S©MR ISSN 1813-3304

СИБИРСКИЕ ЭЛЕКТРОННЫЕ
МАТЕМАТИЧЕСКИЕ ИЗВЕСТИЯ

Siberian Electronic Mathematical Reports
http://semr.math.nsc.ru

Том 18, №2, стр. 1367–1389 (2021) УДК 514.8
DOI 10.33048/semi.2021.18.104 MSC 53-08

ON SEMI-IMPLICIT NUMERICAL METHOD FOR SURFACE DIFFUSION EQUATION FOR TRIANGULATED SURFACES

YU.D. EFREMENKO

Abstract. We propose a semi-implicit numerical approach to computation of solutions to the surface diffusion equation for triangulated surfaces. In addition, an algorithm of re-triangulation of surfaces was developed to handle singularities that appear during surface evolution. A number of numerical solutions for different initial surfaces is presented.

Keywords: surface diffusion equation, triangulated surface, semi-implicit numerical methods.

1. Introduction

In this paper, we study numerical solutions of surface diffusion equation:

$$\begin{cases} V_N = -\Delta H \quad t > 0 \\ S|_{t=0} = S_0 \end{cases} \tag{1}$$

where H is the mean curvature of the surface $S(t)$, Δ is the Laplace–Beltrami operator, $V_N = \frac{\partial \vec{X}}{\partial t} \cdot \vec{N}$ is the normal velocity, and $\vec{X} : \ S(t) \ni \ x \mapsto x \in \mathbb{R}^3$ is the embedding of the surface. This equation is invariant with respect to the change of the orientation of the surface: both the normal and the mean curvature change their signs simultaneously.

This equation was first obtained in work [1] while studying deformation of bodies under the influence of high temperatures. The mechanism of surface diffusion, described by equation (1), plays an important role in the process of sintering of different materials, especially during its first stage [2, 3]. Later, the equation

Efremenko, Yu.D., On semi-implicit numerical method for surface diffusion equation for triangulated surfaces.

The work is supported by Mathematical Center in Akademgorodok, the agreement with Ministry of Science and High Education of the Russian Federation number 075-15-2019-1675.

Received April, 24, 2021, published November, 24, 2021.

of surface diffusion was generalized and studied from geometrical point of view in a number of works. The existence of solution and its uniqueness in a small neighborhood $[0, \varepsilon]$ for closed oriented surfaces were shown in [4], there it was also demonstrated that the surfaces close to a sphere converge to it at an exponential rate. Also, in work [5] it was shown that the surface diffusion flow is able to generate singularities — self-intersections or surface breaks, which are especially important to be modelled numerically.

Solving the above equation means finding a family of surfaces $\{S(t) \mid t \in [0, T]\}$, embedded into $\mathbb{R}^3$, whose normal velocities satisfy equation (1). It is hard to be analyzed analytically as it is a non-linear equation of fourth order with an operator depending on the surface at every point of time. Therefore, in this case numerical methods of study have particular importance. For such formulation, we search for a discreet family of surfaces $\{S^n \mid n = 0, 1, 2, \dots\}$, where S^n approximates $S(n\tau)$ for a chosen time step $\tau > 0$.

Every surface S^n is found by the previous steps S^i, $i \leq n-1$, using a system of difference equations obtained by approximation of equation (1). Correct approximation of the equation and construction of a stable numerical scheme is the most important problem of the process of solving the equation numerically. But for equation (1), due to its complexity we are not able to prove the convergence and stability, hence, a special attention is paid to the properties of solution, such as volume preservation and nonincrease of surface area [6]. We denote the volume, bounded by the surface $S(t)$, and the area of the surface $S(t)$ by $V(t)$ and $A(t)$. Then we have:

$$\begin{aligned} &\frac{d}{dt}V(t) = \int_{S(t)} V_N ds = -\int_{S(t)} \Delta H ds = -\int_{S(t)} div(grad\ H) = 0; \\ &\frac{d}{dt}A(t) = -\int_{S(t)} V_N H ds = -\int_{S(t)} \Delta H H ds = -\int_{S(t)} |\nabla H|^2 \leq 0. \end{aligned} \tag{2}$$

The last equality in the first line (2) is true by Green's theorem (see [7]) for a closed oriented surface $S(t)$. The fulfillment of these properties, taking into account approximation errors, is often required for the constructed numerical schemes.

In [1], an explicit scheme for the case of a surface of revolution was proposed: in this case, the problem becomes one-dimensional and reduces to a curve, which is a profile of the considered surface. This scheme is unstable for large steps, which imposes a constraint on the size of the step τ. It is worth mentioning that this equation is highly sensitive to calculation accuracy, therefore, constructing a numerical scheme is definitely challenging. In paper [6], a completely implicit scheme for the case of triangulated surface in $\mathbb{R}^3$ was proposed. The considered examples show that it is stable, but the theoretic estimate was not presented. Although this scheme allows to perform calculations with a large step, compared to the explicit scheme, its size is still bounded from above for invertibility of the matrix in the scheme.

Also in work [8], an approach to numerical solution of equation (1) by the «level-set» method using semi-implicit scheme was proposed. In this case, the surface is given as the boundary of the evolving area in a three-dimensional space, split into standard cubical cells. This approach has some advantages, for example, it admits processing of topology changes, but due to necessity of discretization of the whole

three-dimensional space, it involves significant restrictions of the area by size (or by level of detail).

Therefore, for a triangulated surface there exist only two schemes for solving the considered equation and both of them have particular disadvantages. In this paper, a semi-implicit scheme for numerical solution of the surface diffusion equation for triangulated surfaces is proposed and the results of numerical experiments performed using the constructed scheme are presented.

2. Construction of the numerical scheme

2.1. **Approximation of the equation.** The main goal of the work is to construct a semi-implicit numerical scheme for equation (1) with a sufficiently good invertible operator. To do that, we will need the following formula [9]:

$$\Delta\vec{X} = 2H\vec{N} \tag{3}$$

Note that relation (3) does not depend on the choice of orientation due to the presence of the product of H and $\vec{N}$. Using (3), we can express the mean curvature: $H = \frac{1}{2}\Delta\vec{X}\cdot\vec{N}$. Therefore, (1) can be rewritten in the form

$$V_N = -\frac{1}{2}\Delta(\Delta\vec{X}\cdot\vec{N}). \tag{4}$$

In local coordinates, the Laplace–Beltrami operator can be written in the form $\Delta = \sum_{i,j}\frac{1}{\sqrt{g}}\frac{\partial}{\partial x_i}(g^{ij}\sqrt{g}\frac{\partial}{\partial x_j})$. We denote arbitrary functions on the surface S with values in $\mathbb{R}$ by $\vec{u},\vec{v}$ and use the formula for the Laplace–Beltrami operator on the scalar product [9]:

$$\Delta(\vec{u}\cdot\vec{v}) = (\Delta\vec{u}\cdot\vec{v}) + (\vec{u}\cdot\Delta\vec{v}) + 2g^{ij}(\frac{\partial\vec{u}}{\partial x_i}\cdot\frac{\partial\vec{v}}{\partial x_j}).$$

With the help of the above equality, we can transform the equation (4) in the following way:

$$\begin{aligned}
V_N = -\frac{1}{2}\Delta(\Delta\vec{X}\cdot\vec{N}) = \\
= -\frac{1}{2}\Delta^2(\vec{X}\cdot\vec{N}) + \frac{1}{2}\Delta(\vec{X}\cdot\Delta\vec{N}) + \Delta\left(g^{ij}(\frac{\partial\vec{X}}{\partial x_i}\cdot\frac{\partial\vec{N}}{\partial x_j})\right) = \\
= -\frac{1}{2}\Delta^2(\vec{X}\cdot\vec{N}) + \frac{1}{2}\Delta(\vec{X}\cdot\Delta\vec{N}) + \Delta\left(g^{ij}\frac{\partial}{\partial x_j}(\frac{\partial\vec{X}}{\partial x_i}\cdot\vec{N}) - g^{ij}(\frac{\partial^2\vec{X}}{\partial x_i\partial x_j}\cdot\vec{N})\right) = \\
= -\frac{1}{2}\Delta^2(\vec{X}\cdot\vec{N}) + \frac{1}{2}\Delta(\vec{X}\cdot\Delta\vec{N}) - \Delta(g^{ij}b_{ij}) = \\
= -\frac{1}{2}\Delta^2(\vec{X}\cdot\vec{N}) + \frac{1}{2}\Delta(\vec{X}\cdot\Delta\vec{N}) - \Delta H
\end{aligned}$$

In the penultimate equality, the orthogonality of the tangent vector $\frac{\partial\vec{X}}{\partial x_i}$ and the normal vector $\vec{N}$ was used, and, moreover, we used the notation b_{ij} for the components of the second quadratic form. Based on the obtained equation

$$V_N = -\frac{1}{2}\Delta^2(\vec{N}\cdot\vec{X}) + \frac{1}{2}\Delta(\vec{X}\cdot\Delta\vec{N}) - \Delta H, \tag{5}$$

a numerical scheme can be constructed. We will approximate the normal velocity $V_N = \vec{N}\cdot\frac{\partial\vec{X}}{\partial t}$ by the finite difference $\vec{N}^n\cdot\frac{\vec{X}^{n+1}-\vec{X}^n}{\tau}$, where the index n means that

the corresponding vector belongs to the surface on the n-th time step, that is, to the moment $\tau \cdot n$.

On the right-hand side, all the values and operators will be substituted by their discrete analogues. Their approximation is presented in detail in the next section. The right-hand side of equation (5) will not fully correspond to the n-th step. To obtain in the numerical scheme a invertible operator with "good" properties, in the expression $-\frac{1}{2}\Delta^2(\vec{N}\cdot\vec{X})$ we will consider the value $\vec{X}$ on the $(n+1)$-the step. The rest of the summands will remain on the n-th one. Then the equation can be transformed into a numerical scheme:

$$\vec{N}^n \cdot \frac{\vec{X}^{n+1}-\vec{X}^n}{\tau} = -\frac{1}{2}\Delta^2(\vec{N}^n\cdot\vec{X}^{n+1}) + \frac{1}{2}\Delta(\vec{X}^n\cdot\Delta\vec{N}^n) - \Delta H^n$$
$$\vec{N}^n\cdot\vec{X}^{n+1} - \vec{N}^n\cdot\vec{X}^n = -\frac{\tau}{2}\Delta^2(\vec{N}^n\cdot\vec{X}^{n+1}) + \frac{\tau}{2}\Delta(\vec{X}^n\cdot\Delta\vec{N}^n) - \tau\Delta H^n$$
$$\left(I+\frac{\tau}{2}\Delta^2\right)(\vec{N}^n\cdot\vec{X}^{n+1}) = \vec{N}^n\cdot\vec{X}^n + \frac{\tau}{2}\Delta(\vec{X}^n\cdot\Delta\vec{N}^n) - \tau\Delta H^n$$

Thus, a semi-implicit numerical scheme is obtained:

$$\vec{N}^n\cdot\vec{X}^{n+1} = \left(I+\frac{\tau}{2}\Delta^2\right)^{-1}\left(\vec{N}^n\cdot\vec{X}^n + \frac{\tau}{2}\Delta(\vec{X}^n\cdot\Delta\vec{N}^n) - \tau\Delta H^n\right)$$

But if we recall the equality $\Delta^2(\vec{N}\cdot\vec{X}) = \Delta(\vec{X}\cdot\Delta\vec{N})$, following from the derivation of equation (5), we can further transform the scheme.

$$\begin{aligned}\vec{N}^n\cdot\vec{X}^{n+1} &= \left(I+\frac{\tau}{2}\Delta^2\right)^{-1}\left(\vec{N}^n\cdot\vec{X}^n + \frac{\tau}{2}\Delta(\vec{X}^n\cdot\Delta\vec{N}^n) - \tau\Delta H^n\right) = \\ &= \left(I+\frac{\tau}{2}\Delta^2\right)^{-1}\left(\vec{N}^n\cdot\vec{X}^n + \frac{\tau}{2}\Delta^2(\vec{X}^n\cdot\vec{N}^n) - \tau\Delta H^n\right) = \\ &= \left(I+\frac{\tau}{2}\Delta^2\right)^{-1}\left(\left(I+\frac{\tau}{2}\Delta^2\right)\vec{X}^n\cdot\vec{N}^n - \tau\Delta H^n\right) = \\ &= \vec{N}^n\cdot\vec{X}^n - \tau\left(I+\frac{\tau}{2}\Delta^2\right)^{-1}\Delta H^n.\end{aligned}$$

Finally, the scheme has the form

$$\vec{N}^n\cdot\vec{X}^{n+1} = \vec{N}^n\cdot\vec{X}^n - \tau\left(I+\frac{\tau}{2}\Delta^2\right)^{-1}\Delta H^n. \tag{6}$$

It is correctly defined, since the operator $\left(I+\frac{\tau}{2}\Delta^2\right)$ is invertible for every $\tau > 0$. The discrete analogue of this operator will be invertible given sufficiently regular triangulation. This issue is described in more detail in Section 2.2.3. The inverse operator is smoothing one, its application to the explicit scheme allows to increase its stability for all problems of interest and time steps. A similar scheme was proposed in [8], but this article considered a classical Laplace operator in $\mathbb{R}^3$, and the surface was defined by the level of a function approximated on the standard cubical lattice of a three-dimensional space. We will say more about the stability of the constructed scheme, but now we can already make the following

Remark 1. *It seems that we can substitute the construction of the scheme by adding the summand $\beta\Delta^2(\vec{N}\cdot\vec{X}) - \beta\Delta^2(\vec{N}\cdot\vec{X})$ and obtain a more general case with the parameter β instead of $\frac{1}{2}$. But as the numerical experiments presented in our paper and in paper [8] show, the value $\beta = \frac{1}{2}$ turns out to be important. It is a*

stability threshold of the numerical scheme: given $\beta < \frac{1}{2}$ *it is not stable, and given* $\beta \geq \frac{1}{2}$ *it is stable.*

After calculating $\vec{N}^n \cdot \vec{X}^{n+1}$, the surface on the step $n+1$ is obtained in the form:

$$\vec{X}^{n+1} = \vec{X}^n + \tau \vec{N}^n V_N = \vec{X}^n + \vec{N}^n \left(\vec{N}^n \cdot \vec{X}^{n+1} - \vec{N}^n \cdot \vec{X}^n \right).$$

Now, it only remains to calculate all the values and operators belonging to the scheme.

2.2. **Computation of the auxiliary values.** The scheme is implemented on the closed triangulated surface S^n in $\mathbb{R}^3$ with a given orientation. In this case, that means that every vertex $v \in S^n$ has a given set of vertices $\{v_i\}_{i=0}^{m-1}$, connected to it by the edges. Moreover, for this set the forward direction, consistent across all vertices of triangulation, is given. It is necessary so that the normals $\vec{N}_v$ constructed at every vertex v have the same direction with respect to the surface S^n.

For simplicity, we introduce some auxiliary notations. We denote by e_i the edge vv_i, the angle $\angle v_i v v_{(i+1)mod\ m}$ by α_i, the angles by the side $v_i v_{(i+1)mod\ m}$ in the triangle $v_i v v_{(i+1)mod\ m}$ by β_i and γ_i respectively. We will refer to the set of vertices $\{v_i\}_{i=0}^{m-1}$, connected by edges with the given vertex v as a *star* of the vertex v and denote it by $St(v)$.

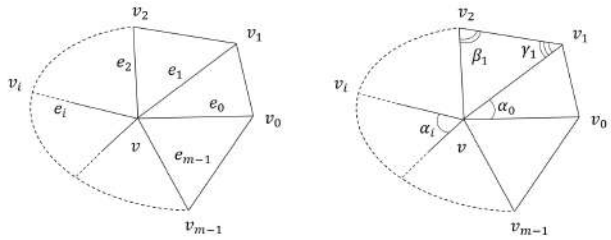


FIG. 1. The notations on the star of the vertex

To implement the numerical scheme, it is necessary to be able to compute the following values:

- The unit normal vector $\vec{N}$;
- The mean curvature H;
- The Laplace–Beltrami operator Δ.

All these values have a local character, hence, to compute them in some vertex of triangulation it suffices to use only adjacent vertices. It is also necessary to be able to construct a correct triangulation that will define the orientation of the surface. The following section will be dedicated to this issue.

2.2.1. *Computation of the normal vector.* There exists a significant amount of methods for calculating the normal vector in the vertex of a triangulated surface, but they all use the same idea — averaging of the normals to triangles of the star of the given vertex with some coefficients. The normals to the triangles of the star are calculated as vector products:

$$\vec{N}_i^v = \frac{[\vec{e}_i \times \vec{e}_{(i+1)mod\ m}]}{\left\| [\vec{e}_i \times \vec{e}_{(i+1)mod\ m}] \right\|}.$$

Then the unit normal vector to the surface at the point v is found in the form

$$\vec{N}'_v = \sum_{i=0}^{m-1} a_i^v \vec{N}_i^v; \quad \vec{N}_v = \frac{\vec{N}'_v}{\left\|\vec{N}'_v\right\|}, \tag{7}$$

with some coefficients a_i^v, depending on the vertex and the triangle considered. In our work, we chose the variant

$$a_i^v = \frac{\sin\alpha_i}{\left\|\vec{e}_i\right\|\left\|\vec{e}_{(i+1)mod\ m}\right\|},$$

since the result turns out to be more accurate compared to other methods in the cases when the exact normal vector is known (see [10]). Also with the given coefficients, the normal is calculated using a simple formula:

$$\vec{N}'_v = \sum_{i=0}^{m-1} \frac{[\vec{e}_i \times \vec{e}_{(i+1)mod\ m}]}{\left\|\vec{e}_i\right\|^2\left\|\vec{e}_{(i+1)mod\ m}\right\|^2}; \quad \vec{N}_v = \frac{\vec{N}'_v}{\left\|\vec{N}'_v\right\|}. \tag{8}$$

It is worth mentioning that in the majority of cases it turns out that averaging with equal coefficients — $a_i^v = \frac{1}{m}$ is sufficient, but for our case we should not neglect accuracy.

Thus, taking into account the given numbering of vertices in $St(v)$, the normals pointing in the same direction with respect to the surface are calculated at every point. Hence, the orientation of the surface is considered to be given.

2.2.2. *Computation of the mean curvature.* The sign of the mean curvature depends on the choice of surface orientation, therefore, for its computation we will use the normal vectors obtained above. To calculate the mean curvature, we have chosen the method proposed in work [11], which is based on the formula:

$$\int_0^{2\pi} \kappa_n(\varphi) d\varphi = 2\pi H. \tag{9}$$

Here $\kappa_n(\varphi)$ is a normal curvature in the vertex v, depending on the rotation angle φ of the normal plane around $\vec{N}_v$. This integral in the vertex v can be approximately found in the form of a sum by all vertices adjacent to it:

$$H = \frac{1}{2\pi} \sum_{i=0}^{m-1} \kappa_n^i \left(\frac{\alpha_{(i-1)mod\ m} + \alpha_i}{2} \right),$$

where $\kappa_n^i = \frac{2\vec{N}_v \cdot \vec{e}_i}{\left\|\vec{e}_i\right\|^2}$ is an approximate value of the normal curvature in the direction of the edge e_i, and $\frac{\alpha_{(i-1)mod\ m}+\alpha_i}{2} \approx d\varphi(v_i)$. The value of the normal curvature κ_n^i is calculated with the help of decomposition of the normal section into the Taylor series (the detailed derivation can be found in [11]).

Apart from the simplicity of calculation, this method has one more advantage, which is its accuracy. In paper [12], an analysis of different methods for calculation of mean and Gaussian curvatures was performed. For surfaces with a given mean curvature, this method turned out to be more accurate in average in comparison with other known methods. Also, to compute H, the unit normal vector $\vec{N}_v$ is explicitly used, which ensures that the obtained value corresponds to the chosen orientation.

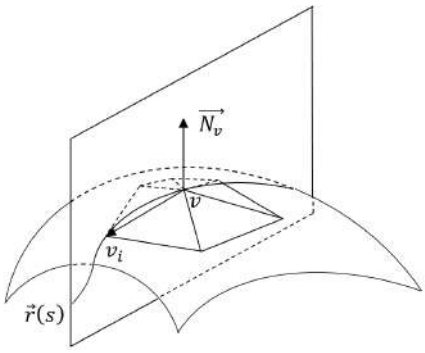


FIG. 2. Normal section of a triangulated surface

2.2.3. *Discrete Laplace–Beltrami operator.* For the Laplace–Beltrami operator, its approximation by a modified cotangent method was chosen:

$$\Delta f(v) = \frac{1}{2A} \sum_{i=0}^{m-1} \omega_i (f(v_i) - f(v)), \tag{10}$$

where

$$\omega_i = \begin{cases} ctg\beta_i + ctg\gamma_{(i-1)mod\ m}, & \text{for inner vertices;} \\ ctg\beta_i, & \text{for the vertices on the boundary.} \end{cases}$$

Here β_i and $\gamma_{(i-1)mod\ m}$ are the angles opposite to the edge e_i in the star of the vertex v. A is the area of the neighbourhood of the vertex v. By the neighbourhood of the vertex we mean the Voronoi region — the set of points of a triangulated area, which are located closer to this vertex than to the rest of the vertices of triangulation.

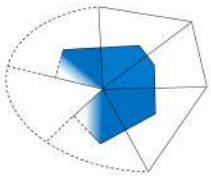


FIG. 3. Voronoi region of the vertex v

This method was proposed in [13], where it was analytically shown that such neighbourhood choice helps to achieve the best approximation of the operator. It is a more accurate version of a well-known cotangent method that was first proposed in [14].

Consider the issue of invertibility of the operator $\left(I + \frac{\tau}{2}\Delta^2\right)$. Let λ be an eigenvalue of the operator Δ^2, and f be an eigenfunction corresponding to λ. Then:

$$0 \leq \int_S (\Delta f)(\Delta f) ds = \int_S (\Delta^2 f) f ds = \int_S \lambda f^2 ds = \lambda \int_S f^2 ds.$$

Therefore, the eigenvalues of the operator Δ^2 are non-negative, and hence, the operator $\left(I + \frac{\tau}{2}\Delta^2\right)$ is invertible when $\tau > 0$. As for the discrete analogue by formula (10), its matrix would be symmetrical given equality of the areas of neighbourhoods of every vertex. For a symmetrical matrix, it is possible to prove a similar property using the scalar product in $\mathbb{R}^3$. In practice, the matrix will no be symmetrical, but given sufficiently regular triangulation, the areas will be close to each other, which allows to expect the discrete operator to be invertible. For our computations in Section 4, the operator has always been invertible.

In the form (10), the Laplace–Beltrami operator can be represented as a sparse $(N \times N)$-matrix where N is the amount of triangulation vertices. In the $i-$th row of this matrix, only $|St(v_i)|$ of N elements are nonzero. The function f can be represented as a vector of its values on the vertices. Then for the surface S^n, we obtain:

$$\Delta_{S^n} \longmapsto \Delta_n \in M_N(\mathbb{R}); \qquad f \longmapsto f^n = \begin{pmatrix} f_1 \\ \vdots \\ f_N \end{pmatrix} \in \mathbb{R}^N.$$

Therefore, the vector $\vec{N}^n \cdot \vec{X}^{n+1}$ can be found from (6) as a solution of the system of linear equations. The solution was found by the method of QR-decomposition for sparse matrices.

2.2.4. *Computation of the area, volume, and mean curvature value.* To control the fulfillment of properties (2) of the flow, it is necessary to compute the area and volume of the triangulated surface. The surface area was calculated in a trivial way — as the sum of areas of all triangles of triangulation.

Computation of the volume can be reduced to computation of the integral over the surface. To do that, we used the Stokes formula:

$$\int_\Omega d\omega = \int_{\partial\Omega} \omega$$

We put $\omega = xdy \wedge dz$, then

$$V(S^n) = \int_{\Omega^n} dx \wedge dy \wedge dz = \int_{S^n} xdy \wedge dz = \sum_{j=1}^{M} \int_{\Delta_j} xdy \wedge dz.$$

Here Δ_j are the triangles of triangulation. For an arbitrary triangle, the integral of the form ω is calculated explicitly and depends on the coordinates of its vertices. Therefore, the whole integral can be computed in one cycle by the triangles of triangulation.

In order to detect the emergence of surfaces of constant mean curvature — the stationary surfaces of the flow, it is necessary to watch for the behaviour of deviation of the curvature from its average value. The average value $\tilde{H}$ was calculated in the following way:

$$\tilde{H} = \frac{1}{A(S^n)} \int_{S^n} H ds \approx \frac{1}{A(S^n)} \sum_{i=1}^{N} H_i A(U_i)$$

The deviation of the curvature from its average value was calculated in the L_1-norm:

$$H_{dev} = \frac{1}{A(S^n)} \int_{S^n} |H - \tilde{H}| ds \approx \frac{1}{A(S^n)} \sum_{i=1}^{N} |H_i - \tilde{H}| A(U_i).$$

For clarity, on the graphs we displayed not the H_{dev} itself, but the percentage that it amounts from $|\tilde{H}|$. In both integrals, $A(U_i)$ is the area of Voronoi neighbourhood of the vertex v_i.

3. Program implementation

The program for implementation of the described scheme was written in C++ language with the use of a number of libraries. For the work with vectors and matrices, and to solve systems of linear equations with sparse matrices of a large size, the Eigen library was used.

To construct triangulations and reconstruct a surface, we used the library of computational geometry CGAL. The package 3D Surface Mesh Generation provides a possibility to construct triangulation for an arbitrary surface given by some function. Moreover, for smooth functions the algorithm ensures homeomorphism of the triangulated surface and the initial one, and Hausdorff distance constraint.

For reconstruction of the surface, the Polygon Mesh Processing (PMP) and Scale-space Surface Reconstruction (SSSR) packages were employed. The first one constitutes a set of functions for processing of polygonal surfaces, including the triangulated ones. The second one serves for restoration of surface triangulation by a set of points in space. It turned out to be more convenient compared to the other ones, since it allows to restore triangulation for the surfaces with boundary. The detailed description will be provided below.

The computations of normals during the reconstruction of triangulation (but not during the step of the scheme) and their subsequent orientation were made using functions from the Point Set Processing (PSP) package. Note that the normals obtained by formula (8) turn out to be more accurate than the normals calculated by the functions of the PSP package. The reason is that in the first case it is known exactly which vertices are adjacent to the given one.

3.1. Construction of triangulation.

We input in the algorithm a smooth function that defines a closed smooth surface and the three parameters: a_b, r_b, d_b. Here a_b is the lowest boundary of angle in triangulation triangles, r_b is the upper boundary of radii of Delaunay balls, and d_b is the upper boundary of distance between the centre of the circumscribed circle of a triangle and the centre of its Delaunay ball. A Delaunay ball is a ball that passes through the vertices of a triangulation triangle, whose centre belongs to the surface. The algorithm is guaranteed to terminate if $a_b \leqslant 30^\circ$, but for the work with a discrete Laplace–Beltrami operator obtuse triangles are undesirable, hence, a_b is taken equal to 30°. The rest of the parameters are selected in a way that the triangulation is sufficiently small and the theoretical guarantees from [15] are fulfilled.

As a result of work of the algorithm by the given surface S, a triangulation close in its properties to a regular one is constructed. An example is shown on Figure 4. This example demonstrates that the obtained triangles are close to each other in size, and the triangulation is regular on some domains.

However, as the algorithm runs, the triangles change their size and position in space: in some places they stretched, in others they shrank. Therefore, for correct approximation it is necessary to reconstruct the triangulation.

Separately note that the constructed surface is oriented — in every triangle, the direction of passing by the vertices is defined, moreover, the orientations of adjacent

triangles are consistent. Hence, knowing the orientation of every triangle it is easy to restore the direction of passing by in the star of the vertex.

FIG. 4. Examples of triangulation

3.2. **Reconstruction of triangulation.** Reconstruction of triangulation is necessary not only to make it close to a regular one, but also to process the changes in the topology of the surface. As it was theoretically shown in [5], the surface diffusion flow can lead to self-intersection or break of the surface. The algorithm presented below allows to handle both of these kinds of singularities.

3.2.1. *The algorithm.* The main idea of the algorithm contains three steps: correction of triangulation, finding self-intersections of the surface and removing the corresponding triangles, restoration of the triangulation by points. Since both breaks into two components and self-intersections are reduced to processing of intersections of triangles, this algorithm allows to process both of these kinds of singularities.

The algorithm itself consists of the following items:

(1) The input is a triangulated surface after passing through several number of steps of the scheme. Most often the work of the scheme leads to appearance of triangles that are too narrow or too large. Hence, first of all, the function *isotropic_remeshing* from the PMP package is applied, that makes the triangulation close to regular by reconstructing some edges and vertices with maximal preservation of geometry. This function is a program realization of the algorithm proposed in [17]. It takes only two arguments: the number of iterations and the length of the edge, to which the algorithm will approximate the lengths of the edges of triangulation.

(2) Further, with the help of function *self_intersections* from the PMP package, all pairs of intersecting triangles of triangulation are detected and removed. After the removal of triangles, it is necessary to remove the points that correspond to self-intersections.

(3) To do that, in every vertex, a normal by k nearest vertices is calculated with the help of function *jet_estimate_normals* from the PSP package. Further, by function *mst_orient_normals*, the normals by all points are oriented in the way that the direction of the normals in the point is consistent with the directions of normals of its neighbours. This function uses the method proposed in work [16]. Vertices for which it was not successful are removed. These points exactly correspond to the places of self-intersection of surface.

(4) The next stage is the construction of triangulation by a given set of points. To do this, function *reconstruct_surface* from the SSSR package was used. Note that despite the absence of guarantees for preservation of the topology (if the points belong to the surface S, then the surface constructed by them may appear to be not homeomorphic to S), for sufficiently smooth

surfaces and regular triangulation the initial surface is restored accurately. We should mention that due to step 1, the points lie almost regularly, hence, the obtained triangulation only locally differs from the initial one.

(5') (Optional) Unless the surface is sufficiently smooth and the formation of holes differing from the initial boundary is possible, one more step can be performed. The PMP package provides a possibility to triangulate holes in the surface, satisfying a particular condition. For example, the length of a boundary cycle should not exceed a given number (the length of the initial boundary). To do this, function *refine_and_fair_hole* is used.

Remark 2. *The algorithm that seals the holes in the surface only works provided that the boundary cycle is found. If after removal of intersecting triangles, the boundary has a more complex shape (for example, a figure eight on Figure 5), then such boundary will not be sealed. Therefore, stage* $5'$ *cannot be used instead of* 4.

FIG. 5. An example of a boundary cycle after removal of triangles

This algorithm starts after passing through a particular number of steps of the numerical scheme. In order to adapt to the changes of topology, the moments for its application are chosen manually. But it can also be applied after some fixed but sufficiently small number of steps. Its only limitation is that in the break neighbourhood the calculations have to be performed with a small interval to process the rupture in time. For the calculations in Chapter 4, the algorithm was applied after each step of the scheme.

3.2.2. *Examples.* An example of a break is the evolution of the left surface from Figure 4. It is obtained by rotation of two semicircles of radius 6, connected by a parabolic isthmus that has a width 1 in the most narrow place. After some amount of steps, the isthmus becomes so thin that its triangles intersect. Then they are removed, and the surface gets triangulated again. As a result, it breaks into two connected components, each of which evolves separately from the other one. The rupture process is presented on Figure 6.

The situation is similar when the surface self-intersects. Here, an example is the evolution of a two-dimensional torus with radii 6 and 3, presented in Section 4.1. The length of the larger radius of the torus gradually decreases while the one of the smaller radius increases, until a self-intersection happens. After the intersecting triangles are removed, the hole in the torus is sealed. Further evolution transforms it into a sphere.

4. NUMERICAL EXPERIMENTS

4.1. **A two-dimensional torus.** The first example of work of the algorithm is modeling of evolution of a two-dimensional torus with radii 3 and 6. Its initial

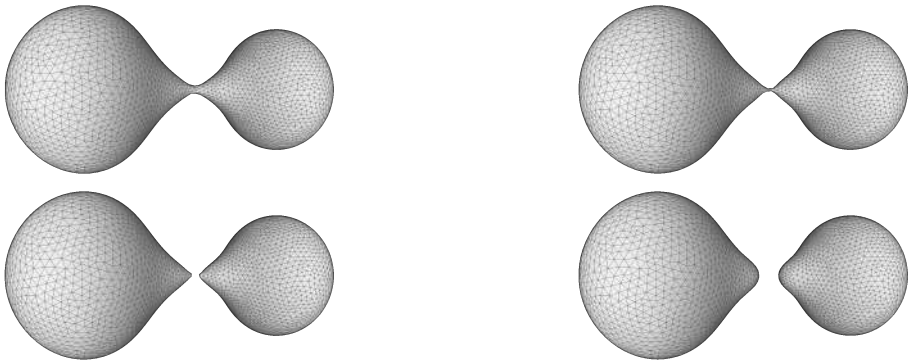


FIG. 6. An example of surface break

triangulation consists of 15250 vertices and 30500 triangles. A step of the scheme has changed over time from 0.01 to 0.1, its change is described in detail in Table 1.

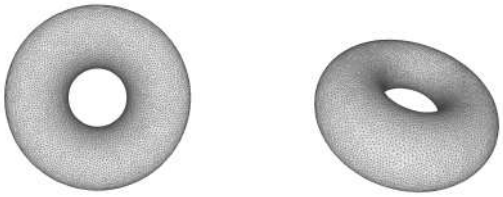


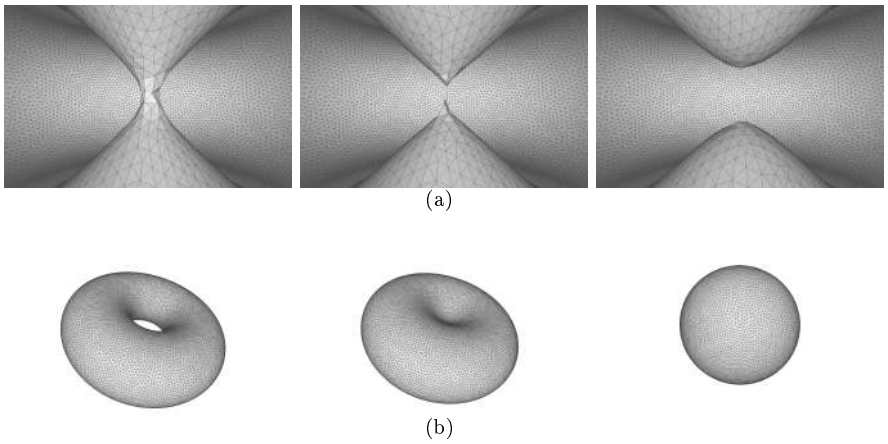
FIG. 7. A two-dimensional torus

A torus is a closed surface, hence, its evolution should result into a closed surface of constant mean curvature surface — a two-dimensional sphere. Indeed, over time the hole in the torus starts to decrease, until at the point of time 83.2 it gets sealed. At this moment, a self-intersection of the surface happens, which is then processed by the algorithm. But in order for this processing to be correct it is necessary to perform calculations with a sufficiently small step. After that, the obtained surface evolves into a sphere. The process of sealing the hole and the whole evolution of the torus are presented on Figure 8.

Step number	$0-50$	$51-100$	$101-150$	
Step τ	0.01	0.02	0.05	
Step number	$151-1000$	$1001-1100$	$1101-1200$	$1201-1600$
Step τ	0.1	0.2	0.5	1

TABLE 1. The value of the step of the scheme depending of its number

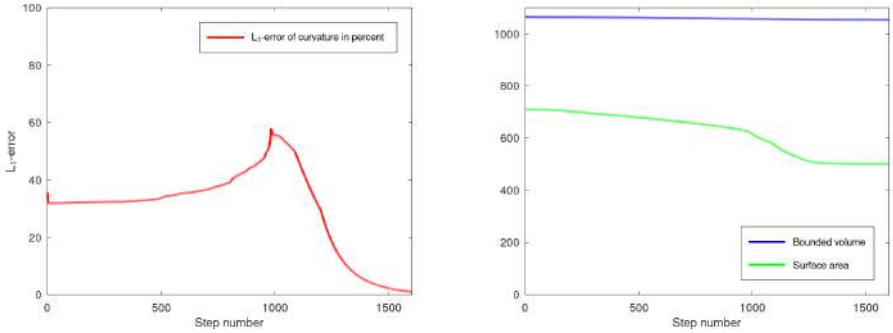
As it has been theoretically shown, the surface area decreases monotonously. By the end of calculations, it constitutes 70% from its initial size. As for the volume, it does not change, but up to computational error — its change is only 1%. Also on every step, the average value of curvature and the deviation of the curvature from that value was calculated. Both graphs are presented on Figure 9. The peak on the deviation graph corresponds to the moment of singularity emergence, further, the deviation monotonously tends to zero. On the last step, it constitutes 1%.



(a)

(b)

FIG. 8. Hole sealing (a) and torus evolution (b) at points of time $t = 49$, $t = 88$ и $t = 559$



(a) (b)

FIG. 9. A graph showing the deviation of the curvature from its average value as a percentage (a) and the change of the surface area and the volume bounded by the surface (b)

4.2. **Sphere sintering.** Here, 8 spheres touching each other is the initial surface. All the spheres have radii 3 and are located in the points of the form $(\pm 3, \pm 3, \pm 3)$. The touch points are approximated by cylindrical regions of radius $\sqrt{59}/10$. Such number was selected for convenience in searching for the points of intersection with the spheres. The initial triangulation is constructed by 51311 points, it contains 102638 triangles. During the starting stages, due to insufficient smoothness the algorithm was applied without the re-triangulation stage, only correction of triangulation was performed. The calculations were initially conducted with a step $\tau = 0.001$, which was gradually increased to 0.1. The detailed process of change of the step of the scheme is presented in the table below. The total number of steps during the calculation was 840. The evolution process before emergence of singularities is presented on Figure 11. At this time, surface irregularities gradually get smoothed

FIG. 10. The surface at the point of time $t = 0$

Step number	$0-30$	$31-60$	$61-100$	
Step τ	0.001	0.002	0.005	
Step number	$101-150$	$151-200$	$201-300$	$301-840$
Step τ	0.01	0.02	0.05	0.1

TABLE 2. The value of the step of the scheme depending of its number

and the holes narrow down. After that, all 6 holes start closing until the initial

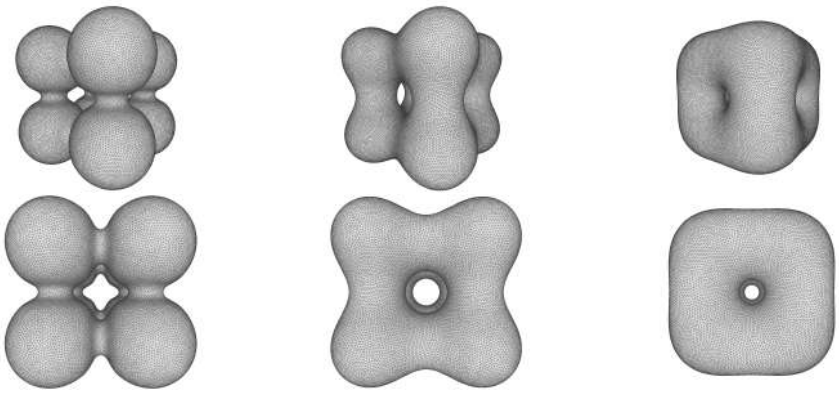


FIG. 11. The surface at points of time $t = 0.025$, $t = 1.79$ and $t = 16.79$

surface breaks into two connected components: the outer surface that resembles a cube, and the inner one with its shape close to a sphere. All breaks happen similarly to the ones presented in Section 3.2.2. Since the initial surface is sufficiently large, the step $\tau = 0.1$ was suitable for correct processing of the singularities. In what follows, the algorithm is applied without step $5'$. Moreover, due to the smoothness of the surface and regular location of the points, no extra holes appear, the surface remains being closed. When the last hole is sealed, the surface breaks into two parts. At this moment, the triangulation consists of 37310 points and 75012 triangles. The shape of the outer surface resembles a cube. After the rupture, both of the obtained surfaces independently evolve into spheres. The curvature of a two-dimensional sphere is constant, hence, a sphere is a stationary point in the given flow. At the end of calculations, the outer sphere consists of 28334 vertices and 56664 triangles, and the inner one has 5185 vertices and 10366 triangles. During each step, the surface area and the volume bounded by it were calculated. The corresponding graph is

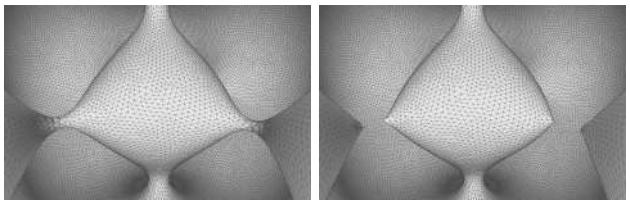


FIG. 12. Sealing of holes in the surface

FIG. 13. The form of the outer surface

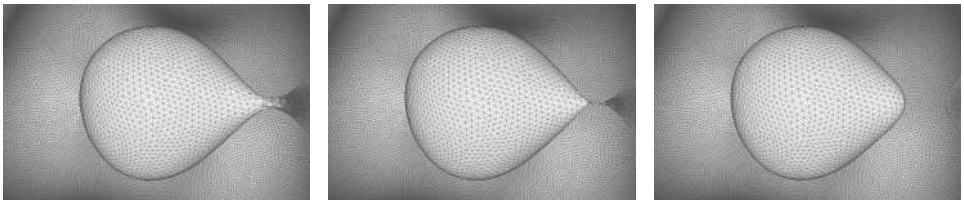


FIG. 14. Surface rupture, inside view

provided on Figure 16. The area monotonously decreases as it has been predicted theoretically. Due to computational error, the volume cannot be preserved, but its change over the whole calculation time constitutes only 0.6% from the initial one.

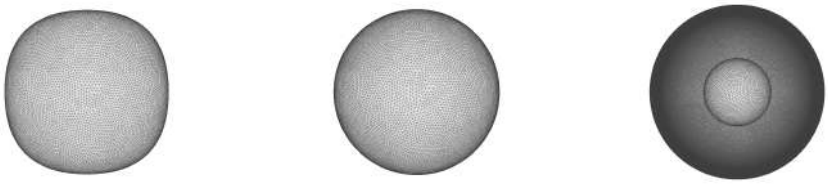
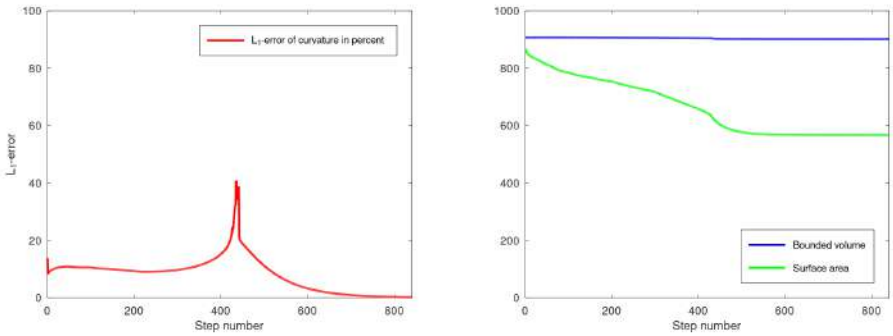


FIG. 15. The surface at the point of time 31.8 and the surface obtained as a result

Figure 16 also presents a graph of the curvature deviation from its average value as a percentage. One peak can be observed on the graph, which corresponds to the moments of breaking of the surface into two connected components. After this rupture, the curvature gradually approaches the constant one.

4.3. **Three spheres and a plane.** The third example of calculations is the evolution of three spheres of radius 3, touching each other, and a plane $z = -3$. This surface is presented on Figure 17. Touch points of the spheres and the plane are approximated



(a) (b)

FIG. 16. A graph showing the deviation of the curvature from its average value as a percentage (a) and the change of the surface area and the volume bounded by the surface (b)

by cylinders of radius 1, and the points there the spheres touch each other by cylinders of radius $\frac{1}{2}$. The plane which we consider is, of course, bounded, it is a circular domain of radius 24. It has to be sufficiently large so that the surface disturbances during the evolution would not reach the boundary of the plane.

To calculate the values on the boundary, some modifications were performed:

- If the boundary vertex only has two vertices in the star, then the normal is calculated as a vector product of the corresponding edges. If it has more than two vertices, then the normal was calculated by formula (7) with the coefficients $a_i^v = \frac{1}{m}$;
- Mean curvature at boundary vertices is assumed to be zero;
- For the Laplace–Beltrami operator, formula (10) was used, since it takes boundary vertices into account.

The initial triangulation consisted of 95657 vertices and 190322 triangles. First, step τ due to a large curvature at touch points was equal 0.001, later it gradually increased to 0.2. After all the emerging singularities were processed, the step of the scheme increased up to $\tau = 2$. Such great value results from the fact that the surface was close to a plane — the mean curvature was almost constant and close to zero. But the moments of singularity processing were passed by with small steps. Detailed change of the step is presented in Table 3. During the first ten steps, the triangulation algorithm was applied only in the form of stage 1.

First, the hole between the spheres at the point of time $t = 0.84$ is sealed. This process is presented on Figure 18. Further, the holes between the spheres and the plane expand until the holes between the pairs of spheres and the plane are sealed. Seen from below, this process looks as an expansion of circles, by which the spheres and the plane were intersecting. At the point of time $t = 9.1$, the holes between the

[1]At this moment the intersection of two surfaces happens, hence, the region of high curvature emerges.

[2]The closer the surface is to a plane, the closer the mean curvature is to zero. Hence, the step of the scheme was getting larger and larger.

Step number	$1-30$	$31-60$	$61-100$	$101-200$	
Step τ	0.001	0.002	0.005	0.01	
Step number	$201-300$	$301-400$	$401-500$	$501-601$	
Step τ	0.02	0.05	0.1	0.2	
Step number	$601-850$	$851-935$[1]	$936-950$	$951-980$	$1001-1030$
Step τ	0.5	1	0.0005	0.001	0.005
Step number	$1031-1060$	$1061-1100$	$1101-1140$	$1141-1170$	$981-1000$
Step τ	0.01	0.02	0.05	0.1	0.002
Step number	$1171-1200$	$1201-1250$[2]	$1251-1350$	$1351-1450$	$1451-1600$
Step τ	0.2	0.5	1	1.5	2

TABLE 3. The value of the step of the scheme depending on its number

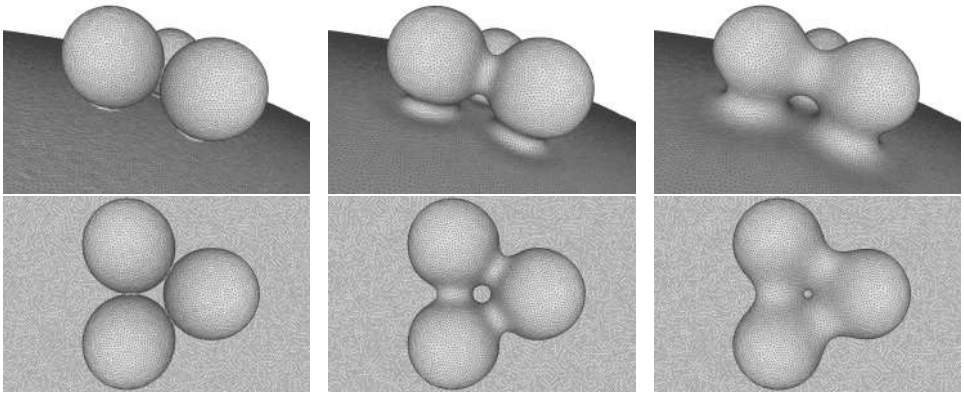


FIG. 17. The surface at points of time $t = 0$, $t = 0.07$ and $t = 0.8$.

spheres and the plane start sealing. The isthmus breaks on Figure 20 correspond to those.

After these holes are sealed, the surface breaks into two parts: a closed surface homeomorphic to the sphere, and a surface with a boundary homeomorphic to a region of the plane. The mean curvature of the plane equals zero, hence, it is a stationary point of the flow of surface diffusion. It is to the plane that the large surface tends. But the general configuration here is such that these two surfaces necessarily intersect when evolution happens. An intersection of a ball with a larger

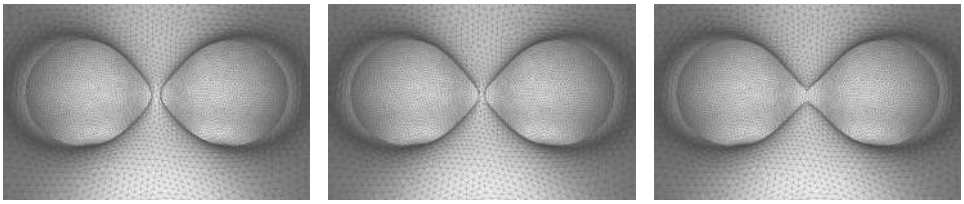


FIG. 18. Sealing of the first hole when $t = 0.84$

surface (see Appendix B) happens at the point of time $t = 187.3$. At this moment, the step of the scheme decreases again to 0.0005, since the touch point has a high

curvature. When the surface smooths, the step of the scheme gradually increases to 0.1. Further, due to a significant slowdown in the evolution speed, the step of the scheme was increasing more often and on the last steps its value achieved 2.

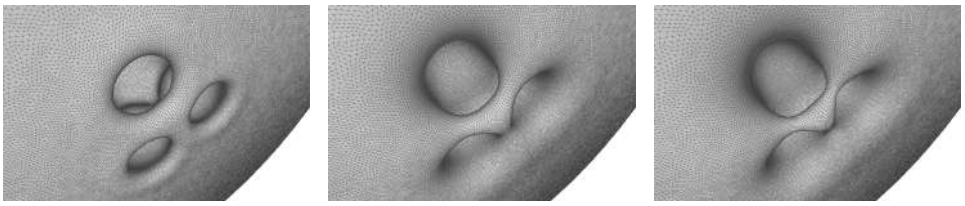


FIG. 19. The view on the surface from below at the moments $t = 0.29$, $t = 5.79$, and $t = 8.29$

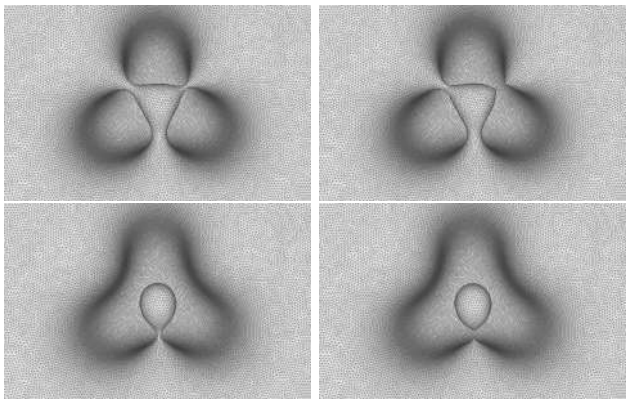


FIG. 20. Surface break for $t \in [9.1, 9.5]$

As a result, at the point of time $t = 774.627$, the process stops, since the surface turns into a plane, which is a stationary point of the flow. The intermediate stages of the surface evolution are presented on Figures 19, 20, and 21. For clarity, the rupture moments are presented separately.

After every step of the scheme, the surface area and the deviation of the curvature from the average value were calculated. In this case, not as a percentage, since the average value of the curvature tends to zero. The corresponding graphs are presented on Figure 22. Although the decrease of the area is guaranteed only for a closed surface, for the considered surface it also takes place, moreover, it happens monotonously. The graph showing deviations of the curvature from the average value, as it was seen before, has peaks, which represent the moments of division of the surface and gluing of two components.

4.4. **Accuracy of the obtained solutions.** At the moment, we have not found exact nontrivial solutions of the equation of surface diffusion. Only the constant mean curvature surfaces are known. Since the theoretical estimates of accuracy for the discrete analogue of the Laplace–Beltrami operator, mean curvature, and the normal vector could not be found, there is no theoretical accuracy estimate for any of the existing numerical schemes for solving the equation (1).

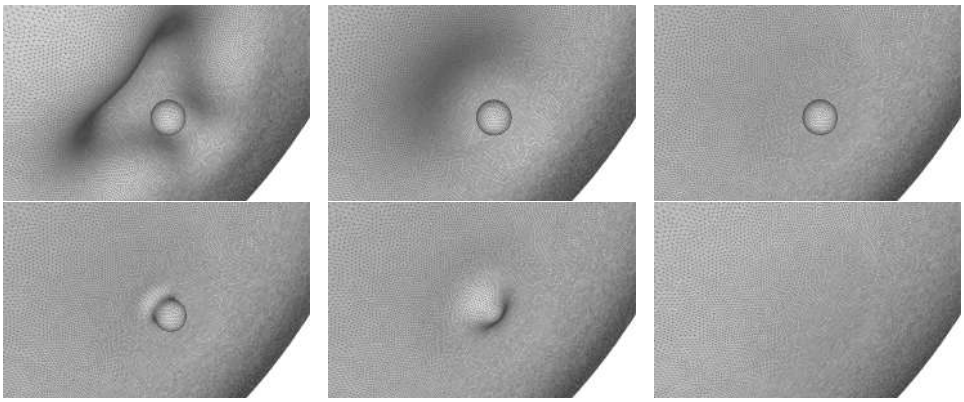
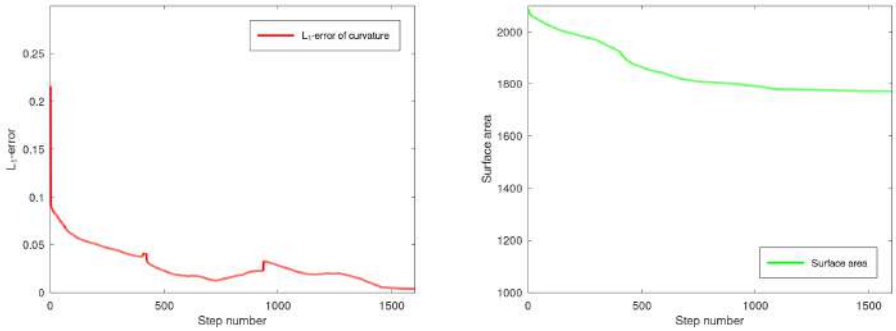


FIG. 21. The surface evolution from the moment $t = 13.3$ until the end of calculations



(a) (b)

FIG. 22. A graph showing the deviation of the curvature from its average value as a percentage (a) and change of the surface area (b)

Therefore, accuracy of the obtained approximated solutions can be estimated only by indirect evidence: the behaviour of area, volume, and mean curvature. We can also make a comparison with a trivial solution. A unit sphere was taken as the initial surface $S(0)$. Therefore, it will be the solution at every moment of time. But due to the error in the initial triangulation, not all of the points belong to the surface of the sphere, but some of them are located inside or outside of it. To estimate this error, after each step of the scheme, two values were calculated.

The first value $D_1 = \frac{1}{A(S^n)} \sum_{i=1}^{N} ||\vec{v}_i| - 1|A(U_i)$ — the average deviation of the length of the radius vector of the vertex from unit— characterizes the closeness of the surface to a unit sphere. The second one — $D_2 = \frac{1}{A(S^n)} \sum_{i=1}^{N} |||\vec{v}_i| - 1| - D_1|A(U_i)$ — characterizes the difference between the vertices in deviations of the lengths of their radii vectors from unit. The equality $D_2 = 0$ means that all deviations are similar and equal D_1, that is, the surface is a sphere with a radius other than unit.

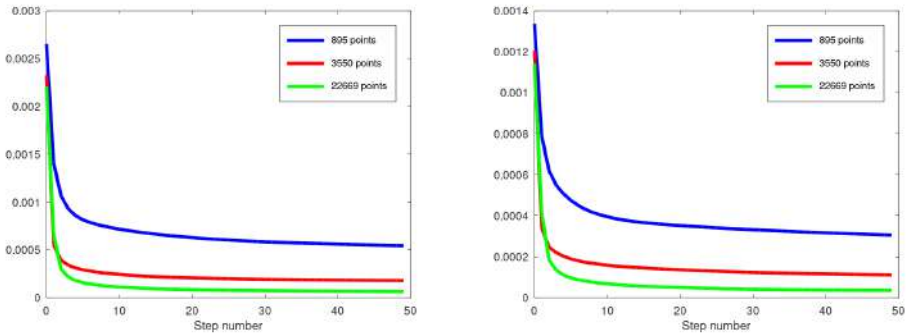


FIG. 23. The change D_1 (on the left) and D_2 (on the right) during the evolution given a different number of vertices of initial triangulation.

Three cases were considered: 895, 3550, and 22669 vertices of triangulation. For them, calculations were conducted with a step 0.00005, with the total number of 50 steps. The results are presented on Figure 23. As it can be seen on the graph, over time and as the triangulation gets smaller, both values tend to zero. That means that the surface tends to a sphere, and the radius of this sphere tends to unit.

4.5. **Computational stability.** In Section 2.1, a remark has been made about the choice of the parameter β for construction of a numerical scheme. In this section, the reasoning on the choice of the value $\beta = \frac{1}{2}$ is provided. As we have already mentioned, a similar construction was encountered in [8] for a square of Laplace operator, and a similar value was chosen on the basis of experiments and some reasoning, although that was not strict. No experiments with various parameters were provided there.

Numerical experiments with the scheme (6) constructed here show a similar result: for $\beta < \frac{1}{2}$, the scheme is not stable, and for $\beta \geq \frac{1}{2}$ it is computationally stable. The experiments, on whose basis the conclusion was made, are provided on Figures 24 and 25. The initial surface is an ellipsoid with semi-axis 5, 4, and 3. The calculations were performed with a step $\tau = 0.5$, and the initial number of points in the four cases was different, all around 3500. It can be seen that the explicit scheme corresponding to $\beta = 0$, for the considered step is absolutely unstable — the surface "exploads" in the majority of its points after three steps of the explicit scheme (Figure 24). The first two shots were made on the same scale, and the third one shows the obtained surface entirely.

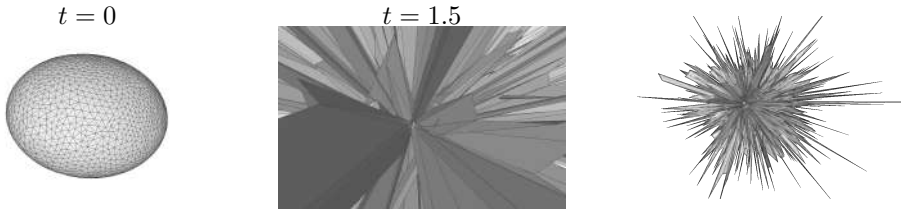


FIG. 24. Computations with using of the explicit scheme

The scheme with a parameter $\beta = \frac{1}{4}$ computes steadily for some time, but at some point waves appear on the surface, which do not stop even as the surface approximates the sphere. We can conclude that this scheme is not stable. Moreover, its instability is not caused by triangulation, since it is reconstructed on every step.

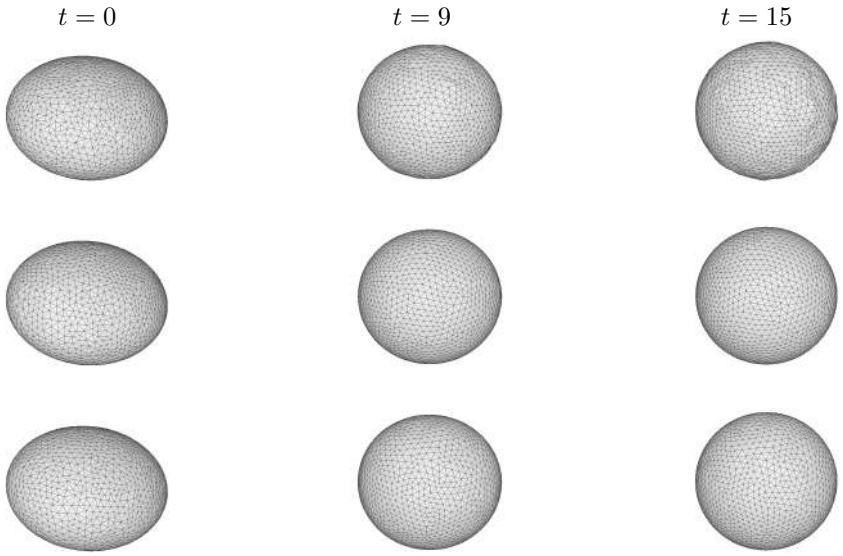


FIG. 25. Computational stability for different parameters; The first line — $\beta = \frac{1}{4}$, the second one — $\beta = \frac{1}{2}$, the third one — $\beta = \frac{3}{4}$.

As for the scheme with the parameter $\beta = \frac{1}{2}$, it turns out to be computationally stable on each of the considered steps and triangulations. For $\beta > \frac{1}{2}$, the scheme also shows stability, but does not differ in its properties from $\beta = \frac{1}{2}$. No significant differences were found by the speed of convergence to a sphere and the change of area and volume.

APPENDIX A. DETECTION OF CONNECTED COMPONENTS

As it was seen from the numerical experiments, during the evolution the surface can break into several connected components. But when finding the next position of $\vec{X}_i^{n+1}$ of some vertex v_i, scheme (6) uses all the other vertices of the surface. Hence, before computing the matrices, it is necessary to divide the surface into connected components, so that they evolve independently from each other. Also when the surface is re-triangulated, at the break points some points that lie separately can emerge, which are not connected to other edges. It is necessary to be able to find and remove them. We will consider this separately.

To detect the connected components, the PMP package provides function *split_connected_components*, which breaks a given (disjoint) triangulated surface T into an array of its connected components $T[i]$. This function starts after the re-triangulation stage in the algorithm from Section 3.2.1. If there are several connected components, then the algorithm outputs an array of surfaces, and further

the calculations are performed on each of the components separately from the other ones.

In the case of emergence of isolated points, there is also a possibility to remove them. To do that, function *keep_large_connected_components* was applied, that removes connected components whose size is smaller than the given one. Here, by size we mean the number of triangles in the connected component. This parameter is selected depending on the number of triangles of the initial triangulation. In the case when the exact number of connected components is known, function *keep_largest_connected_components* can be used, which only preserves a given number of the largest connected components.

Appendix B. Processing of intersection of two surfaces

All ruptures and self-intersections of the surfaces considered above happened smoothly, hence, for their processing the algorithm 3.2.1 was sufficient. The situation when two separate surfaces intersect is different. That is much harder to process, since the triangulation was not prepared to that in advance. If we just remove the intersecting triangles, there is a chance that we do not obtain a required surface as a result of re-triangulation. Therefore, to process the intersection in Section 4.3, a different approach was used.

At the moment when two surfaces intersect (that can be tracked by combining surfaces pairwise into one and calling function *does_self_intersect*), it is necessary to change the triangulations of both surfaces such that the intersection line was a subset of the edges of both triangulations. That can be achieved using function *corefine_and_compute_union*, which finds a closed surface generated by an intersection of the given surfaces.

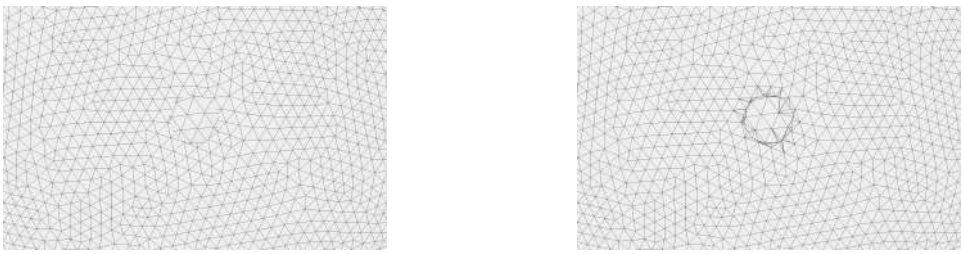


Fig. 26. Intersection of surfaces and further subdivision of the intersection.

Further, all three surfaces are combined into one and, as it was done before, all intersecting triangles are removed. The obtained surface has a hole in the area of intersection, since exactly the surface by which they intersect is removed. Geometrically everything is finished, but data issues remain — vertices and edges on the line of intersection are duplicated for both surfaces. To remove the duplicating vertices and edges and for orientation of the obtained surface (if the initial ones were oriented in a different way), function *repair_polygon_soup* was used. Without this step, both surfaces will not be glued by the intersection line. After that, the computational process starts again.

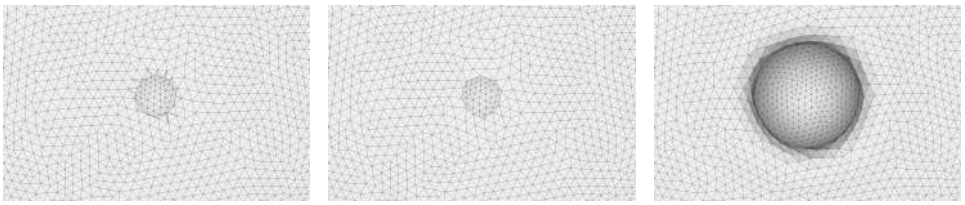


FIG. 27. Removal of intersection, fixing of the triangulation, and evolution of the surface

REFERENCES

[1] W. Mullins, *Theory of Thermal Grooving*, J. Appl. Physics, **28**:3 (1957), 333–339.

[2] Ya.E. Gegusin. *Fizika spekaniya*, Nauka, Moscow, 1984.

[3] Ya.V. Bazaikin, V.S. Derevschikov, E.G. Malkovich, A.I. Lysikov, A.G. Okunev, *Evolution of sorptive and textural properties of CaO-based sorbents during repetitive sorption/regeneration cycles: Part II. Modeling of sorbent sintering during initial cycles*, Chemical Engineering Science. **199** (2019), 156–163.

[4] J. Escher, U.F. Mayer, G. Simonett, *The surface diffusion flow for immersed hypersurfaces*, SIAM J. Math. Anal., **29**:6 (1998), 1419–1433. Zbl 0912.35161

[5] U.F. Mayer, G. Simonett, *Self-intersections for the surface diffusion and the volume-preserving mean curvature flow*, Differ. Integral Equ., **13**:7-9 (2000), 1189–1199. Zbl 1013.53045

[6] U.F. Mayer, *Numerical solutions for the surface diffusion flow in three space dimensions*, Comput. Appl. Math., **20**:3 (2001), 361–379. Zbl 1125.35422

[7] S. Kobayashi, K. Nomizu, *Foundations of differential geometry, Vol. II*, John Wiley and Sons, New York etc., 1969. Zbl 0175.48504

[8] P. Smereka, *Semi-implicit level set methods for curvature and surface diffusion motion*, J. Sci. Comput., **19**:1-3 (2003), 439–456. Zbl 1035.65098

[9] S. Kobayashi, K. Nomizu, *Foundations of differential geometry. I*, Wiley & Sons, New York-London, 1963. Zbl 0119.37502

[10] M. Cenanovic, P. Hansbo, M.G. Larson, *Finite element procedures for computing normals and mean curvature on triangulated surfaces and their use for mesh refinement*, arXiv:1703.05745v1 [math.NA], (2017).

[11] K. Watanabe, A.G. Belyaev, *Detection of salient curvature features on polygonal surfaces*, EUROGRAPHICS, **20**:3 (2001), 385–392.

[12] E. Magid, O. Soldea, E. Rivlin. *A comparison of Gaussian and mean curvature estimation methods on triangular meshes of range image data*, Computer Vision and Image Understanding, **107**:3 (2007), 139–159.

[13] M. Meyer, M. Desbrun, P. Schroder, A.H. Barr, *Discrete differential-geometry operators for triangulated 2-manifolds*, in Hege, Hans-Christian (ed.) et al., *Visualization and Mathematics III*, Springer, Berlin, 2003, 35–57. Zbl 1069.53004

[14] U. Pinkall, K. Polthier, *Computing discrete minimal surfaces and their conjugates*, Exp. Math., **2**:1 (1993), 15–36. Zbl 0799.53008

[15] J.-D. Boissonnat, S. Oudot, *Provably good sampling and meshing of surfaces*, Graph. Models, **67**:5 (2005), 405–451. Zbl 1087.68114

[16] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, W. Stuetzle, *Surface reconstruction from unorganized points*, Computer Graphics, **26**:2 (1992), 71–77.

[17] M. Botsch, L. Kobbelt, *A remeshing approach to multiresolution modeling*, In *Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry Processing*, (2004), 189–196.

YURY DANIILOVICH EFREMENKO
NOVOSIBIRSK STATE UNIVERSITY,
1, PIROGOVA STR.,
NOVOSIBIRSK, 630090, RUSSIA
Email address: i.efremenko@g.nsu.ru