Variational Feature Extraction in Scientific Visualization

NICO DASSLER and TOBIAS GÜNTHER, Friedrich-Alexander-Universität Erlangen-Nürnberg, Germany

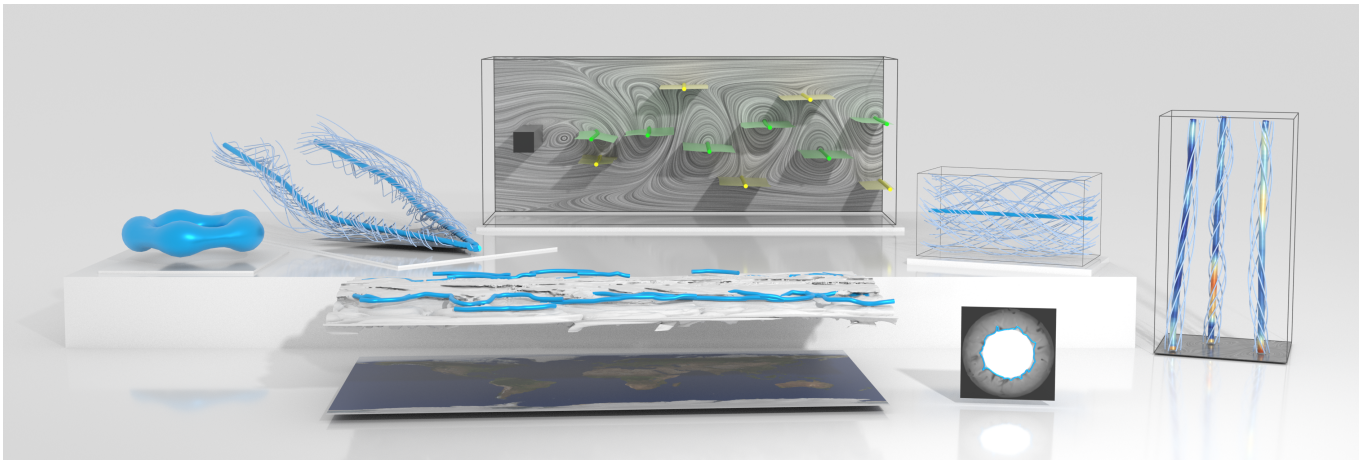


Fig. 1. Results of variational feature extraction for common feature curves and surfaces in scalar fields, vector fields, and tensor fields, including ridge surfaces, vortex corelines, atmospheric jet streams, paths of vortex and bifurcation lines, tensor corelines, isocontours, and paths of ocean eddies.

Across many scientific disciplines, the pursuit of even higher grid resolutions leads to a severe scalability problem in scientific computing. Feature extraction is a commonly chosen approach to reduce the amount of information from dense fields down to geometric primitives that further enable a quantitative analysis. Examples of common features are isolines, extremal lines, or vortex corelines. Due to the rising complexity of the observed phenomena, or in the event of discretization issues with the data, a straightforward application of textbook feature definitions is unfortunately insufficient. Thus, feature extraction from spatial data often requires substantial pre- or post-processing to either clean up the results or to include additional domain knowledge about the feature in question. Such a separate pre- or post-processing of features not only leads to suboptimal and incomparable solutions, it also results in many specialized feature extraction algorithms arising in the different application domains. In this paper, we establish a mathematical language that not only encompasses commonly used feature definitions, it also provides a set of regularizers that can be applied across the bounds of individual application domains. By using the language of variational calculus, we treat features as variational minimizers, which can be combined and regularized as needed. Our formulation not only encompasses existing feature definitions as special case, it also opens the path to novel feature definitions. This work lays the foundations for many new research directions regarding formal definitions, data representations, and numerical extraction algorithms.

CCS Concepts: • **Human-centered computing** → **Visualization techniques**; • **Computing methodologies** → *Computer graphics.*

Additional Key Words and Phrases: Feature extraction, extremal lines, critical lines, parallel vectors, variational, Euler-Lagrange equations

Authors' address: Nico Daßler, nico.dassler@fau.de; Tobias Günther, tobias.guenther@fau.de, Friedrich-Alexander-Universität Erlangen-Nürnberg, Cauerstr. 11, Erlangen, Bavaria, 91058, Germany.

0730-0301/2024/7-ART109
https://doi.org/10.1145/3658219

ACM Reference Format:
Nico Daßler and Tobias Günther. 2024. Variational Feature Extraction in Scientific Visualization. *ACM Trans. Graph.* 43, 4, Article 109 (July 2024), 16 pages. https://doi.org/10.1145/3658219

1 INTRODUCTION

In many scientific disciplines, dynamical systems such as the evolution of the atmosphere or the blood flow in our veins, are described by spatio-temporal fields, i.e., spatially and temporally varying functions, such as temperature, pressure, wind velocity, etc. Due to the increasing availability of computational resources, and the ambition to resolve ever more detail, the grid resolutions and time scales are rapidly increasing, which entails a serious scalability problem. To meet this challenge, a data reduction is vitally needed. A common approach is the extraction of so-called *features* [Post et al. 2003], which are structures of interest that are needed for a particular data analysis. Most contemporary feature definitions in scalar, vector, and tensor fields are criteria that can be evaluated locally, i.e., we can test for a given point if it is part of a feature. Examples are extremal lines and surfaces, vortex corelines, isocontours, parallel vector lines, or critical lines to name a few. Depending on the amount of noise in the data and the quality of derivative estimations, these feature lines (or surfaces) are often pre- or post-processed to become more smooth, to align with a vector field, or to be close to a structure of interest. Usually, feature extraction and clean-up are done separately, for example by smoothing the input fields before feature extraction, or by smoothing the resulting line or surface geometry afterwards. In this paper, we aim for a more rigorous definition of features, in which the regularization is directly built into the feature definition. For this, we need a conceptually different approach. Finding the optimal feature that not only meets the feature definition, but also meets all the regularizations as best as possible, leads to an energy minimization approach that seeks for an optimal curve (or surface)

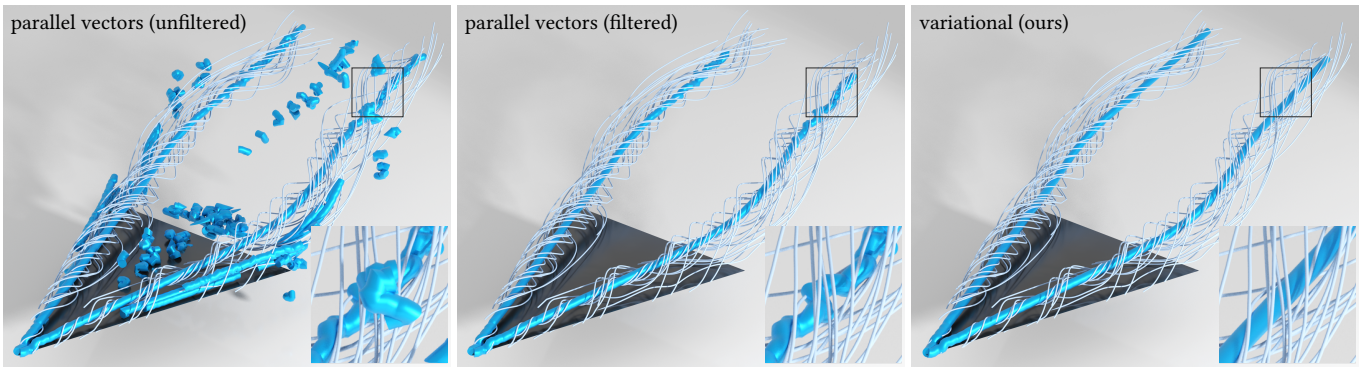


Fig. 2. Vortex coreline extraction from vector fields is numerically challenging, due to its dependence on derivative estimations. In the left image, a conventional vortex coreline extraction technique is shown, namely the reduced velocity criterion of Sujudi and Haimes [1995], which is computed by a Newton-Raphson based parallel vector solver [Peikert and Roth 1999]. The result is known to be spurious [Günther and Theisel 2018] and is cleaned up in the middle image by only showing the largest connected curve. In this flow over a delta wing, the discretization of the piece-wise linear vector field is too coarse in the wake of the vortex, which results in bumpy feature curves. With our variational formulation in the right image, the feature definition is complemented with a smoothness regularization using a suitably chosen regularization weight. Different parameter choices and the advantages/disadvantages are discussed later in Section 7.1.

among many possible curves (or surfaces). In this paper, we phrase the most-frequently used feature definitions and regularizations in the language of variational calculus. Using the Euler-Lagrange equations, we derive ODEs and PDEs, which are the building blocks for the necessary conditions that need to hold for an energy minimum. All derivations are done in a continuous formulation, which makes the (regularized) feature definition independent of the underlying discretization of both the data and the feature curve or surface. We demonstrate how the ODEs and PDEs can be discretized and propose a numerical algorithm to solve for a given initial condition and/or boundary condition. We apply the approach to feature extraction in several scientific disciplines, including aerodynamics, geophysics, atmospheric sciences, wind engineering, fluid dynamics, oceanography, and Morse theory. Results of our method are shown in Fig. 1 for the various disciplines and Fig. 2 demonstrates the benefits of combining the feature definition with a regularizer at the example of vortex corelines above the leading edge of a delta wing surface. In summary, we make the following contributions. We provide:

- a general mathematical formulation of the most commonly used feature definitions as variational energy minimizers,
- a continuous necessary condition for each feature and regularizer that the optimal geometry must fulfill,
- a numerical extraction algorithm that extracts optimal lines and surfaces iteratively from initial conditions.

The variational formulation opens many possible research directions for future work, including further formal feature definitions, discretization schemes, numerical extraction algorithms, and more applications in practice. The approach is not free of parameters, and thus more research on energy weight exploration, and suitable seeding can follow in the future, as well.

Notation. Throughout this paper, we utilize the following notation. Scalar-valued quantities are denoted by lowercase italic letters, such as s. Bold letters denote column-vectors, such as $\mathbf{v}$. The symbol ∇ refers to the nabla operator, which computes the vector of partial derivatives. Note that our Jacobian matrices contain the partial derivatives in the columns. Functions receive their arguments in round brackets $(\dots)$, while functionals (i.e., 'functions' that receive other functions as input) specify their arguments in square brackets $[\dots]$. The symbol ∂ denotes the partial derivative, while δ denotes a functional derivative.

2 RELATED WORK

In the following section, we give an overview of conventional and variational feature extraction methods. Afterwards, we briefly summarize the foundations of variational calculus.

2.1 Conventional Feature Extraction

In scientific visualization, commonly-used feature definitions can be categorized into extremal features and iso features.

2.1.1 Extremal Features. An extremal feature is a geometric structure along which a scalar value is sectionally minimized or maximized. For a formal definition, consider an m-dimensional steady scalar field $s(\mathbf{y}) : \mathbb{R}^m \to \mathbb{R}$, with the $m \times m$ Hessian matrix $\nabla(\nabla s(\mathbf{x})) : \mathbb{R}^m \to \mathbb{R}^{m \times m}$, which has eigenvectors $\mathbf{c}_i(\mathbf{y})$ and eigenvalues $\lambda_i(\mathbf{y})$ for $i \in \{1, \dots, m\}$, i.e., $\nabla(\nabla s(\mathbf{x}))\ \mathbf{c}_i(\mathbf{y}) = \lambda_i(\mathbf{y})\ \mathbf{c}_i(\mathbf{y})$. The eigenvalues are sorted in ascending order $\lambda_1 \leq \dots \leq \lambda_m$. A k-dimensional ridge structure is assembled by all locations at which the gradient is orthogonal to the $m - k$ eigenvectors with smallest eigenvalues, which are in addition required to be negative:

$$\{\mathbf{y} \ : \ \nabla s(\mathbf{y})^{\mathrm{T}} \mathbf{c}_i(\mathbf{y}) = 0,\ \lambda_i(\mathbf{y}) < 0,\ 1 \leq i \leq m - k\}. \tag{1}$$

A valley structure is the opposite and is found by extracting ridge structures of the negated field $-s(\mathbf{y})$. This definition includes the widely used ridge line definition of Eberly [1996], which locates points at which the eigenvector $\mathbf{c}_m$ with largest eigenvalue λ_m is parallel to the gradient $\nabla s(\mathbf{y})$. Many application-specific feature definitions have been phrased, including ridge lines of region-based vortex measures [Sahner et al. 2007], potential vorticity banners [Bader

et al. 2020], and jet streams [Bösiger et al. 2022; Kern et al. 2018]. For further discussions of ridges, we refer to Lindeberg [1998], Peikert and Sadlo [2008], and the framework of Kindlmann et al. [2018].

2.1.2 Iso Features. Another common approach to describe geometric structures of significant relevance is to define them implicitly as (intersections) of level sets in one or multiple scalar fields. Consider a field $\mathbf{v}(\mathbf{y}) : \mathbb{R}^m \to \mathbb{R}^d$, which maps an m-dimensional point to a d-dimensional vector. Iso features are defined as locations at which the function maps to a specific isovalue $\mathbf{c}_0 \in \mathbb{R}^d$:

$$\{\mathbf{y} \ : \ \mathbf{f}(\mathbf{y}) = \mathbf{c}_0\} \tag{2}$$

For scalar fields ($d = 1$), such features are called isocontours (or level sets), which can be extracted with marching cubes [Lorensen and Cline 1987], flying edges [Schroeder et al. 2015], or more recent learning-based approach such as neural marching cubes [Chen and Zhang 2021] that better reconstruct surface details. In time-dependent 2-dimensional vector fields ($m = 3$, $d = 2$), critical points move over time along so-called critical lines [Weinkauf et al. 2007]. For bi-variate scalar fields in 3-dimensional space ($m = 3$, $d = 2$), the intersection of two isocontours is called a fiber, which sweeps out fiber surfaces [Carr et al. 2015] when varying the isovalues.

2.1.3 Parallel Vectors. An important instance of an iso feature is the solution to the parallel vectors operator [Peikert and Roth 1999]. Given the 3-dimensional steady vector fields $\mathbf{v}_1(\mathbf{y}), \mathbf{v}_2(\mathbf{y}) : \mathbb{R}^3 \to \mathbb{R}^3$, the parallel vectors operator returns the set of all locations at which the two vector fields are parallel:

$$\{\mathbf{y} \ : \ \mathbf{v}_1(\mathbf{y}) \parallel \mathbf{v}_2(\mathbf{y})\}, \tag{3}$$

which generally assembles line structures. Two vector fields are parallel if their cross product vanishes, which makes this an iso feature. In steady flow, the operator can be used to extract vortex corelines using the reduced velocity criterion [Sujudi and Haimes 1995]. This concept has been extended to time-dependent flows in multiple ways [Fuchs et al. 2008; Weinkauf et al. 2007] and was generalized to model bent curves [Roth and Peikert 1998], see Günther and Theisel [2018] for a discussion. An objective vortex coreline extraction is achieved when performing an unsteadiness minimization of the reference frame beforehand [Baeza Rojo and Günther 2019; Günther et al. 2017; Hadwiger et al. 2019; Rautek et al. 2020, 2024; Theisel et al. 2021; Zhang et al. 2021]. Closely related to vortex corelines is the concept of bifurcation lines [Perry and Chong 1987; Roth 2000], which was later extended by Machado et al. [2013] in 3D and in 2D space-time [Machado et al. 2016]. The detection of saddle-like behavior in time-dependent vector fields has seen considerable attention. We refer to Bujack [2022]; Hofmann and Sadlo [2020] for a recent discussion of distinguished hyperbolic trajectories. The numerically stable extraction of the parallel vectors operator has been subject to much research [Guo and Peterka 2021; Ju et al. 2014; Pagot et al. 2011; Van Gelder and Pang 2009; Weinkauf et al. 2010]. The operator itself has been extended to higher dimensions [Hofmann and Sadlo 2019] and to the parallel eigenvectors problem [Oster et al. 2018b], which had applications in tensor fields [Oster et al. 2018a].

2.2 Variational Feature Extraction

In the following, we visit past feature definitions that were founded on variational principles.

2.2.1 Ridge Lines and Vortex Cores. Quapp and Schmidt [2011] defined *reaction paths* as shortest paths along which the gradient norm of a scalar is minimized. This definition collapses a feature to a point. Thus, they carefully fixed the end points to a saddle and a minimum to obtain a curve. Instead, we apply an established feature definition that is minimized along ridge lines. Finn and Boghosian [2006] defined a vortex coreline as a tangent curve in the vorticity field along which vorticity is maximized. They required a smoothness term to reach stability since their definition had undesirable degrees of freedom, leading to what they referred to as accordion folding. Our vortex coreline definition is consistent with established definitions and does not require regularizations to work.

2.2.2 Lagrangian Coherent Structures. In flow visualization, Lagrangian coherent structures (LCS) are material lines and surfaces that separate the flow into regions of coherent motion. They are categorized into parabolic (jets), elliptic (vortex boundaries), and hyperbolic (transport barriers) LCS. Haller et al. [Farazmand and Haller 2012; Haller 2011; Oettinger et al. 2016; Oettinger and Haller 2016] introduced a variational formulation for all three types of features. Since we add regularizers to our features, we cannot follow their approach of using Noether's theorem to derive a feature extraction. Instead, we derive a general iterative feature extraction algorithm for our feature definitions.

2.2.3 Flow Representations. Weißmann et al. [2014] represented the vorticity field of a given vector field by means of a set of vortex filaments with a given filament strength. For this, a complex-valued field is searched in which the filaments arise as zero level set, i.e., as curves along which the real part and the imaginary part vanish to zero. Such an implicit representation leads to closed loops. In contrast, we will represent feature curves explicitly, which can represent non-closed structures. Chern et al. [2017] presented spherical Clebsch maps to represent/approximate the fluid state of vector fields using a set of fields, for which a Dirichlet type of energy is minimized. The map is used for flow processing, vortex tube visualization using isocontours, and it delivers initial conditions for the Schrödinger's smoke fluid solver [Chern et al. 2016].

2.3 Variational Calculus Review

Fermat's theorem in differential calculus provides the means to find a point that minimizes (or maximizes) a target function. In a similar way, *variational calculus* allows us to find functions that minimize a target *functional*, i.e., a function that receives functions as input [Gelfand and Fomin 1963]. Consider the functional $\mathcal{F}[f_1, \ldots, f_m]$, which receives m differentiable n-dimensional univariate functions $f_i(x_1, \ldots, x_n)$ for $i \in \{1, \ldots, m\}$ as input and returns a scalar that is computed as integral over the domain $\mathbb{X}$:

$$\mathcal{F}[f_1, \ldots, f_m] = \int_{\mathbb{X}} \mathcal{L}\left(x_1, \ldots, x_n, f_1, \ldots, f_m, \frac{\partial f_1}{\partial x_1}, \ldots, \frac{\partial f_m}{\partial x_n}\right) \mathrm{d}\mathbf{x}, \tag{4}$$

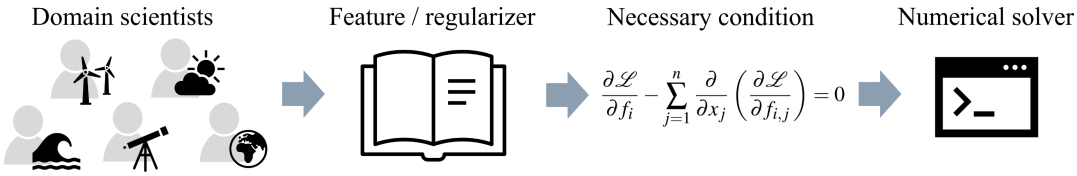


Fig. 3. Overview of the intended variational feature extraction pipeline from the eyes of a domain scientist. A domain scientist may select and combine feature definitions and regularizations as desired by looking them up in our collection of definitions. For example, they might choose to compute vortex corelines that are smoothed. For the chosen features and regularizers, the necessary condition can be read from this paper, which gives ODEs and PDEs that the optimal feature must fulfill (Sections 4 and 5). Depending on the properties of the resulting necessary condition (e.g., is it a linear or non-linear equation), a suitable numerical extraction algorithm can be chosen (Section 6).

where the functions $f_i(\mathbf{x}) : \mathbb{X} \to \mathbb{R}$ are defined on the domain $\mathbb{X} \subseteq \mathbb{R}^n$, with $\mathbf{x} = (x_1, \ldots, x_n)^\mathrm{T} \in \mathbb{X}$. For notational convenience, we later combine the m scalar-valued functions into one vector-valued function $\mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), \ldots, f_m(\mathbf{x}))^\mathrm{T} : \mathbb{X} \to \mathbb{R}^m$ and place its first-order partial derivatives column-wise in the Jacobian matrix $\nabla\mathbf{f}(\mathbf{x}) \in \mathbb{R}^{m \times n}$. Then, Eq. (4) can be rephrased more concisely:

$$\mathcal{F}[\mathbf{f}(\mathbf{x})] = \int_{\mathbb{X}} \mathcal{L}\left(\mathbf{x}', \mathbf{f}(\mathbf{x}'), \nabla\mathbf{f}(\mathbf{x}')\right) \mathrm{d}\mathbf{x}'. \tag{5}$$

The function $\mathcal{L}$ that is integrated over the domain is called the *Lagrangian* and measures what is to be minimized over the function domain. The Lagrangian could be extended to include higher derivatives, as well, but for the rest of the paper, we only need first-order partials of the unknown functions.

Euler-Lagrange Equation. As a necessary condition, the scalar-valued functions $f_i(x_1, ..., x_n)$ minimize Eq. (4) when their *functional derivatives* $\frac{\delta\mathcal{F}}{\delta f_i}$ vanish, which are known as the Euler-Lagrange equations. Their derivation is included in the supplemental material. There is one equation for each function f_i in $i \in \{1, \ldots, m\}$:

$$\frac{\delta\mathcal{F}}{\delta f_i} = \frac{\partial\mathcal{L}}{\partial f_i} - \sum_{j=1}^{n} \frac{\partial}{\partial x_j}\left(\frac{\partial\mathcal{L}}{\partial f_{i,j}}\right) = 0, \tag{6}$$

with the short-hand notation $f_{i,j} := \frac{\partial f_i}{\partial x_j}$. In the remainder of the paper, we stack those derivatives into a row vector $\frac{\delta\mathcal{F}}{\delta\mathbf{f}}$:

$$\frac{\delta\mathcal{F}[\mathbf{f}(\mathbf{x})]}{\delta\mathbf{f}(\mathbf{x})} = \left(\frac{\delta\mathcal{F}[\mathbf{f}(\mathbf{x})]}{\delta f_1(\mathbf{x})}, \ldots, \frac{\delta\mathcal{F}[\mathbf{f}(\mathbf{x})]}{\delta f_m(\mathbf{x})}\right) = \mathbf{0}. \tag{7}$$

As sufficient condition, the functional $\mathcal{F}[\mathbf{f}(\mathbf{x})]$ has a minimum if the second functional derivatives are strongly positive. Depending on the Lagrangian $\mathcal{L}$, Eq. (7) typically results in a second-order differential equation in $\mathbf{f}(\mathbf{y})$. This differential equation holds for every point on every feature. To select a particular solution, appropriate boundary conditions need to be set.

3 VARIATIONAL FEATURE EXTRACTION

In this paper, we lay the foundations for the consistent modeling of features in the language of variational calculus. Thus, our goal is to phrase many commonly-used features as minimizer of a specific energy that can be subject to regularizations that model domain knowledge. Figure 3 gives a conceptual overview of the intended workflow. A domain scientist who works with spatio-temporal field data, may select and combine feature definitions and regularizations as desired. For each of the feature definitions and regularizations, the necessary condition for the optimum, i.e., the Euler-Lagrange equation of each term, can be looked up from this paper. Depending on the properties of the resulting Euler-Lagrange equations, i.e., dependent on whether they are linear or non-linear, a suitable numerical algorithm for the feature extraction is selected. To model this, we need four ingredients, as introduced in the following.

Feature (i.e., the 'unknown'). We begin our formal description by introducing the unknown in our optimization, i.e., a continuous feature such as a curve or a surface. We formally describe a continuous feature as a parametric function $\mathbf{f}(\mathbf{x}) : \mathbb{X} \to \mathbb{Y}$, where the parameter-space coordinate $\mathbf{x} \in \mathbb{X} \subseteq \mathbb{R}^n$ (for example the uv coordinate on a surface) maps to the vertex positions in the spatial domain $\mathbb{Y} \subseteq \mathbb{R}^m$. We assume that $\mathbf{f}(\mathbf{x})$ is differentiable in $\mathbb{X}$.

Lagrangian (i.e., the 'energy'). Each main type of feature comes with a variational definition to which regularizers can be added. Some regularizers are commonly applicable (such as smoothness), while others may be tailored to model specific domain knowledge. To model this formally, we split the Lagrangian $\mathcal{L}$, i.e., our objective function, into a feature term $\mathcal{M}$ and its regularization terms Γ_k, each with an energy weight λ_k that can be adjusted carefully:

$$\mathcal{L}(\mathbf{x}, \mathbf{f}, \nabla\mathbf{f}) = \mathcal{M}(\mathbf{x}, \mathbf{f}, \nabla\mathbf{f}) + \sum_k \lambda_k \Gamma_k(\mathbf{x}, \mathbf{f}, \nabla\mathbf{f}). \tag{8}$$

Euler-Lagrange Equation (i.e., the 'necessary condition'). The resulting Euler-Lagrange equations in Eq. (7), which provide a necessary condition for an optimal solution, are then likewise split:

$$\frac{\delta\mathcal{F}[\mathbf{f}(\mathbf{x})]}{\delta\mathbf{f}(\mathbf{x})} = \frac{\delta\mathcal{M}[\mathbf{f}(\mathbf{x})]}{\delta\mathbf{f}(\mathbf{x})} + \sum_k \lambda_k \frac{\delta\Gamma_k[\mathbf{f}(\mathbf{x})]}{\delta\mathbf{f}(\mathbf{x})} = \mathbf{0}. \tag{9}$$

Gradient Descent (i.e., the 'solver'). Our optimization requires an initial guess $\mathbf{f}^{(0)}(\mathbf{x})$ for the feature $\mathbf{f}(\mathbf{x})$, for which several options are discussed later in Section 6.2. To improve the estimate of a feature point that is not pinned by a boundary condition, we employ a gradient descent using the functional derivative:

$$\frac{\partial\mathbf{f}(\mathbf{x})}{\partial t} = -\frac{\delta\mathcal{F}[\mathbf{f}(\mathbf{x})]}{\delta\mathbf{f}}^\mathrm{T} \approx \frac{\mathbf{f}^{(n+1)}(\mathbf{x}) - \mathbf{f}^{(n)}(\mathbf{x})}{h}. \tag{10}$$

Rearranging the finite difference for the next position $\mathbf{f}^{(n+1)}(\mathbf{x})$, we obtain for the step size h the following update rule:

$$\mathbf{f}^{(n+1)}(\mathbf{x}) \approx \mathbf{f}^{(n)}(\mathbf{x}) - h \cdot \frac{\delta \mathcal{F}[\mathbf{f}^{(n)}(\mathbf{x})]}{\delta \mathbf{f}}^{\mathrm{T}}. \tag{11}$$

In practice, stochastic optimizers, such as Adam [Kingma and Ba 2017], can be used to adjust the step size h. In the following, we derive the Euler-Lagrange equations for common feature definitions and regularizers. Afterwards, we discuss the discretization and propose iterative numerical algorithms for estimating initial conditions and for solving the equations.

4 FEATURE DEFINITIONS

In the following, we introduce the variational formulation of several commonly-used feature definitions. We keep the descriptions concise, such that users of the variational framework can simply 'lookup' the needed definitions. Thus, we only list the formal input fields, the Lagrangian that is minimized, and the corresponding functional derivatives for each feature. For future users of the variational framework, it is of course important to understand how those equations are derived, such that new feature definitions can be introduced. For this reason, we present the full derivations of every term in the supplemental material, where we continue to list and demonstrate further feature definitions, as well.

4.1 Ridge Lines

Given is a three-dimensional scalar field $s(\mathbf{y}) : \mathbb{Y} \to \mathbb{R}$ with the gradient vector $\nabla s(\mathbf{y})$ and the Hessian matrix $\mathbf{H}(\mathbf{y})$, which has at each location $\mathbf{y}$ the real-valued, sorted eigenvalues $\lambda_1(\mathbf{y}) \leq \lambda_2(\mathbf{y}) \leq \lambda_3(\mathbf{y})$ and the corresponding eigenvectors $\mathbf{c}_1(\mathbf{y})$, $\mathbf{c}_2(\mathbf{y})$, $\mathbf{c}_3(\mathbf{y})$, i.e., $\mathbf{H}(\mathbf{y})\mathbf{c}_i(\mathbf{y}) = \lambda_i(\mathbf{y})\mathbf{c}_i(\mathbf{y})$ for all $i \in \{1, 2, 3\}$. The ridge line criterion of Eberly [1996] from Eq. (1) requires that the eigenvector $\mathbf{c}_3$, corresponding to the largest eigenvalue λ_3, is parallel to the gradient ∇s. Thus, the 1-dimensional ridge line $\mathbf{f}(x) = \big(f_1(x), \dots, f_3(x)\big)^{\mathrm{T}}$ minimizes the functional:

$$\mathcal{M}^{rl} = \frac{1}{2} \left\| \nabla s(\mathbf{f}(x)) \times \mathbf{c}_3(\mathbf{f}(x)) \right\|^2 \quad \text{s.t.} \quad \lambda_2(\mathbf{f}(x)) < 0, \tag{12}$$

$$\frac{\delta \mathcal{M}^{rl}}{\delta \mathbf{f}(x)} = (\nabla s \times \mathbf{c}_3)^{\mathrm{T}} \cdot \begin{pmatrix} (\frac{\partial \nabla s}{\partial f_1} \times \mathbf{c}_3 + \nabla s \times \frac{\partial \mathbf{c}_3}{\partial f_1})^{\mathrm{T}} \\ \vdots \\ (\frac{\partial \nabla s}{\partial f_m} \times \mathbf{c}_3 + \nabla s \times \frac{\partial \mathbf{c}_3}{\partial f_m})^{\mathrm{T}} \end{pmatrix}. \tag{13}$$

For notational convenience we dropped the dependencies, i.e., $\nabla s := \nabla s(\mathbf{f}(x))$ and $\mathbf{c}_3 := \mathbf{c}_3(\mathbf{f}(x))$. Valley lines of a scalar field $s(\mathbf{y})$ are analogously found as ridge lines of the negated scalar field $-s(\mathbf{y})$. In a two-dimensional domain ($m = 2$), the parallel vectors may either be lifted by appending a zero, or the ridge line may be searched as locations where the dot product of the gradient and the eigenvector with smallest eigenvalue vanishes. A vanishing dot product is also the minimizer for ridge surfaces, as introduced next.

4.2 Ridge Surfaces

Given is a three-dimensional scalar field $s(\mathbf{y}) : \mathbb{Y} \to \mathbb{R}$ as in Section 4.1. Following Eberly [1996], a ridge surface is characterized by locations at which the eigenvector $\mathbf{c}_1(\mathbf{y})$ to the smallest eigenvalue $\lambda_1(\mathbf{y})$ is orthogonal to the gradient $\nabla s(\mathbf{y})$ and where the corresponding eigenvalue is negative, i.e., $\lambda_1(\mathbf{y}) < 0$. The 2-dimensional ridge surface $\mathbf{f}(x_1, x_2) = \big(f_1(x_1, x_2), \dots, f_3(x_1, x_2)\big)^{\mathrm{T}}$ minimizes:

$$\mathcal{M}^{rs} = \frac{1}{2} \left\| \nabla s(\mathbf{f}(\mathbf{x}))^{\mathrm{T}} \mathbf{c}_1(\mathbf{f}(\mathbf{x})) \right\|^2 \quad \text{s.t.} \quad \lambda_1(\mathbf{f}(\mathbf{x})) < 0, \tag{14}$$

$$\frac{\delta \mathcal{M}^{rs}}{\delta \mathbf{f}(\mathbf{x})} = \left(\nabla s^{\mathrm{T}} \mathbf{c}_1 \right) \cdot \begin{pmatrix} \frac{\partial \nabla s}{\partial f_1}^{\mathrm{T}} \mathbf{c}_1 + \nabla s^{\mathrm{T}} \frac{\partial \mathbf{c}_1}{\partial f_1} \\ \vdots \\ \frac{\partial \nabla s}{\partial f_m}^{\mathrm{T}} \mathbf{c}_1 + \nabla s^{\mathrm{T}} \frac{\partial \mathbf{c}_1}{\partial f_m} \end{pmatrix}^{\mathrm{T}}, \tag{15}$$

where we dropped the dependencies for brevity, i.e., $\nabla s := \nabla s(\mathbf{f}(\mathbf{x}))$ and $\mathbf{c} := \mathbf{c}_1(\mathbf{f}(\mathbf{x}))$. Valley surfaces of a scalar field $s(\mathbf{y})$ are analogously found as ridge surfaces of the negated scalar field $-s(\mathbf{y})$.

4.3 Isocontours

In an m-dimensional steady scalar field $s(\mathbf{y}) : \mathbb{R}^m \to \mathbb{R}$, isocontours are the union of locations $\mathbf{y}$ at which the scalar value $s(\mathbf{y})$ is equal to a user-defined isovalue c_0, cf. Eq. (2). In 2D domains, isocontours are referred to as isolines. Analogously, they are called isosurfaces in 3D. We refer to the n-dimensional isocontour as $\mathbf{f}(\mathbf{x}) = \big(f_1(x_1, \dots, x_n), \dots, f_m(x_1, \dots, x_n)\big)^{\mathrm{T}}$ with $n = m - 1$, which minimizes for isovalue c_0 the following Lagrangian, cf. Eq. (2):

$$\mathcal{M}^{i} = \frac{1}{2} \| s(\mathbf{f}(\mathbf{x})) - c_0 \|^2, \tag{16}$$

$$\frac{\delta \mathcal{M}^{i}}{\delta \mathbf{f}(\mathbf{x})} = (s(\mathbf{f}(\mathbf{x})) - c_0) \cdot \frac{\partial s(\mathbf{f}(\mathbf{x}))}{\partial \mathbf{f}(\mathbf{x})}, \tag{17}$$

with $\frac{\partial s(\mathbf{f}(\mathbf{x}))}{\partial \mathbf{f}(\mathbf{x})} = \nabla s(\mathbf{f}(\mathbf{x}))^{\mathrm{T}}$. The term $\frac{\delta \mathcal{M}^{i}}{\delta \mathbf{f}(\mathbf{x})}$ is the corresponding feature term in the Euler-Lagrange equation in Eq. (9). This describes isolines for $m = 2$, $n = 1$ and isosurfaces for $m = 3$, $n = 2$.

4.4 Critical Lines

Locations at which a vector field vanishes to zero are called critical points [Helman and Hesselink 1989]. In a 2-dimensional time-dependent vector field $\mathbf{v}(\mathbf{y}, t) : \mathbb{R}^2 \times \mathbb{R} \to \mathbb{R}^2$, critical points move over time. Their path is called a critical line, which is assembled by all points $(\mathbf{y}, t)$ at which the velocity vanishes, i.e., $\mathbf{v}(\mathbf{y}, t) = \mathbf{0}$. The paths of critical points are a key ingredient in the extension of classic steady vector field topology [Helman and Hesselink 1989, 1991] to the unsteady case [Bujack et al. 2020].

Given a 2-dimensional, time-dependent vector field $\mathbf{v}(\mathbf{y}, t) : \mathbb{Y} \times \mathbb{R} \to \mathbb{R}^2$ with $\mathbf{v}(\mathbf{y}, t) = (u(y_1, y_2, t), v(y_1, y_2, t))^{\mathrm{T}}$ a 1-dimensional critical line in space-time $\mathbf{f}(x) = \big(f_1(x), \dots, f_3(x)\big)^{\mathrm{T}}$ is characterized by

$$\mathcal{M}^{c} = \frac{1}{2} \| \mathbf{v}(\mathbf{f}(x)) \|^2, \tag{18}$$

$$\frac{\delta \mathcal{M}^{c}}{\delta \mathbf{f}(x)} = \mathbf{v}(\mathbf{f}(x))^{\mathrm{T}} \cdot \frac{\partial \mathbf{v}(\mathbf{f}(x))}{\partial \mathbf{f}(x)}, \tag{19}$$

where $\frac{\partial \mathbf{v}(\mathbf{f}(x))}{\partial \mathbf{f}(x)} = \nabla \mathbf{v}(\mathbf{f}(x))$ is the space-time Jacobian matrix at $\mathbf{f}(x)$. Note that the third dimension $f_3(x)$ denotes the time coordinate.

4.5 Parallel Vector Lines

The parallel vectors operator [Peikert and Roth 1999] locates curves along which two vector fields are parallel, cf. Eq. (3). Given are two

3-dimensional vector fields $\mathbf{v}_1(\mathbf{y}), \mathbf{v}_2(\mathbf{y}) : \mathbb{Y} \to \mathbb{R}^3$, a 1-dimensional parallel vectors solution $\big(f_1(x), \dots, f_3(x)\big)^\mathrm{T}$ is defined by

$$\mathcal{M}^p = \frac{1}{2} \left\| \mathbf{v}_1(\mathbf{f}(x)) \times \mathbf{v}_2(\mathbf{f}(x)) \right\|^2, \tag{20}$$

$$\frac{\delta \mathcal{M}^p}{\delta \mathbf{f}(x)} = (\mathbf{v}_1 \times \mathbf{v}_2)^\mathrm{T} \cdot \begin{pmatrix} (\frac{\partial \mathbf{v}_1}{\partial f_1} \times \mathbf{v}_2 + \mathbf{v}_1 \times \frac{\partial \mathbf{v}_2}{\partial f_1})^\mathrm{T} \\ \vdots \\ (\frac{\partial \mathbf{v}_1}{\partial f_m} \times \mathbf{v}_2 + \mathbf{v}_1 \times \frac{\partial \mathbf{v}_2}{\partial f_m})^\mathrm{T} \end{pmatrix}^\mathrm{T}, \tag{21}$$

where we dropped the dependencies for brevity, i.e., $\mathbf{v}_1 := \mathbf{v}_1(\mathbf{f}(\mathbf{x}))$ and $\mathbf{v}_2 := \mathbf{v}_2(\mathbf{f}(\mathbf{x}))$. The operator is also used in the next section.

4.6 Vortex Corelines and Bifurcation Lines

In flow visualization, vortex corelines are lines around which particles are rotating. Given is a 3-dimensional steady vector field $\mathbf{v}(\mathbf{y}) : \mathbb{R}^3 \to \mathbb{R}^3$, with its Jacobian $\nabla \mathbf{v}(\mathbf{y})$, which has three eigenvectors $\mathbf{c}_i(\mathbf{y})$ with corresponding eigenvalues $\lambda_i(\mathbf{y})$, i.e., $\nabla \mathbf{v}(\mathbf{y})\ \mathbf{c}_i(\mathbf{y}) = \lambda_i(\mathbf{y})\ \mathbf{c}_i(\mathbf{y})$. In the presence of two complex eigenvalues (w.l.o.g. let λ_1 and λ_2 be complex-valued), Sujudi and Haimes [1995] characterized the vortex coreline as the curve formed by locations $\mathbf{y}$ at which the flow vector $\mathbf{v}(\mathbf{y})$ points in direction of the eigenvector $\mathbf{c}_3(\mathbf{y})$ with real eigenvalue $\lambda_3(\mathbf{y})$, which implies that $\mathbf{v}$ is an eigenvector and hence $\mathbf{v} \parallel \mathbf{J}\mathbf{v}$.

Given a 3-dimensional steady vector field $\mathbf{v}(\mathbf{y}) : \mathbb{Y} \to \mathbb{R}^3$, 1-dimensional vortex corelines (and bifurcation lines) are feature curves $\big(f_1(x), \dots, f_3(x)\big)^\mathrm{T}$ that minimize:

$$\mathcal{M}^v = \frac{1}{2} \left\| \mathbf{v}(\mathbf{f}(x)) \times (\nabla \mathbf{v}(\mathbf{f}(x)))\mathbf{v}(\mathbf{f}(x)) \right\|^2, \tag{22}$$

$$\frac{\delta \mathcal{M}^v}{\delta \mathbf{f}(x)} = (\mathbf{v} \times (\nabla \mathbf{v})\mathbf{v})^\mathrm{T} \tag{23}$$

$$\cdot \begin{pmatrix} (\frac{\partial \mathbf{v}}{\partial f_1} \times ((\nabla \mathbf{v})\mathbf{v}) + \mathbf{v} \times (\frac{\partial \nabla \mathbf{v}}{\partial f_1}\mathbf{v} + (\nabla \mathbf{v})\frac{\partial \mathbf{v}}{\partial f_1}))^\mathrm{T} \\ \vdots \\ (\frac{\partial \mathbf{v}}{\partial f_m} \times ((\nabla \mathbf{v})\mathbf{v}) + \mathbf{v} \times (\frac{\partial \nabla \mathbf{v}}{\partial f_m}\mathbf{v} + (\nabla \mathbf{v})\frac{\partial \mathbf{v}}{\partial f_m}))^\mathrm{T} \end{pmatrix}^\mathrm{T}. \tag{24}$$

An eigenanalysis distinguishes between vortex corelines [Sujudi and Haimes 1995] and bifurcation lines [Roth 2000]. For time-dependent flow, it was proposed to subtract the feature flow field [Theisel and Seidel 2003; Weinkauf et al. 2007] first for Galilean invariant features, or to perform a reference frame optimization [Baeza Rojo and Günther 2019; Günther et al. 2017; Hadwiger et al. 2019; Rautek et al. 2020, 2024] first for objective features.

4.7 Parallel Eigenvector Lines

Similar to the parallel vectors operator of vector fields, the parallel eigenvectors operator [Oster et al. 2018b] is used for the analysis of tensor fields. Given the two three-dimensional tensor fields $\mathbf{S}(\mathbf{y}), \mathbf{T}(\mathbf{y}) : \mathbb{Y} \to \mathbb{R}^{3\times3}$, the parallel eigenvectors operator searches for locations $\mathbf{y}$ at which there exists a direction vector $\mathbf{r} \neq \mathbf{0} \in \mathbb{R}^3$ for which holds:

$$\big\{ \mathbf{y} : (\exists \mathbf{r} \in \mathbb{R}^3)\ [\ \mathbf{r} \parallel \mathbf{S}(\mathbf{y})\mathbf{r} \parallel \mathbf{T}(\mathbf{y})\mathbf{r}\ \wedge\ \mathbf{r} \neq \mathbf{0}\ ] \big\}. \tag{25}$$

The vector $\mathbf{r}$ thereby becomes an eigenvector of both $\mathbf{S}$ and $\mathbf{T}$. We introduce a 6D feature $\mathbf{f}(x) = (\mathbf{g}(x), \mathbf{r}(x))$ where $\mathbf{g}(x)$ is the unknown position and $\mathbf{r}(x)$ is the unknown direction:

$$\mathcal{M}^{pe} = \frac{1}{2} \left(\|\mathbf{S}(\mathbf{g})\mathbf{r} \times \mathbf{r}\|^2 + \|\mathbf{T}(\mathbf{g})\mathbf{r} \times \mathbf{r}\|^2 + \frac{1}{2} \left(\|\mathbf{r}\|^2 - 1 \right)^2 \right), \tag{26}$$

$$\frac{\delta \mathcal{F}^{pe}}{\delta \mathbf{g}(x)} = (\mathbf{S}(\mathbf{g})\mathbf{r} \times \mathbf{r})^\mathrm{T} \cdot \left(\frac{\partial \mathbf{S}(\mathbf{y})}{\partial f_1}\mathbf{r} \times \mathbf{r}, \dots, \frac{\partial \mathbf{S}(\mathbf{y})}{\partial f_3}\mathbf{r} \times \mathbf{r} \right) \tag{27}$$

$$+ (\mathbf{T}(\mathbf{g})\mathbf{r} \times \mathbf{r})^\mathrm{T} \cdot \left(\frac{\partial \mathbf{T}(\mathbf{y})}{\partial f_1}\mathbf{r} \times \mathbf{r}, \dots, \frac{\partial \mathbf{T}(\mathbf{y})}{\partial f_3}\mathbf{r} \times \mathbf{r} \right) \tag{28}$$

$$\frac{\delta \mathcal{F}^{pe}}{\delta \mathbf{r}(x)} = (\mathbf{S}(\mathbf{g})\mathbf{r} \times \mathbf{r})^\mathrm{T} \cdot \begin{pmatrix} (\mathbf{S}(\mathbf{g})\mathbf{r} \times \hat{\mathbf{e}}_1 + \mathbf{S}(\mathbf{g})\hat{\mathbf{e}}_1 \times \mathbf{r})^\mathrm{T} \\ (\mathbf{S}(\mathbf{g})\mathbf{r} \times \hat{\mathbf{e}}_2 + \mathbf{S}(\mathbf{g})\hat{\mathbf{e}}_2 \times \mathbf{r})^\mathrm{T} \\ (\mathbf{S}(\mathbf{g})\mathbf{r} \times \hat{\mathbf{e}}_3 + \mathbf{S}(\mathbf{g})\hat{\mathbf{e}}_3 \times \mathbf{r})^\mathrm{T} \end{pmatrix}^\mathrm{T} \tag{29}$$

$$+ (\mathbf{T}(\mathbf{g})\mathbf{r} \times \mathbf{r})^\mathrm{T} \cdot \begin{pmatrix} (\mathbf{T}(\mathbf{g})\mathbf{r} \times \hat{\mathbf{e}}_1 + \mathbf{T}(\mathbf{g})\hat{\mathbf{e}}_1 \times \mathbf{r})^\mathrm{T} \\ (\mathbf{T}(\mathbf{g})\mathbf{r} \times \hat{\mathbf{e}}_2 + \mathbf{T}(\mathbf{g})\hat{\mathbf{e}}_2 \times \mathbf{r})^\mathrm{T} \\ (\mathbf{T}(\mathbf{g})\mathbf{r} \times \hat{\mathbf{e}}_3 + \mathbf{T}(\mathbf{g})\hat{\mathbf{e}}_3 \times \mathbf{r})^\mathrm{T} \end{pmatrix}^\mathrm{T} \tag{30}$$

$$+ \mathbf{r}^\mathrm{T}(\|\mathbf{r}\|^2 - 1). \tag{31}$$

The third term in Eq. (26) prevents the direction $\mathbf{r}$ from vanishing to zero. In the equations above, we removed the dependencies $\mathbf{g} := \mathbf{g}(x)$ and $\mathbf{r} := \mathbf{r}(x)$ for brevity. The symbols $\hat{\mathbf{e}}_1 = (1,0,0)^\mathrm{T}$, $\hat{\mathbf{e}}_2 = (0,1,0)^\mathrm{T}$, $\hat{\mathbf{e}}_3 = (0,0,1)^\mathrm{T}$ represent the unit basis vectors.

5 REGULARIZERS

The feature definitions of the previous section can be complemented with regularizers that allow for the modeling of additional domain knowledge. Note that each of the feature definitions above can also be added as regularizer, for example when looking for a compromise between two feature definitions. We discuss such a setting later when studying the extraction of atmospheric jet streams.

5.1 Proximity

If for each feature point $\mathbf{f}(\mathbf{x})$ a reference position $\mathbf{c}(\mathbf{x}) : \mathbb{X} \to \mathbb{Y}$ is known that the feature should remain close to, the following regularizer can be added with its functional derivative:

$$\Gamma^p = \frac{1}{2}\|\mathbf{f}(\mathbf{x}) - \mathbf{c}(\mathbf{x})\|^2, \qquad \frac{\delta \Gamma^p}{\delta \mathbf{f}} = (\mathbf{f}(\mathbf{x}) - \mathbf{c}(\mathbf{x}))^\mathrm{T}. \tag{32}$$

This regularizer requires vertex correspondence between the feature and the reference geometry by means of a parameterization.

5.2 Smoothness

For both feature curves and feature surfaces, a smooth solution is obtained when minimizing the Dirichlet energy:

$$\Gamma^s = \frac{1}{2}\|\nabla \mathbf{f}(\mathbf{x})\|^2, \qquad \frac{\delta \Gamma^s}{\delta \mathbf{f}} = -\nabla^2 \mathbf{f}(\mathbf{x})^\mathrm{T}. \tag{33}$$

A gradient descent based minimization of the variational problem leads to the well-known Laplacian smoothing, which uses the Laplace operator $\nabla^2 \mathbf{f}(\mathbf{x})$.

5.3 Flow Alignment

The tangent $\nabla \mathbf{f}(x) = \frac{\mathrm{d}\mathbf{f}(x)}{\mathrm{d}x}$ of a feature curve ($n = 1$) can be aligned with a vector field $\mathbf{v}(\mathbf{y}) : \mathbb{Y} \to \mathbb{R}^m$ by using the regularizer:

$$\Gamma^a = \frac{1}{2}\|\mathbf{v}(\mathbf{f}(x)) - \nabla \mathbf{f}(x)\|^2, \tag{34}$$

$$\frac{\delta \Gamma^a}{\delta \mathbf{f}} = (\mathbf{v}(\mathbf{f}(x)) - \nabla \mathbf{f}(x))^{\mathrm{T}} \cdot \frac{\partial \mathbf{v}(\mathbf{f}(x))}{\partial \mathbf{f}(x)} \tag{35}$$

$$+ \left(\frac{\partial \mathbf{v}(\mathbf{f}(x))}{\partial \mathbf{f}(x)} \nabla \mathbf{f}(x) \right)^{\mathrm{T}} - \nabla^2 \mathbf{f}(x)^{\mathrm{T}}. \tag{36}$$

This regularizer requests features to be flow-aligned, making them behave like streamlines in steady flow or pathlines in unsteady flow. As a hard constraint, the alignment can be requested with Neumann boundary conditions $\nabla \mathbf{f}(x_0) = \mathbf{v}(\mathbf{f}(x_0))$, for example at $x_0 \in \mathbb{X}$.

6 NUMERICAL EXTRACTION

In our variational framework, we compute optimal feature curves and surfaces by applying a gradient descent scheme, cf. Eq. (11). First, we describe how the feature curves and surfaces are discretized and how derivatives are estimated. Second, we discuss two methods for the initialization of the gradient-based optimization. And third, we propose a grow-refine algorithm, which is inspired from predictor-corrector methods [Banks and Singer 1995]. The approach starts from a single feature point and then incrementally grows the feature. After each growing step, the current feature is iteratively refined to keep a valid solution to the Euler-Lagrange equation. We provide a demo implementation at https://github.com/fau-vc/vfe-demo.

6.1 Discretization

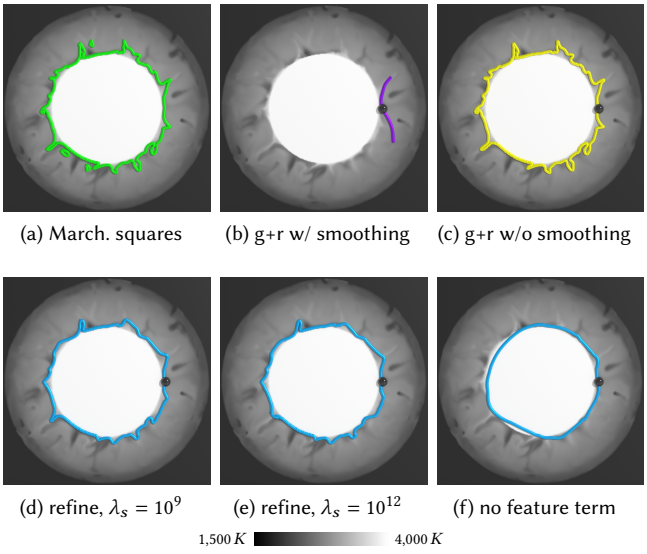
Up until here, our approach could be described without any mentioning of the underlying discretization, since all features and their necessary conditions have been introduced in a continuous setting. For a given choice of discretization, three operations need to be defined: the evaluation of the function $\mathbf{f}(\mathbf{x})$ at a given location $\mathbf{x}$, its first derivative $\nabla \mathbf{f}(\mathbf{x})$, and its second derivative $\nabla^2 \mathbf{f}(\mathbf{x})$.

6.1.1 Curve Discretization. We discretize 1-dimensional curve features ($n = 1$) as piece-wise linear polylines with uniform parameterization. The function value $\mathbf{f}(x)$ is linearly interpolated from the curve vertices, and the derivatives $\nabla \mathbf{f}(x)$ and $\nabla^2 \mathbf{f}(x)$ are estimated with second-order accurate finite differences. Due to the uniform parameterization, all three can be computed in constant time. For non-uniform parameterizations, standard algorithms based on Newton interpolation [Fornberg 1988] would be available, too.

6.1.2 Surface Discretization. We discretize surfaces ($n = 2$) using triangle meshes with piece-wise linear (barycentric) interpolation. At triangle vertices $\mathbf{f}_i$, we calculate the derivatives using the commonly-used cotangens formulas [Pinkall and Polthier 1993; Reuter et al. 2009], which iterate the vertices j in the 1-ring $\mathcal{N}(i)$ of vertex i:

$$\nabla \mathbf{f}_i \approx \frac{1}{2A_i} \sum_{j \in \mathcal{N}(i)} \omega_{ij}(\mathbf{f}_i - \mathbf{f}_j)^2, \quad \nabla^2 \mathbf{f}_i \approx \frac{1}{A_i} \sum_{j \in \mathcal{N}(i)} \omega_{ij}(\mathbf{f}_i - \mathbf{f}_j) \tag{37}$$

with $\omega_{ij} = \frac{1}{2}(\cot \alpha_{ij} + \cot \beta_{ij})$ and A_i being the Voronoi area of vertex i. In the two triangles that share the edge (i, j), the angles α_{ij} and β_{ij} are measured on the corners opposite to the shared edge.



(a) March. squares (b) g+r w/ smoothing (c) g+r w/o smoothing

(d) refine, $\lambda_s = 10^9$ (e) refine, $\lambda_s = 10^{12}$ (f) no feature term

Fig. 4. Isocontour extraction for isovalue $T_0 = 3{,}000\,K$: (a) marching squares [Lorensen and Cline 1987], (b) grow-and-refine (g+r) with too strong smoothness regularization fails, (c) growth without smoothness regularization succeeds. The second row shows subsequent refinement of (c) to obtain a smooth curve: (d) and (e) contain a feature term, while in (f) there is no feature term which lets the contour move incorrectly into the Earth core.

6.2 Initialization

The iterative gradient descent procedure in Eq. (11) requires an initial guess $\mathbf{f}_i^{(0)}$ for each vertex of the feature. In the following, we discuss two alternative approaches at the example of isoline extraction in Fig. 4, which shows (smoothed) isocontours of the temperature field in an Earth mantle convection simulation. Both of the approaches have certain advantages and disadvantages, and hence there is an application-dependent trade-off to be made.

6.2.1 Start from Baseline. If an existing baseline algorithm delivers results that are already close to the desired regularized feature, then the result of the baseline algorithm is a suitable starting point. For example, we could begin with the largest connected component of the marching squares algorithm in Fig. 4a and apply a smoothness regularization to it if we are looking for smooth boundaries of hot plumes. The benefit of this approach is that no feature is missed that the baseline approach would have been able to find. The approach, however, is not applicable if the desired feature is far from the output of a simple baseline algorithm. This occurs when the feature is a mixture of multiple competing constraints. Further, starting from an already formed initial feature curve or surface complicates the adaption of the topology during the optimization, since our ideal feature might require splitting and/or merging. And lastly, the implementation of a baseline algorithm is required, which may be complicated depending on the type of feature we are interested in.

6.2.2 Grow and Refine. Alternatively, one may also grow the feature from one single seed point in a predictor-corrector manner. In a

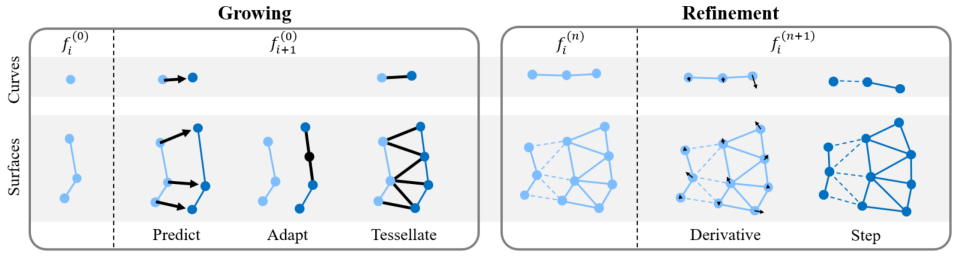


Fig. 5. Schematic overview of our grow and refine feature extraction algorithm. To numerically extract curve and surface features, we alternate between two steps: a growing step and a refinement step. For each feature, we keep an active front, which is moved forward in the prediction step. In case of surface extraction, the front is subdivided when adjacent feature points move too far apart. Afterwards, the space between the previous front and the new front is tessellated. In the subsequent refinement step, all feature points of the k last fronts are iteratively updated by evaluating the functional derivative $\delta\mathcal{F}/\delta\mathbf{f}$ for each feature point, and by applying a gradient descent step. Multiple refinement iterations are used until the feature converges. The feature is iteratively grown and refined until it reaches the desired size, reaches a domain boundary, or forms a closed loop.

growth step, the feature is expanded, for example by adding points in tangential direction at the feature boundary. In a subsequent refinement step, the gradient descent of Eq. (11) is iterated until the feature has converged. Alternating, the feature can then be grown and refined further. A benefit of this approach is that the feature can be grown from just a single point. The downside, however, is that too strong regularizers during growth might prevent the feature from fulfilling the feature term well. Consider, for example, the setting in Fig. 4b, where the seed point is depicted by a black sphere. Here, the smoothness regularization is so strong that the feature curve is unable to stay close to the isocontour. If, however, the feature curve is first grown with low regularization as in Fig. 4c, which gives results equivalent to marching squares, then the smoothness term can be increased afterwards, as shown in Figs. 4d-4e for different smoothness weights. This shows that regularizers have to be applied with care. Not only can a too strong regularizer prevent the extraction of the desired feature, a user may also be biased to confirm their believes by steering the parameters to see the shape they want to see. For this reason, it is valuable to explore the impact of energy weight variations. Despite those limitations, this approach has the advantage that it does not require the implementation of a baseline algorithm to obtain an initial guess. When the parameters are chosen carefully, it can extract features that are compromises of multiple constraints. To demonstrate the benefit of having a feature term, Fig. 4f shows post-smoothing without a feature term, which has the curve shrink into the Earth's core.

6.3 Grow and Refine

A schematic illustration of the grow and refine procedure is presented in Fig. 5 for both curves and surfaces. In the following, we explain the individual steps in more detail.

6.3.1 Growing. When growing a feature, we keep track of an active front of points, which is similar to stream surface computation algorithms [Hultquist 1992; McLoughlin et al. 2010]. The placement of seed points is application-dependent, for example local extrema for an extremal line computation might be calculated to start the variational extraction from local scalar field extrema. For both curve and surface features, the growing procedure consists of three steps.

Predict. For each active point $\mathbf{f}_i^{(0)}$ on the feature boundary, the curve or surface is grown by predicting the next location $\mathbf{f}_{i+1}^{(0)}$:

$$\mathbf{f}_{i+1}^{(0)} \approx \mathbf{f}_i^{(0)} + h \cdot \frac{\partial \mathbf{f}_i^{(0)}}{\partial x_n}. \tag{38}$$

In principle, different direction choices for $\frac{\partial \mathbf{f}_i^{(0)}}{\partial x_n}$ are imaginable. We chose to grow the features tangentially (for curve features), or orthogonal to both the active front and the surface normal (for surface features). When tracking structures over time, such as critical points, feature flow fields [Theisel et al. 2005; Theisel and Seidel 2003] can be used to estimate the next feature point in space-time.

Adapt. For surface features, points on the active front might drift apart, which requires adaptive refinement. If the distance between two adjacent points exceeds a threshold, then the line segment is subdivided in the middle into two segments. Such a front line refinement is again similar to the computation of stream surfaces [McLoughlin et al. 2010]. For line features, no front adaptation is needed, since fronts contain only a single point. At present we assume that features are manifold and differentiable. Modeling topological changes (bifurcations and reconnections) could be desirable in the future, depending on the feature type.

Tessellate. Lastly, in the free space between the previous active front and the current active front, line segments (for curve features) or triangles (for surface features) are formed by iterating both fronts and by deciding the orientation of the next edge to insert by choosing the edge with shortest length. This approach is a key ingredient in the stream ribbon triangulation of Hultquist [1992].

Termination. The growing procedure is terminated when a maximum number of growing steps was performed, when the prediction leaves the domain $\mathbb{Y}$, when the feature forms a closed loop, or when the additional constraints of the feature definition no longer hold, see Section 4. For example, vortex corelines are terminated, when the eigenvalues of the Jacobian matrix are no longer complex, i.e., when no swirling motion is present.

6.3.2 Refinement. Similar to predictor-corrector methods [Banks and Singer 1995], the prediction of the next feature point is not perfect and requires an iterative improvement. Thus, for each prediction step a series of refinement steps follows.

Derivative. For a given feature point $\mathbf{f}_i$, the direction in which the feature point should move is obtained from the functional derivative of the feature $\frac{\delta \mathcal{F}}{\delta \mathbf{f}}$, i.e., its Euler-Lagrange equation, cf. Eq. (6). The computation of the change in the feature position is local and only needs $\mathbf{f}_i$, $\nabla \mathbf{f}_i$ and $\nabla^2 \mathbf{f}_i$, cf. Section 6.1.

Step. To improve the estimate of a feature point $\mathbf{f}_i$ that is not pinned by a boundary condition, we employ gradient descent using the functional derivatives, see Eq. (11).

Termination. The refinement procedure is terminated when a maximum number of iterations is reached or when the magnitude of the functional derivative (i.e., the gradient of the optimization) falls below a user-defined threshold.

Once the refinement is completed, the next active front can be estimated in a growing step, which leads to an iterative alternation between growing and refinement steps. A key difference to the predictor-corrector method [Banks and Singer 1995] is that in our formulation all vertices are able to refine their position in the refinement step, since newly added feature points might require a change in former feature points to meet for example smoothness regularization conditions. The refinement of the entire feature, however, entails an algorithmic problem due to quadratic time complexity: The loop over *all* feature points after each single growth step is quickly becoming too expensive when the number of feature points increases. We accelerate this by only updating the last k active fronts, since we observed that a feature point is unlikely to move later when regularizers have local support.

7 APPLICATIONS

Our approach is applicable in many scientific disciplines. In the following, we describe how variational optimizations can be formulated using examples from aerodynamics, geophysics, atmospheric sciences, fluid dynamics, and oceanography. Further, we compare with baseline approaches, analyze the impact of smoothness in the grow-and-refine method, perform a parameter study, analyze the minimization of individual energy terms, and conduct performance measurements. In the additional material, we present further applications in wind engineering and Morse theory, we demonstrate a multi-scale optimization to deal with noisy field data, we discuss the seed dependence, and we examine the convergence behavior.

7.1 Aerodynamics

We begin with vortex coreline extraction in aerodynamics. When air flows at a high angle of attack over the leading edge of a delta wing, a vortex generates that is parallel to the leading edge, see Fig. 2. The vortex remains nearly stationary above the wing and causes an air flow pattern that lowers air pressure and thereby generates so-called vortex lift. The numerical simulation that is available to us resolves the domain using a tetrahedral grid, which is adaptively subdivided. In the regime behind the delta wing, where the outflow of the vortex is transported by the passing flow, the resolution is lower than near the wing geometry. This difference in resolution causes instabilities with conventional vortex coreline extraction methods. By incorporating a smoothness regularization, we are able to extract a smooth vortex coreline, minimizing:

$$\mathcal{L}^{\text{vortex}} = \underbrace{\frac{1}{2}\|\mathbf{v}(\mathbf{f}(\mathbf{x})) \times (\nabla \mathbf{v}(\mathbf{f}(\mathbf{x})))\mathbf{v}(\mathbf{f}(\mathbf{x}))\|^2}_{\text{vortex coreline}} + \underbrace{\frac{\lambda_s}{2}\|\nabla \mathbf{f}(\mathbf{x})\|^2}_{\text{smoothness}}, \quad (39)$$

where $\mathbf{v}(\mathbf{y})$ is a steady 3-dimensional vector field. We set the smoothness weight to $\lambda_s = 10^8$ and used Adam (learning rate $h = 10^{-4}$) starting from the parallel vectors solution. Note that the domain of this data set is large, which entails the large energy weight. Weight normalization is discussed later in Section 8.

Comparison with Parallel Vectors. In Fig. 2 (left), we show the extraction result using the parallel vectors operator [Peikert and Roth 1999], which notoriously leads to spurious results [Günther and Theisel 2018]. In Fig. 2 (middle), the largest connected component is shown, which removes the short line segments. However, the vortex coreline remains noisy in the wake area, which is especially visible in the close-up. By using our variational approach, we are able to determine a smooth vortex coreline. The experiment is continued in Fig. 6, where pre- and post-processing of a parallel vectors solution are compared with our variational method. For all methods, close-ups with different parameter choices are shown. Pre-processing the input fields by applying Gaussian smoothing in Fig. 6(a) causes strong artifacts where the vortex coreline is close to the leading edge of the delta wing. Post-processing by Laplacian smoothing of the parallel vectors curve in Fig. 6(b) pulls the curve away (b.3), which is visible by the yellow curve shining through, especially where the parallel vectors curve is bent. The variational method in Fig. 6(c), on the other hand, obtains a smooth solution that remains close to the parallel vectors curve, due to the combination of a feature term with a smoothing term. In the close-ups, it can be seen that all three methods require careful selection of the smoothing parameters. With our approach, however, the resulting feature stays close to the feature longer, due to the feature term (c.3).

7.2 Geophysics

Geophysical simulations of the Earth mantle model convection processes of material over millions of years [Gülcher et al. 2021]. Rising from the core's surface, which is at over 5,000 K, there are hot plumes that reach into the mantle. To locate plumes that rise from the core, we extract isocontours using:

$$\mathcal{L}^{\text{plume}} = \underbrace{\frac{1}{2}\|T(\mathbf{f}(x)) - T_0\|^2}_{\text{isocontour}} + \underbrace{\frac{\lambda_s}{2}\|\nabla \mathbf{f}(x)\|^2}_{\text{smoothness}}, \quad (40)$$

where $T(\mathbf{y})$ is the temperature field and $T_0 = 2{,}800\,K$ is the isovalue.

Impact of Smoothness on Growing. In Section 6.2, we discussed the impact of the smoothing regularization on the ability of the grow and refine procedure to stay close to the desired feature. We have shown in Fig. 4 that a split into first tracing out the feature with low smoothness regularization, followed by an increase of the smoothness afterwards, made it possible to extract smooth isocontours. Here, gradient descent (step size $h = 10^{-2}$) was used.

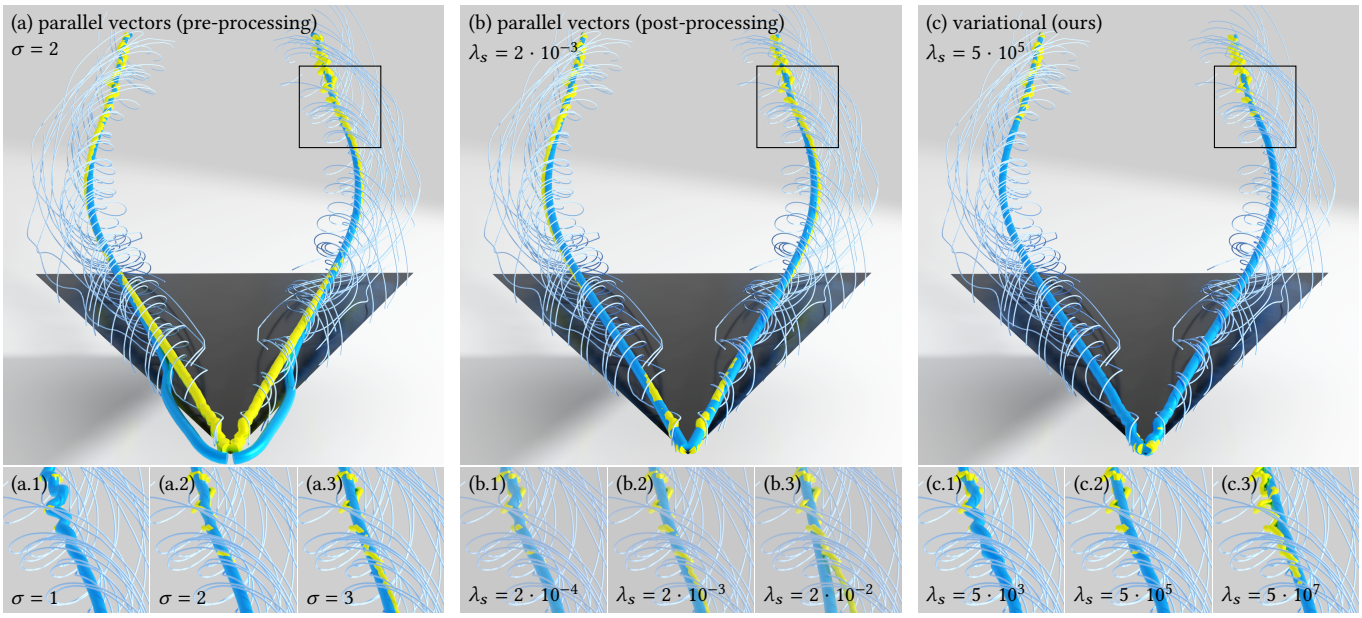


Fig. 6. Comparing three different strategies that lead to smoother vortex corelines. The top row shows the vortex coreline (blue) alongside the unsmoothed parallel vectors solution (yellow). Below, the smoothness parameters were varied to show how this affects the noisy part of the extraction. From left to right, we see a prior Gaussian smoothing of the input vector field, a Laplacian smoothing of the extract parallel vector curve, and our variational method, which combines feature and regularizer. The latter results in smooth vortex corelines that remain close to the underlying parallel vectors solution.

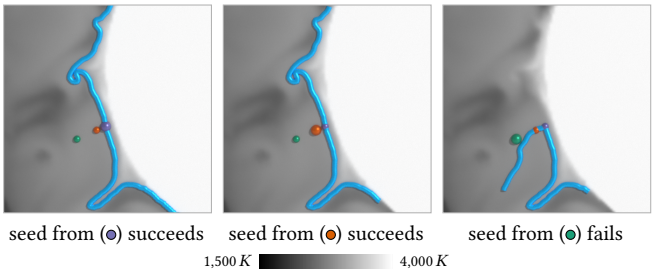


Fig. 7. Study of seed point sensitivity during isocontour extraction in the Earth mantle. From left to right, the seed point was moved further away from the true isocontour. The used seed point is enlarged while the other two seed points are shown to provide context.

Seed Point Sensitivity. In Fig. 7, we varied the seed point location to examine the robustness. If the seed point is close enough to the ground truth, the refinement pulls the seed onto the correct feature. In the third image, the seed point is too far away, and hence half of the feature curve is missed. This shows that seeding is important.

7.3 Atmospheric Sciences

In the atmospheric sciences, atmospheric jet streams are a highly relevant feature curve [Bösiger et al. 2022; Koch et al. 2006; Manney and Hegglin 2018], since their presence and geometric shape have a significant influence on the mid-latitude weather evolution [Harnik et al. 2016; Nielsen-Gammon 2001]. Knowledge about the geometric shape of a jet is not only interesting for weather forecasters, who are interested in the spread of jet streams across ensemble members, but it is also interesting to study how the shape varies statistically across climate simulation runs to assess the changes for future climate scenarios. To open the path to a quantitative analysis of jets, we numerically extract jets as minimizers of the following terms:

$$\mathcal{L}^{\text{jet}} = \underbrace{\frac{1}{2}\,\|\nabla s(\mathbf{f}(x)) \times \mathbf{c}_3(\mathbf{f}(x))\|^2}_{\text{ridge line}} + \underbrace{\frac{\lambda_s}{2}\,\|\nabla \mathbf{f}(x)\|^2}_{\text{smoothness}} + \underbrace{\frac{\lambda_i}{2}\,\|p(\mathbf{f}(x)) \pm 2\|^2}_{\text{tropopause proximity}} \tag{41}$$

where $s(\mathbf{y})$ is the wind magnitude of the air flow and $p(\mathbf{y})$ is the so-called potential vorticity field [Bader et al. 2020]. We follow Bösiger et al. [2022] and consider the tropopause as isocontour with potential vorticity value of +2 on the Northern hemisphere and a value of −2 on the Southern hemisphere. In the literature, other thresholds have been used, as well [Dameris 2015]. We used a gradient descent (step size $h = 10^{-2}$) in the grow-and-refine method.

Comparison with other Methods. Fig. 8 compares jet extraction results using a local ridge line extraction based on parallel vectors, cf. Eq. (1), a parallel vectors formulation of the jet criterion of Kern et al. [2018], and a recent predictor-corrector method [Bösiger et al. 2022] with our variational approach for $\lambda_s = 10^3$ and $\lambda_i = 0.01$. The integration-based methods trace the jets from local wind extrema that have a wind speed of at least 40 ms^{-1} and that are found at an altitude range between 150 and 400 hPa. Both local parallel vectors approaches results in spurious curves, which is a common problem [Günther and Theisel 2018]. The predictor-corrector method [Bösiger et al. 2022] achieves a smoothing effect by lowering the

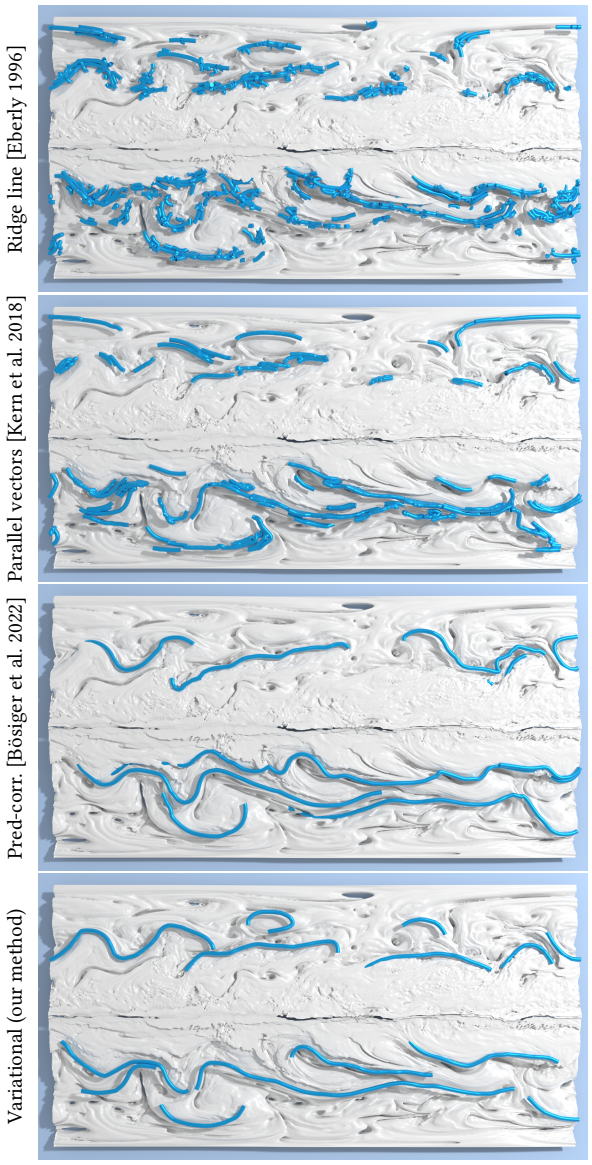


Fig. 8. Comparison with other jet stream extraction methods in a global ERA5 reanalysis simulation [Hersbach et al. 2020] of the atmosphere on Sep 2nd 2016, 23:00 UTC. To overcome spurious results and discretization issues in the data, our variational approach combines three pieces of domain knowledge to formalize the structure of interest as variational minimizer: aim for high wind speed (using Eq. (13)), spatial smoothness (using Eq. (33)) and proximity to the tropopause (using Eq. (17)), shown as gray surface.

number of correction steps, which however, leads to feature curves that are slightly off. Our variational method, on the other hand, is able to compensate for those problems by modeling further domain knowledge, namely smoothness and a general proximity to the tropopause, which is shown as isosurface. Compared to the parallel vectors solutions, the variational approach results in a smaller set of longer lines, which is more suitable for further processing. The

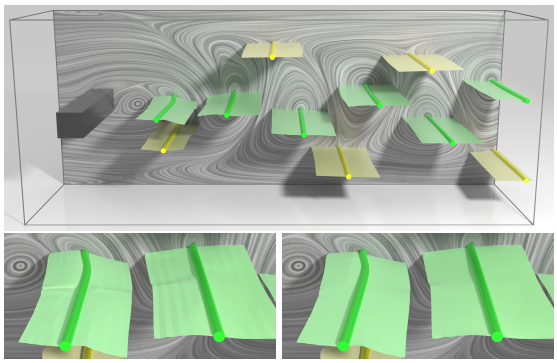


Fig. 9. The top image shows a visualization of the vortex corelines (green) and bifurcation lines (yellow) in the wake of the flow around a square cylinder. In the background, the velocity field is shown in the optimized reference frame using line integral convolution [Cabral and Leedom 1993]. Below, close-ups are shown for different smoothness weights (left: $\lambda_s = 0$, right: $\lambda_s = 10$), demonstrating the utility of the smoothness regularizer.

method, however, requires careful adjustment of energy weights to strike the right balance between the different energy terms. Developing tools that guide the users is an avenue for future work.

7.4 Fluid Dynamics

Vortex corelines and bifurcation lines are flow features that have been studied extensively in the past. Given the time-dependent 3D vector field $\mathbf{v}(\mathbf{y}, t)$ of a von-Kármán vortex street [Camarri et al. 2005], we apply a reference frame optimization [Günther et al. 2017] to subtract the transport direction of the flow features first. This results in a reduced vector field $\mathbf{w}(\mathbf{y}, t)$ that was used in the past to extract vortex corelines and bifurcation lines [Baeza Rojo and Günther 2019]. Our goal is to track the path of the feature curves over time as a surface $\bar{\mathbf{f}}(\mathbf{x}) = (\mathbf{f}(x), t)$ in the space-time domain, using the Lagrangian from Sections 4.6 and 5.2:

$$\mathcal{L}^{\text{track}} = \underbrace{\frac{1}{2}\|\mathbf{w}(\bar{\mathbf{f}}(\mathbf{x})) \times (\nabla \mathbf{w}(\bar{\mathbf{f}}(\mathbf{x}))\mathbf{w}(\bar{\mathbf{f}}(\mathbf{x}))\|^2}_{\text{parallel vectors}} + \underbrace{\frac{\lambda_s}{2}\|\nabla \mathbf{f}(\mathbf{x})\|^2}_{\text{smoothness}} \quad (42)$$

The paths of the feature curves arise as space-time surfaces, which are displayed in Fig. 9 (top). The green surfaces depict the paths of vortex corelines, while the yellow surfaces show the paths of bifurcation lines. The surfaces were extracted using the Adam optimizer (learning rate $h = 10^{-4}$) from a seed line (Sec. 6.2.2). The extraction was stopped after five steps in both forward and backward time to avoid occlusions. Afterwards, the surfaces were optimized (Sec. 6.2.1) using the same learning rate, adding the smoothness regularizer ($\lambda_o = 10$).

Smoothness Study. In Fig. 9 (bottom), the space-time track of a vortex coreline is shown with and without smoothing weight (left: $\lambda_s = 0$, right: $\lambda_s = 10$). By increasing the smoothness regularization, our approach is able to stabilize the feature tracks temporally. In more turbulent flow, vortex breakdown occurs which requires the handling of topological changes such as splits and merges.

7.5 Oceanography

In oceanography, large-scale vortices, also referred to as ocean eddies, are ecosystems that travel across the seas. In an AVISO geostrophic velocity field [Abernathey and Haller 2018], we track the vortex centers of ocean eddies as critical lines in 2D space-time after a prior reference frame optimization [Baeza Rojo and Günther 2019]. With imperfect reference frame optimizations, the resulting vortex corelines are not Lagrangian [Bujack et al. 2020], meaning that they are not pathlines of the flow. Thus, to align vortex corelines with pathlines, we add a flow alignment regularization in space-time. Given the 2-dimensional unsteady vector field, $\mathbf{v}(\mathbf{y}, t)$, we lift the flow into space-time $\overline{\mathbf{v}}(\overline{\mathbf{y}}) = (\mathbf{v}(\mathbf{y}, t), 1)^{\mathrm{T}}$ and denote the reference frame optimized flow as $\mathbf{w}(\overline{\mathbf{y}})$, in which the ambient transport of features was subtracted. We then minimize:

$$\mathcal{L}^{\text{eddy}} = \underbrace{\frac{1}{2}\|\mathbf{w}(\overline{\mathbf{f}}(x))\|^2}_{\text{critical line}} + \underbrace{\frac{\lambda_a}{2}\|\overline{\mathbf{v}}(\overline{\mathbf{f}}(x)) - \nabla\overline{\mathbf{f}}(x)\|^2}_{\text{flow alignment}}. \tag{43}$$

Again, a two-stage process was used to extract the feature lines. First, the lines were extracted with $\lambda_a = 0$ using the grow and refine algorithm. Next, the curves are refined using $\lambda_a = \{1, 2, 3, 4\}$. Both steps used the Adam optimizer (learning rate: $h = 10^{-3}$).

Assessment of Alignment Quality. To assess the impact of the flow alignment regularizer, we varied its weight in Fig. 10. At the top, from left to right, the flow alignment weight is increased. The color-coding shows qualitatively the trend that with growing regularization weight, the feature curves better align with pathlines of the flow. Below, we quantitatively show that the critical line term is well minimized when the flow alignment term is turned off. With increasing flow alignment term, two constraints compete and a compromise between them is reached.

7.6 Extremal Surface

To demonstrate the extraction of a surface feature from an initial guess, we created an extremal surface based on a torus, similar to Kindlmann et al. [2018]. The two-dimensional extremal surface $\mathbf{f}(\mathbf{x})$ is extracted by minimizing the squared norm of the dot product between gradient ∇s and minor eigenvector $\mathbf{c}_1$, cf. Section 4.2:

$$\mathcal{L}^{\text{ridge}} = \underbrace{\frac{1}{2}\left\|\nabla s(\mathbf{f}(\mathbf{x}))^{\mathrm{T}}\mathbf{c}_1(\mathbf{f}(\mathbf{x}))\right\|^2}_{\text{ridge surface}}. \tag{44}$$

The scalar field $s(\mathbf{y})$ is based on the quartic implicit torus equation:

$$t(x, y, z; r, R) = (x^2 + y^2 + z^2 + R^2 - r^2)^2 = 4R^2(x^2 + y^2) \tag{45}$$

where r is the minor (tube) radius of the torus, and $R = 3$ is the major radius. We spatially vary the tube radius using $r' = \frac{1}{2} + \frac{1}{2}\cos(\frac{5}{2}\phi)$, where $\phi = \operatorname{atan2}(y, x)$. From this, we define a scalar field $s(x, y, z) = \exp\left(-t(x, y, z; r', R)^2 \cdot 10^{-5}\right)$, which has a ridge surface at the location of the spatially varied torus. We begin the variational gradient descent (step size $h = 10^{-2}$) from an explicit representation of a simple torus $t(x, y, z; 3/4, R)$. The initial guess and the final converged result are shown in Fig. 11. A convergence plot is shown in the additional material.

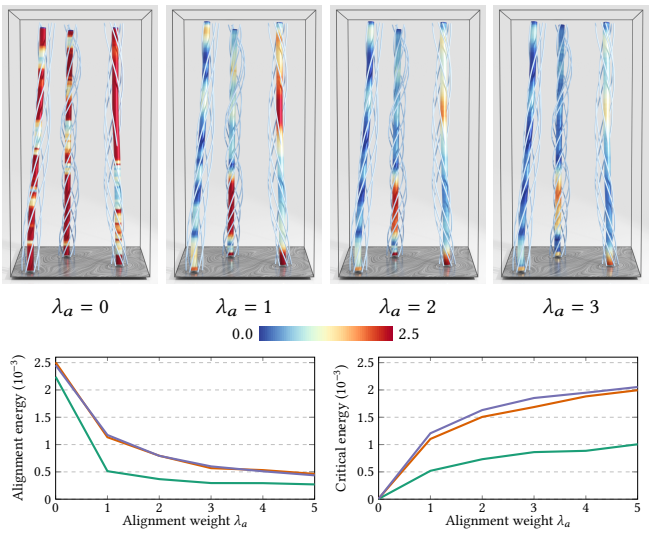


Fig. 10. The top images show the results of the space-time vortex coreline extraction in the AVISO ocean data set. The flow alignment weight λ_a is gradually increased. Swirling pathlines are seeded around the base of the coreline, showing the alignment of the cores with the flow. Additionally, the current energy of the flow alignment term is color-coded on the extracted lines (energy values scaled by 10^4). Non flow-aligned sections are visible for $\lambda_a = 0$ by the red color which corresponds to sections where the vortex coreline is not well-enough aligned with pathlines of the underlying flow. The plots below show the effect of the increased alignment weight on the alignment energy (left) and the critical line energy (right) quantitatively.

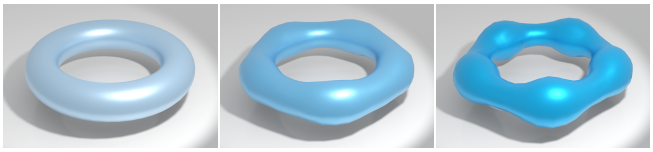


Fig. 11. Here, we demonstrate the convergence of a simple torus shape onto a torus with spatially varying tube radius. The surface arises as ridge surface in an analytic scalar field. The convergence is shown by gradually increasing the number of gradient descent steps. The images show the initialization, an intermediate result (200 steps) and the converged result (1000 steps).

7.7 Tensor Corelines

Tensor fields arise in structural mechanics and fluid mechanics in the form of stress and deformation tensors. In such fields, the behavior of hyperstreamlines in the eigenvector field has been analyzed by extracting feature curves. For a symmetric tensor field $\mathbf{S}(\mathbf{y})$ in a three-dimensional domain $\mathbf{y} \in \mathbb{Y} \subseteq \mathbb{R}^3$, Oster et al. [2018a] introduced the concept of tensor corelines. In analogy to the vortex coreline criterion of Sujudi and Haimes [1995] for steady fluid flow, tensor corelines are curves around which hyperstreamlines of the eigenvector field are swirling. These feature curves are obtained with the parallel eigenvectors operator of Section 4.7 when choosing $\mathbf{T}(\mathbf{y}) = \nabla_{\mathbf{r}}\mathbf{S}(\mathbf{y})$ to be the directional derivative of $\mathbf{S}(\mathbf{y})$ in direction $\mathbf{r}$.

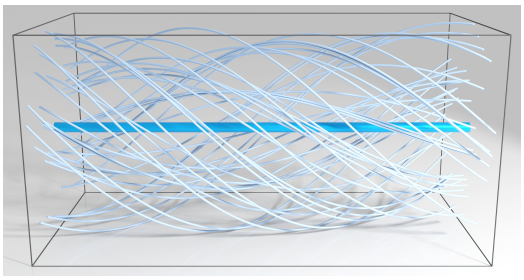


Fig. 12. Tensor corelines (dark blue) are the center lines around which streamlines of the major eigenvector field are swirling (light blue).

We determine such tensor corelines by minimizing:

$$\mathcal{L}^{\text{tensor}} = \underbrace{\frac{1}{2}\left(\|\mathbf{S}(\mathbf{g})\mathbf{r}\times\mathbf{r}\|^2 + \|\mathbf{T}(\mathbf{g})\mathbf{r}\times\mathbf{r}\|^2 + \frac{1}{2}\left(\|\mathbf{r}\|^2 - 1\right)^2\right)}_{\text{tensor coreline}}. \quad (46)$$

with $\mathbf{T}(\mathbf{y}) = \nabla_{\mathbf{r}}\mathbf{S}(\mathbf{y})$ being the directional derivative of $\mathbf{S}(\mathbf{y})$ in direction $\mathbf{r}$. In Fig. 12, we apply our grow-and-refine method with gradient descent (step size $h = 1$) to the analytic tensor field $\mathbf{S}(x, y, z)$, which was provided by Oster et al. [2018a]:

$$\mathbf{S}(x, y, z) = \begin{pmatrix} y^2 & -xy & -y \\ -xy & x^2 & x \\ -y & x & 1 \end{pmatrix} \quad (47)$$

and which we visualized in the domain $[-1, 1]^2 \times [-2, 2]$. In this example, a straight tensor coreline is present around which the streamlines of the major eigenvector field (light blue) are swirling. The sensitivity of the seed point is studied in the additional material.

7.8 Performance

The performance of the variational feature extraction for a particular Lagrangian definition scales linearly in the number of optimization iterations. When initializing from a baseline, the runtime also increases linearly in the size of the input, while the grow-and-refine approach scales with the number of seed points, front size (k) and the number of growing iterations. The extraction time is reported in Table 1 along with the relevant scene parameters for all scenes in the paper, as well as for those in the additional material. The measurements have been taken on a workstation with an Intel Core i9-10980XE CPU with 3.00 GHz. The measurements exclude memory I/O traffic, i.e., the data is already in memory. The table shows an increased runtime when comparing line-based extractions and surface-based extractions due to an increased size of the input (see Sec. 6.2.1). Differences between the two surface extractions are explained by the number of optimization iterations, the type of optimizer, and the definition of the Lagrangian. The extraction of the extremal surface requires the computation of derivatives of the Hessian matrix and derivatives of eigenvectors. This increases the computation time compared to the fluid dynamics example, where $\nabla\mathbf{w}(\bar{\mathbf{f}}(\mathbf{x}))$ was computed in a pre-processing step. Comparing the different line extractions, a similar trend can be observed. Computationally intensive feature definitions such as ridge or valley lines exhibit an increased runtime compared to less intensive definitions such as critical lines or isocontours. For example, the extraction of atmospheric jet streams took 3.76 minutes while the extraction of isocontours in the geophysics data set took only 1.42 seconds. Compared to marching squares (19 milliseconds), the variational approach is slower but customizable. Compared to the input field data, the added memory consumption of our approach is negligible, since we only need to allocate advancing fronts, which are small sets of points/lines. This is a benefit of explicit feature representations, compared to implicit ones [Weißmann et al. 2014], in which features are modeled as level set of an unknown field.

8 DISCUSSION

We proposed a number of variational feature definitions for scalar, vector, and tensor field analysis. There are many opportunities for future research, which we outline in the following.

Alternative Feature Definitions. In this paper, we proposed a first variational formulation for common line and surface features in scalar, vector, and tensor field analysis. We leave the exploration of alternative formulations to future work. For example, other smoothness norms are imaginable, ridge lines could be expressed via dot products, and in the parallel eigenvectors operator the direction could be defined in spherical coordinates. Further, an implicit formulation similar to Weißmann et al. [2014] could be interesting, when aiming for closed feature curves.

Alternative Feature Discretizations. We modeled the features exclusively as piece-wise linear curves (polylines) or piece-wise linear surfaces (triangle meshes). Likewise, it would be possible to generalize this by modeling the curves and surfaces in a linear basis, such as the Bézier Bernstein, monomial, Chebychev, or Fourier basis.

Local vs. Global Optimum. The Euler-Lagrange equation is a necessary condition for all variational minimizers. When the underlying scalar and vector fields are non-linear (which is usually the case on real-world data), then there is no guarantee that a gradient descent can find the best solution. Thus, the result will strongly depend on the initial conditions. In the future, strategies could be developed to choose better initial conditions, including data-driven approaches.

Adaptive Weighting. When attempting to extract closed feature curves, it is counterproductive to begin with a too high smoothness weight, since the feature curve will tend to become straight instead of closed. In Fig. 4 (and Fig. 1 in the additional material), we first grew the feature curve without smoothness regularization and then increased the smoothness afterwards. Interactive tools for aiding the user in adjusting weights during an optimization could be an interesting avenue in the future.

Weight Normalization. The value range of the domain coordinates $\mathbb{Y}$ can vary greatly across different data sets, which impacts the value range of the functional derivatives. A normalization procedure that makes the magnitude of regularization weights comparable across scenes would be an interesting topic for future work.

Linear Solutions. If the discretized Euler-Lagrange equation is linear in $\mathbf{f}_i$, we can solve for $\mathbf{f}_i$ directly. Then, the unknowns are placed in a vector $\mathbf{F} = (\ldots, \mathbf{f}_i, \ldots)$ and a linear system of the form $\mathbf{A} \cdot \mathbf{F} = \mathbf{B}$ is solved with an appropriate linear solver. Such a setting

Table 1. Specification of features, regularizations, and performance measurements for our test scenes. The columns list the number of domain dimensions m, the number of feature dimensions n, the number of active fronts per iteration k, the number of growing steps, the number of variational gradient descent steps (Eq. (11)), and the measured computation time. For baseline initialized optimization, k is given as *all* to indicate that the whole feature was optimized at once.

Data set	Figure	Feature definition	Regularization(s)	m	n	k	#grow	#gradient	time
Aerodynamics	Fig. 2	vortex core	smoothness	3	1	all	–	5,000	10.4 sec
Geophysics	Fig. 4	isocontour	smoothness	2	1	5	800	250	1.42 sec
Atmospheric sciences	Fig. 8	ridge line	smoothness, isocontour	3	1	3	250	100	3.76 min
Fluid Dynamics	Fig. 9	parallel vectors	smoothness	4	2	all	–	500	32.6 sec
Oceanography	Fig. 10	critical line	flow alignment	3	1	all	–	760	7.76 sec
Extremal surface	Fig. 11	ridge surface	none	3	2	all	–	1,000	3.83 min
Tensor corelines	Fig. 12	tensor coreline	none	6	1	20	40	10,000	3.74 sec
Morse theory	Fig. 1 (add. mat.)	Jacobi set	smoothness	2	1	2	80	1,000	2.98 sec
Wind engineering	Fig. 2 (add. mat.)	valley line	orientation	2	1	all	–	300	47.4 sec

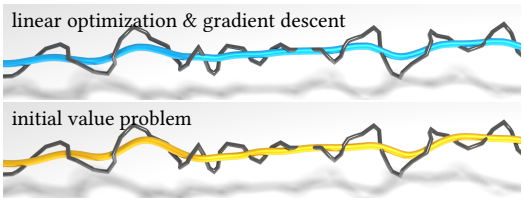


Fig. 13. With a regular curve smoothing problem that only contains a smoothness and a proximity term, the Euler-Lagrange equation turns into a linear equation. Thus, both linear optimization and gradient descent give the same result (blue curve, top). For initial value problems (orange curve, bottom), however, the result strongly depends on the initial position and velocity. The noisy input curve is shown in black.

is shown in Fig. 13 (top), where only a smoothness (Eq. (33)) and a proximity regularizer (Eq. (32)) are used to smooth a curve (gray):

$$\mathcal{L}^{\text{smooth}} = \underbrace{\frac{\lambda_s}{2}\|\nabla \mathbf{f}(x)\|^2}_{\text{smoothness}} + \underbrace{\frac{\lambda_p}{2}\|\mathbf{f}(x) - \mathbf{c}(x)\|^2}_{\text{proximity}}. \tag{48}$$

Linear approximations to non-linear problems could be used to estimate initial conditions for the non-linear solvers.

Solve as Initial Value Problem. For feature curves ($n = 1$), the Euler-Lagrange equation in Eq. (7) is a second-order ODE that can be solved as initial value problem. Thus, by choosing a start point and a start tangent, the entire feature curve can be traced out with a numerical integrator of choice. Fig. 13 (bottom) gives an example of a fourth-order Runge-Kutta integration when phrasing the second-order ODE as set of two coupled first-order ODEs after introducing an auxiliary velocity variable to render the system autonomous, cf. [Günther and Theisel 2017]. Note, however, that the result strongly depends on the chosen initial position and tangent. The initial value problems could be considered further for deriving more accurate growth steps.

Preconditioning. The Euler-Lagrange equations in Eq. (7) phrase a root-finding problem. Thus, it would be natural to explore Newton and quasi-Newton methods, such as L-BFGS, in order to accelerate convergence and to pre-condition the descent in order to avoid oscillations. It should be noted, however, that Newton methods require a sufficient condition, since unlike the gradient descent, which will always walk 'down' towards a variational minimum, the Newton approaches can converge to roots of any type.

Sufficient Condition. The Euler-Lagrange equation is a necessary condition and holds for both functional minima and maxima. The second variation is needed to distinguish the two. Since we used a gradient descent, we converge to a minimal solution. If a Newton method was used instead as mentioned above, the second variation should be observed to exclude false positives.

GPU Acceleration. A GPU implementation of the gradient-based optimization would be a natural next step to accelerate the computation in order to enable an interactive exploration of weights.

9 CONCLUSIONS

We established a common mathematical language to approach the definition and extraction of feature curves and surfaces from scalar, vector, and tensor fields. Using variational calculus, features can be treated as unknown functions that minimize customizable energy terms for which necessary conditions can be derived. The variational formulations allows combining features not only with each other but also with regularizers that enable the incorporation of valuable domain knowledge. This was previously not possible, due to incompatible modelling. In the future, we expect more feature definitions to be introduced in the variational formulation, further numerical extraction algorithms will be developed, and other feature discretizations are imaginable. Our approach requires suitable seed points, and the setting of energy weights is in the hands of the user. Developing an interactive system that guides the user through the feature extraction process is another direction for future work.

ACKNOWLEDGMENTS

The authors would like to thank Markus Rütten from DLR for the Delta wing flow, Anna Gülcher of NASA JPL for the Earth mantle convection, Michael Sprenger of ETH Zürich for the atmospheric air flow, Tino Weinkauf of KTH Stockholm for the square cylinder flow, Helen Schottenhamml of FAU for the wind turbine simulation, and Nathan Lerzer of FAU for early experiments on the jet streams.

REFERENCES

Ryan Abernathey and George Haller. 2018. Transport by Lagrangian Vortices in the Eastern Pacific. *Journal of Physical Oceanography* 48, 3 (2018), 667 – 685. https://doi.org/10.1175/JPO-D-17-0102.1

Robin Bader, Michael Sprenger, Nikolina Ban, Stefan Rüdisühli, Christoph Schär, and Tobias Günther. 2020. Extraction and Visual Analysis of Potential Vorticity Banners around the Alps. *IEEE Transactions on Visualization and Computer Graphics (Proc. IEEE Scientific Visualization)* 26, 1 (2020), 259–269. https://doi.org/10.1109/TVCG.2019.2934310

Irene Baeza Rojo and Tobias Günther. 2019. Vector field topology of time-dependent flows in a steady reference frame. *IEEE Transactions on Visualization and Computer Graphics* 26, 1 (2019), 280–290. https://doi.org/10.1109/TVCG.2019.2934375

David C. Banks and Bart A. Singer. 1995. A Predictor-Corrector Technique for Visualizing Unsteady Flow. *IEEE Transactions on Visualization and Computer Graphics* 1 (1995), 151–163. https://doi.org/10.1109/2945.468404

Lukas Bösiger, Michael Sprenger, Maxi Boettcher, Hanna Joos, and Tobias Günther. 2022. Integration-based extraction and visualization of jet stream cores. *Geoscientific Model Development* 15, 3 (2022), 1079–1096. https://doi.org/10.5194/gmd-15-1079-2022

Roxana Bujack. 2022. Discussion and Visualization of Distinguished Hyperbolic Trajectories as a Generalization of Critical Points to 2D Time-dependent Flow. In *2022 Topological Data Analysis and Visualization (TopoInVis)*. IEEE, IEEE, Oklahoma City, OK, USA, 59–69. https://doi.org/10.1109/TopoInVis57755.2022.00013

Roxana Bujack, Lin Yan, Ingrid Hotz, Christoph Garth, and Bei Wang. 2020. State of the Art in Time-Dependent Flow Topology: Interpreting Physical Meaningfulness Through Mathematical Properties. *Computer Graphics Forum* 39, 3 (2020), 811–835. https://doi.org/10.1111/cgf.14037

Brian Cabral and Leith Leedom. 1993. Imaging Vector Fields Using Line Integral Convolution. *Computer Graphics (Proc. SIGGRAPH)* 27 (1993), 263–272. https://doi.org/10.1145/166117.166151

Simone Camarri, Angelo Iollo, Marcelo Buffoni, and Maria Vittoria Salvetti. 2005. Simulation of the three-dimensional flow around a square cylinder between parallel walls at moderate Reynolds numbers. In *XVII Congresso di Meccanica Teorica ed Applicata*. Firenze University Press, Florence, Italy, 1–12. https://doi.org/10.1400/56904

Hamish Carr, Zhao Geng, Julien Tierny, Amit Chattopadhyay, and Aaron Knoll. 2015. Fiber surfaces: Generalizing isosurfaces to bivariate data. *Computer Graphics Forum* 34, 3 (2015), 241–250. https://doi.org/10.1111/cgf.12636

Zhiqin Chen and Hao Zhang. 2021. Neural marching cubes. *ACM Trans. Graph.* 40, 6, Article 251 (2021), 15 pages. https://doi.org/10.1145/3478513.3480518

Albert Chern, Felix Knöppel, Ulrich Pinkall, and Peter Schröder. 2017. Inside fluids: clebsch maps for visualization and processing. *ACM Trans. Graph.* 36, 4, Article 142 (jul 2017), 11 pages. https://doi.org/10.1145/3072959.3073591

Albert Chern, Felix Knöppel, Ulrich Pinkall, Peter Schröder, and Steffen Weißmann. 2016. Schrödinger's smoke. *ACM Trans. Graph.* 35, 4, Article 77 (2016), 13 pages. https://doi.org/10.1145/2897824.2925868

Martin Dameris. 2015. Stratosphere/Troposphere Exchange and Structure | Tropopause. In *Encyclopedia of Atmospheric Sciences (Second Edition)* (second edition ed.), Gerald R. North, John Pyle, and Fuqing Zhang (Eds.). Academic Press, Oxford, 269–272. https://doi.org/10.1016/B978-0-12-382225-3.00418-7

David Eberly. 1996. *Ridges in image and data analysis*. Kluwer Academic Publishers, Dordrecht. https://doi.org/10.1007/978-94-015-8765-5

Mohammad Farazmand and George Haller. 2012. Computing Lagrangian coherent structures from their variational theory. *Chaos: An Interdisciplinary Journal of Nonlinear Science* 22, 1 (2012), 013128. https://doi.org/10.1063/1.3690153

Lucas I. Finn and Bruce M. Boghosian. 2006. A global variational approach to vortex core identification. *Physica A: Statistical Mechanics and its Applications* 362, 1 (2006), 11–16. https://doi.org/10.1016/j.physa.2005.09.013 Discrete Simulation of Fluid Dynamics.

Bengt Fornberg. 1988. Generation of finite difference formulas on arbitrarily spaced grids. *Mathematics of computation* 51, 184 (1988), 699–706. https://doi.org/10.1090/S0025-5718-1988-0935077-0

Raphael Fuchs, Ronald Peikert, Helwig Hauser, Filip Sadlo, and Philipp Muigg. 2008. Parallel vectors criteria for unsteady flow vortices. *IEEE Transactions on Visualization and Computer Graphics* 14, 3 (2008), 615–626. https://doi.org/10.1109/TVCG.2007.70633

Izrail Moiseevitch Gelfand and S. V. Fomin. 1963. *Calculus of variations*. Prentice-Hall Inc., Englewood Cliffs, N. J. https://doi.org/10.1017/S0008439500032045

Anna Johanna Pia Gülcher, Maxim Dionys Ballmer, and Paul James Tackley. 2021. Coupled dynamics and evolution of primordial and recycled heterogeneity in Earth's lower mantle. *Solid Earth* 12, 9 (2021), 2087–2107. https://doi.org/10.5194/se-12-2087-2021

Tobias Günther, Markus Gross, and Holger Theisel. 2017. Generic Objective Vortices for Flow Visualization. *ACM Transactions on Graphics (Proc. SIGGRAPH)* 36, 4 (2017), 1–11. https://doi.org/10.1145/3072959.3073684

Tobias Günther and Holger Theisel. 2017. Backward Finite-Time Lyapunov Exponents in Inertial Flows. *IEEE Transactions on Visualization and Computer Graphics (Proc. IEEE SciVis 2016)* 23, 1 (2017), 970–979. https://doi.org/10.1109/TVCG.2016.2599016

Tobias Günther and Holger Theisel. 2018. The State of the Art in Vortex Extraction. *Computer Graphics Forum* 37, 6 (2018), 149–173. https://doi.org/10.1111/cgf.13319

Hanqi Guo and Tom Peterka. 2021. Exact Analytical Parallel Vectors. In *2021 IEEE Visualization Conference (VIS)*. IEEE, New Orleans, LA, USA, 101–105. https://doi.org/10.1109/VIS49827.2021.9623310

Markus Hadwiger, Matej Mlejnek, Thomas Theußl, and Peter Rautek. 2019. Time-Dependent Flow seen through Approximate Observer Killing Fields. *IEEE Transactions on Visualization and Computer Graphics* 25, 1 (Jan 2019), 1257–1266. https://doi.org/10.1109/TVCG.2018.2864839

George Haller. 2011. A variational theory of hyperbolic Lagrangian Coherent Structures. *Physica D: Nonlinear Phenomena* 240, 7 (2011), 574–598. https://doi.org/10.1016/j.physd.2010.11.010

Nili Harnik, Chaim I Garfinkel, and Orli Lachmy. 2016. The influence of jet stream regime on extreme weather events. *Dynamics and Predictability of Large-Scale, High-Impact Weather and Climate Events* 2 (2016), 79–94. https://doi.org/10.1017/CBO9781107775541.007

James L. Helman and Lambertus Hesselink. 1989. Representation and Display of Vector Field Topology in Fluid Flow Data Sets. *Computer* 22, 8 (1989), 27–36. https://doi.org/10.1109/2.35197

James L. Helman and Lambertus Hesselink. 1991. Visualizing Vector Field Topology in Fluid Flows. *IEEE Computer Graphics and Applications* 11 (May 1991), 36–46. https://doi.org/10.1109/38.79452

Hans Hersbach, Bill Bell, Paul Berrisford, Shoji Hirahara, András Horányi, Joaquín Muñoz-Sabater, Julien Nicolas, Carole Peubey, Raluca Radu, Dinand Schepers, Adrian Simmons, Cornel Soci, Saleh Abdalla, Xavier Abellan, Gianpaolo Balsamo, Peter Bechtold, Gionata Biavati, Jean Bidlot, Massimo Bonavita, Giovanna De Chiara, Per Dahlgren, Dick Dee, Michail Diamantakis, Rossana Dragani, Johannes Flemming, Richard Forbes, Manuel Fuentes, Alan Geer, Leo Haimberger, Sean Healy, Robin J. Hogan, Elías Hólm, Marta Janisková, Sarah Keeley, Patrick Laloyaux, Philippe Lopez, Cristina Lupu, Gabor Radnoti, Patricia de Rosnay, Iryna Rozum, Freja Vamborg, Sebastien Villaume, and Jean-Noël Thépaut. 2020. The ERA5 global reanalysis. *Quarterly Journal of the Royal Meteorological Society* 146, 730 (2020), 1999–2049. https://doi.org/10.1002/qj.3803

Lutz Hofmann and Filip Sadlo. 2019. The Dependent Vectors Operator. *Computer Graphics Forum* 38, 3 (2019), 261–272. https://doi.org/10.1111/cgf.13687

Lutz Hofmann and Filip Sadlo. 2020. Extraction of Distinguished Hyperbolic Trajectories for 2D Time-Dependent Vector Field Topology. *Computer Graphics Forum* 39, 3 (2020), 303–316. https://doi.org/10.1111/cgf.13982

Jeffrey P. M. Hultquist. 1992. Constructing Stream Surfaces in Steady 3D Vector Fields. In *Proc. Visualization* (Boston, Massachusetts). IEEE, Boston, MA, USA, 171–178. https://doi.org/10.1109/VISUAL.1992.235211

Tao Ju, Minxin Cheng, Xu Wang, and Ye Duan. 2014. A robust parity test for extracting parallel vectors in 3D. *IEEE Transactions on Visualization and Computer Graphics* 20, 12 (2014), 2526–2534. https://doi.org/10.1109/TVCG.2014.2346412

Michael Kern, Tim Hewson, Filip Sadlo, Rüdiger Westermann, and Marc Rautenhaus. 2018. Robust Detection and Visualization of Jet-Stream Core Lines in Atmospheric Flow. *IEEE Transactions on Visualization and Computer Graphics* 24, 1 (2018), 893–902. https://doi.org/10.1109/TVCG.2017.2743989

Gordon L. Kindlmann, Charisee Chiw, Tri Huynh, Attila Gyulassy, John Reppy, and Peer-Timo Bremer. 2018. Rendering and Extracting Extremal Features in 3D Fields. *Computer Graphics Forum* 37, 3 (2018), 525–536. https://doi.org/10.1111/cgf.13439

Diederik P. Kingma and Jimmy Ba. 2017. Adam: A Method for Stochastic Optimization. arXiv:1412.6980 [cs.LG]

Patrick Koch, Heini Wernli, and Huw C. Davies. 2006. An event-based jet-stream climatology and typology. *International Journal of Climatology* 26, 3 (2006), 283–301. https://doi.org/10.1002/joc.1255

Tony Lindeberg. 1998. Edge Detection and Ridge Detection With Automatic Scale Selection. *International Journal of Computer Vision* 30 (09 1998), 117–154. https://doi.org/10.1023/A:1008097225773

William E. Lorensen and Harvey E. Cline. 1987. Marching Cubes: A High Resolution 3D Surface Construction Algorithm. *SIGGRAPH Comput. Graph.* 21, 4 (Aug 1987), 163–169. https://doi.org/10.1145/37402.37422

Gustavo M. Machado, Sebastian Boblest, Thomas Ertl, and Filip Sadlo. 2016. Space-Time Bifurcation Lines for Extraction of 2D Lagrangian Coherent Structures. *Computer Graphics Forum (Proc. EuroVis)* 35, 3 (2016), 91–100. https://doi.org/10.1111/cgf.12885

Gustavo Mello Machado, Filip Sadlo, and Thomas Ertl. 2013. Local Extraction of Bifurcation Lines. In *Vision, Modeling and Visualization*. The Eurographics Association, Lugano, Switzerland, 17–24. https://doi.org/10.2312/PE.VMV.VMV13.017-024

Gloria L. Manney and Michaela I. Hegglin. 2018. Seasonal and Regional Variations of Long-Term Changes in Upper-Tropospheric Jets from Reanalyses. *Journal of Climate* 31, 1 (2018), 423 – 448. https://doi.org/10.1175/JCLI-D-17-0303.1

Tony McLoughlin, Robert S Laramee, Ronald Peikert, Frits H Post, and Min Chen. 2010. Over Two Decades of Integration-Based, Geometric Flow Visualization. *Computer Graphics Forum* 29, 6 (2010), 1807–1829. https://doi.org/10.1111/j.1467-8659.2010.01650.x

John W. Nielsen-Gammon. 2001. A Visualization of the Global Dynamic Tropopause. *Bulletin of the American Meteorological Society* 82, 6 (2001), 1151 – 1168. https://doi.org/10.1175/1520-0477(2001)082<1151:AVOTGD>2.3.CO;2

David Oettinger, Daniel Blazevski, and George Haller. 2016. Global variational approach to elliptic transport barriers in three dimensions. *Chaos: An Interdisciplinary Journal of Nonlinear Science* 26, 3 (2016), 033114. https://doi.org/10.1063/1.4944732

David Oettinger and George Haller. 2016. An autonomous dynamical system captures all LCSs in three-dimensional unsteady flows. *Chaos: An Interdisciplinary Journal of Nonlinear Science* 26, 10 (2016), 103111. https://doi.org/10.1063/1.4965026

Timo Oster, Christian Rössl, and Holger Theisel. 2018a. Core Lines in 3D Second-Order Tensor Fields. *Computer Graphics Forum* 37, 3 (2018), 327–337. https://doi.org/10.1111/cgf.13423

Timo Oster, Christian Rössl, and Holger Theisel. 2018b. The Parallel Eigenvectors Operator. In *Vision, Modeling and Visualization*, Fabian Beck, Carsten Dachsbacher, and Filip Sadlo (Eds.). The Eurographics Association, Eindhoven, The Netherlands, 39–46. https://doi.org/10.2312/vmv.20181251

Christian Pagot, D Osmari, Filip Sadlo, Daniel Weiskopf, Thomas Ertl, and João Comba. 2011. Efficient parallel vectors feature extraction from higher-order data. *Computer Graphics Forum* 30, 3 (2011), 751–760. https://doi.org/10.1111/j.1467-8659.2011.01924.x

Ronald Peikert and Martin Roth. 1999. The "Parallel Vectors" Operator – A Vector Field Visualization Primitive. In *Proc. IEEE Visualization*. IEEE, San Francisco, CA, USA, 263–270. https://doi.org/10.1109/VISUAL.1999.809896

Ronald Peikert and Filip Sadlo. 2008. Height ridge computation and filtering for visualization. In *2008 IEEE Pacific Visualization Symposium*. IEEE, IEEE, Kyoto, japan, 119–126. https://doi.org/10.1109/PACIFICVIS.2008.4475467

Anthony E Perry and Min S Chong. 1987. A description of eddying motions and flow patterns using critical-point concepts. *Annual Review of Fluid Mechanics* 19, 1 (1987), 125–155. https://doi.org/10.1146/annurev.fl.19.010187.001013

Ulrich Pinkall and Konrad Polthier. 1993. Computing discrete minimal surfaces and their conjugates. *Experimental mathematics* 2, 1 (1993), 15–36. https://doi.org/10.1080/10586458.1993.10504266

Frits H. Post, Benjamin Vrolijk, Helwig Hauser, Robert S. Laramee, and Helmut Doleisch. 2003. The State of the Art in Flow Visualisation: Feature Extraction and Tracking. *Computer Graphics Forum* 22, 4 (2003), 775–792. https://doi.org/10.1111/j.1467-8659.2003.00723.x

Wolfgang Quapp and Benjamin Schmidt. 2011. An empirical, variational method of approach to unsymmetric valley-ridge inflection points. *Theoretical Chemistry Accounts* 128 (2011), 47–61. https://doi.org/10.1007/s00214-010-0749-z

Peter Rautek, Matej Mlejnek, Johanna Beyer, Jakob Troidl, Hanspeter Pfister, Thomas Theußl, and Markus Hadwiger. 2020. Objective observer-relative flow visualization in curved spaces for unsteady 2D geophysical flows. *IEEE Transactions on Visualization and Computer Graphics* 27, 2 (2020), 283–293. https://doi.org/10.1109/TVCG.2020.3030454

Peter Rautek, Xingdi Zhang, Bernhard Woschizka, Thomas Theußl, and Markus Hadwiger. 2024. Vortex Lens: Interactive Vortex Core Line Extraction using Observed Line Integral Convolution. *IEEE Transactions on Visualization and Computer Graphics (Proceedings IEEE VIS 2023)* 30, 1 (2024), 55–65. https://doi.org/10.1109/TVCG.2023.3326915

Martin Reuter, Silvia Biasotti, Daniela Giorgi, Giuseppe Patanè, and Michela Spagnuolo. 2009. Discrete Laplace–Beltrami operators for shape analysis and segmentation. *Computers & Graphics* 33, 3 (2009), 381–390. https://doi.org/10.1016/j.cag.2009.03.005

Martin Roth. 2000. *Automatic extraction of vortex core lines and other line-type features for scientific visualization*. Vol. 9. ETH Zurich, Zurich, Switzerland. https://doi.org/10.3929/ethz-a-004016407

Martin Roth and Ronald Peikert. 1998. A higher-order method for finding vortex core lines. In *Proc. IEEE Visualization*. IEEE, Research Triangle Park, NC, USA, 143–150. https://doi.org/10.1109/VISUAL.1998.745296

Jan Sahner, Tino Weinkauf, Nathalie Teuber, and Hans-Christian Hege. 2007. Vortex and Strain Skeletons in Eulerian and Lagrangian Frames. *IEEE Transactions on Visualization and Computer Graphics* 13, 5 (2007), 980–990. https://doi.org/10.1109/TVCG.2007.1053

William Schroeder, Rob Maynard, and Berk Geveci. 2015. Flying edges: A high-performance scalable isocontouring algorithm. In *IEEE Symposium on Large Data Analysis and Visualization (LDAV)*. IEEE, 33–40. https://doi.org/10.1109/LDAV.2015.7348069

David Sujudi and Robert Haimes. 1995. *Identification of Swirling Flow in 3D Vector Fields*. Technical Report. Departement of Aeronautics and Astronautics, MIT. https://doi.org/10.2514/6.1995-1715 AIAA Paper 95-1715.

Holger Theisel, Markus Hadwiger, Peter Rautek, Thomas Theußl, and Tobias Günther. 2021. Vortex criteria can be objectivized by unsteadiness minimization. *Physics of Fluids* 33, 10 (2021), 107115. https://doi.org/10.1063/5.0063817

Holger Theisel, Jan Sahner, Tino Weinkauf, Hans-Christian Hege, and Hans-Peter Seidel. 2005. Extraction of parallel vector surfaces in 3D time-dependent fields and application to vortex core line tracking. In *Proc. IEEE Visualization*. IEEE, Minneapolis, MN, USA, 631–638. https://doi.org/10.1109/VISUAL.2005.1532851

Holger Theisel and Hans-Peter Seidel. 2003. Feature Flow Fields. In *Proc. Symposium on Data Visualisation*. The Eurographics Association, Grenoble, France, 141–148. https://doi.org/10.2312/VisSym/VisSym03/141-148

Allen Van Gelder and Alex Pang. 2009. Using PVsolve to Analyze and Locate Positions of Parallel Vectors. *IEEE Transactions on Visualization and Computer Graphics* 15, 4 (2009), 682–695. https://doi.org/10.1109/TVCG.2009.11

Tino Weinkauf, Jan Sahner, Holger Theisel, and Hans-Christian Hege. 2007. Cores of Swirling Particle Motion in Unsteady Flows. *IEEE Transactions on Visualization and Computer Graphics (Proc. Visualization)* 13, 6 (2007), 1759–1766. https://doi.org/10.1109/TVCG.2007.70545

Tino Weinkauf, Holger Theisel, Allen Van Gelder, and Alex Pang. 2010. Stable feature flow fields. *IEEE Transactions on Visualization and Computer Graphics* 17, 6 (2010), 770–780. https://doi.org/10.1109/TVCG.2010.93

Steffen Weißmann, Ulrich Pinkall, and Peter Schröder. 2014. Smoke rings from smoke. *ACM Trans. Graph.* 33, 4, Article 140 (2014), 8 pages. https://doi.org/10.1145/2601097.2601171

Xingdi Zhang, Markus Hadwiger, Thomas Theußl, and Peter Rautek. 2021. Interactive exploration of physically-observable objective vortices in unsteady 2D flow. *IEEE Transactions on Visualization and Computer Graphics* 28, 1 (2021), 281–290. https://doi.org/10.1109/TVCG.2021.3115565