Towards optimal advection using stretch-maximizing stream surfaces

Michael Bartoň[*,a], Jiří Kosinka[b]

[a]BCAM – Basque Center for Applied Mathematics, Bilbao, Basque Country, Spain
[b]Johann Bernoulli Institute, University of Groningen, The Netherlands

Abstract

We investigate a class of stream surfaces that expand in time as much as possible. Given a vector field, we look for seed curves that locally propagate in time in a stretch-maximizing manner, i.e., curves that infinitesimally expand most progressively. We show that such a curve is generically unique at every point in an incompressible flow and offers a very good initial guess for a stretch-maximizing stream surface. With the application of efficient fluid advection-diffusion in mind, we optimize fluid injection towards optimal advection and show several examples on benchmark datasets.

Key words: Stream surface, seed curve, advection, flow visualization.

1. Introduction

We investigate a special class of stream surfaces generated by seed curves that maximize a certain arc-length energy. We aim at an efficient advection process where one typically requires to minimize the cost of injection while speeding up (or maximizing) the advection effect. To achieve that, we focus on globally optimizing the geometric advection of the injected seed curve since this is closely related to seed curves that expand in time with the maximum stretching rate. The corresponding stream surface exhibits the maximum possible advection of the injected fluid [19].

Recent research in this area has been devoted to oil extraction from porous media by gas diffusion in both fractured and unfractured reservoirs, see e.g. [17] and the references cited therein. The efficiency in terms of the time required as well as the amount of solvent injected into a porous medium was studied in [30]. Another possible application of our research points to efficient crop spraying [20], where the trajectory of an aircraft is optimized to maximize the area covered by the sprayed pesticide.

With these applications in mind, we seek injection (or seed) curves [24] that propagate in time in the most progressive manner, by stretching their lengths as much as possible. For a steady state vector field, our goal is to find a seed curve such that the corresponding stream surface facilitates efficient advection.

We extend previous work on stretch-minimizing stream surfaces [3] where a different special class of stream surface was investigated. Those surfaces possess the property that their seed curves propagate in time such that their arc-lengths are as constant as possible, and therefore help to detect parts of the domain where the given divergence-free vector field acts not only in a volume-preserving but also in a length-preserving manner. In contrast, in this work, we seek stream surfaces arising from seed curves that advect in time with the maximum stretching rate, see Fig. 1, left.

Problem formulation. Given a vector field $\mathbf{v}$ in some domain $\Omega \subset \mathbb{R}^3$, find a *stream surface* such that the *seed curve* that defines it propagates in time such that its arc-length expands as much as possible.

We combine theoretical results and a practical algorithm for finding such surfaces. The main contributions of our method are:

- We theoretically investigate families of seed curves based on a certain stretch-maximizing energy (Section 3).

[*] Corresponding author

Email addresses: mbarton@bcamath.org (Michael Bartoň), j.kosinka@rug.nl (Jiří Kosinka)

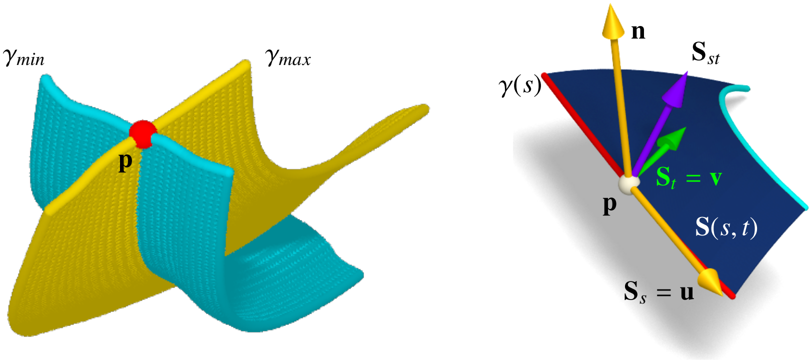


Figure 1: Left: Let $\mathbf{p}$ be a seed point in a vector field $\mathbf{v}$. A stretch-minimizing curve γ_{min} advects infinitesimally in $\mathbf{v}$ with a zero stretching rate, while a stretch-maximizing curve γ_{max} stretches with the maximum possible rate and consequently maximum local advection. Right: Differential geometry of stream surfaces. The stream surface $\mathbf{S}(s,t)$ (dark blue) is generated by the seed curve $\gamma(s)$ (red) by integrating it through $\mathbf{v}$.

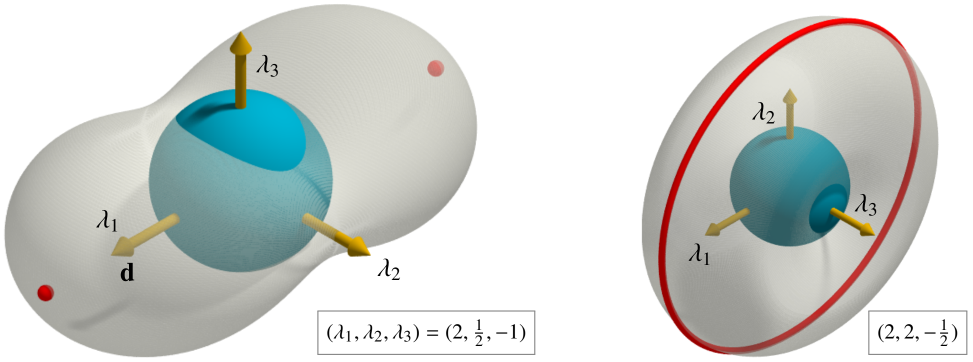


Figure 2: The scalar function $\mathbf{uJu}^\top$ (transparent grey) is defined on the unit sphere (blue), and we seek its maximum (red); see (3). The stretch-maximizing direction $\mathbf{d}$ is given by the eigenvector associated to the largest eigenvalue, λ_1, of $\mathbf{J}^+$. The normalized eigenvectors are shown in yellow. Generically, $\mathbf{d}$ is unique (left). In the case when $\lambda_1 = \lambda_2$, a one parameter family of stretch-maximizing directions exists (right).

- These candidate seed curves are used to initialize stream surfaces, which are subsequently globally optimized (Section 4).

We provide implementation details of our method in Section 5. Our results are presented and discussed in Section 6. We include standard benchmark datasets and we also validate our algorithm on an analytic vector field with known stretch-maximizing stream surface solutions. The paper is concluded in Section 7.

2. Related work

Vector fields are widely used in geometry processing for object propagation and deformation [16, 32, 4]. For example for deformation purposes, appropriate vector fields are iteratively sought-after such that an object under consideration (curve, solid) preserves its certain measure (arc-length, volume). Visualization of vector fields is a very active area; see [5, 21, 9, 10], and the survey paper [24] and the references cited therein. Among flow visualization techniques, stream surfaces play an important role [31, 29, 27, 23, 11, 8].

Given a vector field, the main goal is to select a seed curve such that its stream surface captures well the characteristic features of the field. While classical results like [18] admit user's intervention to set seed curves in a trial-and-error manner, recent research focuses on fully *automatic* stream surface seeding [11, 8, 3]. Our work belongs to this category of automatic methods.

As shown in Fig. 1, our work on optimal advection can be seen as a modification of the concept of stretch-minimizing stream surfaces [3]. Our framework differs from that of [3] in several aspects, which require a separate treatment as detailed below.

Our problem formulation is closely related to the concept of Lyapunov stability. In fact, the sought-after seed curves could be seen as loci of points where the corresponding dynamical system (represented by the input vector field) is as unstable as possible. This is formalized by the notion of Lyapunov exponents [25, 33]. In our approach, we employ the local variant of the Lyapunov exponent [7].

3. Stretch-energy maximizing curves

Let $\mathbf{v}(\mathbf{p})$ be a steady differentiable vector field defined over a domain $\Omega \subset \mathbb{R}^3$. Let $\mathbf{J}$ be the Jacobian matrix of $\mathbf{v}(\mathbf{p})$, i.e., $\mathbf{J}_{ij} = \frac{\partial \mathbf{v}_i}{\partial p_j}$ with $\mathbf{p} = (p_1, p_2, p_3)$. Consider a regular *seed curve* $\gamma(s)$ parametrized by arc-length, $s \in [s_0, s_1]$, that gives rise to a *stream surface* $\mathbf{S}(s,t)$ defined on $[s_0, s_1] \times [t_0, t_1]$, with $\mathbf{S}(s,0) = \gamma(s)$; see Fig. 1, right. The partial derivatives of $\mathbf{S}$ will be denoted $\mathbf{S}_s$, $\mathbf{S}_{st}$, etc.

The Taylor expansion of the arc-length of the timelines of $\mathbf{S}(s,t)$ with respect to t at $t = 0$ (which corresponds to γ) is given by

$$\int_{s_0}^{s_1} \|\mathbf{S}_s(s,t)\| \, \mathrm{d}s = (s_1 - s_0) + ct + O(t^2) \tag{1}$$

for some $c \in \mathbb{R}$. By definition (cf. Lemma 1 in [3]), $c = C(0)$, where $C(t) = \frac{\partial}{\partial t} \int_{s_0}^{s_1} \|\mathbf{S}_s(s,t)\| \, \mathrm{d}s$. We have that

$$C(t) = \int_{s_0}^{s_1} \frac{\partial}{\partial t} \|\mathbf{S}_s(s,t)\| \, \mathrm{d}s = \int_{s_0}^{s_1} \frac{\partial}{\partial t} \sqrt{\langle \mathbf{S}_s, \mathbf{S}_s \rangle} \, \mathrm{d}s = \int_{s_0}^{s_1} \frac{\langle \mathbf{S}_s, \mathbf{S}_{st} \rangle}{\sqrt{\langle \mathbf{S}_s, \mathbf{S}_s \rangle}} \, \mathrm{d}s.$$

Since $\sqrt{\langle \mathbf{S}_s, \mathbf{S}_s \rangle}|_{t=0} \equiv 1$ it follows that

$$c = \int_{s_0}^{s_1} \langle \mathbf{S}_s(s,t), \mathbf{S}_{st}(s,t) \rangle \, \mathrm{d}s|_{t=0}. \tag{2}$$

Based on this, our aim is to identify curves γ in Ω for which $c \to \max$.

Denoting $\mathbf{u} := \dot{\gamma}(s) = \mathbf{S}_s(s,0)$, we have $\langle \mathbf{S}_s, \mathbf{S}_s \rangle|_{t=0} \equiv \langle \mathbf{u}, \mathbf{u} \rangle \equiv 1$. Differentiation with respect to t, and requiring that the arc-length of γ expands by the maximum possible factor, yields

$$\langle \mathbf{S}_s, \mathbf{S}_{st} \rangle|_{t=0} \equiv \mathbf{u}\mathbf{J}\mathbf{u}^\top \to \max, \tag{3}$$

where $\mathbf{u}^\top$ denotes the transpose of $\mathbf{u}$. A unit vector $\mathbf{d}$ that maximizes $\mathbf{d}\mathbf{J}\mathbf{d}^\top$ will be called *stretch-maximizing*. Note that $\mathbf{d}\mathbf{J}\mathbf{d}^\top$ is then equal to the (largest) local Lyapunov exponent [7].

We denote *stretch rate* $\mathbf{J}^+ := (\mathbf{J} + \mathbf{J}^\top)/2$ and *vorticity* $\mathbf{J}^- := (\mathbf{J} - \mathbf{J}^\top)/2$ as the symmetric and antisymmetric parts of $\mathbf{J}$, respectively.

Since $\mathbf{u}\mathbf{J}^-\mathbf{u}^\top = 0$ for any $\mathbf{u}$, we are searching for curves γ that maximize $\mathbf{u}\mathbf{J}^+\mathbf{u}^\top$ at every $s \in [s_0, s_1]$. Assume that the eigenvalues λ_1, λ_2, and λ_3 of $\mathbf{J}^+$ satisfy $\lambda_1 > \lambda_2 > \lambda_3$ in some neighborhood $\mathcal{N}$ of a point $\mathbf{p} \in \Omega$. Then the stretch-maximizing vector $\mathbf{d}$ is, up to orientation, unique at all points in $\mathcal{N}$ and equal to the normalized eigenvector associated to λ_1. This implies that the curve γ, obtained by integrating $\mathbf{d}$, passing through $\mathbf{p}$ is unique. Any such curve will be called *stretch-maximizing*.

We remark that while in the setting of [3] also second order analysis was applied, i.e., the quadratic term in (1) was considered as well, only first-order analysis can be used in our case as generically the solution space of stretch-maximizing vectors $\mathbf{d}$ arising from our first order analysis is zero-dimensional. Therefore the integral seed curves that arise from the first order analysis are weaker initial guesses for the optimization than those in [3], and yet our method finds the exact solution as tested on an analytical example (see Section 6 and Fig. 11).

Note that $\lambda > 0$ corresponds to (infinitesimal) stretching, $\lambda = 0$ to no change, and $\lambda < 0$ to shrinking of arc-length in the associated eigenvector direction; see Fig. 2. For our application, we require that $\lambda_1 > 0$. While this is not guaranteed for general vector fields (where one can even have $\lambda_1 < 0$, in which case we stop integration), incompressible vector fields satisfy $\lambda_1 + \lambda_2 + \lambda_3 = 0$ and thus, unless $\mathbf{J}^+$ is singular (case of measure zero), fit our framework well. Further, the stretch-maximizing vector may not be unique (Fig. 2, right). Nevertheless, this is again a case of measure zero and can be easily dealt with.

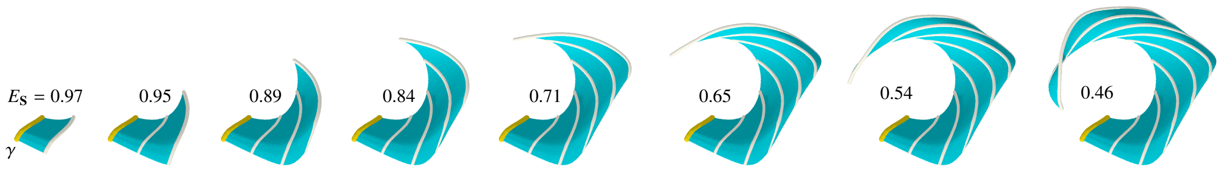


Figure 3: Evolution of the stretch energy $E_\mathbf{S}$ of the stream surface $\mathbf{S}$ in time. As t grows, $E_\mathbf{S}$ decreases. The seed curve γ is stretch-maximizing. By construction, $E_\mathbf{S} = 1$ only for infinitesimally small surfaces (i.e., for $t \to 0$); see (4) and (5).

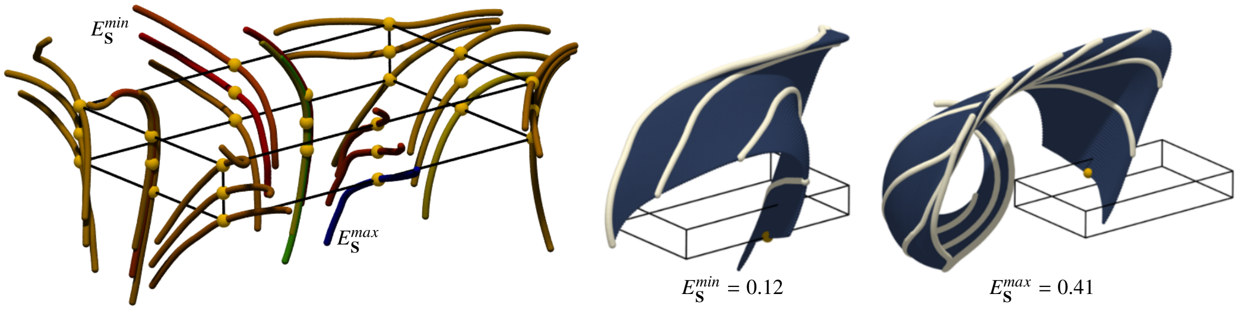


Figure 4: Seed curve ranking. Left: A voxel (black) is sampled by 3^3 seed points (yellow) and stretch-maximizing seed curves of a fixed length are computed. The seed curves are ranked (and colored) according to $E_\mathbf{S}$; see (5). Middle and right: The worst ($E_\mathbf{S}^{min}$) and the best ($E_\mathbf{S}^{max}$) stream surface found. Observe that while the stretching of the timelines is progressive only in the later stage of the propagation in the case of the worst ranked stream surface ($E_\mathbf{S}^{min}$), the timelines of the best ranked stream surface ($E_\mathbf{S}^{max}$) stretch more uniformly in time.

To be able to classify the stretch-maximizing property of curves (and later also surfaces), we define the *stretch energy* of the seed curve $\gamma(s) = \mathbf{S}(s, 0)$ as

$$E(\gamma) = \frac{1}{s_1 - s_0} \int_{s_0}^{s_1} \frac{\langle \mathbf{S}_s(s,t), \mathbf{S}_{st}(s,t) \rangle}{\lambda_1(s,t)} \, \mathrm{d}s|_{t=0}. \tag{4}$$

Turning back to (1) and the definition of c, we see that the energy is well defined, i.e., $E(\gamma) \leq 1$ and the value of 1 is attained if γ is a stretch-maximizing (seed) curve.

4. Stream surfaces and optimization

We seek stream surfaces with timelines that expand with the maximum possible stretch rate. Our candidate seed curves satisfy this condition at $t = 0$. However, our goal is to find stream surfaces whose generating seed curves propagate *globally* in a stretch-maximizing manner. Therefore, we define the *stretch energy* of a stream surface $\mathbf{S}$ by

$$E_\mathbf{S} = \frac{1}{\mathrm{area}(\mathbf{S})} \int_{s_0}^{s_1} \int_{t_0}^{t_1} \frac{\langle \mathbf{S}_s(s,t), \mathbf{S}_{st}(s,t) \rangle}{\lambda_1(s,t)} \, \mathrm{d}t \, \mathrm{d}s. \tag{5}$$

$E_\mathbf{S}$ measures the *relative* stretching given by the deformation of the seed curve of $\mathbf{S}$ through the field by accumulating the relative stretch energies of all timelines of $\mathbf{S}$. Observe that $E_\mathbf{S} \leq 1$ and the equality occurs when each of the timelines of $\mathbf{S}$ is a stretch-maximizing curve, i.e., when $E(\mathbf{S}(s,t)) = 1$ for every $t \in [t_0, t_1]$. The evolution of the stretch energy $E_\mathbf{S}$ of a surface emanating from a stretch-maximizing curve $\gamma = \mathbf{S}(s, 0)$ is shown in Fig. 3.

We use this energy to measure the stretching of stream surfaces. We subdivide the domain into voxels. Each voxel is sampled, and initial stretch-maximizing seed curves are computed; see Fig. 4. Integration of the seed curves in the vector field gives initial stream surfaces that are ranked according to (5). Neighborhoods of seed points with highest $E_\mathbf{S}$ are then sampled more densely.

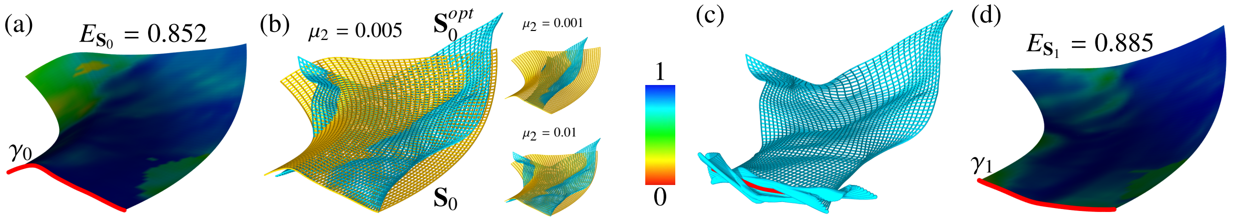


Figure 5: One cycle of our global optimization. (a) The initial stretch-maximizing seed curve γ_0 (red) is integrated, resulting in the initial stream surface $\mathbf{S}_0$, which is color-coded by the stretch energy E of (5). (b) $\mathbf{S}_0$ is, as a dense quad mesh (yellow), optimized according to (6); three outcomes of $\mathbf{S}_0^{opt}$ with fixed $\mu_1 = 0.1$ and varying μ_2 are shown. (c) The timelines of the optimized (non-stream) surface are back-integrated to $t = 0$ (cyan curve bundle) and the best L_2 fit defines the new seed curve γ_1 (red). (d) The new stream surface $\mathbf{S}_1$ emanating from γ_1. Note that $E_{\mathbf{S}_1} > E_{\mathbf{S}_0}$.

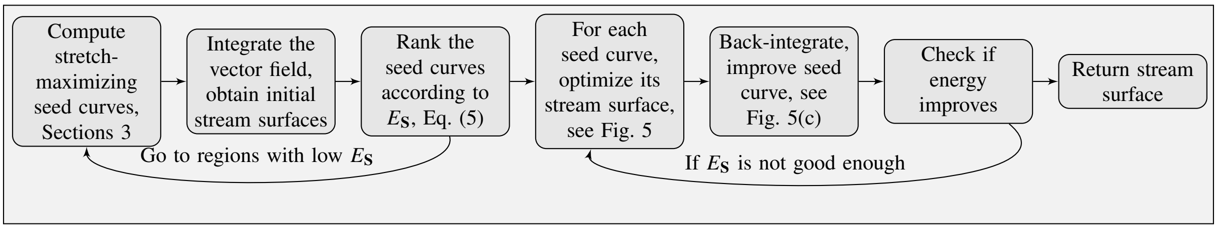

Compute stretch-maximizing seed curves, Sections 3
Integrate the vector field, obtain initial stream surfaces
Rank the seed curves according to $E_\mathbf{S}$, Eq. (5)
For each seed curve, optimize its stream surface, see Fig. 5
Back-integrate, improve seed curve, see Fig. 5(c)
Check if energy improves
Return stream surface
Go to regions with low $E_\mathbf{S}$
If $E_\mathbf{S}$ is not good enough

Figure 6: Algorithm overview.

Optimization. Candidate seed curves generate stream surfaces with relatively high stretch energies. However, their effect is still only *local*, whilst we are interested in maximizing the stretch energy *globally*. This is achieved by employing an optimization procedure; see Fig. 5.

Given a stream surface $\mathbf{S}$, we want to maximize its stretch energy (5). We discretize $\mathbf{S}$ by sampling m points in the s-direction (seed curve) and n points in the t-direction (time), resulting in a quad mesh $\mathbf{Q}$ with $m \times n$ vertices $\mathbf{q}_{i,j}^0$. Denoting the vertices of the optimized surface by $\mathbf{q}_{i,j}$ and setting $\mathbf{d}_{i,j}$ to the eigenvector of $\mathbf{J}^+$ that corresponds to its maximum eigenvalue, the desired improvement is formulated as a non-linear least squares optimization problem with the objective function

$$F(\mathbf{Q}) = F_{stretch}(\mathbf{Q}) + \mu_1 F_{fair}(\mathbf{Q}) + \mu_2 F_{prox}(\mathbf{Q}) \tag{6}$$

with three individual terms

$$\begin{aligned} F_{stretch}(\mathbf{Q}) &= \textstyle\sum_{i,j} \|(\mathbf{q}_{i+1,j} - \mathbf{q}_{i,j}) \times \mathbf{d}_{i,j}\|^2, \\ F_{fair}(\mathbf{Q}) &= \textstyle\sum_{i,j} \|\mathbf{q}_{i+1,j} - 2\mathbf{q}_{i,j} + \mathbf{q}_{i-1,j}\|^2 + \\ &\quad + \|\mathbf{q}_{i,j+1} - 2\mathbf{q}_{i,j} + \mathbf{q}_{i,j-1}\|^2, \\ F_{prox}(\mathbf{Q}) &= \textstyle\sum_{i,j} \|\mathbf{q}_{i,j} - \mathbf{q}_{i,j}^0\|^2. \end{aligned} \tag{7}$$

$F_{stretch}$ reflects the stretch maximizing condition of (5), F_{fair} is a fairness term that penalizes highly curved stream surfaces, and F_{prox} is a proximity term, a regularizer that forces the vertices of the optimized mesh to stay near the input mesh. The optimization problem was solved using the Gauss-Newton method for all the examples in the paper.

Experimentally, the weights were set to $\mu_1 = 0.1$, $\mu_2 = 0.005$. In particular, higher values of μ_2 forced the quad-mesh vertices to move from their initial position very little, which resulted in many optimization rounds with only marginal improvement of F. On the other hand, tiny weights μ_2 allowed the vertices to move considerably, which resulted in a very incoherent bundle of back-integrated timelines, see Fig. 5(c), and consequently the optimization failed. Both of these extreme cases are aimed to be avoided by setting an intermediate value for μ_2. Another fact that affects the optimization is the fineness of the surface discretization. However, in our experiments, this issue turned out to be less influential than the choice of weights in (6), see also Fig. 5(b).

An overview of our framework is shown in Fig. 6.

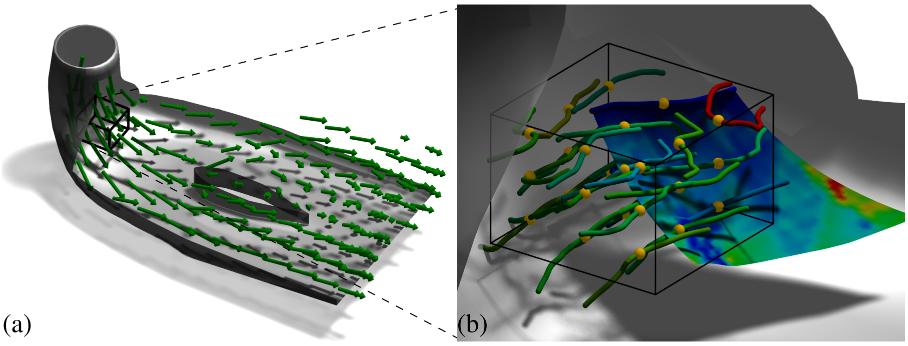


Figure 7: Turbine data of [5]. (a) One time instant of the unsteady vector field generated by the Francis turbine with one sampling subdomain (black) is shown. (b) A zoom-in on the subdomain with the initial seed points (yellow). Stretch-minimizing and stretch-maximizing stream surfaces for this example are shown in Fig. 10.

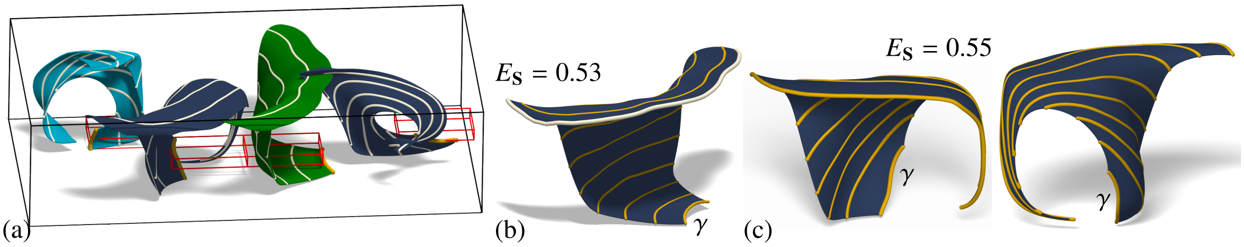


Figure 8: Rayleigh-Bénard heat convection. The vector field used corresponds to one time instant of the unsteady vector field from [1]. (a) Four best initial stretch maximizing surfaces in the second subdivision level. Each surface emanates from a stretch-maximizing curve (yellow), the best candidate from the corresponding subdomain (smaller red boxes). (b) The second highest ranked surfaces according to (4) after applying our optimization; see (6). (c) Two different views on the best stretch-maximizing surface.

5. Implementation

We now discuss important implementation aspects of the proposed framework for finding stretch-maximizing surfaces.

Seed curve computation. Starting with a sampled seed point, the direction that maximizes (3) is computed (see Figs. 2 and 4) and integrated. We employed the classical fourth order Runge-Kutta method with constant step-size, which was set to 1% of the domain's diameter, i.e., $\mathrm{diam}(\Omega)/100$. This parameter was set experimentally and can be changed, as well as other parameters discussed in the paragraphs below, using our graphical user interface.

In the non-generic cases when the stretch-maximizing direction is not unique (or in a non-incompressible vector field), the seed curve integration continues by choosing the next tangent direction so that it minimally deviates from the one in the previous integration step. This strategy is compatible with our goal of finding stretch-maximizing surfaces.

Table 1: Default parameters and thresholds of our algorithm for a domain of unit diameter. In the *'Curve tracing'* columns, Δs is the tracing stepsize, l_{min} is the minimum length of a candidate curve, and $\#n_{max}$ is the maximum number of tracing steps, both for t and s directions. In the *'Adaptive sampling'* columns, $\#S$ is the number of subdivisions, δ is the percentage of the new subdomain relative to the old one, and b is the percentage of the best candidate curves selected for the next iteration. In the last two columns, the optimization parameters include μ_1 and μ_2 (see Eq. (6)).

Curve tracing			Adaptive sampling			Optimisation	
Δs	l_{min}	$\#n_{max}$	$\#S$	δ	b	μ_1	μ_2
0.01	0.1	500	3	10%	5%	0.1	0.005

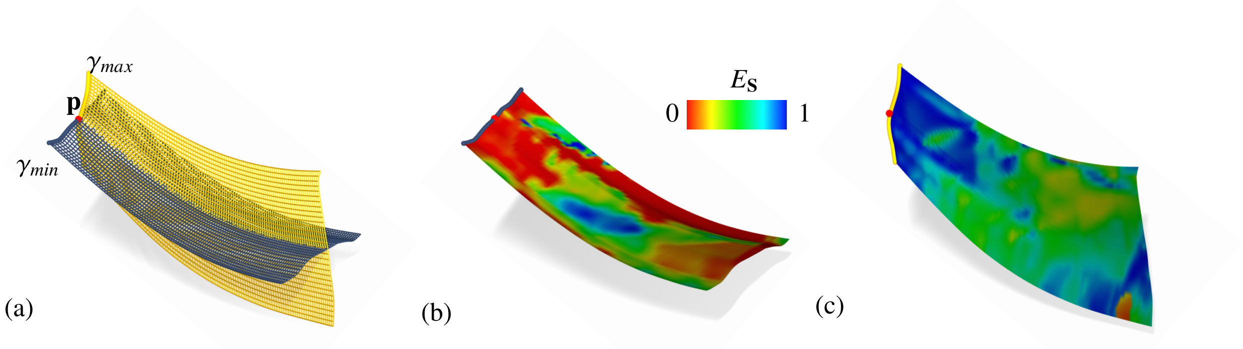


Figure 9: Stretch minimizing vs. maximizing seed curves. (a) At a seed point $\mathbf{p}$, two extremal seed curves are integrated: the second order stretch-minimizing curve [3] and the stretch maximizing curve found by our approach. These curves propagate infinitesimally with no stretching (γ_{min}) and maximum possible stretching (γ_{max}), respectively. Both seed curves are of the same arc-length; the corresponding stream surfaces are discretized as quad meshes. (b,c) The stream surfaces are color-coded by the stretch energy (5).

Candidate seed curves. The interior of Ω is sampled and seed curves are computed. These are ranked by (5) and regions with high values of $E_\mathbf{S}$ are recursively sampled with higher density. That is, the neighborhoods of points giving rise to good seed curves are taken as new subdomains. The diameter of the new domain is set to 10% of the old one and the best 5% of seed points are taken as input for the next iteration. This sampling is repeated recursively three times and the overall best 5% are taken as candidate curves. The length of the curves is set to $\mathrm{diam}(\Omega)/10$, but this can be easily made application-dependent.

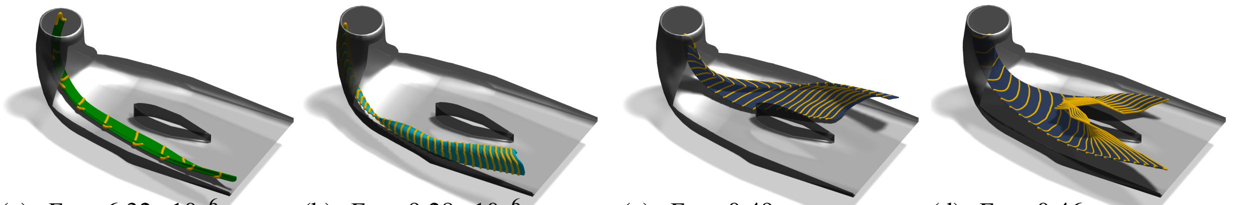
Seed curve optimization. One step of our optimization cycle is illustrated in Fig. 5. A candidate seed curve γ_0 gives rise to the stream surface $\mathbf{S}_0$, which is optimized into $\mathbf{S}_0^{opt}$ according to (6). The timelines of $\mathbf{S}_0^{opt}$ are back-integrated to $t = 0$. As $\mathbf{S}_0^{opt}$ is not, in general, a stream surface, this yields a family of space curves. The seed curve γ_0 is replaced by the least squares approximation [12] of this family. This process yields γ_1 and its stream surface $\mathbf{S}_1$. If $E_{\mathbf{S}_1} > E_{\mathbf{S}_0}$, the process is repeated with $\gamma_0 \leftarrow \gamma_1$ as input. Otherwise, optimization is terminated. In our discrete setting, the surface stretch energy (5) is computed at the vertices of the quad mesh and normalized by the mesh area.

Surface parameter domain. The parametric rectangle $[s_0, s_1]$ and $[t_0, t_1]$ is set as follows. In the s direction (along seed curve/timelines), $\widetilde{s} = s_1 - s_0$ is set to $\mathrm{diam}(\Omega)/10$, but it is truncated in case integration reaches the boundary of Ω. The time interval $[t_0, t_1]$ directly maps to the desired resolution and number of time steps, and $\widetilde{t} = t_1 - t_0$ is also set as a parameter. Again, integration is terminated at the boundary of Ω. In our implementation, the parameters $\widetilde{s}$ and $\widetilde{t}$ are constants in the optimization framework, but enable the user to control the desired length and time. In other words, we seek the best stretch-maximizing surface that can be generated by a seed curve of a given length in a fixed time.

Stream surface computation and discretization. To advect the seed curve, we again employed the classical fourth order Runge-Kutta method with constant step-size. The surface integration terminates once the front timeline reaches the domain boundary. The stream surfaces were discretized by $m \times n$ quad meshes, m and n being the number of samples in s and t direction, respectively. The discretization was realized by uniform sampling of the parameter domain $[s_0, s_1] \times [t_0, t_1]$. The concrete values of m vary as they are computed independently for each seed curve. The maximum value of n was by default set to 500, see also Table 2, but this value is smaller when the timeline integration reaches the domain's boundary (e.g., $n = 461$ in Fig. 10(d)). For the sake of simplicity, the quad mesh was kept in this rectangular pattern. Alternatively, one could apply topology-aware techniques like [18, 27, 28, 14, 26, 22] to refine the mesh, e.g., in the neighborhood of critical points. The parameters of our algorithm were set as shown in Table 1.

6. Results and discussion

Results obtained by our method are shown in Figs. 7 and 10 for the Francis turbine dataset (kindly provided by Maik Schulze) and in Fig. 8 for Rayleigh-Bénard heat convection. Table 2 shows statistics for these two examples.



(a) $E_\mathbf{S} = 6.32 \cdot 10^{-6}$ (b) $E_\mathbf{S} = 9.28 \cdot 10^{-6}$ (c) $E_\mathbf{S} = 0.49$ (d) $E_\mathbf{S} = 0.46$

Figure 10: A comparison between stretch-maximizing and stretch-minimizing stream surfaces on the Francis turbine dataset. Two best stretch-minimizing stream surfaces (a,b) and two best stretch-maximizing stream surfaces (c,d) together with the values of the corresponding stretching energies given by Eq. (11) in [3] and (5) in this paper, respectively.

Stretch-minimizing vs. stretch-maximizing seed curves. A comparison with [3] is made in Fig. 9. At a given seed point $\mathbf{p}$, the stretch-maximizing and the second order stretch-minimizing seed curves are computed and advected. In addition, Fig. 10 shows a comparison between the two best stretch-minimizing and the two best stretch-maximizing stream surfaces on the Francis turbine dataset.

We emphasize here that the first order analysis of stretch-minimizing directions gives rise to a one parameter family of admissible directions (see Section 3.1 in [3]) and thus second-order analysis can be utilized in that case. In contrast, the first order stretch-maximizing direction is, in general, unique, and therefore higher order analysis makes no sense in our case. Fig. 9 shows the stream surfaces emanating from the stretch-minimizing and stretch-maximizing seed curves. Note the significant difference in the area and the stretch energy of the two stream surfaces.

Validation against exact solution. We tested our algorithm on the analytic vector field $\mathbf{v} = [x, 2y - \frac{z^2}{2}, yz]$ over $\Omega = [-1, 1]^3$, see Fig. 11. Since $\mathbf{J}^+ = \text{diag}(1, 2, y)$, the stretch-maximizing direction is constant (y-axis) and the stretch-maximizing seed curves are lines parallel to y. Thus, the (two-parameter family of) stretch-maximizing ($E_\mathbf{S} \equiv 1$) surfaces are planar patches emanating from seed curves in the $z = 0$ plane. The exact solution was recovered by our method within a very fine energy threshold, see Fig. 11(d,e).

Singular cases. The seed curve remains well-defined (and unique) as long as $\mathbf{J}^+$ is regular and the dominant eigenvalue is unique. In regions where $\lambda_1 \doteq \lambda_2$, the stretch-maximizing direction becomes ill-posed (Fig. 2, right), and the seed curve may exhibit sudden changes in its tangent direction, while in general it is smooth; see Fig. 12. In the case when $\mathbf{J}^+$ is singular, the integration is terminated.

Unsteady flows and unstructured grids. We considered steady vector fields for the sake of simplicity. Conceptually, the generalization to unsteady flows is straightforward: one needs to sample the space-time domain. Another simplification relates to the fact that the input vector fields were given over fine (voxelized) grids, which simplifies the computation of velocities outside grid-points. An extension of the algorithm towards unstructured input is also possible.

Table 2: A summary of our results. In the *'Seed curves'* columns, #$\mathbf{p}$ is the number of initial sampled seed points and $E_\mathbf{S}^{ini}$ is the stretch energy (5) of the best candidate before optimization. In the *'Optimization'* columns, #$\mathbf{S}$ is the number of surfaces processed and $E_\mathbf{S}^{opt}$ is the energy of the best optimized stream surface. The last column lists total computation times in seconds (obtained on a double core CPU (2.67GHz) machine with 24GB RAM).

	Seed curves		Optimization		Time
Fig.	#$\mathbf{p}$	$E_\mathbf{S}^{ini}$	#$\mathbf{S}$	$E_\mathbf{S}^{opt}$	(sec.)
8	300	0.47	83	0.55	565
10 (c,d)	216	0.44	62	0.49	405
10 (a,b)	216	$3.17 \cdot 10^{-4}$	54	$6.32 \cdot 10^{-6}$	385

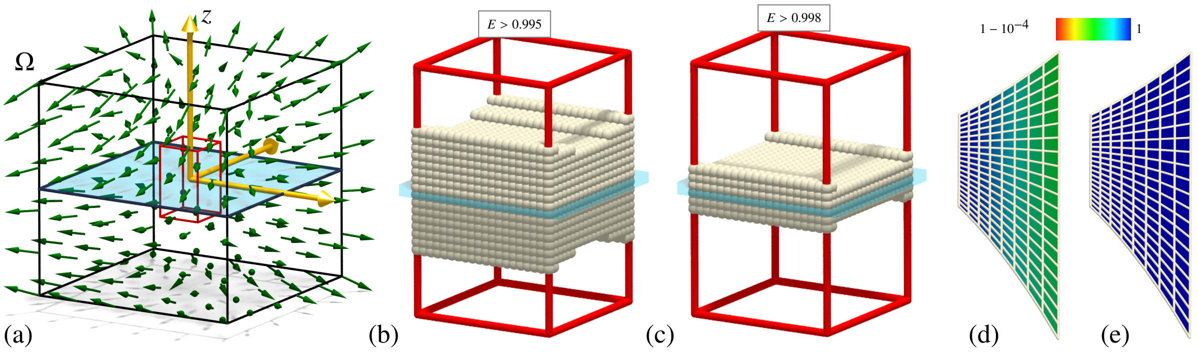


Figure 11: Validation. (a) An analytic vector field (green) over Ω possesses exact stretch-maximizing surfaces in $z = 0$ plane (transparent). (b,c) Seed points from a sub-domain (red) that induce seed curves (and consequently surfaces) satisfying fine energy thresholds. The best stretch-maximizing surface computed by our algorithm (d) and its analytic counterpart (e); (rotated top view).

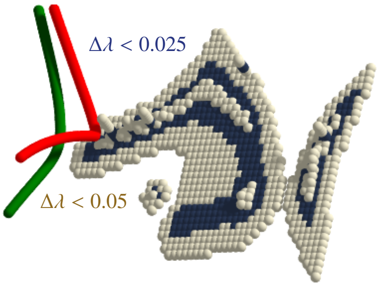


Figure 12: When $\Delta\lambda = |\lambda_1 - \lambda_2|$ is small, the seed curve (red) may suddenly change direction, whereas it is smooth (provided that $\mathbf{v}$ is smooth) otherwise (green).

Initialization and optimization strategy. The approach we took is common in geometry processing literature where an optimal solution is sought-after in high-dimensional spaces (shape space) [34, 6, 2]. The concept is to detect parts of the (typically huge) space, compute the initial solution, and optimize it by minimizing certain energy functional.

We use adaptive sampling with all pros and cons of this approach. The main advantage is that it is relatively computationally cheap and it admits user's intervention to continue the search if the result (expressed in term of stretch energy) is not satisfactory. On the other hand, there is no guarantee that the optimal solution is found. Alternatively, one could use the more sophisticated approach via scattered data fitting, see e.g. [13]. However, such an approach does not provide optimality guarantees either and is computationally more demanding.

Divergence-free vs. general vector fields. We have restricted ourselves to divergence-free vector fields since these fields guarantee the existence of a direction that stretches with a positive rate, i.e., seed curves locally expand. Of course, the concept of the best stretch-maximizing direction can be considered for general vector fields as well. However, for example when the signature of $\mathbf{J}^{+}$ is $(-,-,-,)$, the 'best' direction reflects the minimum possible shrinking rate, which is at odds with our application context of efficient advection.

Time-minimizing advection. Our approach seeks seed curves that globally expand as much as possible, but it ignores the time needed for advection. It might be interesting to search for the most progressive advection within a time unit. That is, to seek a seed curve that propagates also as fast as possible.

Stretch-maximizing surfaces and Lyapunov theory. In the context of dynamical systems, our problem seeks one-parameter families of initial values (seed curves) such that their solutions (determined in our case by the input vector field) are, in Lyapunov sense, as unstable as possible (stretch-maximizing evolution). Our seed curves are integral curves that correspond to the eigenvectors associated to the largest local Lyapunov exponents. As a promising avenue for future research, it would be interesting to look at finite-time Lyapunov exponents [25, 15]. However, one would need to also investigate the influence of the selected time-interval on the obtained results, something we expect to be considerably more expensive than our less global method given the complexity of the computation of the non-local Lyapunov exponents. And as an additional challenge, non-local exponents do not directly provide a direction along which to integrate to obtain ‘optimal’ seed curves.

7. Conclusion

We have investigated a family of stream surfaces that possess the property of maximum timeline expansion admitted by a given vector field. The surfaces are initialized by stretch-maximizing seed curves that always exist in divergence-free vector fields and are generically unique. We illustrated our algorithm on several examples and validated it on a field with known solutions. We believe our analysis will serve not only as a visualization tool, but also help in applications such as efficient fluid advection-diffusion and dissemination.

Acknowledgments. The first author has been partially supported by the Basque Government through the BERC 2014-2017 program, by Spanish Ministry of Economy and Competitiveness MINECO: BCAM Severo Ochoa excellence accreditation SEV-2013-0323, and the Project of the Spanish Ministry of Economy and Competitiveness with reference MTM2016-76329-R (AEI/FEDER, EU).

We thank the anonymous referees of GMP 2017 for their helpful insights which helped us improve this paper.

References

[1] NaSt3DGP (v2.08) – A Parallel 3D Flow Solver. http://wissrech.iam.uni-bonn.de/research/projects/nast3dgp.

[2] M Bartoň, Helmut Pottmann, and Johannes Wallner. Detection and reconstruction of freeform sweeps. *Computer Graphics Forum*, 33(2):23–32, 2014.

[3] Michael Bartoň, Jiří Kosinka, and Victor M Calo. Stretch-minimising stream surfaces. *Graphical Models*, 79:12–22, 2015.

[4] Michael Bartoň, Ling Shi, Martin Kilian, Johannes Wallner, and Helmut Pottmann. Circular arc snakes and kinematic surface generation. *Computer Graphics Forum*, 32(2):1–10, 2013.

[5] Dirk Bauer and Ronald Peikert. Vortex tracking in scale-space. In *Proceedings of the Symposium on Data Visualisation*, pages 233–242. Eurographics Association, 2002.

[6] Bailin Deng, Sofien Bouaziz, Mario Deuss, Juyong Zhang, Yuliy Schwartzburg, and Mark Pauly. Exploring local modifications for constrained meshes. *Computer Graphics Forum*, 32(2pt1):11–20, 2013.

[7] Bruno Eckhardt and Demin Yao. Local Lyapunov exponents in chaotic systems. *Physica D: Nonlinear Phenomena*, 65(1–2):100–108, 1993.

[8] M. Edmunds, , R.S. Laramee, G. Chen, E. Zhang, and N. Max. Advanced, automatic stream surface seeding and filtering. *Theory and Practice of Computer Graphics*, pages 53–60, 2012.

[9] M. Edmunds, R.S. Laramee, R. Malki, I. Masters, T.N. Croft, G. Chen, and E. Zhang. Automatic stream surface seeding: A feature centered approach. *Comp. Graph. Forum*, 31:1095–1104, 2012.

[10] M. Edmunds, R.S. Laramee, R. Malki, I. Masters, T.N. Croft, G. Chen, and E. Zhang. Surface based flow visualization. *Computer & Graphics*, 36(8):974–990, 2012.

[11] M. Edmunds, T. McLoughlin, R.S. Laramee, G. Chen, N. Max, and E. Zhang. Automatic stream surface seeding. *EUROGRAPHICS 2011 Short Papers*, pages 53–56, 2011.

[12] Gerald Farin, Josef Hoschek, and Myung-Soo Kim. *Handbook of Computer Aided Geometric Design*. Elsevier, Amsterdam, 2002.

[13] Richard Franke and Gregory M Nielson. Scattered data interpolation and applications: A tutorial and survey. In *Geometric Modeling*, pages 131–160. Springer, 1991.

[14] Christoph Garth, Xavier Tricoche, Tobias Salzbrunn, Tom Bobach, and Gerik Scheuermann. Surface techniques for vortex visualization. In *Proceedings of the Sixth Joint Eurographics - IEEE TCVG Conference on Visualization*, VISSYM'04, pages 155–164, Aire-la-Ville, Switzerland, Switzerland, 2004. Eurographics Association.

[15] Tobias Günther, Alexander Kuhn, and Holger Theisel. MCFTLE: Monte Carlo Rendering of Finite-Time Lyapunov Exponent Fields. *Computer Graphics Forum*, 35(3):381–390, 2016.

[16] Klaus Hildebrandt, Konrad Polthier, and Eike Preuss. Evolution of 3D curves under strict spatial constraints. In *CAD/Graphics*, pages 6 pp.–. IEEE, 2005.

[17] Hussein Hoteit, Abbas Firoozabadi, et al. Numerical modeling of diffusion in fractured media for gas-injection and-recycling schemes. *Spe Journal*, 14(02):323–337, 2009.

[18] J. P. M. Hultquist. Constructing stream surfaces in steady 3D vector fields. *Proceedings of VIS '92, IEEE Transactions on Visualization and Computer Graphics*, pages 171–178, 1992.

[19] Willem Hundsdorfer and Jan G Verwer. *Numerical solution of time-dependent advection-diffusion-reaction equations*, volume 33. Springer Science & Business Media, 2013.

[20] Syed Shariq Kamal, Patrick Jackman, and Bruce Grieve. Minimizing flight time and fuel consumption for airborne crop spraying. *Agricultural Engineering International: CIGR Journal*, 16(2):89–96, 2014.

[21] Michael Lentine, Wen Zheng, and Ronald Fedkiw. A novel algorithm for incompressible flow using only a coarse grid projection. *ACM Trans. Graph.*, 29(4):114:1–114:9, July 2010.

[22] Kewei Lu, Han-Wei Shen, and Tom Peterka. Scalable computation of stream surfaces on large scale vector fields. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, SC '14, pages 1008–1019, Piscataway, NJ, USA, 2014. IEEE Press.

[23] J. Martinez Esturo, M. Schulze, C. Rössl, and H. Theisel. Global selection of stream surfaces. *Computer Graphics Forum*, 32(2):113–122, 2013.

[24] Tony McLoughlin, Robert S. Laramee, Ronald Peikert, Frits H. Post, and Min Chen. Over Two Decades of Integration-Based, Geometric Flow Visualization. *Computer Graphics Forum*, 29(6):1807–1829, 2010.

[25] Ronald Peikert, Armin Pobitzer, Filip Sadlo, and Benjamin Schindler. A comparison of finite-time and finite-size Lyapunov exponents. In Peer-Timo Bremer, Ingrid Hotz, Valerio Pascucci, and Ronald Peikert, editors, *Topological Methods in Data Analysis and Visualization III: Theory, Algorithms, and Applications*, pages 187–200. Springer International Publishing, 2014.

[26] Tobias Schafhitzel, Eduardo Tejada, Daniel Weiskopf, and Thomas Ertl. Point-based stream surfaces and path surfaces. In *Proceedings of Graphics Interface 2007*, GI '07, pages 289–296, New York, NY, USA, 2007. ACM.

[27] Dominic Schneider, Wieland Reich, Alexander Wiebel, and Gerik Scheuermann. Topology aware stream surfaces. *Computer Graphics Forum*, 29(3):1153–1161, 2010.

[28] Dominic Schneider, Alexander Wiebel, and Gerik Scheuermann. Smooth stream surfaces of fourth order precision. *Computer Graphics Forum*, 28(3):871–878, 2009.

[29] Maik Schulze, Janick Martinez Esturo, Tobias Günther, Christian Rössl, Hans-Peter Seidel, Tino Weinkauf, and Holger Theisel. Sets of globally optimal stream surfaces for flow visualization. *Computer Graphics Forum*, 33(3):1–10, 2014.

[30] JJ Trivedi and T Babadagli. Efficiency of diffusion controlled miscible displacement in fractured porous media. *Transport in Porous Media*, 71(3):379–394, 2008.

[31] W. von Funck, T. Weinkauf, H. Theisel, and H.-P. Seidel. Smoke surfaces: An interactive flow visualization technique inspired by real-world flow experiments. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1396–1403, 2008.

[32] Wolfram von Funck, Holger Theisel, and Hans-Peter Seidel. Vector field based shape deformations. *ACM Trans. Graph.*, 25(3):1118–1125, July 2006.

[33] Franz Waldner and Rainer Klages. Symmetric Jacobian for local Lyapunov exponents and Lyapunov stability analysis revisited. *Chaos, Solitons & Fractals*, 45(3):325–340, 2012.

[34] Yong-Liang Yang, Yi-Jun Yang, Helmut Pottmann, and Niloy J Mitra. Shape space exploration of constrained meshes. *ACM Trans. Graph.*, 30(6):124, 2011.