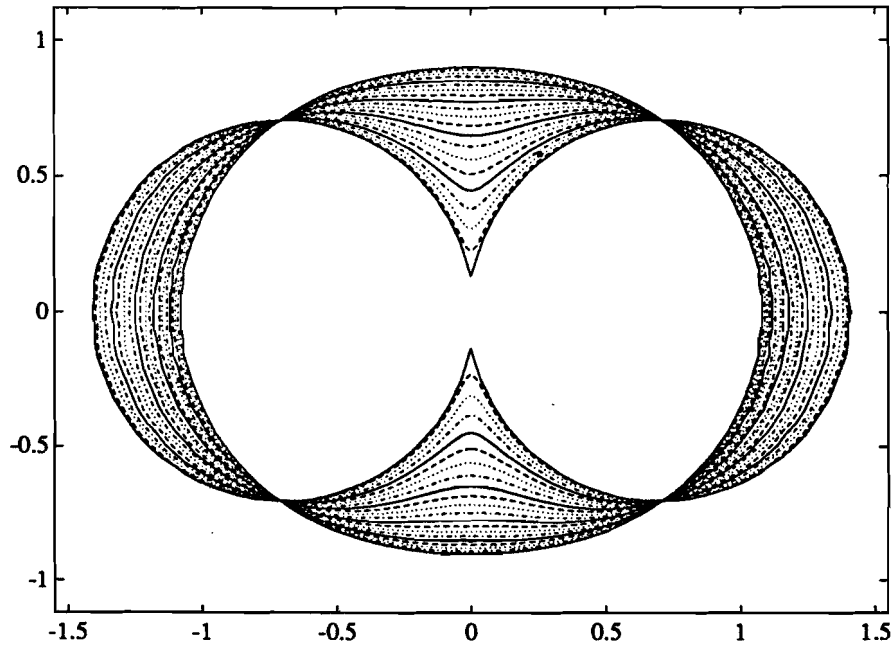


Figure 2: Two sintering cylinders with equal diameters. A mesh verification is done at each time step. The boundary curves refer to values of time t=0.0(0.1)2.0.

This way we can derive the curvature at all nodal points. The curvature at any point of an element is obtained from the polynomial fit (16) through the approximate curvature (23) of these nodal points.

As mentioned above, when approximating the curvature, problems are arising if a cusp occurs. Especially when we make an extra refinement of the mesh in the neighbourhood of such a point, larger errors can be made when computing the curvature there using equation (23). This can be seen as follows: we can assume that the *spatial* discretization error, i.e. the error that is made by the approximation of the boundary by a polygon, is smaller than the *time* discretization error, i.e. the error that arises from the time stepping scheme (15). This Forward Euler scheme gives a global time discretization error $\mathcal{O}(\Delta t)$. Furthermore, the time step Δt can not be made very small, because then the computing time would become prohibitively large. Thus we can say that the computed boundary deviates $\mathcal{O}(\Delta t)$ from the exact curve. When approximating the curvature at points where the mesh is fine, we loose some accuracy because some points are necessarily very close to each other. The quotient of the approximate curvature of equation (23) will then be of $\mathcal{O}(\Delta t)$ over $\mathcal{O}(\Delta t)$. Hence the computed curvature can deviate considerably from the exact curvature. Thus the mesh has to be checked every time step in such a way that the collocation points do not get to close to each other. On the other hand, it seems reasonable that the collocation points have to lie close to each other at places on the boundary where the curvature is large; since there we are expecting large variations of the velocity field $\mathbf{v}$ of the boundary. These two conflicting aspects have to be wrought together in an algorithm that takes care of the mesh adaptation and verification.

The implementation of the latter algorithm is done in the following way. Assume that a suitable initial mesh is given. If at a certain time step Δt is satisfying the conditions, which will follow shortly, nothing is done. If this is not the case, the mesh has to be updated locally.

We introduce two positive constants h_{min} and h_{max}, which are given bounds for the distance between two successive collocation points, i.e.

$$h_{min} \leq \delta_i \leq h_{max}, \tag{24}$$

where δ_i denotes the actual distance between two such points: $\delta_i = \|\mathbf{x}^{i+1} - \mathbf{x}^i\|$. The value of h_{min} is taken of $\mathcal{O}(\Delta t)$. Furthermore, only when a "cusp" arises at a collocation point, say $\mathbf{x}^j$, the distance between the two neighbours of $\mathbf{x}^j$ must be of order Δt:

$$\|\mathbf{x}^{j-1} - \mathbf{x}^{j+1}\| \geq h_{min}. \tag{25}$$

Otherwise, we lose some digits when computing the curvature at this point using equation (23). Thus the approximate curvature will certainly have large errors then. The restriction (25) also limits the magnitude of the curvature. From equation (23) we derive that the maximum value of the curvature is $\mathcal{O}(\Delta t^{-1})$. The requirement of a fine mesh at places where the curvature is large, can be fulfilled through verifying that the straight line between two contiguous nodal points is

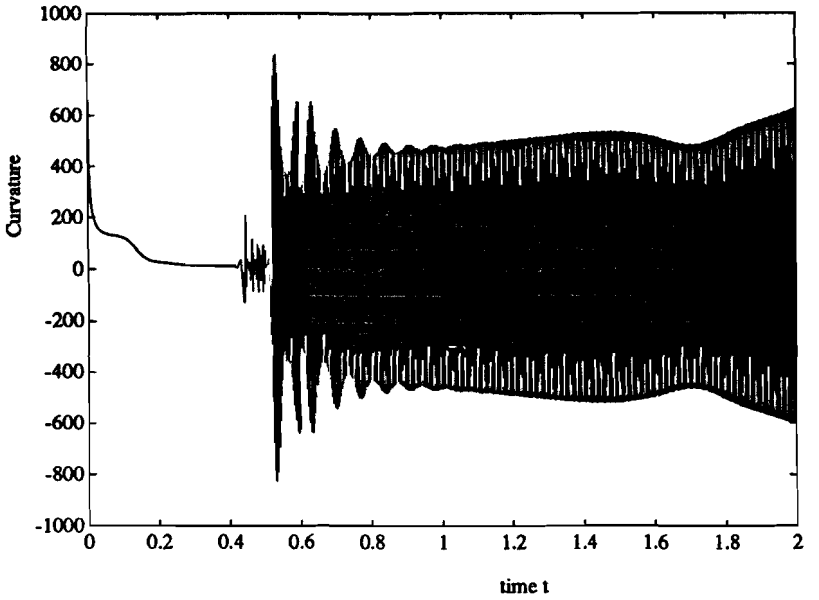


Figure 3: The curvature of the neck point. The same mesh is used throughout the simulation. Large oscillations develop when the trajectories of the nodal points in the neck region come close together.

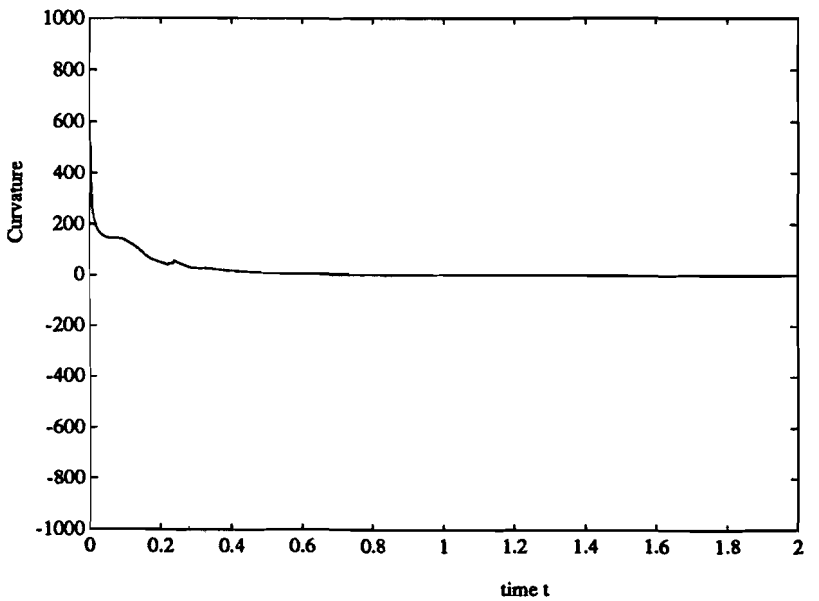


Figure 4: The curvature of the neck point. The mesh is "updated" each time step.

not deviating too much (in a relative sense), from that part of the boundary which lies between those points; i.e. we look for a kind of equidistribution of the curvature over the elements. In [10] we used the following implementation for this criterion:

$$\max\left(\mathrm{h}_i^i, \mathrm{h}_i^{i+1}\right) \leq \epsilon_c \delta_i, \tag{26}$$

where

$$\mathrm{h}_i^k = \frac{1}{|\kappa(\mathbf{x}^k)|}\left(1 - \sqrt{1 - \frac{1}{4}\delta_i^2[\kappa(\mathbf{x}^k)]^2}\right), \quad k = i, i+1 \tag{27}$$

and ϵ_c is a certain threshold value. At least we want the collocation points to lie fairly uniformly distributed on the boundary, i.e.

$$\epsilon_s \leq \frac{\delta_i}{\delta_{i-1}} \leq \frac{1}{\epsilon_s}, \tag{28}$$

where ϵ_s is a given parameter, greater than 1.

In decreasing order of importance of the conditions which have to be fulfilled are: (i) the distance condition (24) and (25), (ii) the curvature criterion (26) and at least, (iii), the smooth variation of the mesh (28). We assume the initial mesh satisfies those conditions. When, after some steps, one of the conditions is not fulfilled any more at a certain nodal point, we update the mesh locally. We try to *shift* this badly behaving point along the neighbouring elements. This is done in such a way that the mesh points do not get too close for those neighbouring collocation points. If this is somehow not possible, then an extra collocation point is introduced and put into the mesh, using a locally quadratic interpolation; or otherwise this badly behaving point is removed. Only when this point is a *cusp*, than the point may **not** be shifted along an element or removed. In this case, we only shift or remove the two neighbouring collocation points. In order to ensure that the curvature does not change in the cusp, the cusp is shifted slightly in the normal direction; hereby care is taken that the distance between the "new" and the "old" cusp point is smaller than the discretization error.

5 Numerical results

In this section we show a number of results for some simply connected surfaces. Since, as we said in section 1, the driving force for sintering arises from the excess of free surface energy, a 2-dimensional viscous fluid region Ω transforms itselfs into a circle when the time t is going to infinity. This is because the length of the boundary curve attains its minimum then. Thus a circle must be the result of the simulation considered. Also, the fluid is assumed to be incompressible. In 2-dimensions this means that the *total* surface of the moving fluid region must be constant in time.

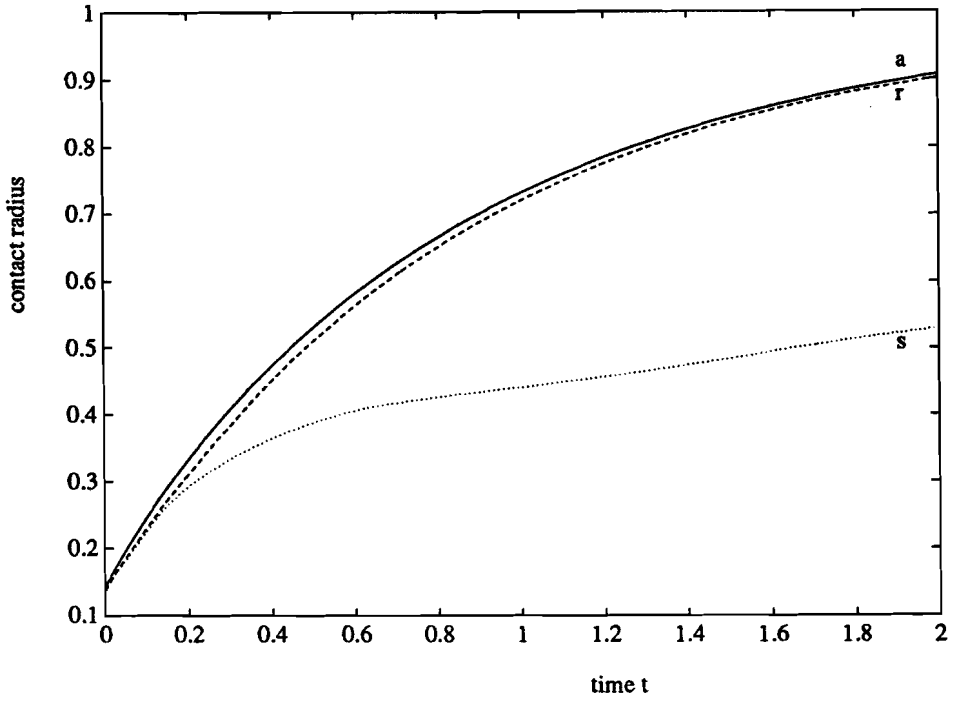


Figure 5: The contact radius of the sintering cylinders, obtained by both numerical simulations compared with the analytical solution (**a**), i.e. the same mesh (**s**) and when the mesh is updated (**r**) at each time step.

To demonstrate the usefulness of the mesh distribution algorithm, described in section 4, we consider the case of two *sintering* spheres in 2-dimensions, fairly shortly after they have made contact. In 2-dimensions these are two circles and in 3-dimensions this models two infinity long cylinders. In the figures 1 and 2 the numerical results obtained at different time steps for the coalescence of the two circles are shown. As an initial time step we chose $\Delta t = 0.002$. During the simulation this time step is automatically updated using the formula we had derived in [10]. The centre of mass is taken as the reference point, $\mathbf{x}^r$, which is chosen to be the origin.

In figure 1 we show the results obtained without mesh verification. Thus the trajectories of the nodal points of the initial mesh are followed. When a mesh verification is applied at each time step (as proposed in section 4) we derive the shapes as shown in figure 2. The shapes of both graphs are plotted at the same time points. When comparing these graphs, it can be seen that the shape evolution of both curves, completely differ already at an early stage, especially near the contact region of both circles. (In sintering literature, this contact region is usually denoted as the *neck* region, see Exner [2]).

The reason that the shape evolutions, develop so differently is due to an oscillation in the approximation of the curvature from the boundary in the neck

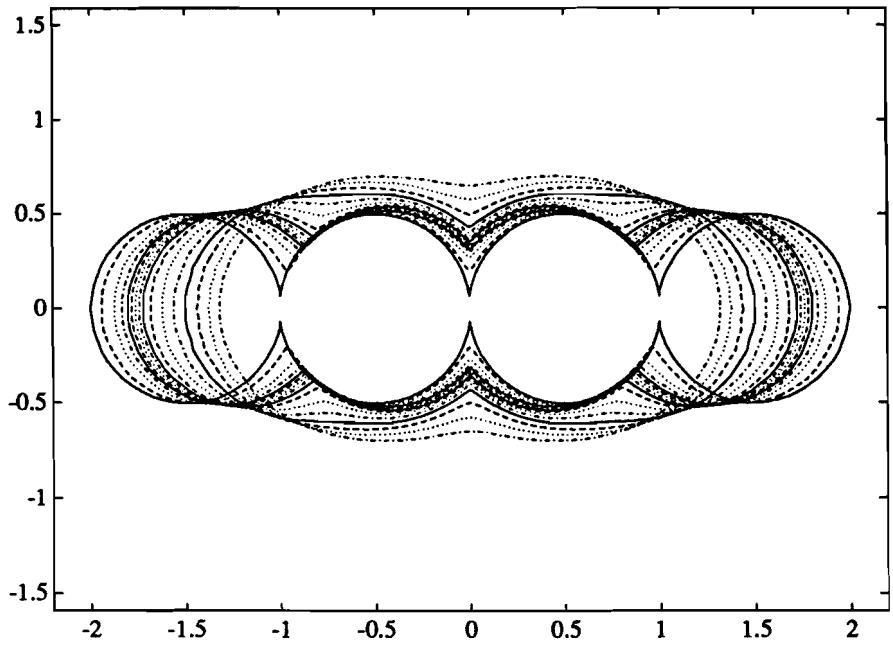


Figure 6: Four sintering cylinders with equal diameters. Here the mesh is kept constant used during the simulation. The boundary curves refer to time values t=0.0(0.2)3.0.

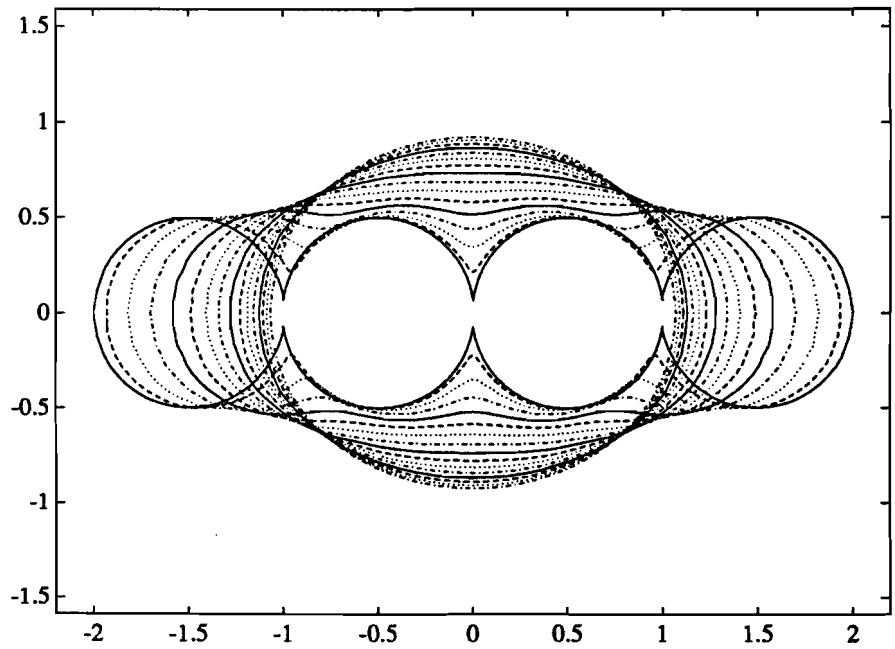


Figure 7: Four sintering cylinders with equal diameters. A mesh verification is done at each time step. The boundary curves refer to time values t=0.0(0.2)3.0.

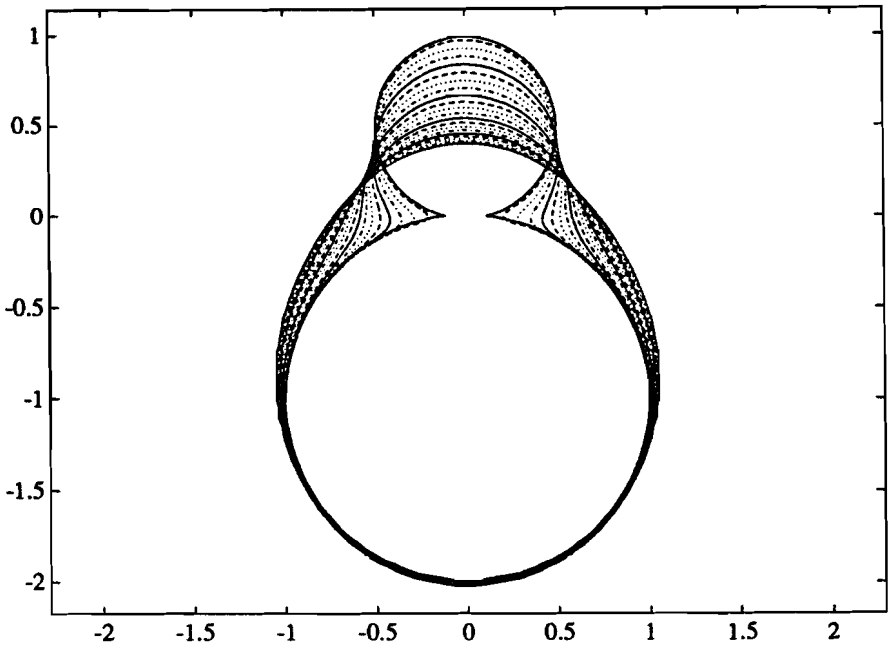


Figure 8: The transformation of two sintering cylinders with different diameters. The diameters of the cylinders are taken to be 0.5 and 1, respectively. The boundary curves refer to values of time t=0.0(0.1)2.0.

region. In figure 3 we show the curvature of the point at the boundary where both circles are making contact. Here, the mesh is not changed throughout the numerical process. As can be seen, a very rapid oscillation is occuring in the approximate curvature, at a time when the trajectories of the nodal points come very close to each other. This oscillation is caused, as is mentioned in section 4, by the loss of accurate digits when computing the approximate curvature. These oscillations of the curvature do not disturb the movement of the boundary, in the sense of a brake-down of our numerical algorithm. The fact that these oscillations are bounded is due to the "natural" damping of such viscous fluid. Kuiken [6] has shown that a small disturbance of the boundary is damped out rapidly. In figure 4 we again give the curvature of this contact point. Now a mesh verification is done at every time step. Here, the oscillations in the (approximate) curvature have disappeared.

The development during sintering of the contact radius, i.e. the neck, is also of physical interest. This contact radius is a measure of how "strong" a sintering compact already is. When this contact radius is small, a smaller force is necessary to brake the contact between both cylinders than at later stages of the sintering process, when the contact radius is bigger. In figure 5 the contact radius is shown, which is obtained in both numerical simulations. In this graph we have

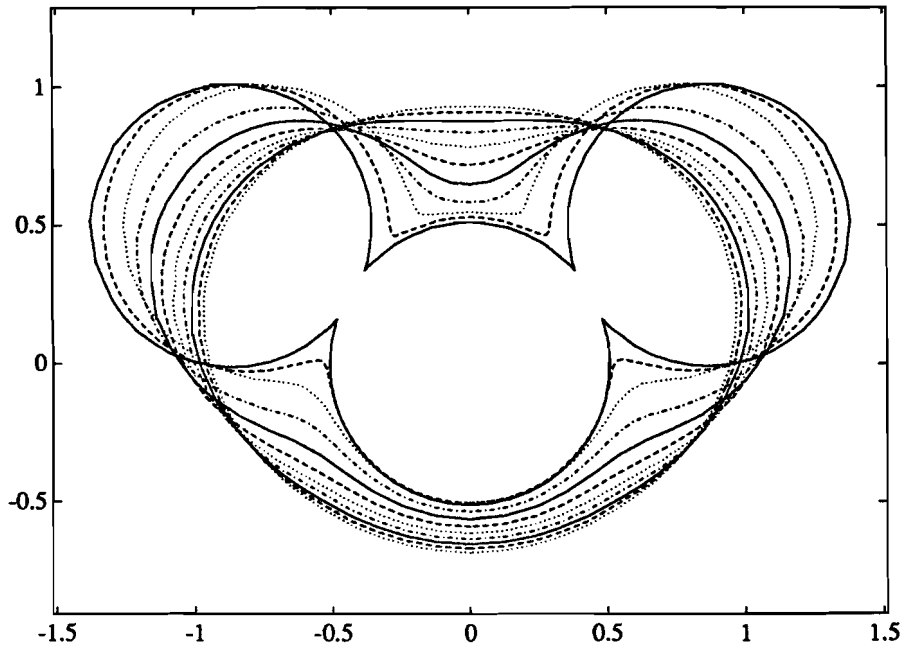


Figure 9: The transformation of three sintering cylinders with a angle between the mid points of the initial cylinders of 120 degrees. The boundary curves refer to values of time t=0.0(0.2)2.0.

also plotted the analytical solution for the sintering of two circles, which has been derived by Hopper [4]. As can be seen the numerical solution, matches well with the exact solution when a mesh verification is done.

Another example of two different numerical simulations for a same initial shape is shown in the figures 6 and 7. Here we simulate the sintering of four equal cylinders which are lying in a row. Again, this example illustrates the unpredictability of the shape evolution, especially when more complex geometries are taken. The shapes obtained by the mesh verification algorithm is the one, which is expected in physical reality.

In figure 8 we show another example of a shape evolution at different time points, of a fluid region with a "near" cusp. The initial shape consists of two cylinders of arbitrary diameters. The diameters of the cylinders is taken 0.5 and 1, respectively. Near the region of contact of the cylinders we have to deal with a large varying curvature. Again, the centre of mass (which is lying somewhere on the y axis) is taken as reference point. Our remeshing algorithm did work well.

At last, we consider the geometry of three equal cylinders, which are making an angle with each other. For the angle between the mid points of the circles we took 120 degrees. In figure 9 we show the obtained shape evolution when the

mesh verification algorithm is used.

In future, we plan to investigate multiply connected regions, i.e. viscous fluid regions with gas bubbles.

ACKNOWLEDGMENT

This research was supported by the Technology Foundation (STW).

References

[1] Brebbia C.A. and Dominguez J., Boundary Elements An Introductory Course, Computational Mechanics Publications, Southampton, 1989.

[2] Exner H.E., Grundlagen von Sintervorgängen, Gebrüder Borntraeger, Berlin-Stuttgart, 1978.

[3] Happel J. and Brenner H., Low Reynolds Number Hydrodynamics with special applications to particulate media, Prentice-Hall, London, 1965.

[4] Hopper R.W., Plane Stokes flow driven by capillarity on a free surface, J. Fluid Mech., Vol. 213, pp. 349-375, 1990.

[5] Jagota A. and Dawson P.R., Simulation of the Viscous Sintering of Two Particles, J. Am. Ceram. Soc., Vol. 73, pp. 173-177, 1990.

[6] Kuiken H.K., Viscous sintering: the surface-tension-driven flow of a liquid form under the influence of curvature gradients at its surface, J. Fluid Mech., Vol. 214, pp. 503-515, 1990.

[7] Ladyzhenskaya O.A., The Mathematical Theory of Viscous Incompressible Flow, Gordon and Beach., New York-London, 1963.

[8] Lorentz H.A., Eene Algemeene Stelling omtrent de Beweging eener Vloeistof met Wrijving en eenige daaruit afgeleide Gevolgen, Verslag Kon. Acad. v. Wetensch. Amsterdam, Vol.5, pp. 168-175, 1896.

[9] Ross J.W., Miller W.A. and Weatherly G.C., Dynamic Computer Simulation of Viscous Flow Sintering Kinetics, J. Appl. Phys., Vol. 52, pp. 3884-88, 1981.

[10] Vorst G.A.L. van de, Mattheij R.M.M., Kuiken H.K., A boundary element solution for 2-dimensional viscous sintering, J. Comput. Phys. (submitted), 1990.