The Dependent Vectors Operator

Article *in* Computer Graphics Forum · July 2019

CITATIONS
0

READS
67

2 authors:

Lutz Hofmann
Universität Heidelberg
2 PUBLICATIONS **0** CITATIONS
SEE PROFILE

Filip Sadlo
Universität Heidelberg
86 PUBLICATIONS **1,057** CITATIONS
SEE PROFILE

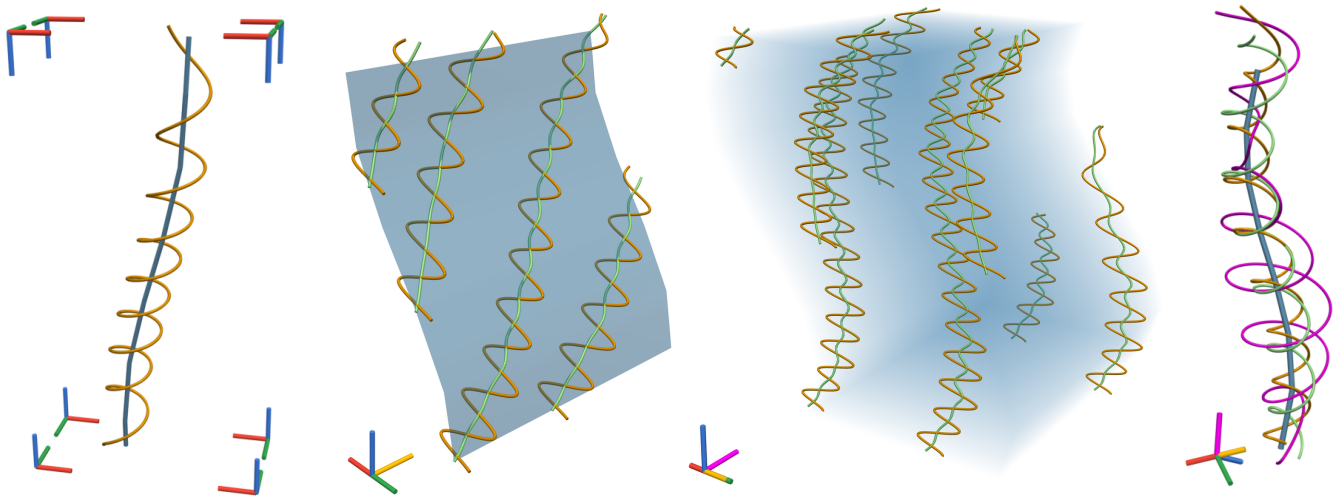
Eurographics Conference on Visualization (EuroVis) 2019
M. Gleicher, H. Leitte, and I. Viola
(Guest Editors)

Volume 38 (2019), Number 3

The Dependent Vectors Operator

Lutz Hofmann and Filip Sadlo

Heidelberg University, Germany



(a) (b) (c) (d)

Figure 1: *Vortex core manifolds (blue) in higher dimensions. 1-vortex core manifolds in 3D (a), 4D (b), and 5D (c), are one-, two-, and three-dimensional, while a 2-vortex core manifold in 5D is one-dimensional (d). The latter has two planes of rotation (orange and green streamlines), i.e., a streamline seeded in between them exhibits a rotation within two planes of rotations simultaneously (magenta streamline).*

Abstract

In this paper, we generalize the parallel vectors operator due to Peikert and Roth to arbitrary dimension, i.e., to four-dimensional fields and beyond. Whereas the original operator tested for parallelism of two (derived) 2D or 3D vector fields, we reformulate the concept in terms of linear dependency of sets of vector fields, and propose a generic technique to extract and filter the solution manifolds. We exemplify our approach for vortex cores, bifurcations, and ridges as well as valleys in higher dimensions.

CCS Concepts

• ***Human-centered computing*** → ***Visualization techniques;*** • ***Applied computing*** → *Mathematics and statistics;*

1. Introduction

One of the striking advantages of feature extraction is that it is often parameter-free. In that sense, features are (typically lower-dimensional) subsets of the domain that exhibit special characteristics, which do not depend on user-defined parameters. A wide range of feature definitions, along with respective extraction techniques, is in use in various fields, and for various applications.

For example, vortex core lines, which represent the "rotation axis" of a vortex, provide a concise representation of vortical flow fields, serve well for providing overview, and lend themselves as seeding structures for particle tracing. The interaction between flow and solids, on the other hand, can be effectively analyzed by means of separation lines and attachment lines. These curves, located on the surface of a solid, can indicate flow separation, for example, the onset of stall in the flow around the wing of an airplane. Finally, in the case of scalar fields, height ridges play an important role, for example, for extracting vessels in medical data, or to represent shape. All these feature types can be effectively defined and extracted by means of the parallel vectors (PV) operator due to Peik-

ert and Roth [PR99], a framework that enables the formulation of these features as sets of those points where two (derived) vector fields are parallel or antiparallel.

Many physical phenomena can be described by 2D or 3D time-independent vector fields and thus are suitable for feature extraction using the PV operator followed by representation in 2D or 3D space. However, there are phenomena that can only be appropriately modeled by higher-dimensional vector fields. Treating time as additional spatial dimension leads to 4D vector fields if the spatial domain is three-dimensional. The motion of inertial objects due to forces in 2D or 3D space is described by respectively 4D or 6D vector fields, called the phase space. Considering time as additional dimension in the phase space leads to 5D and 7D vector fields. In general, any phenomenon that is described by a continuous dynamical system with an n-dimensional phase space has an underlying n-dimensional vector field. Furthermore, higher-dimensional scalar fields can arise from families of 2D or 3D vector fields that depend on one or more additional parameters. For example, scalar fields that are derived from the flow map, which maps particles seeded at an initial time to their position after a certain advection time, add two parameters, initial time and advection time, to the spatial dimensions, and are thus 4D or 5D. Since the projection of higher-dimensional objects onto 3D space usually leads to self-intersection and is intrinsically harder to interpret, the extraction of meaningful features from these higher-dimensional spaces is even more important for their visualization. Features extracted from higher-dimensional vector fields and scalar fields have in common that one can no longer only consider line-type or codimension one features, such as extracted by the PV operator. For example, vortex core manifolds, generalized vortex core lines, are surfaces in 4D vector fields, and lines or volumes in 5D vector fields.

In this paper, we generalize the PV operator to fields with arbitrary dimension larger than one, leading to the dependent vectors (DV) operator. This operator extracts manifolds, where an n-dimensional vector field together with a set of k (derived) nD vector fields is linearly dependent. In the 2D and 3D cases, our DV operator is identical to the PV operator. We show its utility by demonstrating how it is used to define vortex core manifolds, bifurcation manifolds, and height ridge manifolds in higher dimensions. For the extraction of the respective DV solutions (i.e., the manifolds with arbitrary dimension), we present a generic, simple, and effective algorithm, and demonstrate it at these cases. With this work, we hope to pave the way for further research on feature extraction in higher dimensions, and for analysis in novel application domains.

Our contributions include:

- generalization of the PV operator to arbitrary dimension,
- a generic algorithm for extracting the resulting manifolds, and
- application of our approach to generalized vortex core manifolds, bifurcation manifolds, and ridges.

2. Related Work

The parallel vectors operator [PR99] is, together with the feature flow field [TS03, WTVGP11], one of the most successful frameworks for feature definition and feature extraction. As described in the original work by Peikert and Roth, the operator lends itself for the formulation, analysis, and extraction of vortex core lines [DSL90, SH95, RP98], ridge lines and valley lines [EGM*94, PS08], and separation lines and attachment lines [Ken98]. Machado et al. [MSE13] used the PV operator to extract bifurcation lines in 3D steady vector fields, and applied their method to the space-time domain of 2D flows for extracting hyperbolic trajectories [MBES16].

Works that go beyond direct application of the PV operator include the one by Ju et al. [JCWD14], who derive a parity test for the number of PV solutions, and, most closely related to our work, the coplanar vectors operator, defined by Weinkauf and Theisel [WSTH07] for the extraction of vortex core lines in time-dependent flow. Similar to their previous work [TSW*05], which derives a feature flow field formulation for representing the surfaces that a vortex core line sweeps over time, they also operate in space-time there. In that work, they extend the vortex core line concept due to Sujudi and Haimes [SH95] to four-dimensional space-time, i.e., they treat time as additional dimension, resulting in the requirement of three 4D vector fields being coplanar. This coplanarity requirement represents, in our generalization, the special case for $n = 4$ dimensions of the domain, and $k = 2$ dimensions of the resulting manifold. Notice, however, that due to the special properties of space-time vector fields, Weinkauf and Theisel did not need to solve for that manifold in 4D, but instead reformulated the problem as a parallel vectors problem in 3D, and employed the PV operator for the extraction of the respective vortex core lines. Fuchs et al. [FPH*08] also discussed the extension of PV features to unsteady flow. Van Gelder and Pang [VGP09] presented an approach to finding PV lines based on root-finding, which can be applied to arbitrary dimensions, as well as cases where the dimension of the vectors does not match that of the domain. Pagot et al. [POS*11] extended the PV operator to higher-order data. The case of vortices with vanishing longitudinal component was treated by Jung et al. [JHP*17] by transforming the PV problem into a ridge extraction problem. An extension of the PV operator to vector field ensembles was presented by Gerrits et al. [GRT18]. Oster et al. [ORT18a, ORT18b] extracted PV lines in 3D second-order tensor fields, where not only the 3D location but also the eigenvector itself is unknown, leading to a 5D search space. Günther and Theisel [GT18] computed PV lines in a 6D phase space of inertial dynamics, using a generalized cross product, which coincides with the case $n = 6$, $k = 1$ in our generalization.

Higher-dimensional scalar fields have been visualized by slicing [vWvL93] and by extracting isosurfaces [WB96, BWC04]. For visualizing the relationship between multivariate data and geometry, star coordinates [Kan00] and parallel coordinates [Ins85] have been used. Wegenkittl et al. employed parallel coordinates for visualizing trajectories in higher-dimensional dynamical systems [WLG97]. The wedge product has been used for comparing multiple scalar fields [EHM*08], and a generalization of Jacobi sets to arbitrary dimension and number of Morse functions [EH02] has enabled topological feature extraction and tracking in scalar fields [BBD*07]. The notion of a Reeb graph has been extended to time-varying scalar fields [EHNP04]. Extracting features from higher-dimensional spaces, Hofmann et al. [HRS18] used an orthographic 4D camera for visualizing topology and stream manifolds in steady 4D vector fields, and Wilde et al. [WRT18] extracted recirculation surfaces from 4D as well as 5D flow maps.

3. Fundamentals

We consider an n-dimensional vector field $\mathbf{u}(\mathbf{x})$ and a set of k vector fields $\mathbf{w}_1(\mathbf{x}),\dots,\mathbf{w}_k(\mathbf{x})$, all defined on the same domain $\Omega \subset \mathbb{R}^n$, with $\mathbf{u}(\mathbf{x}),\mathbf{w}_i(\mathbf{x}) \in \mathbb{R}^n$ (note that we often write, e.g., $\mathbf{u}$ for $\mathbf{u}(\mathbf{x})$ in our notation, were not ambiguous). For a fixed $\mathbf{x} \in \mathbb{R}^n$, we call the vectors $\mathbf{u}(\mathbf{x}),\mathbf{w}_1(\mathbf{x}),\dots,\mathbf{w}_k(\mathbf{x})$ linearly dependent, if there exist scalars $c_1,\dots,c_k \in \mathbb{R}$ such that

$$\mathbf{u}(\mathbf{x}) = c_1\mathbf{w}_1(\mathbf{x}) + \cdots + c_k\mathbf{w}_k(\mathbf{x})\,. \tag{1}$$

Here, we assume that the vectors $\mathbf{w}_1(\mathbf{x}),\dots,\mathbf{w}_k(\mathbf{x})$ are linearly independent and none of the vectors is zero. This notion directly generalizes that of two vectors being parallel, by which we mean that there is a $c \in \mathbb{R}$, such that $\mathbf{u}(\mathbf{x}) = c\mathbf{w}(\mathbf{x})$. Equation 1 is a set of n equations with unknowns $\mathbf{x} \in \mathbb{R}^n$, and $c_1,\dots,c_k \in \mathbb{R}$, such that its solutions are k-dimensional manifolds. We define the dependent vectors operator as the operator, that, given nD vector fields $\mathbf{u},\mathbf{w}_1,\dots,\mathbf{w}_k$, returns the set $\mathcal{D}$ of solution points of Equation 1.

3.1. Wedge Product

Two three-dimensional vectors $\mathbf{u},\mathbf{w} \in \mathbb{R}^3$ are parallel if and only if their cross product $\mathbf{u} \times \mathbf{w}$ is the zero vector. The appropriate notion of such a cross product in arbitrary dimension and for an arbitrary number of vectors is that of the *wedge product*, which we introduce in the following. Just like the norm of the 3D cross product coincides with the area of the parallelogram spanned by the two vectors, the norm of the wedge product $\mathbf{u} \wedge \mathbf{w}_1 \wedge \cdots \wedge \mathbf{w}_k$ of the $k+1$ vectors $\mathbf{u},\mathbf{w}_1,\dots,\mathbf{w}_k \in \mathbb{R}^n$ coincides with the (k+1)-dimensional volume spanned in n-dimensional space. As such, the wedge product is zero if and only if the vectors $\mathbf{u},\mathbf{w}_1,\dots,\mathbf{w}_k$ are linearly dependent, i.e., they lie in a linear subspace of dimension lower than k+1. Note, however, that two non-parallel four-dimensional vectors $\mathbf{u},\mathbf{w} \in \mathbb{R}^4$ exhibit a full two-dimensional linear subspace of vectors that are orthogonal to $\mathbf{u}$ and $\mathbf{w}$. Therefore, the wedge product of n-dimensional vectors will in general no longer be again an n-dimensional vector.

If the vectors $\mathbf{u},\mathbf{w}_1,\dots,\mathbf{w}_k \in \mathbb{R}^n$ with $k+1 < n$ are linearly independent, i.e., Equation 1 has no solution, they can be extended to a set of n linearly independent vectors by "filling in" appropriately chosen vectors from a basis of $\mathbb{R}^n$, such as the standard basis $\mathcal{E} := \{\mathbf{e}_1,\dots,\mathbf{e}_n\}$. If a choice of $m = n-k-1$ vectors $\mathbf{e}_{i_1},\dots,\mathbf{e}_{i_m}$ does not extend the vectors to a basis, we have

$$\det(\mathbf{u},\mathbf{w}_1,\dots,\mathbf{w}_k,\mathbf{e}_{i_1},\dots,\mathbf{e}_{i_m}) = 0\,. \tag{2}$$

The vectors $\mathbf{u},\mathbf{w}_1,\dots,\mathbf{w}_k$ are thus linearly dependent, if and only if for all choices $i_1,\dots,i_m \in \{1,\dots,n\}$, Equation 2 holds. By symmetry, it suffices to consider the $l := \binom{n}{k}$ choices $1 \le i_1 < \cdots < i_m \le n$. While usually the wedge product is defined as an antisymmetric order-m tensor consisting of terms of Equation 2 indexed by $i_1,\dots,i_m$, we are only interested in its l degrees of freedom. Therefore, we choose coordinates in $\mathbb{R}^l$ by enumerating the subsets $\mathcal{E}_1,\dots,\mathcal{E}_l \subseteq \mathcal{E}$ of the standard basis lexicographically, and define

$$\mathbf{u} \wedge \mathbf{w}_1 \wedge \cdots \wedge \mathbf{w}_k := (\det(\mathbf{u},\mathbf{w}_1,\dots,\mathbf{w}_k,\mathcal{E}_i))_{i=1}^{l} \in \mathbb{R}^l \tag{3}$$

as the wedge product of the vectors $\mathbf{u},\mathbf{w}_1,\dots,\mathbf{w}_k$, i.e., our wedge product is an l-dimensional vector. The wedge product is zero whenever the vectors are linearly dependent, and we will treat this choice of coordinates as the definition of the wedge product for the remainder of this paper.

Note, that we deliberately choose this definition, such that it is consistent with the three-dimensional case, which we aim to generalize. By definition, for two 3D vectors $\mathbf{u},\mathbf{w} \in \mathbb{R}^3$, we have

$$\mathbf{u} \wedge \mathbf{w} = \begin{pmatrix} \det(\mathbf{u},\mathbf{w},\mathbf{e}_1) \\ \det(\mathbf{u},\mathbf{w},\mathbf{e}_2) \\ \det(\mathbf{u},\mathbf{w},\mathbf{e}_3) \end{pmatrix} = \mathbf{u} \times \mathbf{w}\,, \tag{4}$$

i.e., the wedge product of two 3D vectors is identical to their 3D cross product. It can thus be seen, that the choice of a basis $\mathcal{E}$ is independent of the vector fields $\mathbf{u},\mathbf{w}_1,\dots,\mathbf{w}_k$. Furthermore, in the cases $k = n-2$, the wedge product maps a set of $n-1$ n-dimensional vectors to an n-dimensional vector, which is orthogonal to each of them, thus providing a generalized cross product. In the cases $k = n-1$, the wedge product is scalar-valued, with $\mathbf{u} \wedge \mathbf{w}_1 \wedge \cdots \wedge \mathbf{w}_{n-1} = \det(\mathbf{u},\mathbf{w}_1,\dots,\mathbf{w}_{n-1})$.

Using the notion of the wedge product, the solution set $\mathcal{D}$ of the dependent vectors operator can be written as

$$\mathcal{D} = \{\mathbf{x} \in \mathbb{R}^n \mid \mathbf{u}(\mathbf{x}) \wedge \mathbf{w}_1(\mathbf{x}) \wedge \cdots \wedge \mathbf{w}_k(\mathbf{x}) = \mathbf{0}\}\,, \tag{5}$$

keeping in mind, that the vectors $\mathbf{w}_1(\mathbf{x}),\dots,\mathbf{w}_k(\mathbf{x})$ are assumed to be linearly independent. The set $\mathcal{D}$ is thus the intersection of l zero-level sets of scalar functions, each of which are closed, and as such is closed as well.

3.2. Height Ridge Manifolds

A k-dimensional height ridge of an nD scalar field f is the set of points, where f has a local maximum in $n-k$ directions. According to Eberly et al. [EGM*94], these are the points $\mathbf{x}$, where

$$\mathbf{y}_1 \cdot \nabla f = \cdots = \mathbf{y}_{n-k} \cdot \nabla f = 0\,, \tag{6}$$

$$\lambda_1,\dots,\lambda_{n-k} < 0\,, \tag{7}$$

where $\mathbf{y}_1,\dots,\mathbf{y}_n$ denote the eigenvectors of the Hessian $\nabla\nabla f$ at $\mathbf{x}$, with respective real eigenvalues $\lambda_1 \le \cdots \le \lambda_n$.

In other words, the definition requires that the $n-k$ smallest eigenvalues are negative, and that the respective second directional derivatives are negative, i.e., that the directional derivatives of f along the corresponding eigenvectors are zero. Valleys, the opposite of ridges, are obtained by height ridge extraction from $-f$.

The condition of Equation 6 is equivalent to requiring, that the gradient ∇f lies in the linear subspace spanned by the eigenvectors $\mathbf{y}_{n-k+1},\dots,\mathbf{y}_n$. That is, for a fixed $\mathbf{x} \in \mathbb{R}^n$, there exist $c_{n-k+1},\dots,c_n \in \mathbb{R}$, such that

$$\nabla f(\mathbf{x}) = c_{n-k+1}\mathbf{y}_{n-k+1}(\mathbf{x}) + \cdots + c_n\mathbf{y}_n(\mathbf{x})\,. \tag{8}$$

Thus, height ridges can be expressed using the dependent vectors operator with $\mathbf{u} = \nabla f$ and $\mathbf{w}_i = \mathbf{y}_{n-k+i}$, $i = 1,\dots,k$, where the solution set $\mathcal{D}$ is subsequently filtered according to Equation 7.

3.3. Vortex Core Manifolds

By the formulation of Sujudi and Haimes [SH95], a vortex core line in a 3D vector field is the set of points, where the flow direction $\mathbf{u}$ is governed by the real eigenvector of the Jacobian $\nabla\mathbf{u}$, while

the remaining two eigenvectors are complex-valued. The real and imaginary parts of the pair of complex-conjugate eigenvectors span a two-dimensional plane of rotation in 3D space, within which the reduced flow $\mathbf{u} - \mathbf{u}(\mathbf{x})$ rotates. The Sujudi–Haimes criterion can be reformulated using the PV operator by requiring that $\mathbf{u}$ is parallel to the real eigenvector and $\nabla\mathbf{u}$ has a pair of complex eigenvalues.

A rotation in n-dimensional space is defined by one or more two-dimensional planes of rotation. Note, that a plane of rotation in n-space leaves an $(n$-2)-dimensional subspace invariant, which happens to be one-dimensional in 3D, and therefore the notion of a *rotation axis* is in general not available in higher dimensions. For $n \geq 4$, there exist rotations which simultaneously rotate in more than one plane of rotation. We call such rotations *r-rotations*, where r is the number of simultaneous rotation planes. The case $r = 1$ is also called a *simple rotation*.

Accordingly, an *r-vortex core manifold* in an n-dimensional vector field is a $(n$-2$r)$-manifold, within which the flow is mainly governed by its $k = n - 2r$ non-rotational directions. Following the 3D formulation, points that belong to such a manifold are defined by the flow direction $\mathbf{u}$ lying in the linear space spanned by the k real eigenvectors of $\nabla\mathbf{u}$. Thus, r-vortex core manifolds can be reformulated using the DV operator by defining $\mathbf{w}_1, \ldots, \mathbf{w}_k$ to be the k real eigenvectors of $\nabla\mathbf{u}$. Regions that do not have exactly k *real* eigenvectors are omitted (filtered).

Since their definition depends on two-dimensional planes of rotation, different kinds of vortex core manifolds exist, depending on the dimension n of the surrounding space. When n is even, $(n/2)$-vortex core manifolds are zero-dimensional and correspond to critical points of $\mathbf{u}$. Furthermore, vortex core manifolds need to be of even dimension. Vice versa, if n is odd, all vortex core manifolds need to be of odd dimension. For example, in 4-space, there are zero-dimensional 2-vortex core manifolds, and 2-dimensional 1-vortex core manifolds, but no 1-dimensional vortex core lines. Vortex core lines exist, e.g., in 5-space as 2-vortex core manifolds.

3.4. Bifurcation Manifolds

For 3D flows, Perry and Chong [PC87] proposed the notion of *bifurcation lines*, which are streamlines that exhibit an attracting and a repelling two-dimensional manifold of streamlines. Streamlines on the attracting manifold locally converge toward the bifurcation line, while streamlines on the repelling manifold converge to the bifurcation line in reverse time. Bifurcation lines thus locally separate the flow. According to Roth [Rot00], bifurcation lines may be obtained by a modification of the Sujudi–Haimes criterion, where instead of regions of the flow, that have a complex plane of rotation, one seeks those regions, that have only real eigenvalues, and their signs are alternating.

To generalize this notion to arbitrary dimensions, we require, that the Jacobian $\nabla\mathbf{u}$ has real eigenvalues $\lambda_1 \leq \cdots \leq \lambda_n$. A point $\mathbf{x}$ lies on a *bifurcation manifold*, if the flow $\mathbf{u}(\mathbf{x})$ lies in the $(n$-2)-dimensional space spanned by the eigenvectors $\mathbf{y}_2, \ldots, \mathbf{y}_{n-1}$ belonging to the medium eigenvalues and $\lambda_1\lambda_n < 0$. Such a bifurcation manifold then exhibits attracting behavior in direction of the minor eigenvector $\mathbf{y}_1$, and repelling behavior in direction of the major eigenvector $\mathbf{y}_n$. Both eigenvectors define $(n$-1)-dimensional manifolds of streamlines, since integration adds one dimension to the $(n$-2)-dimensional bifurcation manifold. Similarly to vortex core manifolds, bifurcation manifolds can be obtained by the DV operator by rejecting regions of the raw solutions where the Jacobian has complex eigenvalues.

4. Manifold Extraction

We compute the manifolds of linear dependency by triangulating the regions where $\mathbf{u} \wedge \mathbf{w}_1 \wedge \cdots \wedge \mathbf{w}_k = \mathbf{0}$. In those cases, where such solution manifolds have codimension one, i.e., in the cases $k = n - 1$, they could be obtained as contours (remember that the wedge product is a scalar in such cases), using an n-dimensional variant of the marching cubes algorithm [BWC04, LC87]. On the other hand, if $k = 1$ and $n = 3$, one could use the parallel vectors extraction scheme [PR99]. However, in the remaining cases, neither of these approaches is applicable.

Instead of using a combination of these approaches, we propose to follow a generic approach that works for any n and k, inspired by the principles of marching cubes and parallel vectors extraction. We assume that the vector fields $\mathbf{u}, \mathbf{w}_1, \ldots, \mathbf{w}_k$ are given on a common n-dimensional rectilinear grid, with tensor-product multilinear interpolation. Our algorithm consists of four main steps:

1. compute solution points on $(n$-$k)$-faces of the grid (Sec. 4.1),
2. filter solution points (Sec. 4.3),
3. triangulate the solutions in each cell of the grid (Sec. 4.2), and
4. filter resulting k-simplices and connected components (Sec. 4.3).

Except for the computation of connected components, all steps are local within a cell or a cell face, and can thus be trivially parallelized. Also, only the first step needs to traverse all cells, while the subsequent steps depend on the size of the solution set. In most applications, the size of the solution set is much smaller than the number of cells.

The first step, which is the main computational bottleneck, can be sped up in some cases by rejecting those cell faces, where all of the nodes do not adhere to a certain criterion. This can always be employed in the cases of vortex core manifolds and bifurcation manifolds, where a specific number of complex eigenvalues is required. It can further be applied when extracting ridge manifolds, where the cell faces can be filtered by a scalar range in cases, where only ridges within a certain scalar range are of interest (see Section 5.3 for an example).

We restrict our algorithm to rectilinear grids, because its extension to arbitrary dimension is straightforward. In order to adapt the algorithm to unstructured grids, analytical derivatives of the interpolation functions within each cell, as well as the traversal of all $(n$-$k)$-dimensional cell faces would be needed.

4.1. Computation of Solution Points

Since the solution set $\mathcal{D}$ is k-dimensional, its intersection with the $(n$-$k)$-faces of the grid, i.e, the $(n$-$k)$-dimensional cell faces (1-dimensional faces being the cell edges), is a set of isolated points. For example, in 4D space, a 2D manifold intersects a 2D face in at most a single point (see Fig. 2j–2l). We obtain the solution points by minimizing $\|\mathbf{u} \wedge \mathbf{w}_1 \wedge \cdots \wedge \mathbf{w}_k\|$ using Gauss–Newton iteration.

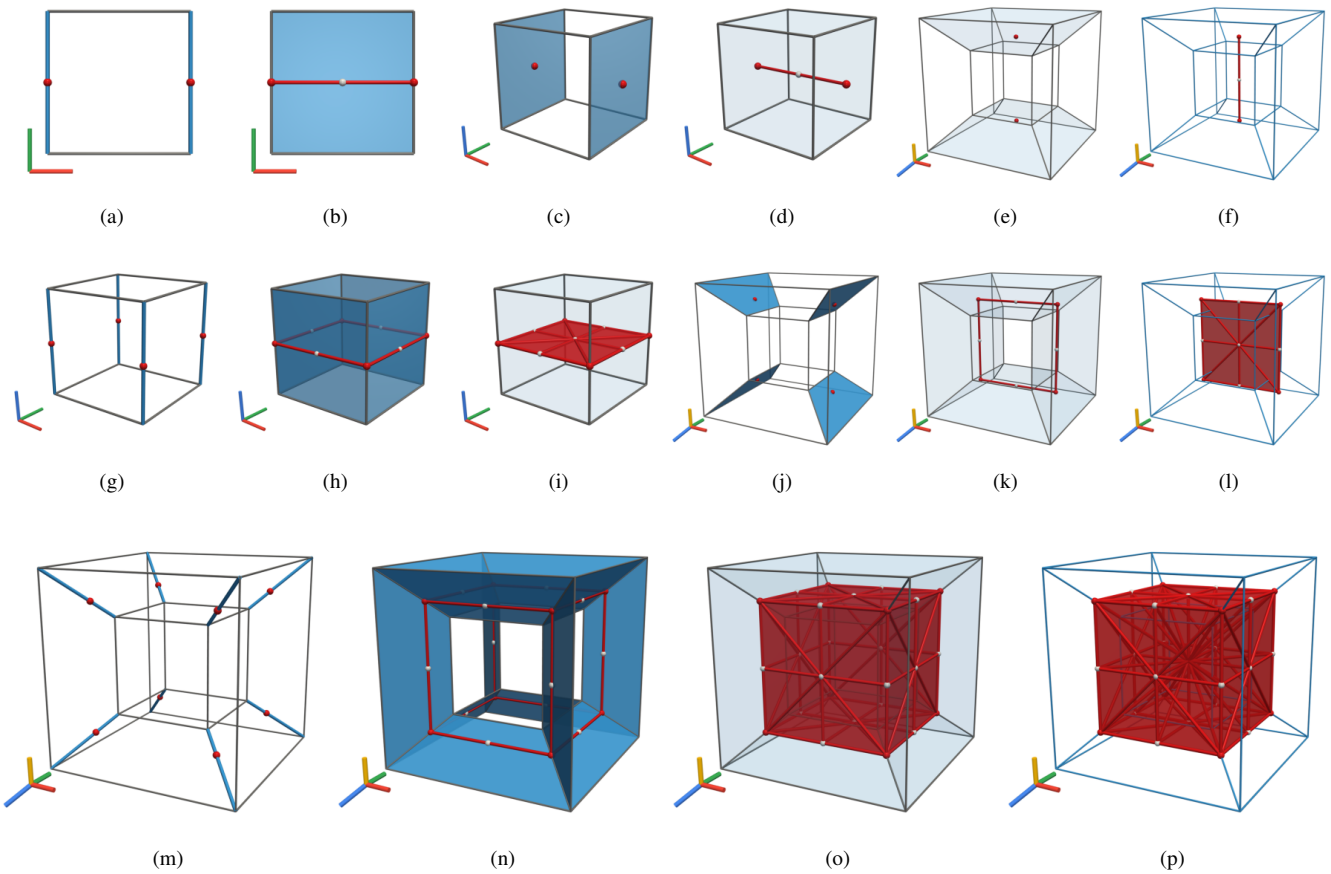


Figure 2: *Triangulation of dependent vectors solution manifolds. Case $k = 1$ in 2D (a)–(b), 3D (c)–(d), and 4D (e)–(f) results in lines, case $k = 2$ in 3D (g)–(i) and 4D (j)–(l) results in surfaces, and case $k = 3$ in 4D (m)–(p) results in volumes. Points (red dots) on codimension k faces (blue) are connected to lines (red lines), and further to triangles and tetrahedra (in the cases $k = 2$ and $k = 3$, respectively), by inserting points at the centers of masses (white dots) of the previous steps. The 4D cases are shown in 3D perspective projection. Currently active faces in each step are marked blue, with lines for 1-faces, quads for 2-faces, transparent cubes for 3-faces, and lines (wireframe) for 4-faces.*

In a face's local coordinates $\boldsymbol{\xi} := (\xi_1, \ldots, \xi_{n-k})^\top$, we choose the initial guess $\boldsymbol{\xi}^{(0)}$ as the center of the cell face, and in each iteration, we update the local coordinates by $\boldsymbol{\xi}^{(i+1)} = \boldsymbol{\xi}^{(i)} - \Delta$, where

$$\Delta = \left(\mathbf{J}^\top \mathbf{J}\right)^{-1} \mathbf{J}^\top \left(\mathbf{u}(\boldsymbol{\xi}^{(i)}) \wedge \mathbf{w}_1(\boldsymbol{\xi}^{(i)}) \wedge \cdots \wedge \mathbf{w}_k(\boldsymbol{\xi}^{(i)})\right), \qquad (9)$$

and where the $l \times (n-k)$ matrix $\mathbf{J}$ is the Jacobian of Equation 3 in local face coordinates. By applying the product rule of differentiation to the Leibniz formula for determinants, it is obtained by differentiating each parameter of the determinant separately, and taking their sum:

$$\mathbf{J}_{ij} = \det\left(\frac{\partial \mathbf{u}}{\partial \xi_j}, \mathbf{w}_1, \ldots, \mathbf{w}_k, \mathcal{E}_i\right) + \cdots + \det\left(\mathbf{u}, \mathbf{w}_1, \ldots, \frac{\partial \mathbf{w}_k}{\partial \xi_j}, \mathcal{E}_i\right).$$

Since we employ multilinear interpolation within each cell face, we obtain these derivatives by analytic differentiation of the interpolation terms. These derivatives are not necessarily continuous across face boundaries, but are only used to locate solutions within the derived fields. After a maximum number of N iterations, we reject any solution, where

$$\|\Delta\| > \vartheta_\Delta \quad \text{or} \quad \left\|\mathbf{u}(\boldsymbol{\xi}^{(N)}) \wedge \mathbf{w}_1(\boldsymbol{\xi}^{(N)}) \wedge \cdots \wedge \mathbf{w}_k(\boldsymbol{\xi}^{(N)})\right\| > \vartheta_\wedge,$$

or if $\boldsymbol{\xi}^{(N)}$ lies outside of the cell face. The threshold ϑ_Δ serves mainly monitoring purposes, i.e., if convergence has not been achieved with respect to $\vartheta_\wedge$ but ϑ_Δ is met, increasing N might improve the result, or the other way round, if $\|\vartheta_\Delta\|$ is too large, this can indicate that the result will not sufficiently converge even with more steps. Using only one initial guess, we may obtain at most one solution point in each cell face, i.e., too high data variation may lead to faces containing more than one solution point. We only compute one solution point per cell face, and leave more elaborate solution methods for future work.

Eigenvector Fields. If the vector fields $\mathbf{w}_1, \ldots, \mathbf{w}_k$ are eigenvectors of a tensor field, such as the Jacobian of a vector field or the Hessian of a scalar field, they need to be sorted and oriented in order to obtain smoothly varying vector fields. In order to consistently orient the eigenvector fields within a cell face, we perform principal

component analysis (PCA) of the vectors, and orient them according to the direction of the major component [FP01].

Enumeration of Cell Faces. To be able to traverse and store all solution points, we need a scheme for enumerating all (n-k)-faces of the n-dimensional grid. We start with an enumeration of the grid nodes, which we choose to be in scanline order. For each cell, we take its "lower left" node (i.e., the cell's node with lowest coordinate in all dimensions) as a reference. We then identify each cell's face that includes this node by the (n-k)-dimensional subspace it is embedded in, i.e., by the subset of the standard basis that spans that space. These subspaces correspond to choices of $n-k$ basis vectors, of which there are $l = \binom{n}{k}$ many. In total, the global ID of a face is thus given by the global scanline ID of the node and the type $t \in \{1,\ldots,l\}$ of the face that shares this node. Notice, how this corresponds to the definition of the wedge product, for example, the types of 2-dimensional faces of a 3D grid conceptually correspond to the wedge products $\mathbf{e}_1 \wedge \mathbf{e}_2$, $\mathbf{e}_1 \wedge \mathbf{e}_3$, and $\mathbf{e}_2 \wedge \mathbf{e}_3$.

4.2. Triangulation

The extracted solution points are triangulated iteratively. Starting from the solution points (zero-dimensional simplices) on the (n-k)-faces, each step connects lower-dimensional simplices, that share the same cell face of one dimension higher, to simplices of one dimension higher. Thus, in the i-th step, we obtain i-dimensional simplices on (n-k+i)-dimensional cell faces. For each (n-k+i)-face, the center of mass of the solution points belonging to the cell face is inserted and connected to all vertices of the (i-1)-simplices from the previous iteration, creating i-simplices. After k steps, the triangulation terminates, resulting in a set of k-simplices defined within the n-dimensional cells.

Six cases of this generic triangulation approach are illustrated in Figure 2, which we further discuss in the following:

- $k = 1$: $n = 2$ (Fig. 2a,2b), $n = 3$ (Fig. 2c,2d), $n = 4$ (Fig. 2e,2f),
- $k = 2$: $n = 3$ (Fig. 2g–2i), $n = 4$ (Fig. 2j–2l), and
- $k = 3$: $n = 4$ (Fig. 2m–2p).

For $k = 1$, it is apparent that the intersection of $\mathcal{D}$ with a cell needs to represent points on the boundary of the cell, and that these need to be subsequently connected to a line. In 2D, the cell's boundary consists of 1-faces (Figure 2a). In 3D, it consists of 2-faces (Figure 2c), and in 4D, it consists of 3-faces (the faces of a 4D cube are 3D cubes, Figure 2e). Our algorithm detects those points, which we denote solution points, and connects them into lines, or in other words, into 1-simplices. We do not connect the points directly, but compute their center of mass (white points in Figure 2) and connect that point with each solution point, generating a 1-simplex for each solution point. This allows for more than two solution points (which are then connected to each other), and can be further applied to the construction of higher-dimensional simplices.

For $k = 2$, the intersection between $\mathcal{D}$ and a 3D cell has to represent points on 1-faces of the 3D cell (Figure 2g). However, we identify on the 2-faces (Figure 2h) of that cell the configuration from $k = 1$ in 2D (Figure 2a). We exploit this analogy by treating the faces of a 3D cube accordingly, i.e., connecting the points into 1-simplices (Figure 2h). Nevertheless, we are not done yet—to obtain a k-manifold, we connect each 1-simplex with the center of mass of all solution points, resulting in a set of 2-simplices (triangles, Figure 2i). In 4D, the intersection between $\mathcal{D}$ and the 4D cell are points on 2-faces of the 4D cell (Figure 2j). Here, we identify on the 3-faces of that cell the configuration from $k = 1$ in 3D (Figure 2c). We exploit this analogy by treating these faces accordingly, i.e., connecting the points into 1-simplices (Figure 2k). Again, we are not done yet—to obtain a k-manifold, we connect each 1-simplex with the center of mass of all solution points, resulting in a set of 2-simplices (Figure 2l).

For $k = 3$, the intersection between $\mathcal{D}$ and a 4D cell has to represent points on 1-faces of the 4D cell (Figure 2m). Here, we identify on the 3-faces of that cell the configuration from $k = 2$ in 3D (Figure 2g). We exploit this analogy by treating this face accordingly, i.e., connecting the points into 1-simplices (Figure 2n) and those to a set of 2-simplices (Figure 2o). And again, as it was in the case before, we are not done yet—to obtain a k-manifold, we connect each 2-simplex with the center of mass of all solution points, resulting in a set of 3-simplices (tetrahedra, Figure 2p).

We observe the following scheme:

- The number of steps in our algorithm depends only on k, i.e., it is independent of n. That is, for $k = 1$ we needed two steps, for $k = 2$ we needed three steps, and for $k = 3$ four steps. This increase is caused by its recursive nature, and the need for an additional step to finally establish k-manifolds.
- The extraction of the manifolds can be achieved in a recursive manner: if $k = 1$, the solutions points are obtained on the faces of the cell and connected into 1-simplices. Otherwise, an nD cube is handled by treating each of its (n-1)-faces independently as an (n-1)D cube, with a subsequent step to combine the intermediate result to the higher-dimensional simplices.

4.3. Filtering

Filtering takes place in two stages of our algorithm: Criteria with a pointwise definition (such as feature strength), are applied at the solution-point level. Criteria that imply connectivity, i.e., that are defined on simplex level (such as feature quality and feature size), are applied after triangulation.

As we have seen in Sections 3.2 and 3.3, applications of the DV operator typically require application-dependent filtering, such as requiring the $n-k$ respective eigenvalues of the Hessian being negative in case of ridge manifolds, or requiring the $n-k$ remaining eigenvalues of the velocity gradient to be complex in the case of vortex core manifolds. These filtering criteria are part of the feature definition, and therefore, typically do not require adjustment by the user. Nevertheless, it is common practice [PR99] to employ optional filtering by means of these criteria, denoted *feature strength*, as well as by additional measures, such as *feature quality*, and *feature size*.

Feature Strength. As mentioned in Section 3.2, $\lambda_1,\ldots,\lambda_{n-k}$ represent the second directional derivatives across the ridge manifold. In other words, the more negative they are, the more the respective profile corresponds to a peak. On the other hand, if these eigenvalues are too close to zero, ridge extraction will not be robust, as

they correspond to "too flat" ridges. Thus, we provide the option of imposing a (nonnegative) threshold ϑ_r on the eigenvalues of the Hessian, i.e., we omit manifold regions where

$$-\lambda_{n-k} \leq \vartheta_r . \tag{10}$$

Setting $\vartheta_r = 0$ results in the unconstrained solution (Equation 7).

In case of vortex core manifolds, we follow the same idea, but now filter with respect to the weakest imaginary part, since imaginary parts relate to rotational strength. Let $\lambda_1, \ldots, \lambda_k$ be the real eigenvalues, we thus omit, with the (nonnegative) threshold ϑ_v, manifold regions where

$$\min_{i=k+1,\ldots,n} |\mathrm{Im}(\lambda_i)| \leq \vartheta_v . \tag{11}$$

Again, setting $\vartheta_v = 0$ results in the unconstrained solution.

Bifurcation manifolds are defined by their locally separating behavior, which requires both the minor eigenvalue λ_1 and the major eigenvalue λ_n to be large at the same time. They also need to be of opposite sign, such that we omit regions, where

$$-\lambda_1 \lambda_n \leq \vartheta_s , \tag{12}$$

and thus exhibit weak separating behavior, or are not bifurcation manifolds at all. Setting the nonnegative threshold $\vartheta_s = 0$ results in the unconstrained solution, since this only forces the signs of the eigenvalues to be alternating.

Feature Quality. Both ridge manifolds and vortex manifolds are, in case of perfect solutions, tangent to $\mathbf{u}$ [PR99]. That is, the manifolds are at the same time integral manifolds of $\mathbf{u}$, e.g., streamsurfaces in nD in case of 2-manifolds. However, local extraction methods, such as our dependent vectors operator, or the PV operator, cannot enforce such global constraints. Peikert and Roth [PR99], for the 3D case, therefore quantify feature quality by the angle between the tangent of the feature line and $\mathbf{u}$.

In our general case, we measure the angle between the n-dimensional vector $\mathbf{u}$ and the k-dimensional feature tangent space using the generalized angle between linear subspaces defined by Gunawan et al. [GNSB05]. For the feature tangent $T = \langle t_1, \ldots, t_k \rangle$, the angle θ between T and $\mathbf{u}$ is given by

$$\cos^2 \theta = \|\pi_T(\mathbf{u})\|^2 / \|\mathbf{u}\|^2 , \tag{13}$$

i.e., the ratio of the lengths of the projection $\pi_T(\mathbf{u})$ of $\mathbf{u}$ onto T, and $\mathbf{u}$ itself. Gunawan et al. have shown, that this definition reduces to the usual angles between two vectors if $k = 1$, and between a vector and a hyperplane if $k = n - 1$. For the remaining cases, this is the largest angle between $\mathbf{u}$ and all possible vectors in T.

For each vertex in the k-dimensional triangulation obtained by our algorithm, we compute its tangent space by PCA of the relative positions of the neighborhood of adjacent vertices in the triangulation. The radius of this neighborhood determines the smoothness of the tangent space across the manifold. The first k components obtained by the PCA are an approximation of the tangent space, and are used to compute the angle to $\mathbf{u}$.

Feature Size. Typically applied last in the filtering stage, one might want to suppress erroneous, small manifolds, that are caused, e.g., by (numerical) noise. To that end, we compute the connected components of the simplicial complex using depth-first search. For each connected component, we sum over the volume of each simplex. The volume $\mu(S)$ of a k-simplex $S = \{\mathbf{v}_0, \ldots, \mathbf{v}_k\}$ is given by the *Cayley-Menger determinant*. It is obtained by computing the squared distances of its vertices, $d_{ij} = \|\mathbf{v}_i - \mathbf{v}_j\|^2$, and finally

$$\mu(S)^2 = \frac{(-1)^{k+1}}{2^k (k!)^2} \det \begin{bmatrix} 0 & 1 & \ldots & 1 \\ 1 & d_{00} & \ldots & d_{0k} \\ \vdots & \vdots & & \vdots \\ 1 & d_{k0} & \ldots & d_{kk} \end{bmatrix} . \tag{14}$$

Components, whose volume is below the nonnegative threshold ϑ_μ, are discarded. Here, $\vartheta_\mu = 0$ provides the unconstrained result.

5. Results

We apply the DV operator to several datasets and for several applications. First, we consider a set of synthetic vortical fields, in order to build intuition to the generalized notion of n-dimensional vortices, and second, we extract features from the 4D phase space of a double pendulum. Finally, we apply our algorithm for several features in 3D time-dependent flow simulations, where time is treated as fourth dimension, such that streamlines in these 4D space-time vector fields correspond to pathlines. All involved streamlines were computed using fourth-order Runge–Kutta integration. When applying the DV operator, all datasets use $N = 10$ as the maximum number of Gauss–Newton iterations, with tolerances $\vartheta_\Delta = \vartheta_\wedge = 10^{-11}$. The cases of the DV operator, from our experiments, are listed in Table 1. The complexity of the datasets, together with performance measurements, are summarized in Table 2.

5.1. n-Dimensional Vortex Core Models

We exemplify the notion of n-dimensional vortex core manifolds by considering synthetic fields, which model n-dimensional vortices based on rigid body rotations. The velocity field is given by

$$\mathbf{u}(\mathbf{x}) = \nabla \boldsymbol{\phi}(\mathbf{x}) \left(\mathbf{W}(\boldsymbol{\phi}^{-1}(\mathbf{x})) \mathbf{v}(\boldsymbol{\phi}^{-1}(\mathbf{x})) \right) , \tag{15}$$

where $\boldsymbol{\phi}$ transforms physical coordinates $\mathbf{x} = (x_1, \ldots, x_n)^\top$ into computational coordinates $\boldsymbol{\xi} = (\xi_1, \ldots, \xi_n)^\top$, $\mathbf{v}(\boldsymbol{\xi})$ describes a rigid

Table 1: *Applied cases (k derived vector fields of dimension n) of the DV operator and extracted features of the datasets in Section 5.*

Dataset	n	k	Feature
1-Vortex Core (3D)	3	1	Vortex Core Line
1-Vortex Core (4D)	4	2	Vortex Core Surface
1-Vortex Core (5D)	5	3	Vortex Core Volume
2-Vortex Core (5D)	5	1	2-Vortex Core Line
Double Pendulum	4	2	Vortex Core Surface
Double Pendulum	4	2	Bifurcation Surface
Double Gyre	4	2	Valley Surface
Vortex Street	4	2	Vortex Core Surface
Convective Flow	4	1	Valley Line
Convective Flow	4	2	Vortex Core Surface

body rotation, and $\mathbf{W}$ interpolates between the identity and a matrix which transforms the rotation into a hyperbolic vector field.

The linear rigid body rotation is given by the linear field

$$\mathbf{v}(\boldsymbol{\xi}) = (-\omega_1\xi_2, \omega_1\xi_1, \ldots, -\omega_r\xi_{2r}, \omega_r\xi_{2r-1}, \nu_1\xi_{n-k}, \ldots, \nu_k\xi_n)^\top,$$

with $2r+k=n$, consisting of r planes of rotation with respective angular velocities $\omega_1, \ldots, \omega_r$ in the first $2r$ components, and k non-rotating components with linear coefficients $\nu_1, \ldots, \nu_k$. The transformation $\mathbf{W}(\boldsymbol{\xi})$ interpolates between the identity function, and a function that swaps the components that belong to rotations (and thus transforming the rotation to a hyperbolic field), according to a window function $w(\boldsymbol{\xi})$, i.e.,

$$\mathbf{W}(\boldsymbol{\xi}) = w(\boldsymbol{\xi})\boldsymbol{\xi} + (1-w(\boldsymbol{\xi}))\,(\xi_2, \xi_1, \ldots, \xi_{2r}, \xi_{2r-1}, \xi_{n-k}, \ldots, \xi_n)^\top,$$

with

$$w(\boldsymbol{\xi}) = \sigma\left(\left|\frac{\xi_{n-k}}{R_1}\right|^p + \cdots + \left|\frac{\xi_n}{R_k}\right|^p - 1\right), \quad \sigma(x) = \frac{1}{1+e^{-qx}},$$

which, for $p, q \to \infty$, takes the value one on the k-dimensional hypercube $\mathbb{R}^{n-k} \times [-R_1, R_1] \times \cdots \times [-R_k, R_k] \subset \mathbb{R}^n$, and zero elsewhere. Choosing sufficiently large $p, q > 1$, makes the function vary smoothly, while having a sharp gradient at the boundaries of the hypercube, i.e., the resulting vector field $\mathbf{u}$ smoothly transitions between a linear saddle field and a linear rigid body rotation, such that we expect to find a k-dimensional r-vortex core manifold within the hypercube described by $w(\boldsymbol{\xi})$.

We employ the coordinate transform $\boldsymbol{\phi}(\mathbf{x}) = \mathbf{x} + \cos(x_3)\mathbf{e}_1$ for the 1-vortex core cases and $\boldsymbol{\phi}(\mathbf{x}) = \mathbf{x} + \cos(x_5)\mathbf{e}_1$ for the 2-vortex core case. This causes the vortex core manifolds to bend along a sinusoidal path in x_1-direction. Further choosing $r=1$, i.e., one plane of rotation, we obtain a 1-vortex core manifold, which is a line in 3D, a surface in 4D, and a volume in 5D (Figures 1a–1c). The surface and the volume can be understood as manifolds of vortex core lines, as reflected by the 5D streamlines in the figure. In 5D space, there exist 2-vortex core lines, which exhibit two planes of rotation (Figure 1d). In this case, we seed two streamlines, offset in direction of each of the two complex eigenplanes, respectively (green and orange), and further a streamline, that is properly contained in the 4-dimensional space spanned by the two complex eigenplanes. The resulting streamline (magenta) shows a 2-rotation. For the computation of the vortex core manifolds, we sampled all cases on 10^n regular grids, which is sufficient due to their nearly linear nature.

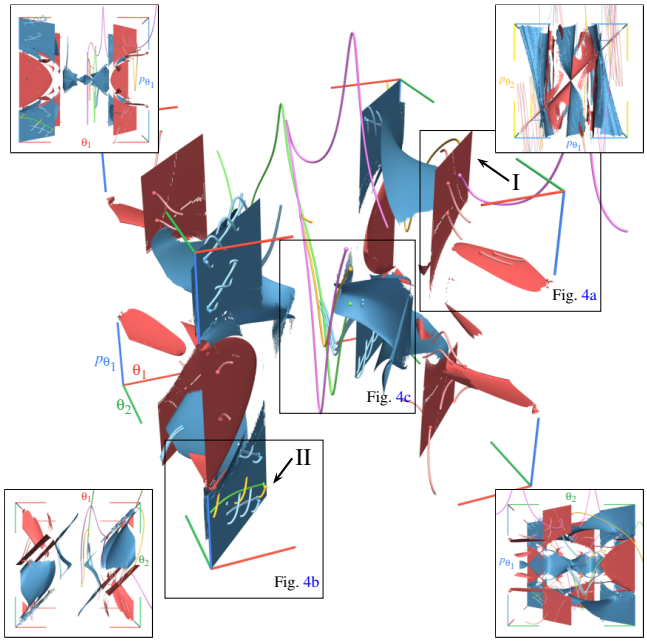


Figure 3: *Phase space of a double pendulum (Figure 4h) with bifurcation manifolds (red) and vortex core manifolds (blue). Shown in 3D orthographic projections (axes: θ_1 red, θ_2 green, p_{θ_1} blue, p_{θ_2} yellow). The marked areas are further discussed in Figure 4.*

The higher-dimensional examples are shown in a 3D orthographic projection, as indicated by the 4D and 5D orientation axes (x_1: red, x_2: green, x_3: blue, x_4: yellow, x_5: magenta).

5.2. Double Pendulum

We consider a double pendulum in the 2D plane, with lengths $l_1 = 10\,\mathrm{m}$, $l_2 = 1\,\mathrm{m}$ and masses $m_1 = 0.1\,\mathrm{kg}$, $m_2 = 1\,\mathrm{kg}$. As gravitational force, we choose $g = 9.81\,\mathrm{m\,s^{-2}}$. The phase space (Figure 3) of this double pendulum is four-dimensional, and we describe it by the two angles θ_1, attached to l_1, θ_2 attached to l_2 (see Figure 4h), and their angular momenta p_{θ_1}, p_{θ_1}, i.e., the dynamical system is given by a 4D vector field $\mathbf{u}(\theta_1, \theta_2, p_{\theta_1}, p_{\theta_2}) = (\dot\theta_1, \dot\theta_2, \dot p_{\theta_1}, \dot p_{\theta_2})^\top$. We sample the vector field on a 150^4 uniform grid on the domain

Table 2: *Details and computational cost of the datasets. Listed are the sizes of the regular grids, the number of processed cell faces, and the number of initial solution points. Timings for computing the derivatives on (all) grid nodes (Deriv.), the dependent vectors operator (DV), filtering (F), triangulation (T), and total computation times (Σ). The timings of the triangulation step include the removal of unused vertices.*

Dataset	Grid Size	Cell Faces	Solutions	Deriv. (s)	DV (s)	F (s)	T (s)	Σ (s)
5D 1-Vortex (Figure 1c)	$10\times10\times10\times10\times10$	400 992	8761	<1	2	<1	<1	2
Double Pendulum (vortex)	$150\times150\times150\times150$	2 316 092 543	444 810	433	6 293	1	19	6 746
Double Pendulum (bifurcation)	$150\times150\times150\times150$	255 913 981	131 082	433	4 608	1	6	5 037
Double Gyre	$1000\times500\times150\times50$	488 933 295	6 316 775	21 703	5 105	3	52	26 863
Vortex Street	$41\times241\times41\times16$	8 742 615	3 064 189	5	47	1	1	54
Convective Flow (valley)	$120\times60\times120\times200$	287 687 572	52 139	737	495	1	1	1 234
Convective Flow (vortex)	$60\times30\times60\times200$	58 646 914	385 800	12	124	<1	<1	136

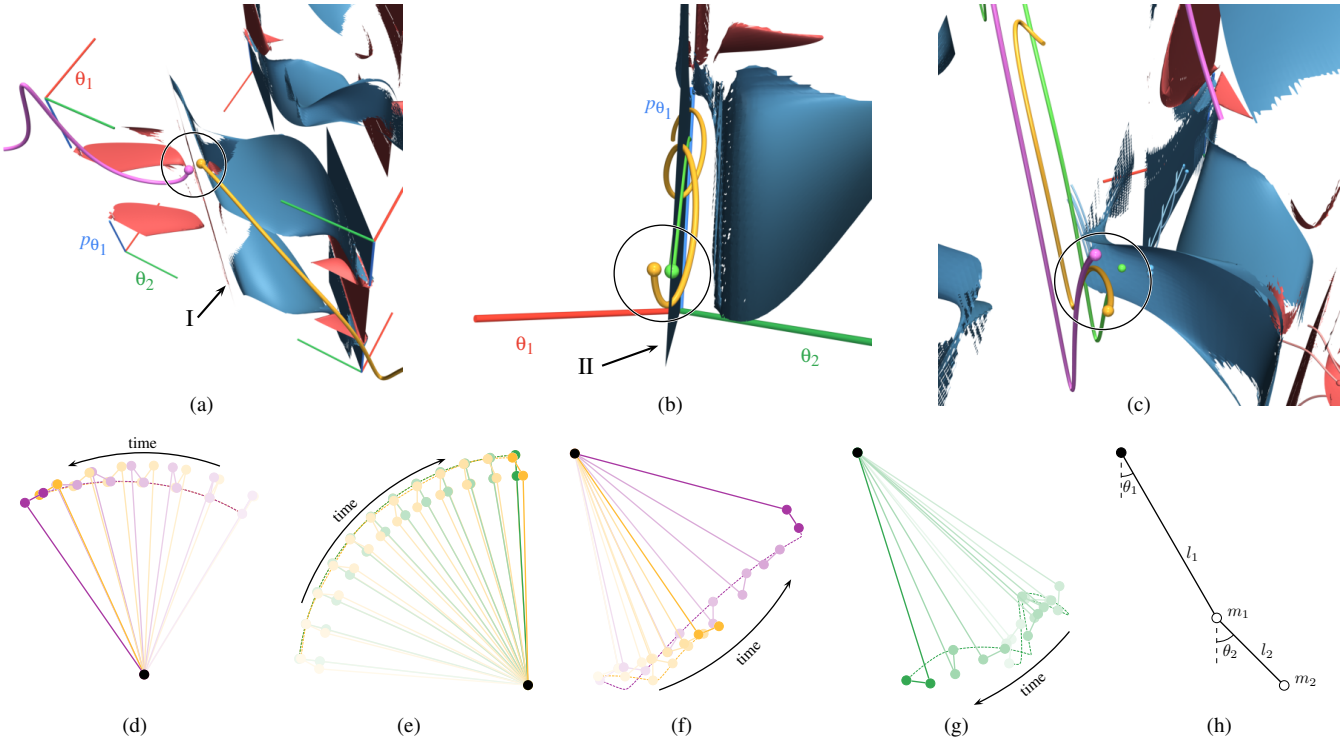


Figure 4: *4D phase space of a double pendulum (h), with bifurcation manifolds (red), and vortex core manifolds (blue) (see Figure 3). The trajectories of the pendulum in Euclidean coordinates are shown for selected seeds, offset in direction of the major eigenvector on a bifurcation manifold (I) (a),(d), and offset within the complex eigenplane on a vortex core manifold (II) (b),(e). Trajectories not near a bifurcation or vortex core manifold ((c),(f), magenta and yellow) exhibit behavior different than near one (a),(b). A trajectory near one of the curved vortex core manifolds ((c), green) diverges from it, while its second arm (θ_2) performs full rotations (g). The colored streamlines started from the circled seeds in (a)–(c) correspond to the motion of the pendulum in Euclidean space, shown in the same color (d)–(g). Temporal positions of the pendulum are indicated by saturation (older are less saturated), and the paths of the second mass by dashed lines.*

$[-\pi,\pi]\times[-\pi,\pi]\times[-1,1]\times[-1,1]$, and extract two-dimensional vortex core manifolds and bifurcation manifolds. Since both features ideally represent manifolds of streamlines, we filter them by discarding results that have an angle to $\mathbf{u}$ larger than 45°. While the phase space is 2π-periodic in the first two dimensions, the second two, representing angular momentum, are not periodic. Our experiments showed, that extending the phase space beyond absolute value one for each momentum, continuously extents the obtained manifolds. An overview of the extracted structures can be found in Figure 3, where we show the 4D structures in orthographic projection onto the axes $(\theta_1,\theta_2,p_{\theta_1})$. We show bifurcation manifolds in red, and vortex core manifolds in blue. On the bifurcation surfaces we choose points, and seed streamlines offset in direction of the major eigenvector (Figure 3, red lines). These streamlines show diverging behavior in vicinity of the bifurcation manifolds, thus verifying the features. We also compute streamlines offset in the complex eigenplane along vortex core manifolds, which exhibit swirling motion (Figure 3, blue lines).

Furthermore, we visualize the motion of the pendulum, that corresponds to a streamline in the phase space, in the plane (e.g., Figure 4d). Two streamlines seeded on opposite sides of the bifurcation surfaces in Figure 4a (green and orange lines) correspond to motions of the pendulum, where the second arm tilts in opposite directions (Figure 4d). Similarly, we choose one seed on the vortex core manifold in Figure 4b (green) and one offset within the complex eigenplane (orange). The motion corresponding to the offset streamline closely follows the corresponding one on the vortex core streamline, but alternates between its opposite sides (Figure 4e). Finally, we investigate streamlines seeded neither in the vicinity of a vortex core surface nor a bifurcation surface (Figure 4c, magenta and yellow). They exhibit none of the above behaviors (Figure 4f). The center of the phase space shows vortex core surfaces with high curvature, which exhibit large angles to the underlying vector field, i.e., streamlines started on them immediately diverge. The second arm of the pendulum corresponding to a streamline started near one of them (Figure 4c, green) performs full rotations (Figure 4g).

5.3. Recirculation Surfaces

Recirculation surfaces are surfaces in the $(n{+}2)$-dimensional space, consisting of the spatial location $\mathbf{x}$ of dimension n together with initial time t and advection time τ. They are characterized by the property, that a trajectory started at $\mathbf{x}$ at time t flows back to its original position after advection time τ. Recirculation surfaces have been recently proposed for flow visualization by Wilde et al. [WRT18].

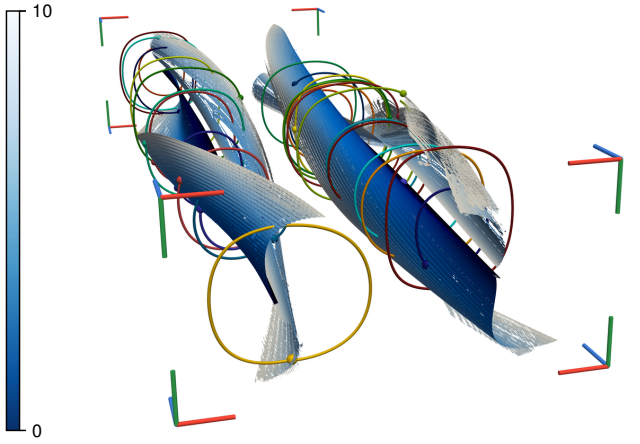


Figure 5: *Recirculation surface of the 2D double gyre in space-time view (x red, y green, time blue axis). The fourth coordinate of the surface, advection time, is shown in shades of blue. Points on this surface in 4D correspond to parameters $(\mathbf{x},t,\tau)$ of a pathline. Randomly chosen points on this manifold, together with their pathlines in their starting time slices, are shown in different colors.*

As an alternative to the authors' approach, we may extract recirculation surfaces as valley surfaces in the scalar field

$$s(\mathbf{x},t,\tau) = \|\boldsymbol{\phi}_t^\tau(\mathbf{x}) - \mathbf{x}\|/\tau, \tag{16}$$

where $\boldsymbol{\phi}_t^\tau(\mathbf{x})$ denotes the flow map, which maps an initial position $\mathbf{x}$ to the endpoint of the trajectory started at time t after advection time τ. While ideally, s would be exactly zero on a recirculation surface, numerically this is infeasible, since s is nonnegative. Valley surfaces in the scalar field s thus allow us to approximate the true recirculation surface, where we filter out those valleys that have a scalar value above a certain threshold.

We demonstrate this approach using the synthetic 2D time-dependent Double Gyre field due to Shadden et al. [SLM05]. The scalar field s is sampled on a regular grid of size $1000 \times 500 \times 150 \times 50$ in the domain $[0,2] \times [0,1] \times [0,5] \times [0,10]$. Its computation took 187 minutes with our prototype in CUDA on a Nvidia Titan Xp. The DV operator took 7.5 hours in parallel on two 14-Core Xeon Gold 6132, where 80% of the time was spent approximating the gradients using least-squares with a 3-ring neighborhood. When computing the DV operator, we only considered those cells, where $s < 0.01$, and we filtered the resulting solutions by $s < 0.006$. We verified the results by computing pathlines with parameters $(\mathbf{x},t,\tau)$ at randomly chosen points on the manifold. Figure 5 shows the projection of the solution manifold on the space-time domain $(\mathbf{x},t)$ together with pathlines, that are projected onto the time-slices corresponding to their respective starting times.

Since the computation of the valley surfaces relies on the Hessian matrix, and thus an approximation of the gradient of the flow map, our resulting surfaces exhibit holes due to aliasing in the computed flow map, especially at locations of high curvature. Wilde et al. [WRT18] on the other hand avoid computations of gradients of the flow map, and employ local refinement of the grid, but their approach is computationally more expensive.

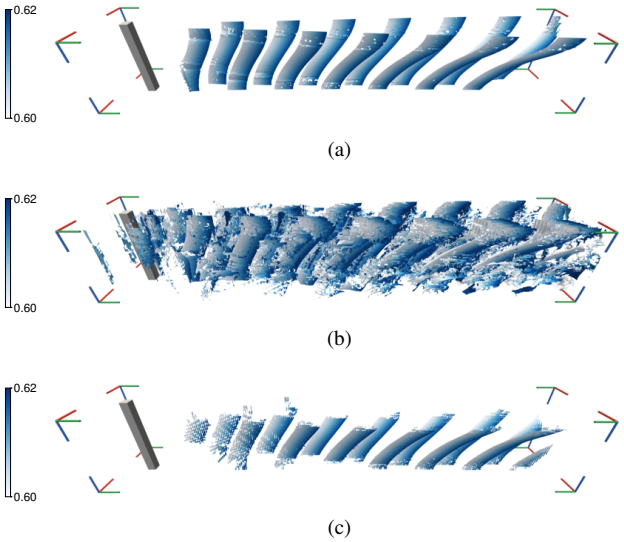
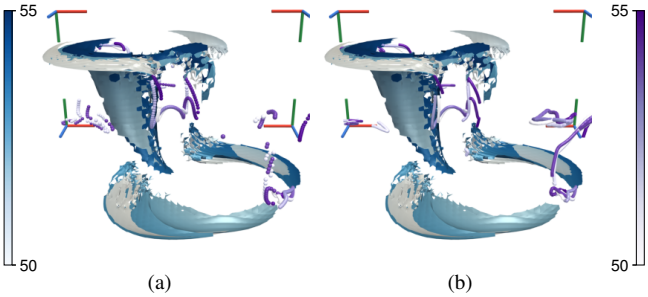


Figure 6: *Vortex core surfaces in the space-time domain of a flow behind an obstacle in the time range 0.60 s to 0.62 s. Projection into the spatial domain (x red, y green, z blue axis), with time colored in shades of blue. Swirling particle cores [WSTH07] are extracted in each time slice and tracked (a) within our framework. Vortex core surfaces extracted by the DV operator in the 4D space-time domain results in noisy raw solutions (b), which requires stronger filters (c).*

5.4. Von Kármán Vortex Street

Now, we demonstrate the DV operator on the time interval of $[0.6, 0.62]$ s in the spatial domain of $[0,10] \times [0,60] \times [0,1]$ m^3 of a computational fluid dynamics (CFD) simulation of a von Kármán vortex street. The vortex street forms behind an obstacle and consists of vortices that move with the flow over time. The Galilean-invariant method by Weinkauf et al. [WSTH07] is able to extract these independently of the frame of reference chosen. Conceptually, the authors extract vortex core surfaces in the 4D space-time domain, which our definition (Section 3.3) generalizes. However, since the time component is constant one, the authors recast the 4D coplanar vectors problem (DV operator with $n = 4$, $k = 2$) as a 3D parallel vectors problem and tracked solution lines over time. We recreate this approach within our framework, by extending the two 3D vector fields $\tilde{\mathbf{u}} = \mathbf{v} + (\nabla\mathbf{v})^{-1}\mathbf{v}_t$, $\tilde{\mathbf{w}}_1 = (\nabla\mathbf{v})\tilde{\mathbf{u}}$, where $\mathbf{v}$ denotes the time-dependent vector field from the CFD simulation, to 4D vector fields via $\mathbf{u} = (\tilde{\mathbf{u}},0)^\top, \mathbf{w}_1 = (\tilde{\mathbf{w}}_1,0)^\top$ in each time slice. Adding a third, constant vector field $\mathbf{w}_2 = (0,0,0,1)^\top$, the DV operator extracts surfaces, which connect the solution lines of $\tilde{\mathbf{u}} \parallel \tilde{\mathbf{w}}_1$ within each time slice to those in adjacent time slices.

We compare the approach by Weinkauf et al. (Figure 6a) to direct solutions in 4D with the DV operator (Figure 6c). The raw features of both approaches exhibit noise close to the obstacle and near the boundaries of the domain. These correspond to weakly-rotating vortices and false positives due to numerical noise, which we filtered by requiring feature strength $\vartheta_v > 10^6$. Since the DV operator not only searches for solutions in space at fixed time slices, but also in time direction, our solutions exhibit more noise (raw



(a) (b)

Figure 7: *Tracking of critical points by means of valley lines of velocity magnitude in the 4D space-time domain of the Convective Flow. Critical points extracted in each time step ((a), spheres) are tracked in the time interval $[50,55]s$ (saturation) by valley lines ((b), tubes). Vortex core manifolds are shown for context (surfaces).*

solutions shown in Figure 6b). This requires to additionally filter the solutions with angle filter of 5°, which, however, leads to false negatives near the top of the domain boundary and close to the obstacle. This is also partially caused by our simple but generic triangulation approach, which is sensitive to numerical noise. A more sophisticated triangulation algorithm could produce more accurate solutions, and less false positives when filtering.

5.5. Convective Flow

Our last example is again a time-dependent CFD simulation, but this time of a thermally driven convective flow. The dataset has a spatial resolution of $60 \times 30 \times 60$. Valley lines of the (spatial) velocity magnitude field in space-time include, among other structures, the motion of critical points of the instantaneous (spatial) field. Because the resolution of the DV operator depends on the grid, and because the original resolution caused the solution lines to be fragmented, we resampled the grid, using trilinear interpolation, in the spatial domain to a resolution of $120 \times 60 \times 120$, and consider the time interval $[50,55]$ s, which results in a 4D grid of size $120 \times 60 \times 120 \times 200$. Since only the valleys close to zero correspond to critical points, we omit features belonging to a scalar value above 0.003. We further impose an angle threshold of 45° to account for false positives, which occur near the boundary. To put the extracted features into context, we furthermore extract vortex core surfaces (see Section 5.4 for a detailed discussion) in the same time interval on the original grid of size $60 \times 30 \times 60 \times 200$, where we require feature strength $\vartheta_v > 5$, and angles below 5°.

We verify the solution lines (Figure 7b) by extracting critical points in selected time steps by recursive subdivision of each 3D cell. The results show, that extracting critical points as valley lines in the space-time magnitude field misses those, that only exist for a short timespan or move farther than one cell within one time step. As the DV operator favors temporal coherence, as it treats space and time equally, critical points that are temporally unstable are missed by the extraction. Figure 7a shows, that some critical points were not tracked by our method, due to filtering false negatives and a lack of temporal coherence. As such, other methods such as (stable) feature flow fields [TS03, WTVGP11] would be suited better for tracking critical points. We have included this example to demonstrate the wide applicability of our method.

6. Discussion and Limitations

Our work focuses on developing a generic algorithm, which extracts locations of dependency of a set of k vector fields of dimension n. Recent work has also focused on generalizations of the parallel vectors operator. Oster et al. [ORT18a, ORT18b] have presented an operator, which extracts parallel vectors locations from tensor fields. The algorithm not only searches for the 3D location, but also the eigenvector itself, adding two degrees of freedom to the search space. While this is a 5D feature extraction method, it does not extract locations in a 5D vector field, where two 5D vectors are parallel (case $n = 5$, $k = 1$ of the DV operator). On the other hand, Günther and Theisel [GT18] extracted locations in a 6D vector field, where two 6D vectors are parallel. This case corresponds to $n = 6$, $k = 1$ of the DV operator, and their formulation of a 6D cross product coincides with the notion of the wedge product used in our generalization.

While we believe, that our generic algorithm enables the visualization of higher-dimensional vector and scalar fields, there are some limitations that come with its generality. Firstly, our very simple triangulation algorithm is sensitive to noise, and thus often produces non-manifold meshes. This is especially a problem for computing feature angles for filtering. An algorithm tailored specifically to, say, surfaces, could yield more accurate solutions. Secondly, the explicit computation of eigenvectors becomes less numerically stable with increasing dimension of the involved matrices, due to the iterative methods that are employed. Furthermore, the curse of dimensionality limits our approach in several ways. As dimension of the domain increases, exponentially more sample points are needed. Even though our algorithms scale linearly with the problem size, especially in 5D and beyond the computations quickly become infeasible, thus limiting us to low sampling rates. For example, the total computation time for the 10^5 dataset shown in Figure 1c is 2 s. With a resolution of 150^5, computation would take approximately 1 518 750 s (421 h), since computation time scales linearly with the number of cells. Secondly, memory access is a limiting factor in our implementation. In order to collect the node data of a single k-dimensional cell face, 2^k data points need to be accessed. Within a uniform grid, which is stored in scanline order, this involves accessing memory k times in a non-linear fashion, thus causing cache misses.

7. Conclusion

In this work, we generalized the parallel vectors operator due to Peikert and Roth to arbitrary dimension. The parallelism property generalized to linear dependency of sets of vector fields, and we classified the types of solution manifolds, and presented a generic algorithm for their extraction. We discussed, and demonstrated, the application of the dependent vectors operator to the extraction of vortex core manifolds, bifurcation manifolds, and ridge manifolds in higher dimensions, and exemplified our technique using synthetic fields, dynamical systems, as well as fields from computational fluid dynamics. As future work, we plan to investigate further applications of our concept for feature definition and extraction.

References

[BBD*07] BREMER P., BRINGA E., DUCHAINEAU M., GYULASSY A., LANEY D., MASCARENHAS A., PASCUCCI V.: Topological feature extraction and tracking. *Journal of Physics: Conference Series 78*, 1 (2007), 012007. 2

[BWC04] BHANIRAMKA P., WENGER R., CRAWFIS R.: Isosurface construction in any dimension using convex hulls. *IEEE Transactions on Visualization and Computer Graphics 10*, 2 (2004), 130–141. 2, 4

[DSL90] DEGANI D., SEGINER A., LEVY Y.: Graphical visualization of vortical flows by means of helicity. *AIAA Journal 28*, 8 (1990), 1347–1352. 2

[EGM*94] EBERLY D., GARDNER R., MORSE B., PIZER S., SCHARLACH C.: Ridges for image analysis. *Journal of Mathematical Imaging and Vision 4*, 4 (1994), 353–373. 2, 3

[EH02] EDELSBRUNNER H., HARER J.: Jacobi sets of multiple Morse functions. *Foundations of Computational Mathematics 312* (2002), 37–57. 2

[EHM*08] EDELSBRUNNER H., HARER J., MASCARENHAS A., PASCUCCI V., SNOEYINK J.: Time-varying Reeb graphs for continuous space–time data. *Computational Geometry 41*, 3 (2008), 149–166. 2

[EHNP04] EDELSBRUNNER H., HARER J., NATARAJAN V., PASCUCCI V.: Local and global comparison of continuous functions. In *Proc. IEEE Visualization '04* (2004), pp. 275–280. 2

[FP01] FURST J. D., PIZER S. M.: Marching ridges. In *Proc. IASTED Internation Conference on Signal and Image Processing* (2001), pp. 22–26. 6

[FPH*08] FUCHS R., PEIKERT R., HAUSER H., SADLO F., MUIGG P.: Parallel vectors criteria for unsteady flow vortices. *IEEE Transactions on Visualization and Computer Graphics 14*, 3 (2008), 615–626. 2

[GNSB05] GUNAWAN H., NESWAN O., SETYA-BUDHI W.: A formula for angles between subspaces of inner product spaces. *Contributions to Algebra and Geometry 46*, 2 (2005), 311–320. 7

[GRT18] GERRITS T., RÖSSL C., THEISEL H.: An approximate parallel vectors operator for multiple vector fields. *Computer Graphics Forum 37*, 3 (2018), 315–326. 2

[GT18] GÜNTHER T., THEISEL H.: Objective vortex corelines of finite-sized objects in fluid flows. *IEEE Transactions on Visualization and Computer Graphics* (2018). 2, 11

[HRS18] HOFMANN L., RIECK B., SADLO F.: Visualization of 4D vector field topology. *Computer Graphics Forum 37*, 3 (2018), 301–313. 2

[Ins85] INSELBERG A.: The plane with parallel coordinates. *The Visual Computer 1*, 2 (1985), 69–91. 2

[JCWD14] JU T., CHENG M., WANG X., DUAN Y.: A robust parity test for extracting parallel vectors in 3D. *IEEE Transactions on Visualization and Computer Graphics 20*, 12 (2014), 2526–2534. 2

[JHP*17] JUNG P., HAUSNER P., PILZ L., STERN J., EULER C., RIEMER M., SADLO F.: Tumble-vortex core line extraction. In *Proc. SIBGRAPI WVIS* (2017). 2

[Kan00] KANDOGAN E.: Star coordinates: A multi-dimensional visualization technique with uniform treatment of dimensions. In *Proc. IEEE Information Visualization Symposium* (2000), vol. 650, p. 22. 2

[Ken98] KENWRIGHT D. N.: Automatic detection of open and closed separation and attachment lines. In *Proc. IEEE Visualization '98* (1998), pp. 151–158. 2

[LC87] LORENSEN W. E., CLINE H. E.: Marching cubes: A high resolution 3D surface construction algorithm. In *Proc. ACM SIGGRAPH Computer Graphics* (1987), vol. 21, pp. 163–169. 4

[MBES16] MACHADO G. M., BOBLEST S., ERTL T., SADLO F.: Space-time bifurcation lines for extraction of 2D Lagrangian coherent structures. *Computer Graphics Forum 35*, 3 (2016), 91–100. 2

[MSE13] MACHADO G. M., SADLO F., ERTL T.: Local extraction of bifurcation lines. In *Proc. Vision, Modeling and Visualization* (2013), pp. 17–24. 2

[ORT18a] OSTER T., RÖSSL C., THEISEL H.: Core lines in 3D second-order tensor fields. *Computer Graphics Forum 37*, 3 (2018), 327–337. 2, 11

[ORT18b] OSTER T., RÖSSL C., THEISEL H.: The parallel eigenvectors operator. In *Proc. Vision, Modeling and Visualization* (2018), pp. 39–46. 2, 11

[PC87] PERRY A. E., CHONG M. S.: A description of eddying motions and flow patterns using critical-point concepts. *Annual Review of Fluid Mechanics 19*, 1 (1987), 125–155. 4

[POS*11] PAGOT C., OSMARI D., SADLO F., WEISKOPF D., ERTL T., COMBA J.: Efficient parallel vectors feature extraction from higher-order data. *Computer Graphics Forum 30*, 3 (2011), 751–760. 2

[PR99] PEIKERT R., ROTH M.: The parallel vectors operator: A vector field visualization primitive. In *Proc. IEEE Visualization '99* (1999), pp. 263–270. 2, 4, 6, 7

[PS08] PEIKERT R., SADLO F.: Height ridge computation and filtering for visualization. In *Proc. IEEE Pacific Visualization Symposium* (2008), pp. 119–126. 2

[Rot00] ROTH M.: *Automatic extraction of vortex core lines and other line-type features for scientific visualization*. PhD thesis, ETH Zurich, No. 13673, 2000. 4

[RP98] ROTH M., PEIKERT R.: A higher-order method for finding vortex core lines. In *Proc IEEE Visualization '98* (1998), pp. 143–150. 2

[SH95] SUJUDI D., HAIMES R.: Identification of swirling flow in 3-D vector fields. In *12th Computational Fluid Dynamics Conference* (1995), p. 1715. 2, 3

[SLM05] SHADDEN S. C., LEKIEN F., MARSDEN J. E.: Definition and properties of Lagrangian coherent structures from finite-time Lyapunov exponents in two-dimensional aperiodic flows. *Physica D: Nonlinear Phenomena 212*, 3-4 (2005), 271–304. 10

[TS03] THEISEL H., SEIDEL H.-P.: Feature flow fields. In *Proc. Symposium on Data Visualisation 2003* (2003), pp. 141–148. 2, 11

[TSW*05] THEISEL H., SAHNER J., WEINKAUF T., HEGE H.-C., SEIDEL H.-P.: Extraction of parallel vector surfaces in 3D time-dependent fields and application to vortex core line tracking. In *Proc. IEEE Visualization '05* (2005), pp. 631–638. 2

[VGP09] VAN GELDER A., PANG A.: Using PVsolve to analyze and locate positions of parallel vectors. *IEEE Transactions on Visualization and Computer Graphics 15*, 4 (2009), 682–695. 2

[vWvL93] VAN WIJK J. J., VAN LIERE R.: HyperSlice. In *Proc. IEEE Visualization '93* (1993), pp. 119–125. 2

[WB96] WEIGLE C., BANKS D. C.: Complex-valued contour meshing. In *Proc IEEE Visualization'96* (1996), pp. 173–180. 2

[WLG97] WEGENKITTL R., LÖFFELMANN H., GRÖLLER E.: Visualizing the behavior of higher dimensional dynamical systems. In *Proceedings of the 8th conference on Visualization'97* (1997), IEEE Computer Society Press, pp. 119–ff. 2

[WRT18] WILDE T., RÖSSL C., THEISEL H.: Recirculation surfaces for flow visualization. *IEEE Transactions on Visualization and Computer Graphics 25*, 1 (2018), 946–955. 2, 9, 10

[WSTH07] WEINKAUF T., SAHNER J., THEISEL H., HEGE H.-C.: Cores of swirling particle motion in unsteady flows. *IEEE Transactions on Visualization and Computer Graphics 13*, 6 (2007), 1759–1766. 2, 10

[WTVGP11] WEINKAUF T., THEISEL H., VAN GELDER A., PANG A.: Stable feature flow fields. *IEEE Transactions on Visualization and Computer Graphics 17*, 6 (2011), 770–780. 2, 11