Topological Methods

Topological Methods

In visualization, topological methods started with vector field visualization. So we will start with that and we will have to introduce the notion of streamlines which trace through the vector field.

Vector Fields

Divergence

Given a vector field $v:R^3 \rightarrow R^3$, the **divergence** of $v=(v_x,v_y,v_z)$ is the scalar quantity

$$div\ v = \frac{\partial v_x}{\partial x} + \frac{\partial v_y}{\partial y} + \frac{\partial v_z}{\partial z}$$

Intuitively, if v is a flow field that transports mass, $div\ v$ characterizes the increase or loss of mass at a given point p in the vector field in unit time.

- positive divergence at p: mass spreads from p outward.
- negative divergence at p: mass gets sucked into p.
- zero divergence at p: mass is transported without getting spread or sucked, i.e. without compression or expansion.

Vector Fields (continued)

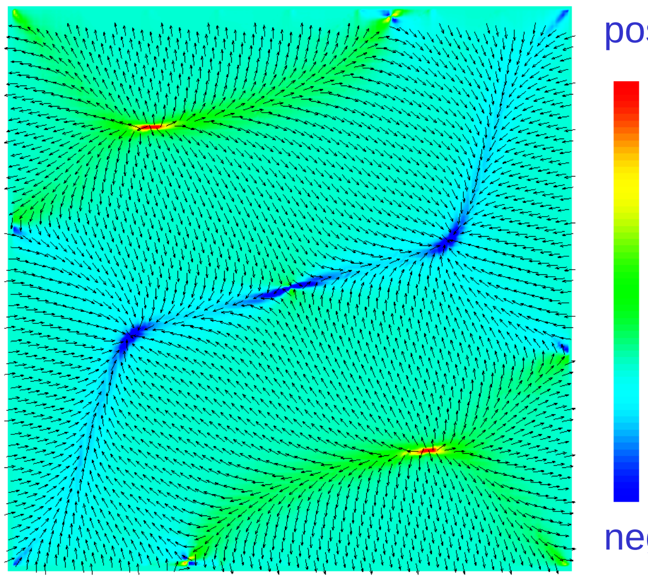
Vorticity

Given a vector field $v: R^3 \to R^3$, the **vorticity** of $v=(v_x,v_y,v_z)$, also called **curl** or **rotor**, is the vector quantity

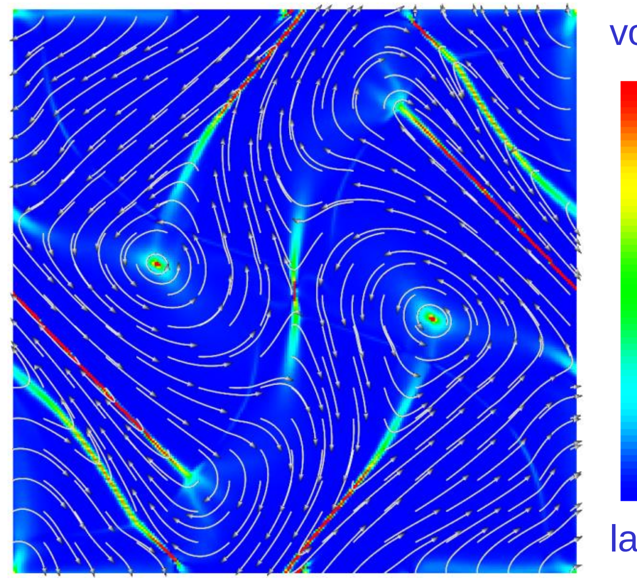
$$rot\ v = \left(\frac{\partial v_z}{\partial y} - \frac{\partial v_y}{\partial z}, \frac{\partial v_x}{\partial z} - \frac{\partial v_z}{\partial x}, \frac{\partial v_y}{\partial x} - \frac{\partial v_x}{\partial y} \right)$$

The vorticity $rot\ v$ of v is a vector field that is locally perpendicular to the plane or rotation of v and whose magnitude expresses the speed of angular rotation of v around $rot\ v$. Hence, the vorticity vector characterizes the speed and direction of rotation of a given vector field at every point. Sometimes $rot\ v$ is also denoted as $curl\ v$.

Vector Algorithms



Divergence of a 2D vector field

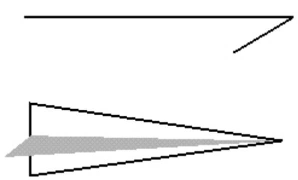


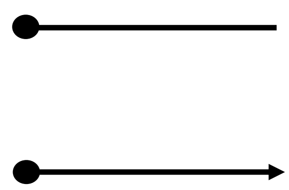
Vorticity of a 2D vector field

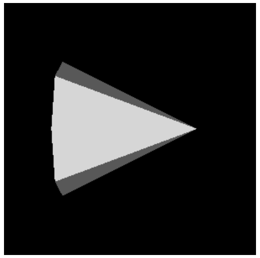
Images courtesy of Alexandru Telea

Vector Algorithms (continued)

A simple vector visualization technique is to draw an oriented, scaled line for each vector. The line begins at the point with which the vector is associated and is oriented in the direction of the vector components. Typically, the resulting line must be scaled up or down to control the size of its visual representation. This technique is often referred to as *hedgehog* because of the bristly result.



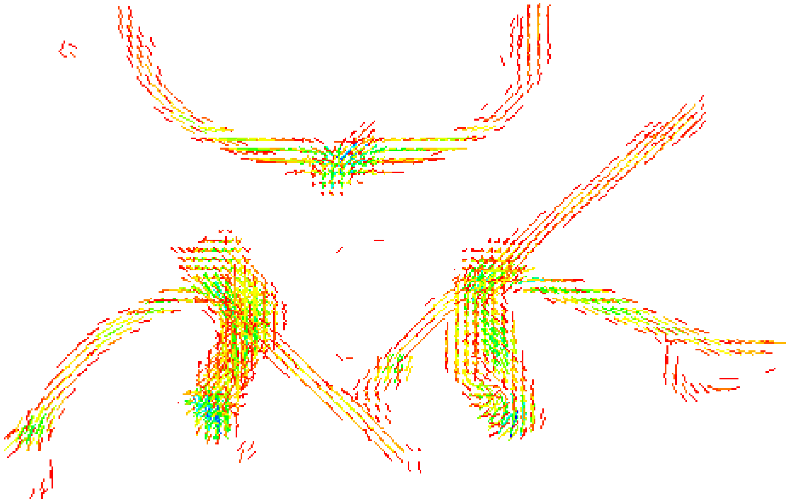


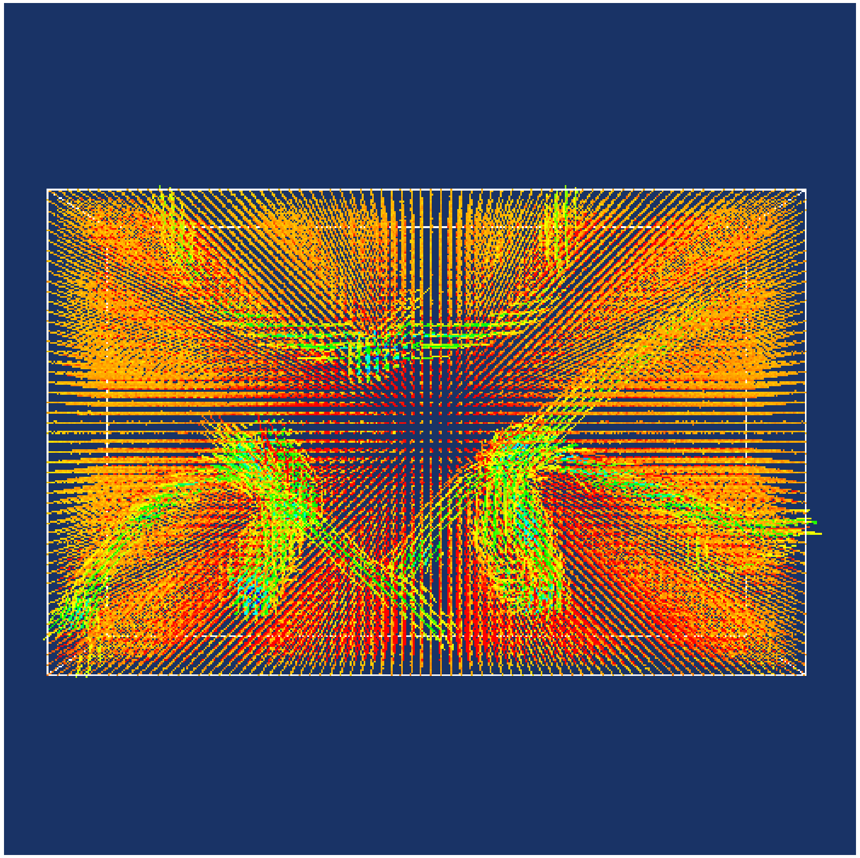


Direction can also be visualized using color coding by using different colors at each ends of the glyph

Vector Algorithms (continued)

The problem with glyphs is that it easily results in clutter.





Vector Algorithms (continued)

Time animation

The idea is to move points (mass-less particles) along the vector field. Basically, the particle is *advected* at every point in direction of the vector at that location (if necessary interpolation needs to be used), i.e. $v = dx/dt$.

Beginning with a sphere S centered about some point, we move S repeatedly to generate the bubbles below:

Vector Algorithms (continued)

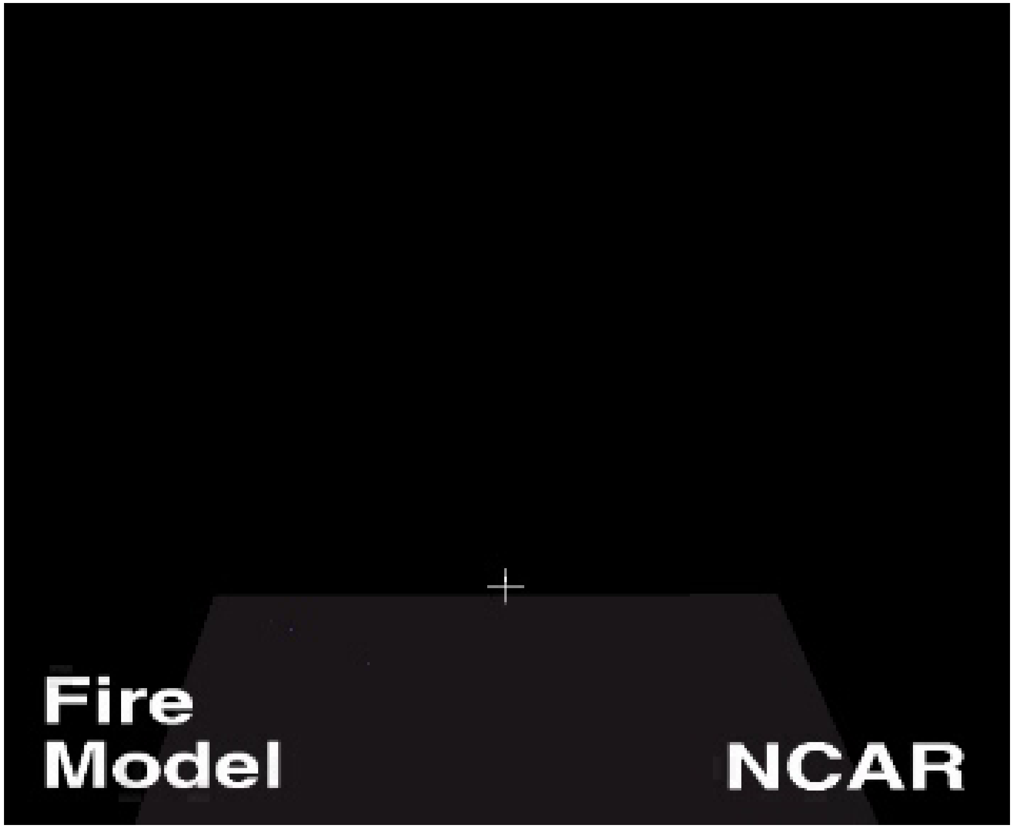
The eye tends to trace out a path by connecting the bubbles, giving the observer a qualitative understanding of the fluid flow in that area. The bubbles may be displayed as an animation over time (giving the illusion of motion) or as a multiple exposure sequence (giving the appearance of a path).

The choice of step size is a critical parameter in constructing accurate visualization of particle paths in a vector field. By taking large steps we are likely to jump over changes in the velocity. Using smaller steps we will end in a different position.

Vector Algorithms (continued)

Example

Particle advection for fire simulation



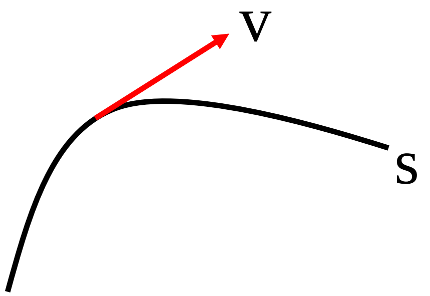
Vector Algorithms (continued)

Tracing particles

In order to determine the locations of a particle previously represented as a bubble, the particle needs to be traced throughout the vector field.

Since we are considering a mass-less particle, the particle basically follows the integral curve, i.e.

$$s'(x,t) = \vec{v}(s(x,t))$$



The initial position is user-defined.

Vector Algorithms (continued)

Although this form cannot be solved analytically for most real world data, its solution can be approximated using numerical integration techniques. Accurate numerical integration is a topic beyond the scope of this class, but it is known that the accuracy of the integration is a function of the step size. Since the path is an integration throughout the data set, the accuracy of the cell interpolation functions, as well as the accuracy of the original vector data, plays an important role in realizing accurate solutions.

Vector Algorithms (continued)

Euler's method

The simples form of numerical integration is Euler's method

$$\vec{x}_{i+1} = \vec{x}_i + \vec{v}(\vec{x}_i) \cdot \Delta t$$

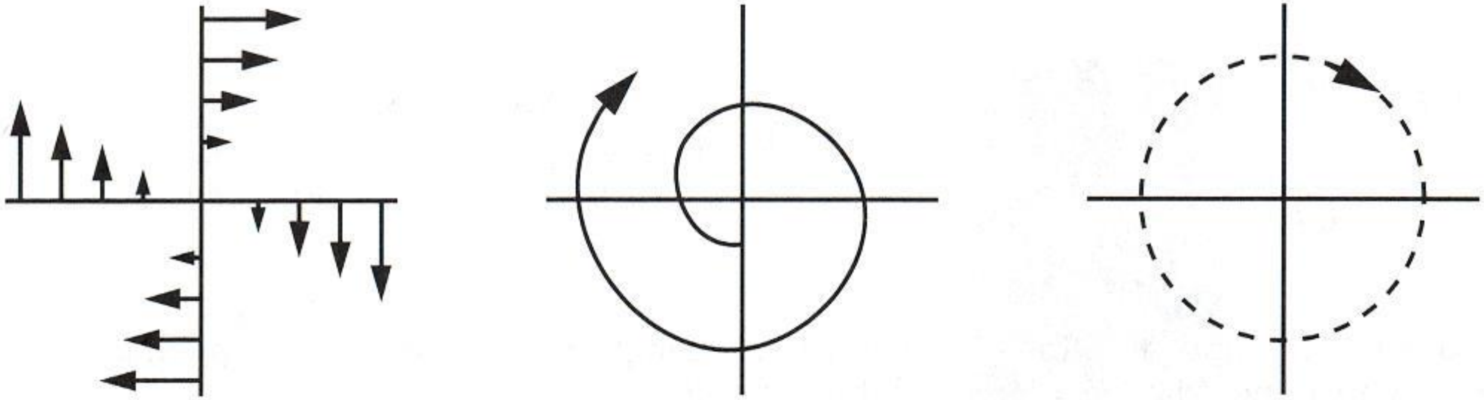
where x_i is the position and Δt the step size.

Euler's method has an error on the order of $O(\Delta t^2)$, which is not accurate enough for some applications.

Vector Algorithms (continued)

Example

Integral curves computed using two different techniques for a rotational vector field



(a) Rotational vector field (b) Euler's method (c) Runge-Kutta

Vector Algorithms (continued)

Runge-Kutta method

The family of explicit Runge-Kutta methods is given by

$$y_{n+1} = y_n + h \sum_{i=1}^{s} b_i k_i,$$

Where

$$\begin{aligned}
k_1 &= f(t_n, y_n), \\
k_2 &= f(t_n + c_2 h, y_n + a_{21} h k_1), \\
k_3 &= f(t_n + c_3 h, y_n + a_{31} h k_1 + a_{32} h k_2), \\
&\;\;\vdots \\
k_s &= f(t_n + c_s h, y_n + a_{s1} h k_1 + a_{s2} h k_2 + \cdots + a_{s,s-1} h k_{s-1}).
\end{aligned}$$

(Note: the above equations have different but equivalent definitions in different texts).

Vector Algorithms (continued)

To specify a particular method, one needs to provide the integer s (the number of stages), and the coefficients a_{ij} (for $1 \le j < i \le s$), b_i (for $i = 1, 2, ..., s$) and c_i (for $i = 2, 3, ..., s$). These data are usually arranged in a mnemonic device, known as a *Runge-Kutta tableau*:

$$
\begin{array}{c|ccccc}
0 & & & & & \\
c_2 & a_{21} & & & & \\
c_3 & a_{31} & a_{32} & & & \\
: & : & : & & & \\
c_s & a_{s1} & a_{s2} & \cdots & a_{s,s-1} & \\
\hline
 & b_1 & b_2 & \cdots & b_{s-1} & b_s
\end{array}
$$

The Runge-Kutta method is consistent if $\sum_{j=1}^{i-1} a_{ij} = c_i \text{ for } i = 2, \ldots, s.$

There are also accompanying requirements if we require the method to have a certain order p, meaning that the truncation error is O(hp+1). These can be derived from the definition of the truncation error itself. For example, a 2-stage method has order 2 if $b_1 + b_2 = 1$, $b_2c_2 = 1/2$, and $b_2a_{21} = 1/2$.

Vector Algorithms (continued)

Runge-Kutta technique of order 2

Hence, we get the following formula for the Runge-Kutta technique of order 2:

$$\vec{x}_{i+1} = \vec{x}_i + \frac{\Delta t}{2}(\vec{v}(\vec{x}_i) + \vec{v}(\vec{x}_{i+1}))$$

Vector Algorithms (continued)

Streamlines

We have seen that the step size is a design parameter. Hence, we can choose the step size in such a way that a line is formed. For a static vector field, i.e. a vector field that does not change over time, the integral curve results in a streamline.

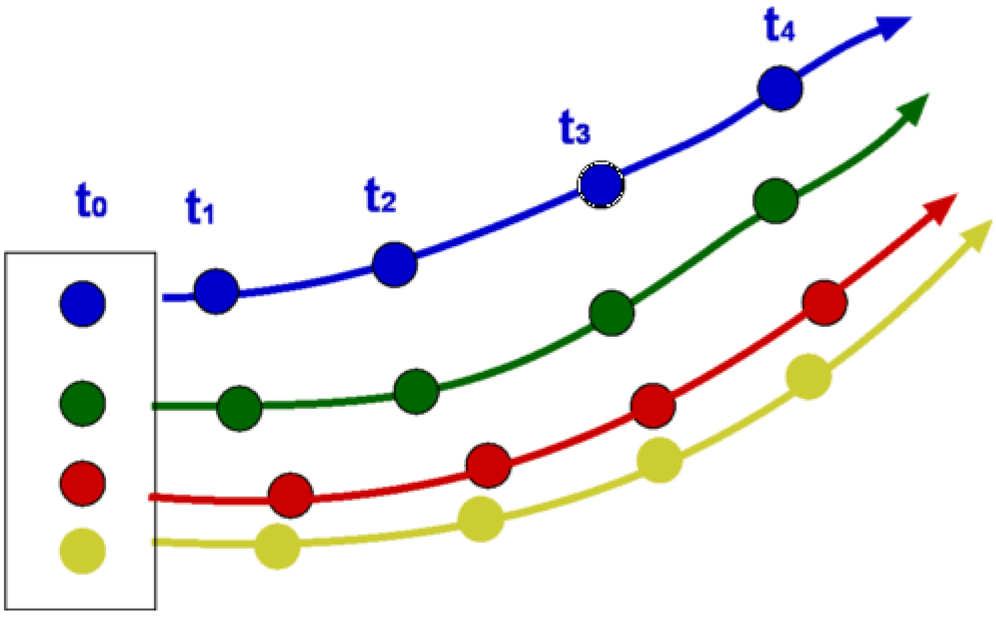
Different type types of integral curves exist:

- Pathlines
- Streaklines
- Streamlines

Vector Algorithms (continued)

Pathline

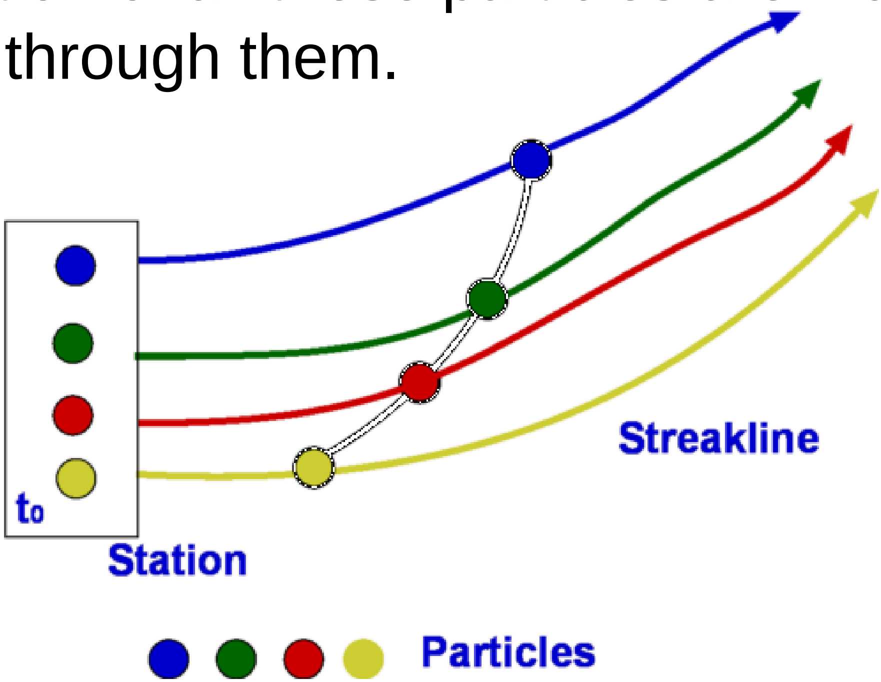
A pathline is the line traced by a given particle. This is generated by injecting a dye into the fluid and following its path by photography or other means



Vector Algorithms (continued)

Streakline

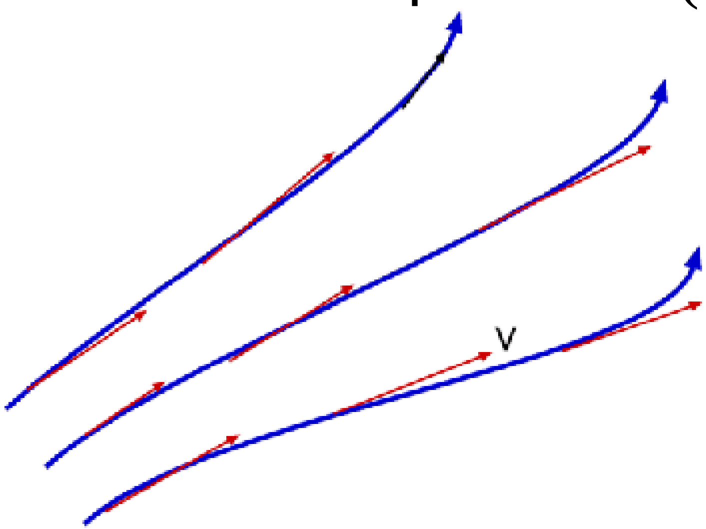
A streakline concentrates on fluid particles that have gone through a fixed station or point. At some instant of time the position of all these particles are marked and a line is drawn through them.

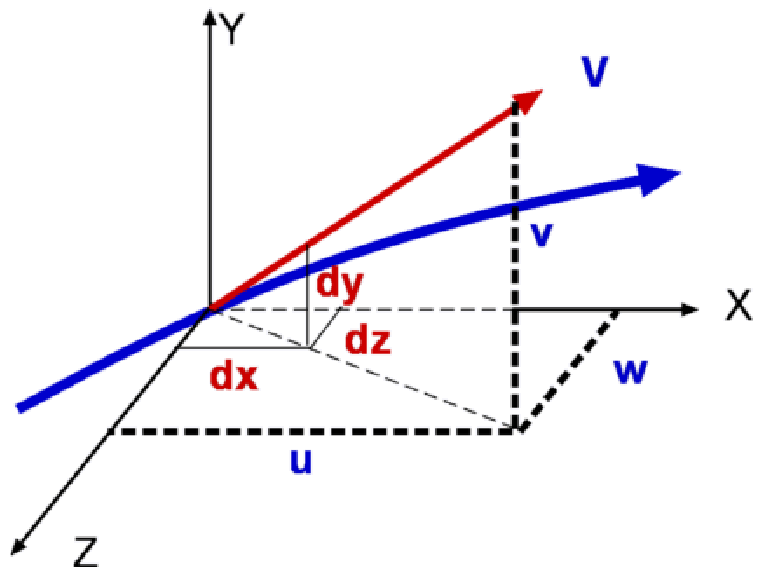


Vector Algorithms (continued)

Streamline

A streamline is one that is drawn tangential to the velocity vector at every point in the flow at a given instant and forms a powerful tool in understanding flows. Thus, it satisfies the equation $s'(x,t) = \vec{v}(s(x,t))$

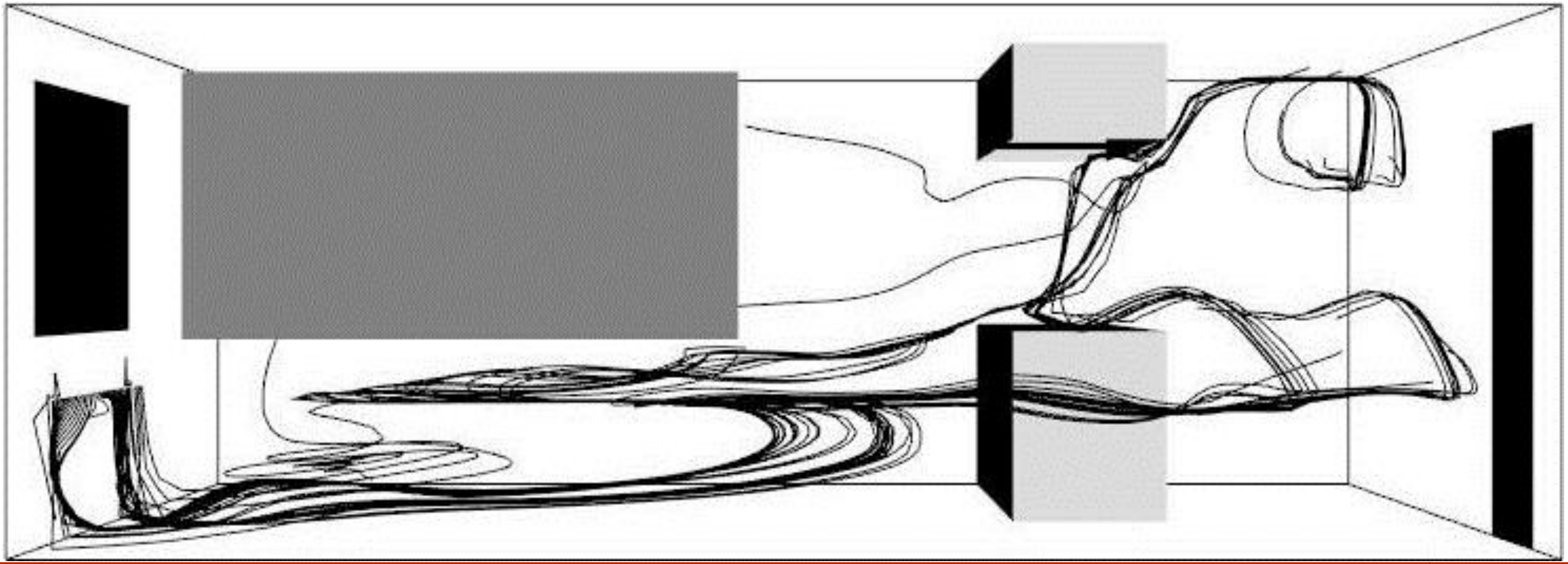




Vector Algorithms (continued)

Example

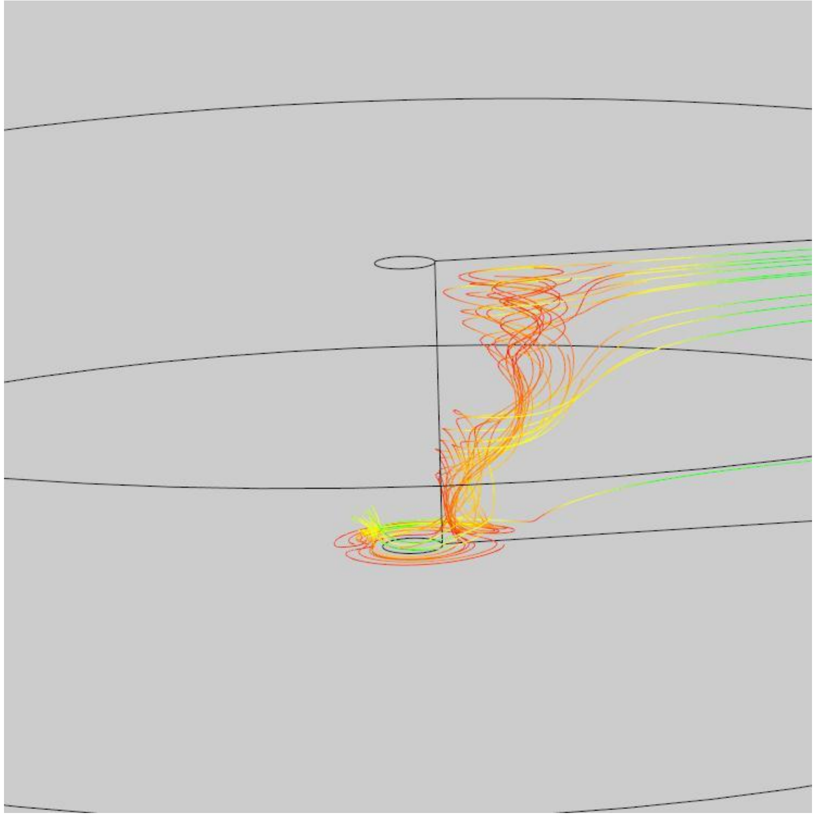
Flow velocity computed for a small kitchen (side view). Forty streamlines start along the rake positioned under the window. Some eventually travel over the hot stove and are convected upwards.



Vector Algorithms (continued)

Example

Flow around NASA's tapered cylinder



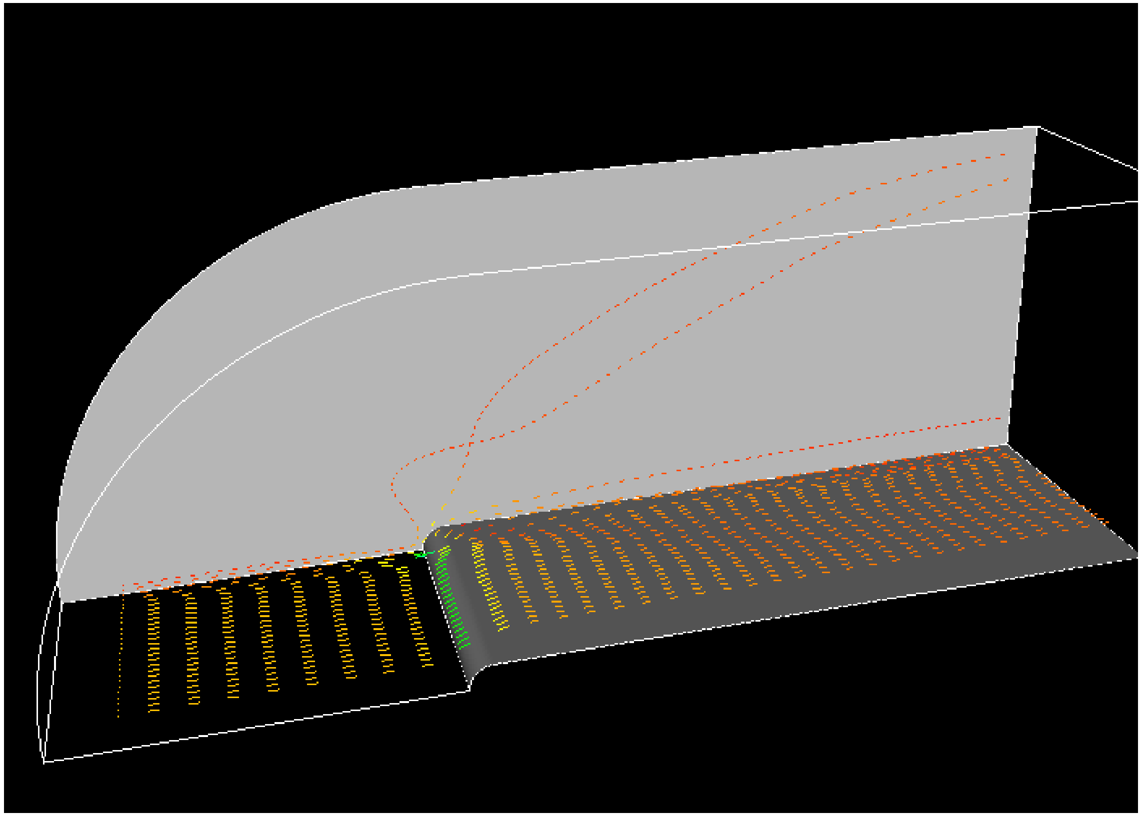
Vector Algorithms (continued)

Many enhancements of streamlines exist. Lines can be colored according to velocity magnitude to indicate speed of flow. Other scalar quantities such as temperature or pressure also may be used to color the lines. We also may create constant time dashed lines. Each dash represents a constant time increment. This, in areas of high velocity, the length of the dash will be greater relative to regions of lower velocity.

Vector Algorithms (continued)

Example

NASA's blunt fin data set



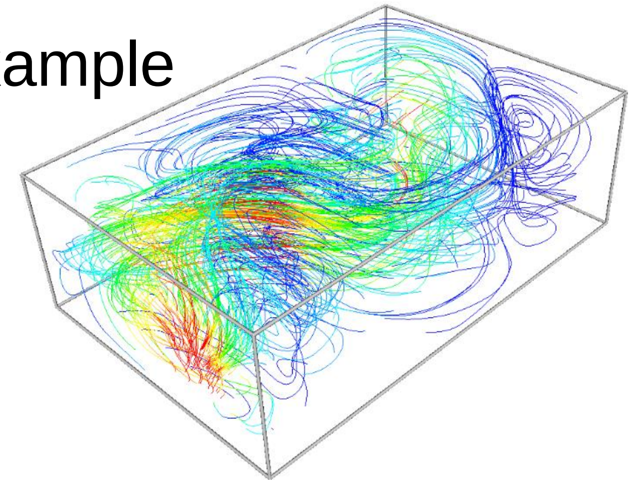
Vector Algorithms (continued)

Choosing an appropriate sampling strategy that solves the coverage, density, and continuity issues well is more critical when tracing streamlines in 3D datasets as compared to 2D datasets.

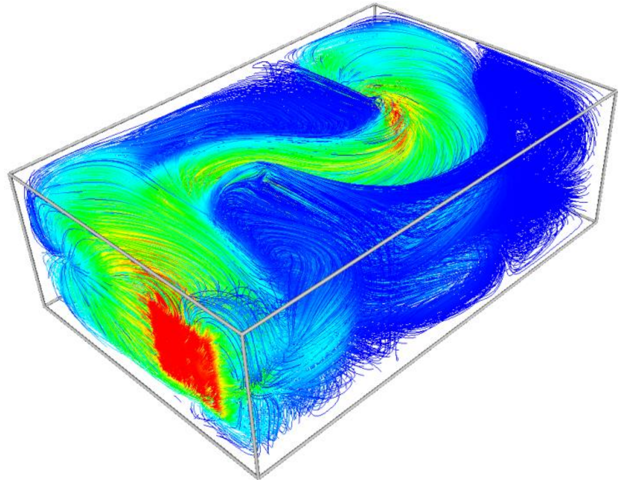
Similar to the using glyphs, streamline visualizations can get cluttered if the parameters for placement and opacity are not chosen properly.

Vector Algorithms (continued)

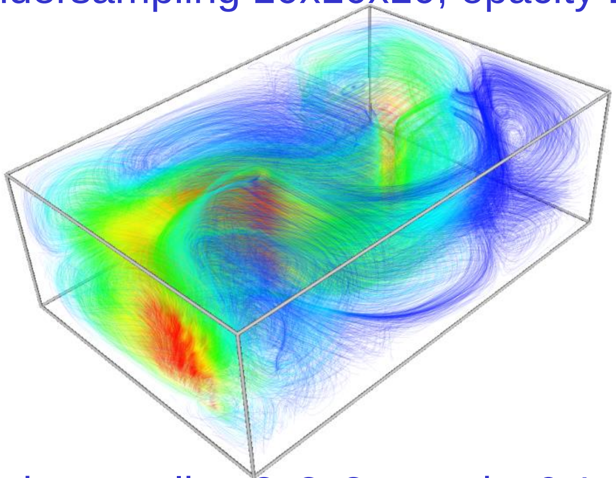
Example



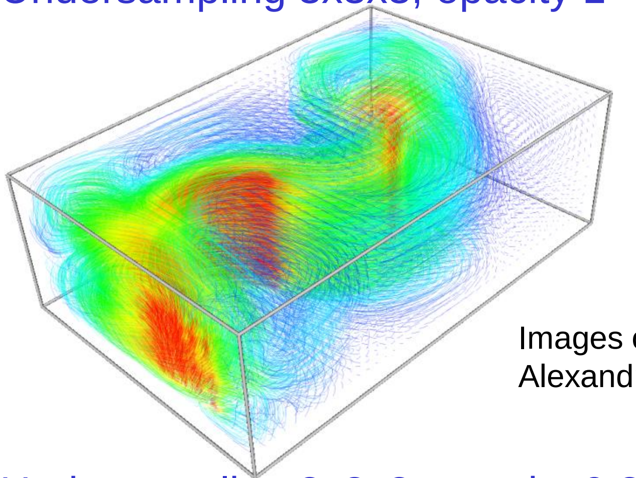
Undersampling 10x10x10, opacity 1



Undersampling 3x3x3, opacity 1



Undersampling 3x3x3, opacity 0.1



Undersampling 3x3x3, opacity 0.3

Images courtesy of Alexandru Telea

Vector Algorithms (continued)

Vector field topology

Vector fields have a complex structure characterized by special features called *critical points*. Critical points are locations in the vector field where the local vector magnitude goes to zero and the vector direction becomes undefined. At these points the vector field either converges or diverges, and/or local circulation around the point occurs. In order to understand critical points better, we take a look at linearly defined vector fields. Since we usually interpolate vector fields linearly, this will result in the most common cases of critical points.

Vector Algorithms (continued)

Critical points

Let *v* be a given vector field $v: W \rightarrow IR^3$ with $W \subset IR^3$ as defined on a face of a tetrahedron. Let further $x_0 \in W$ be a point where the vector field vanishes, i.e. $v(x_0) = 0$. Then x_0 is considered a critical point of the vector field v. Several terms are used synonymously for critical points. These are singularities, singular points, zeros, or equilibrium.

Vector Algorithms (continued)

Linear vector fields

A linear 3-D vector field v can be described by a matrix and a displacement vector. Therefore, a linear map $A: W \rightarrow IR^3$ described by the 3×3 matrix A and a vector $b \in IR^3$ can be found such that it describes the given vector field v, i.e. $v(x) = Ax + b$ for all $x \in W$.

Then, singularities can be found by directly solving the equation $Ax + b = 0$. Obviously, there cannot be more than one singularity located within one triangle when using linear interpolation. For the case $b = 0$ we consider the vector field described by Ax homogenous linear. Without loss of generality we assume homogenous linear vector fields in the further discussion of the theory of vector field topology.

Vector Algorithms (continued)

Classification of critical points

Singularities can be classified using the eigenvalues of the interpolating matrix A regarding their property of attracting or repelling the surrounding flow. If all eigenvalues have negative real parts the singularity is considered a *sink* which attracts the surrounding flow. On the other hand, if all eigenvalues have positive real parts the singularity is a *source* that repels the surrounding flow.

Vector Algorithms (continued)

Computing streamlines

Further analyzing the matrix A leads to a several types of vector fields distinguished by their major properties of the flow, i.e. the behavior of the streamlines within this vector field. In order to compute a streamline, the Cauchy problem has to be solved with initial problem $x(0) = k$, $k \in IR^3$:

$$\frac{d}{dt} x(t) = Ax(t)$$

Vector Algorithms (continued)

Solution to the Cauchy problem

It can be proven that the solution to the Cauchy problem for a linear vector field can be described by an exponential function:

$$x = e^{tA}k \quad \text{with} \qquad e^{A} = \sum_{i=0}^{\infty} \frac{A^{i}}{i!}$$

Different categories of vector fields can then be distinguished weather the matrix A is diagonalizable, resulting in a different behavior of the streamlines in each case. This leads to three main categories described in the following.

Vector Algorithms (continued)

Type 1 vector fields

The matrix A is diagonalizable, i.e. the eigenvalues λ and μ are real. Thus it is similar to a matrix B, i.e. there exist an invertible matrix P with $B = PAP^{-1}$, of the following structure:

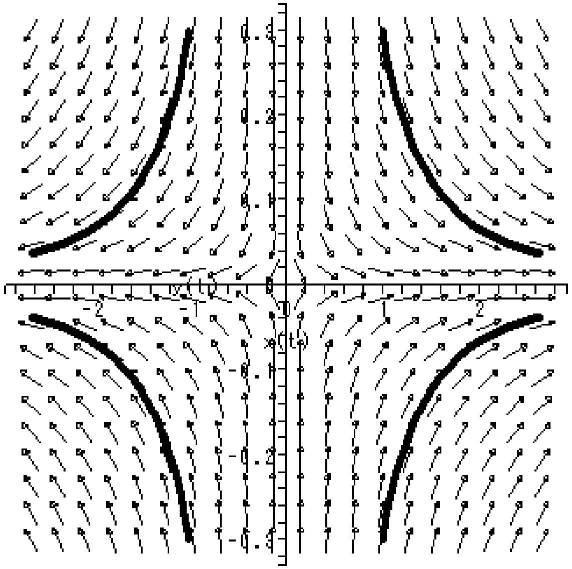
$$B = \begin{pmatrix} \lambda & 0 \\ 0 & \mu \end{pmatrix}$$

Due to the structure of the matrix B, a streamline $x(t)$ with initial condition $k = (k_1, k_2)$ can be computed in a vector field described by such a matrix using the following formula:

$$x(t) = \begin{pmatrix} e^{t\lambda} k_1 \\ e^{t\mu} k_2 \end{pmatrix}$$

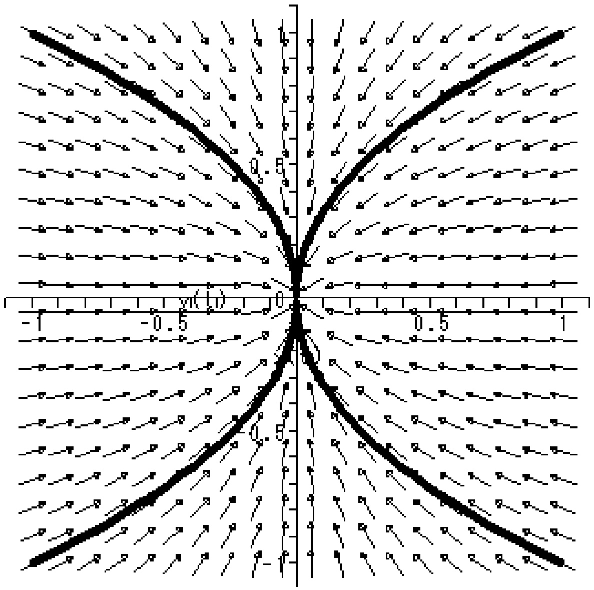
Vector Algorithms (continued)

By computing streamlines we can generate a phase portrait of the different cases of vector fields within this category. Three different types are possible, again distinguished by the eigenvalues of the interpolating matrix A. The first case, where $\lambda > 0 > \mu$, results in a *saddle singularity*:



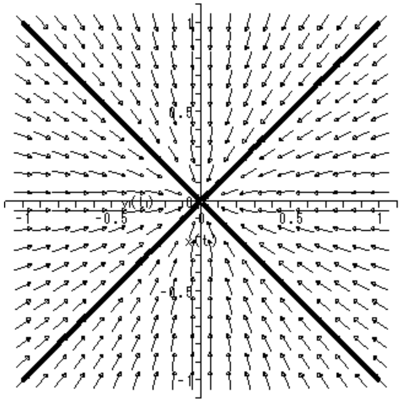
Vector Algorithms (continued)

The second case, described by an eigenvalue configuration of $\lambda < \mu < 0$, describes a *node singularity*:



Vector Algorithms (continued)

The last case with two identical eigenvalues is the *focus singularity*, for example $\lambda = \mu < 0$.



The examples shown here mainly show sinks; however, the same types of singularities occur with sources. The only difference is in the sign of the eigenvalues, i.e. multiplying the eigenvalues by -1 results in the same singularities as sources.

Vector Algorithms (continued)

Type 2 vector fields

The two eigenvalues of the matrix A have a non-imaginary part, i.e. A is similar to the following matrix:

$$B = \begin{pmatrix} a & -b \\ b & a \end{pmatrix}$$

When substituting the values a and b in the above matrix by introducing new values θ and r:

$$r = \sqrt{a^2 + b^2}$$
$$\theta = \arccos(a/r)$$

the matrix B can be rewritten as follows:

$$B = \begin{pmatrix} a & -b \\ b & a \end{pmatrix} = \begin{pmatrix} r & 0 \\ 0 & r \end{pmatrix} \cdot \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}$$

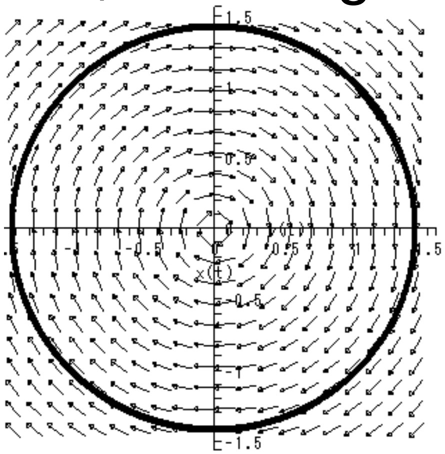
Vector Algorithms (continued)

Obviously, a vector field described by such a matrix has a strong rotational component. Consequently, a streamline $x(t)$ with initial condition $k = (k_1, k_2)$ can be computed using the following formula:

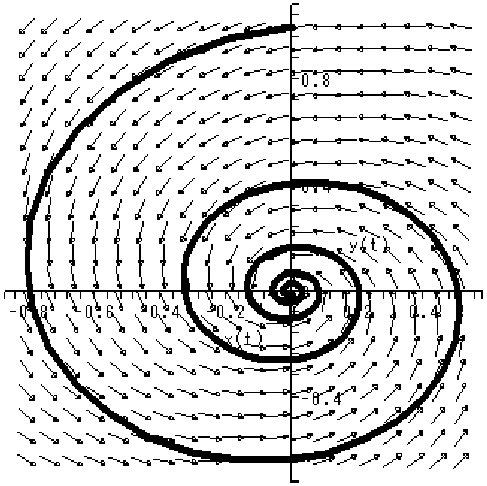
$$x(t) = e^{ta} \cdot \begin{pmatrix} k_1 \cos(tb) - k_2 \sin(tb) \\ k_1 \sin(tb) + k_2 \cos(tb) \end{pmatrix}$$

Vector Algorithms (continued)

For *a=0*, the streamlines describe perfect, concentric circles, resulting in the *center singularity*:



Otherwise, a *spiral singularity* is described with streamlines spiralling around the singularity and then eventually ending up at the singularity itself.



Vector Algorithms (continued)

Type 3 vector fields

The matrix A is not diagonalizable and the two eigenvalues are equal, i.e. $\lambda = \mu$. In this case, A is similar to the following matrix:

$$B = \begin{pmatrix} \lambda & 0 \\ 1 & \lambda \end{pmatrix}$$

By splitting up the matrix *B* into two components

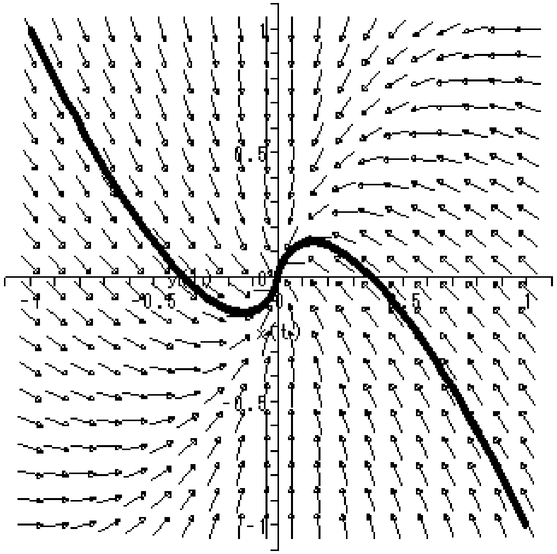
$$B = \begin{pmatrix} \lambda & 0 \\ 1 & \lambda \end{pmatrix} = \lambda \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix}$$

it can be easily seen that a streamline with initial condition $k = (k_1, k_2)$ integrated through such a vector field can be described by:

$$x(t) = e^{t\lambda} \cdot \begin{pmatrix} k_1 \\ k_1 t + k_2 \end{pmatrix}$$

Vector Algorithms (continued)

This case resembles an *improper node singularity*:



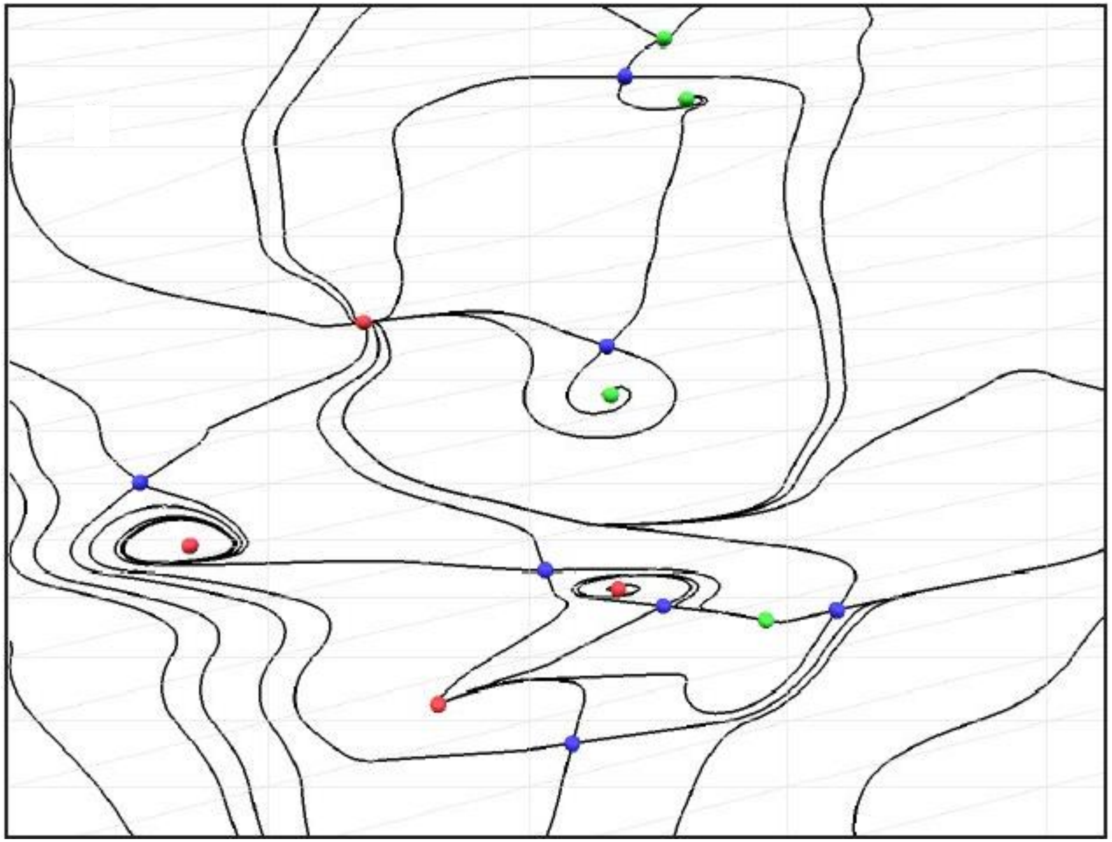
Vector Algorithms (continued)

Topological analysis

The topological graph, or simply topology, of a vector field describes the structure of the flow or phase portrait. Separatrices are used to separate the areas of the flow into regions with similar behavior. Separatrices can be easily computed by integrating streamlines emerging from saddle singularities in direction of the eigenvectors of the interpolating matrix. The topological graph then consists of the singularities and the separatrices.

Vector Algorithms (continued)

Example



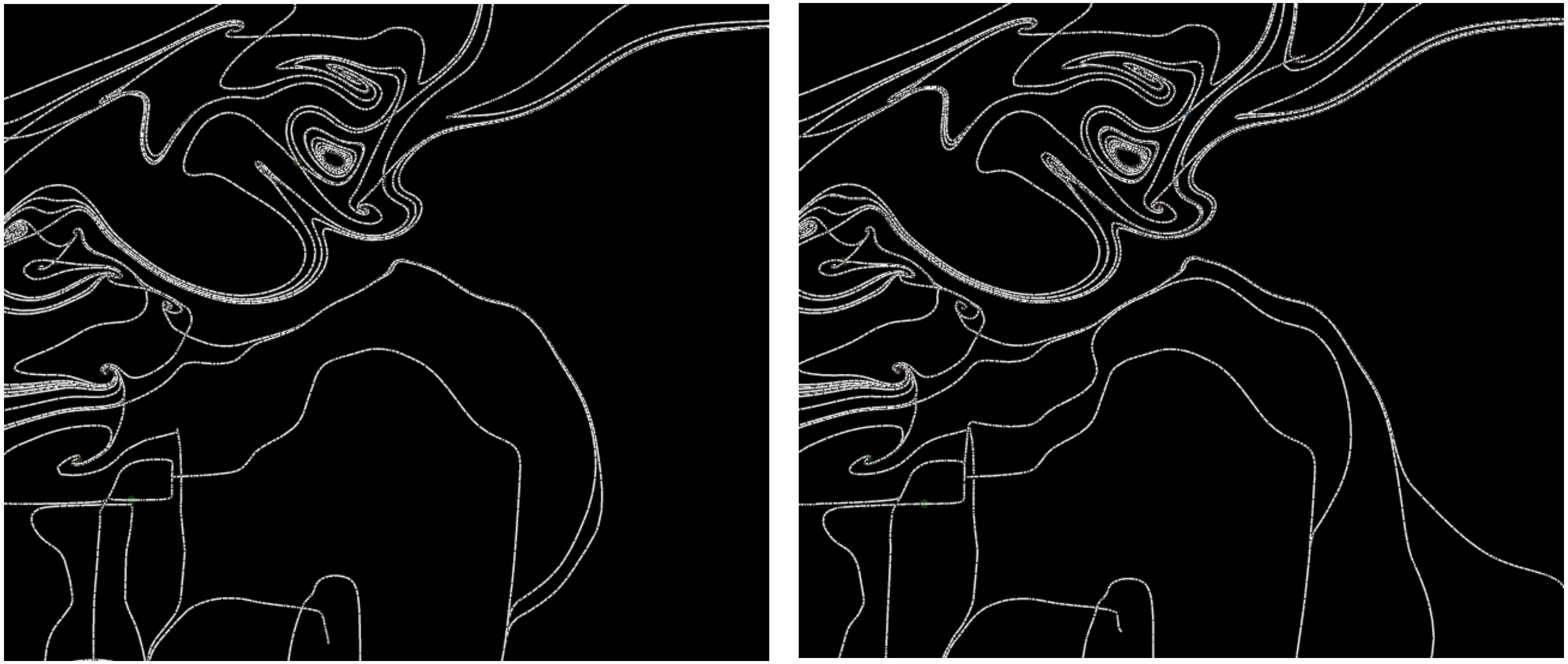
Vector Algorithms (continued)

Topological analysis with different interpolation

The topological graph be different when using different types of interpolations. By changing the cell type alone, for example triangulating the cells by splitting up rectangles into two triangles, the topological graph can change. Hence, the interpolation technique used for integrating the streamlines should be chosen with special care!

Vector Algorithms (continued)

Example

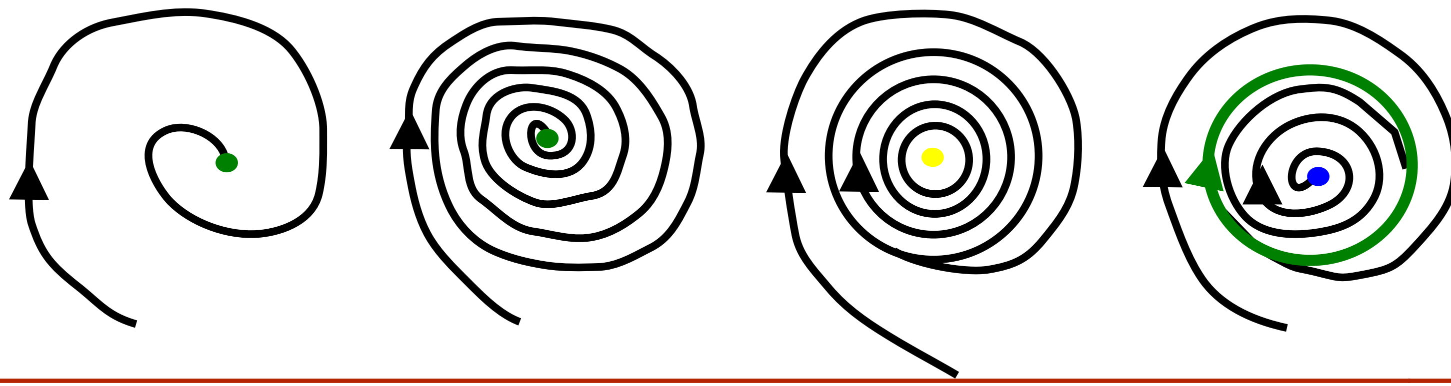


Vector Algorithms (continued)

Closed streamlines

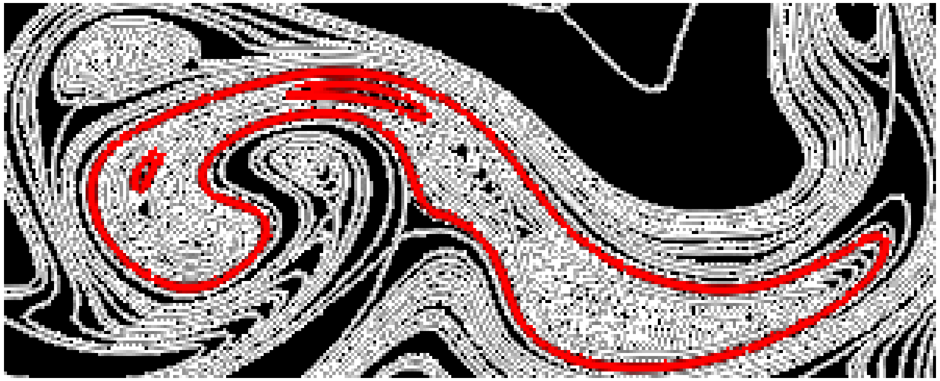
There are more topological features other than separatrices and critical points. The flow can – in the same way as with critical points – be attracted or repelled by a *closed streamline*.

For example, the Hopf bifurcation describes critical point that changes from source to sink resulting in an attracting closed streamline:



Vector Algorithms (continued)

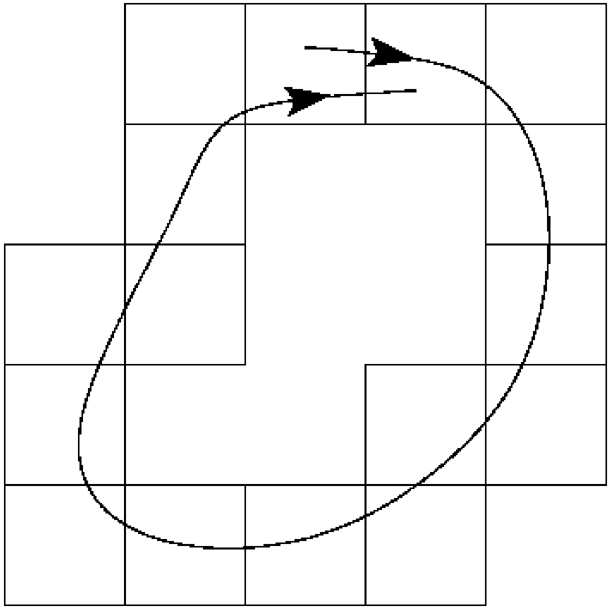
By considering closed streamlines are able to connect parts of the topological skeleton and to complete the topological analysis.



Since closed streamlines are a global feature we cannot identify them by using local properties of the vector field.

Vector Algorithms (continued)

Obviously a closed streamline runs through the same cells of our grid over and over again. The idea for finding a closed streamline is basically to prove that the closed streamline does not leave *cell cycle*.

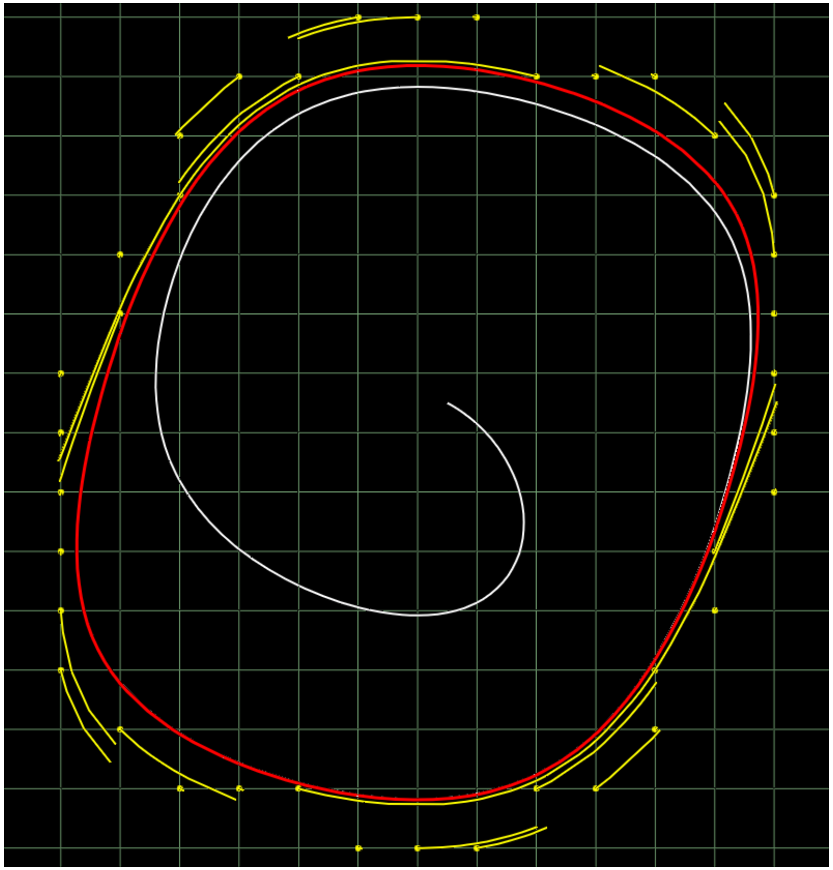


Vector Algorithms (continued)

In order to check if the streamline can leave cell cycle we basically need to start *backward integration* at every point at edge of cell cycle. If the backward integration exists that run toward the streamline then the streamline leaves cell cycle (near the point where the backward integration was started). Otherwise, if such backward integration does not exist then the streamline stays inside the cell cycle forever. The problem, however, is that the number of points at the edge of the cell cycle infinite.

Vector Algorithms (continued)

Therefore, we define potential exits. Potential exits are considered those points that are either vertices of cell cycle or points at edge of cell cycle where the vector field is tangential to that edge. Then, backward integration is only necessary at these potential exits since the integration of two neighboring potential exits cover the entire edge connecting them.



Vector Algorithms (continued)

Example

In a vector field describing a hurricane, the eye of the hurricane is surrounded by a closed streamline if the vector field is projected on to a 2-D plane. Thus, the eye can be identified by finding the closed streamline in the projected vector field.

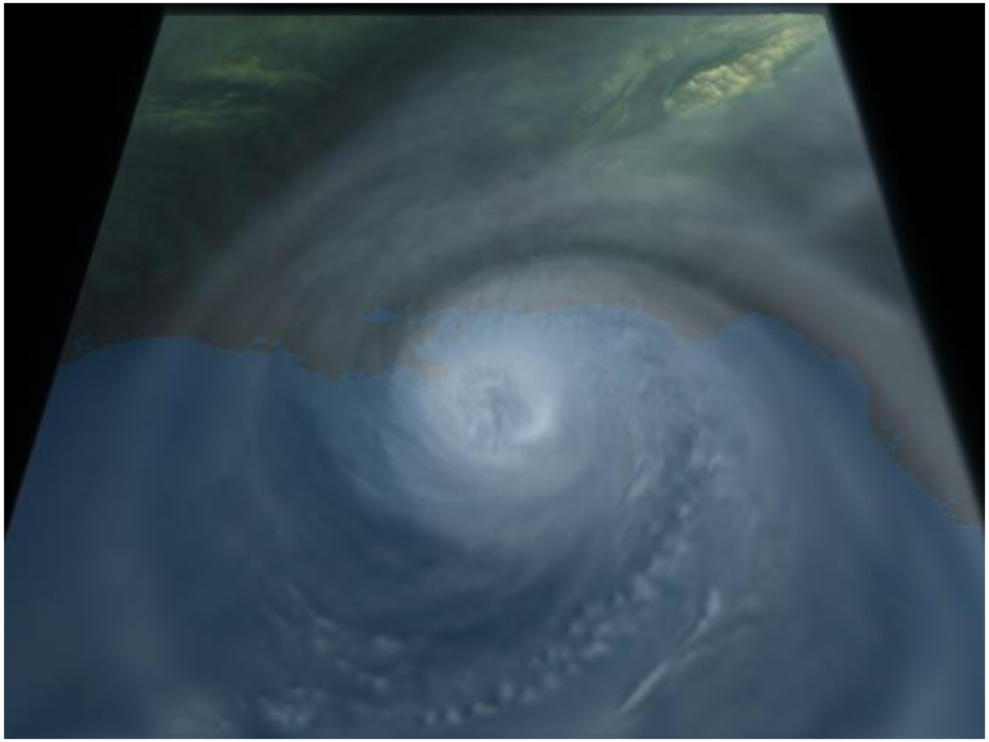
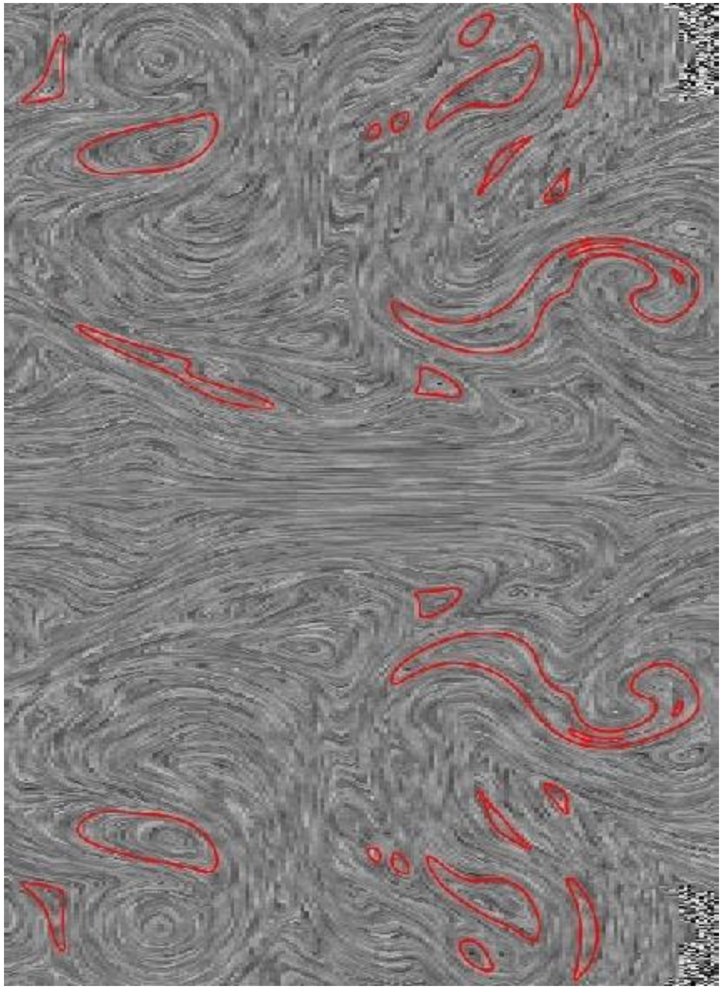


Image courtesy of David Bock, NCSA

Vector Algorithms (continued)

Example

In combustion processes it is important that the gas stays in an area for a certain amount of time for the gas to burn completely. Closed streamlines are a hint for recirculation zones, i.e. areas where the gas stays for longer period of time. Hence, closed streamlines indicate areas with better combustion.



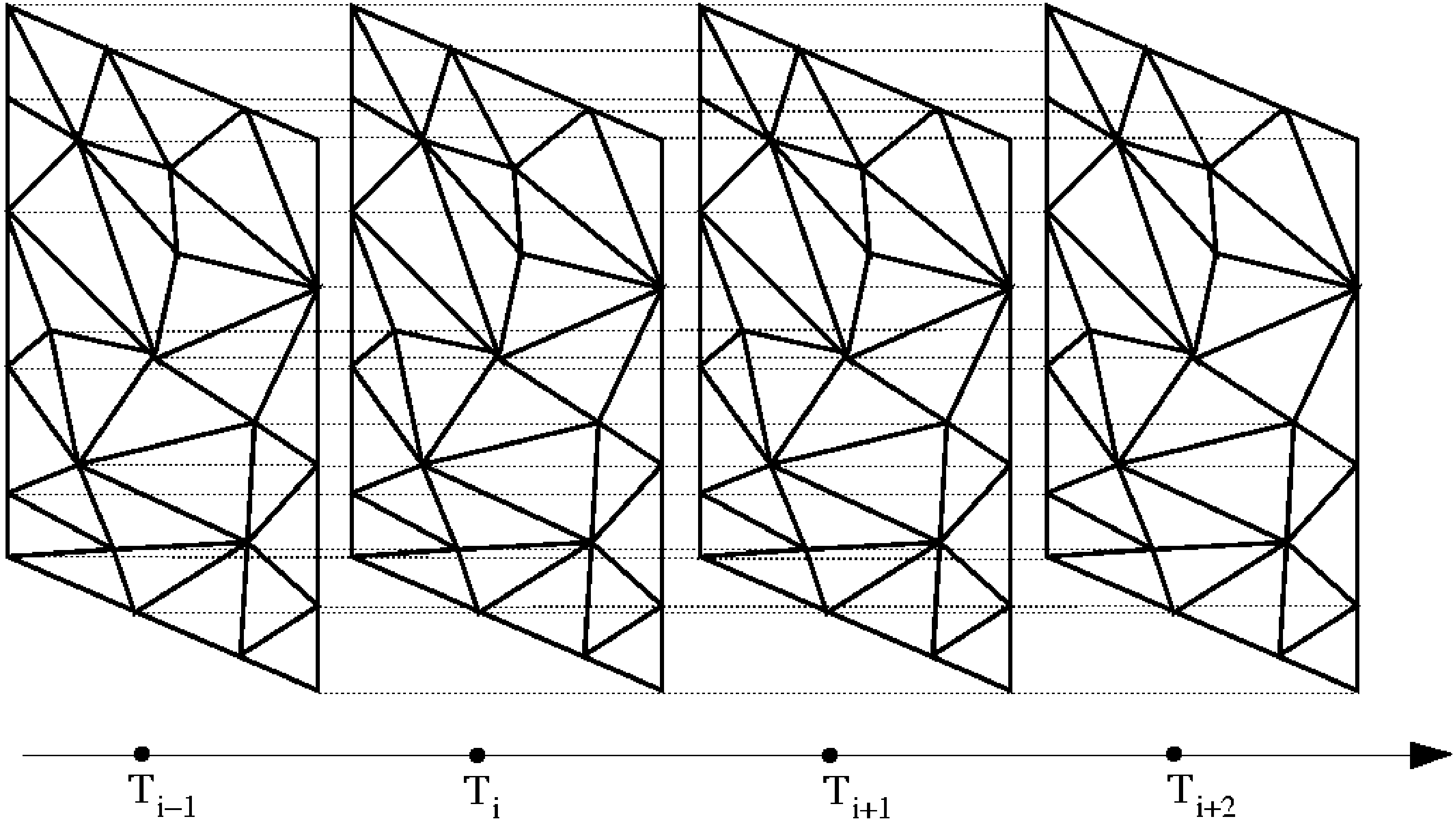
Vector Algorithms (continued)

Tracking closed streamlines over time

In order to understand how and when closed streamlines occur, we can take a look at a vector field that changes over time. At various instances in time, a 2-D vector field is given. Through linear interpolation between consecutive time steps we can compute vectors at every instance in time and any location within the 2-D space defined by the domain of the vector field.

Vector Algorithms (continued)

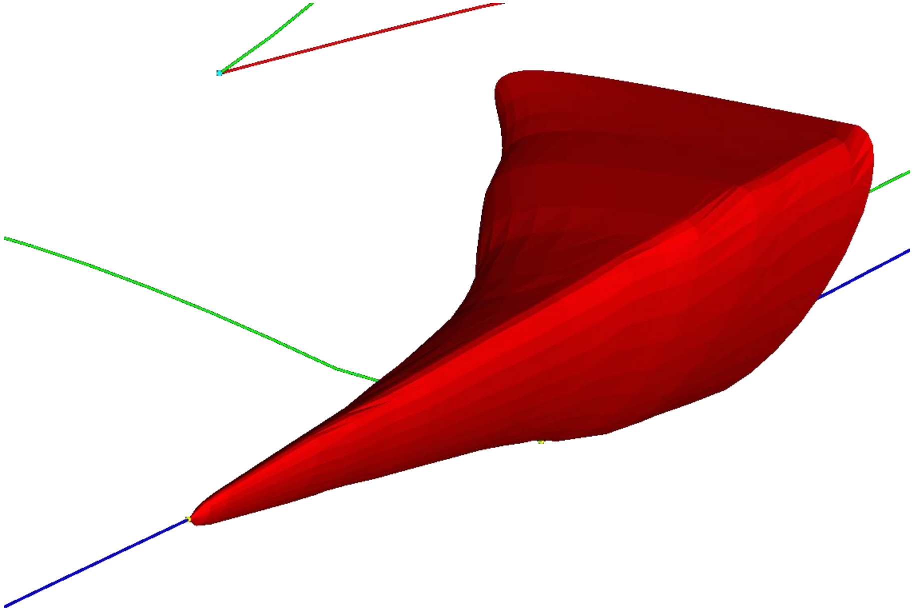
2-D grid structure including time



Vector Algorithms (continued)

Closed streamlines over time

We can now determine the location of the closed streamlines within each of the time steps and then connect the resulting curves with triangles:



Vector Algorithms (continued)

Morse Smale Complex

Vector Algorithms (continued)

Morse Theory

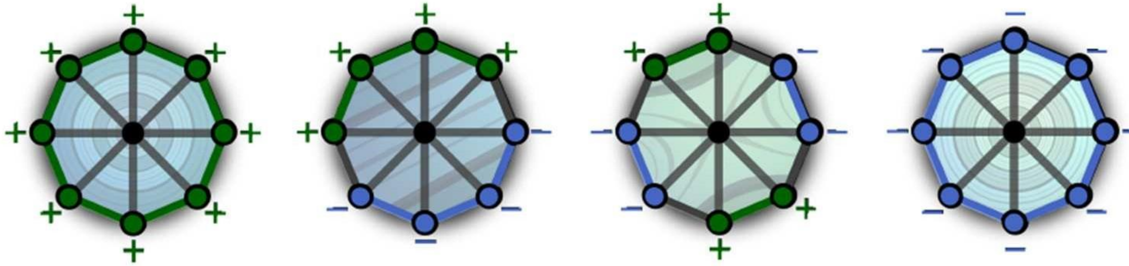
- Investigates the topology of a surface by looking at *critical points* of a function on that surface.

$$\nabla f\,(p) = \left(\frac{\partial f}{\partial x}(p) \quad \frac{\partial f}{\partial y}(p)\right) = 0$$

- A function f is a Morse function if
  - f is smooth
  - All critical points are isolated
  - All critical points are non−degenerate $\det(\mathrm{Hessian}(p)) \neq 0$

Notion of Critical Points and Their Index

- Minima, maxima, and saddles
- Topological changes
- Piecewise linear interpolation
- Barycentric coordinates on triangles

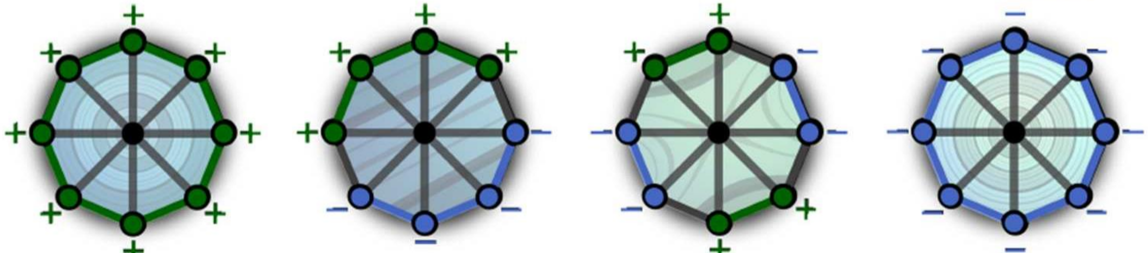


Notion of Critical Points and Their Index

Standard form of a non-degenerate critical point *p* of a function $f\colon M^d \to R$

$$f = -X_1^2 - X_2^2 - \cdots - X_h^2 + X_{h+1}^2 + \cdots + X_d^2 + c$$

Where $(X_1, X_2, \cdots, X_n)$ are some local coordinate system such that *p* is the origin and $f(p) = c$.



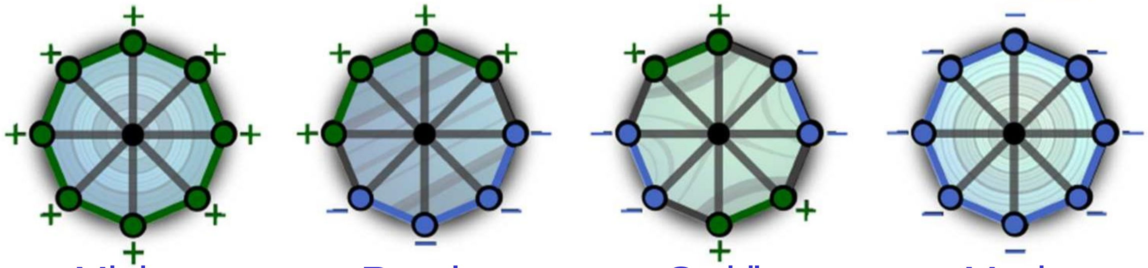
Notion of Critical Points and Their Index

Standard form of a non-degenerate critical point p of a function f: $M^d \rightarrow R$

$$f = -X_1^2 - X_2^2 - \cdots - X_h^2 + X_{h+1}^2 + \cdots + X_d^2 + \quad c$$

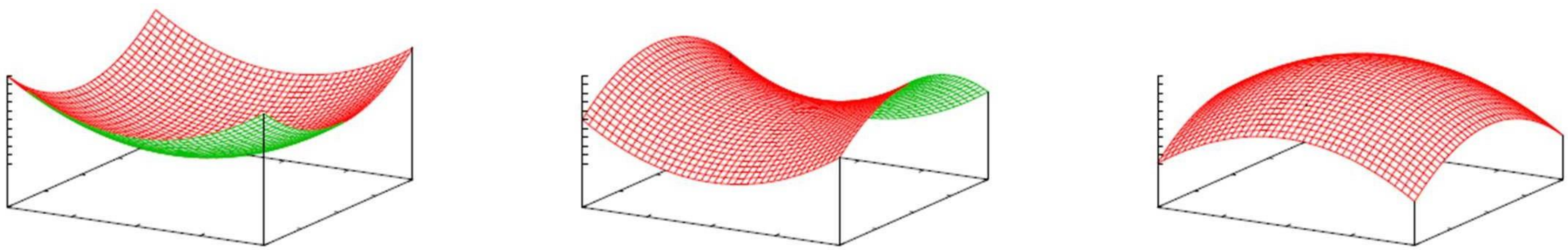
Where $(X_1, X_2, \cdots, X_n)$ are some local coordinate system such that p is the origin and $f(p) = c$.

Then the number of minus signs, λ, is the **index** of p.



Notion of Critical Points and Their Index

Examples of critical points in 2-manfold



Minima

Saddle

Maxima

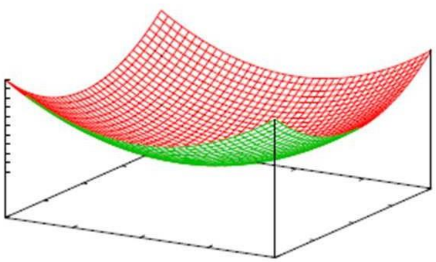
$x^2 + y^2$

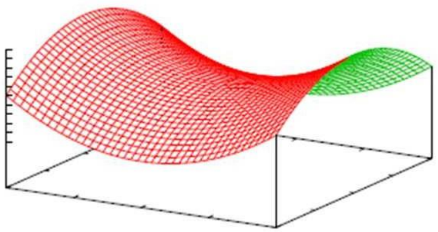
$x^2 - y^2$

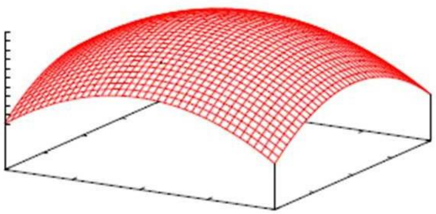
$-x^2 - y^2$

Notion of Critical Points and Their Index

Examples of critical points in 2-manfold







Minimum	Saddle	Maximu m
$x^2 + y^2$	$x^2 - y^2$	$-x^2 - y^2$
$\lambda = 0$	$\lambda = 1$	$\lambda = 2$

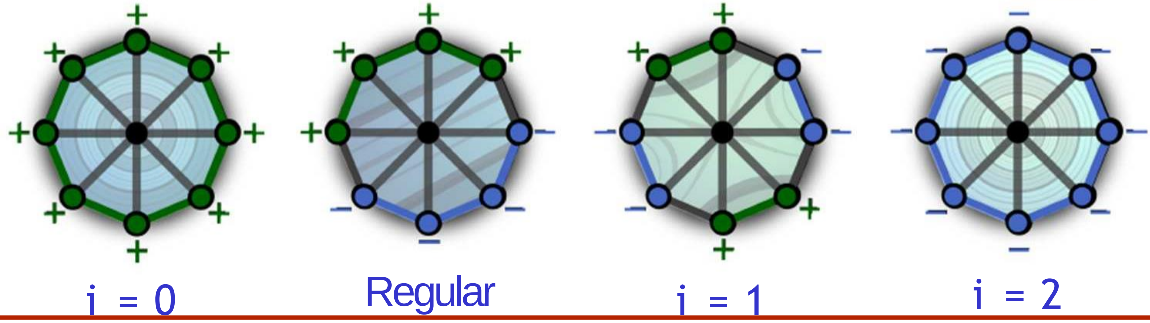
Notion of Critical Points and Their Index

Standard form of a non-degenerate critical point *p* of a function $f: M^d \rightarrow R$

$$f = -x_1^2 - x_2^2 - \cdots - x_h^2 + x_{h+1}^2 + \cdots + x_d^2 + c$$

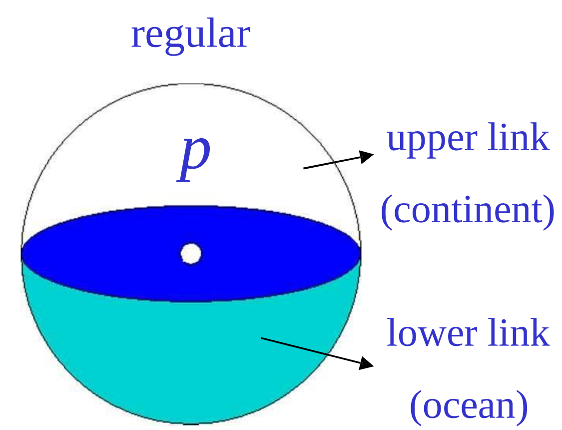
Where $(x_1, x_2, \cdots, x_n)$ are some local coordinate system such that *p* is the origin and $f(p) = c$.

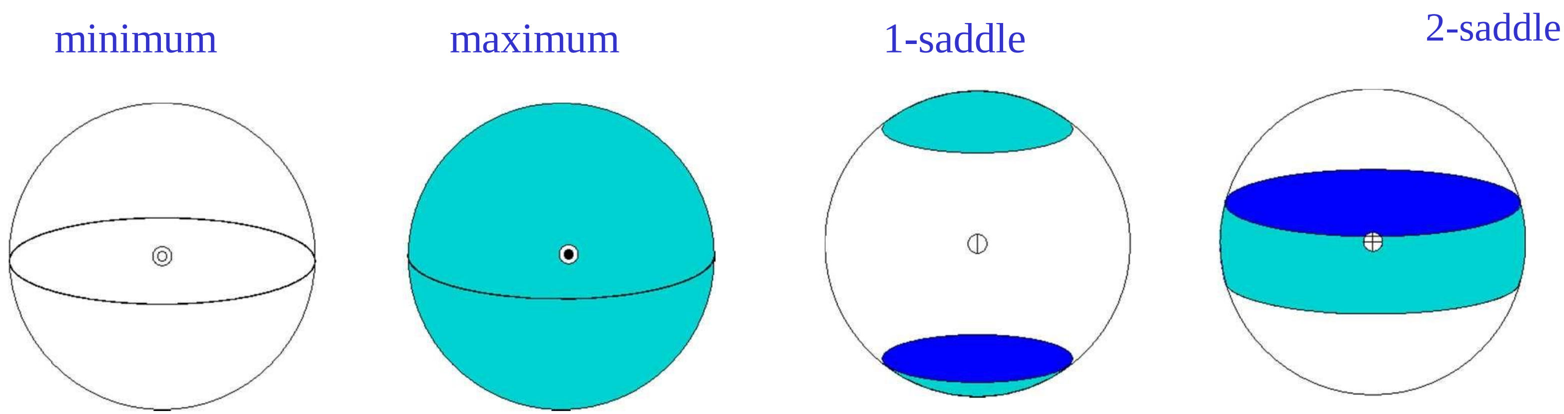
Then the number of minus signs, λ, is the **index** of *p*.



Critical Points in 3D

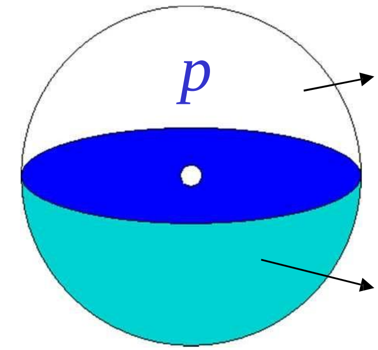
- Have zero gradient
- Characterized by lower link

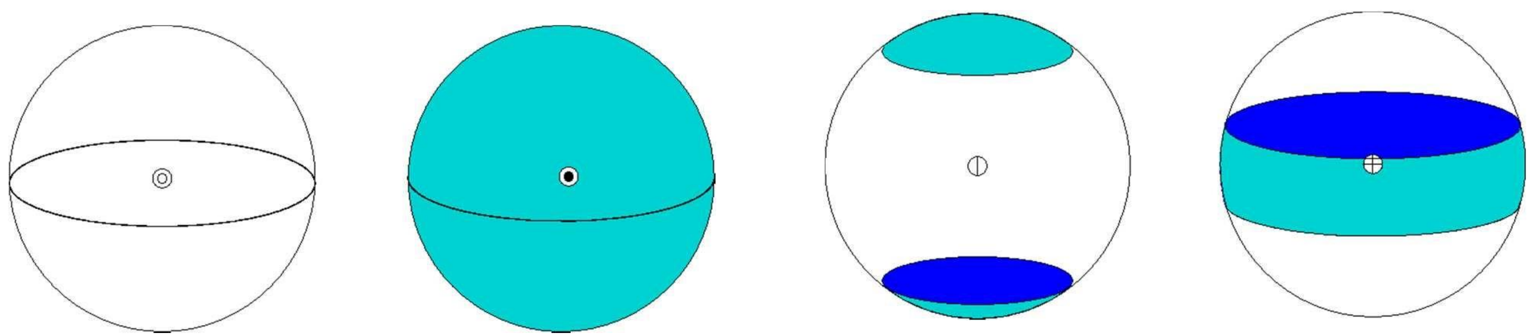




Critical Points in 3D

- Have zero gradient
- Characterized by lower link



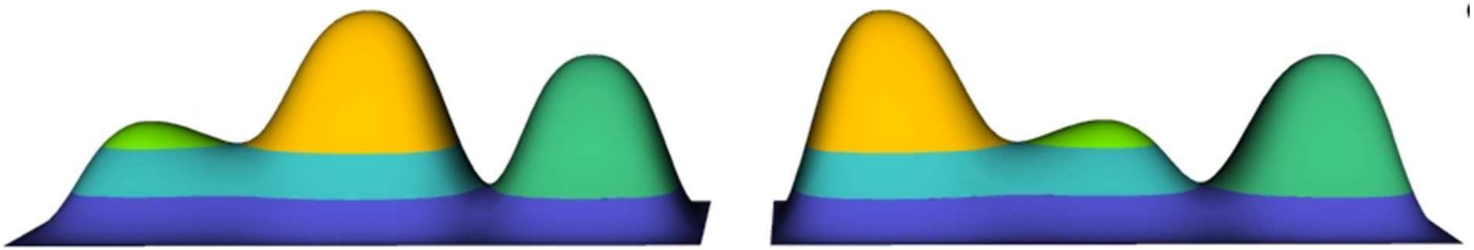


Reeb Graph

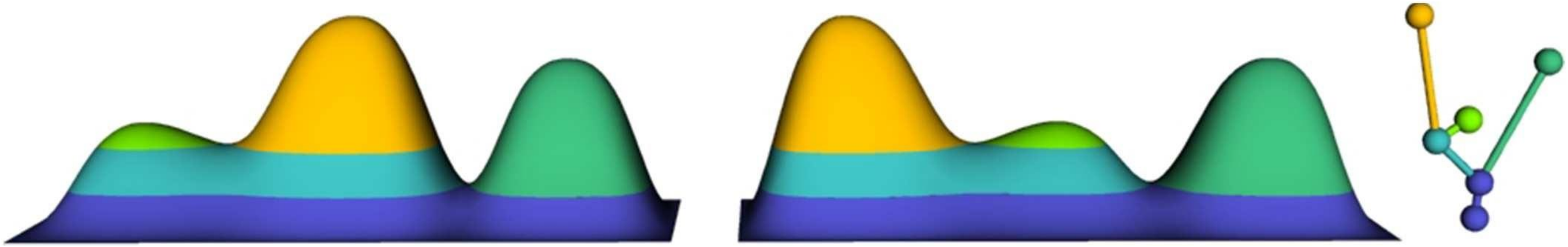
- The Reeb graph maps out the relationship between index–0 and index–1, and index–(d–1) and index–d critical points in a d–dimensional space.
  - In 2-manifold, index(0) to index(1), and index(1) to index(2)
  - In 3-manifold, index(0) to index(1), and index(2) to index(3)

- The **contour tree** is a Reeb graph defined over a simply connected Euclidean space E^d

Limitation of ReebGraph

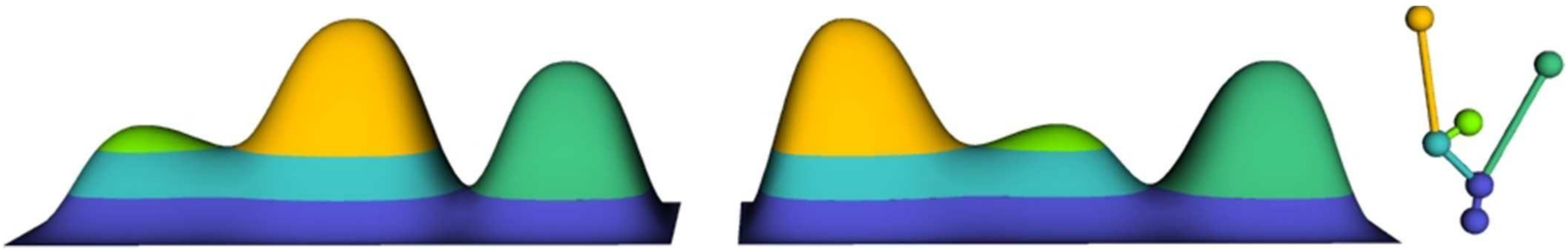


Limitation of ReebGraph



Lacking the geometric connectivity of the features

Limitation of ReebGraph



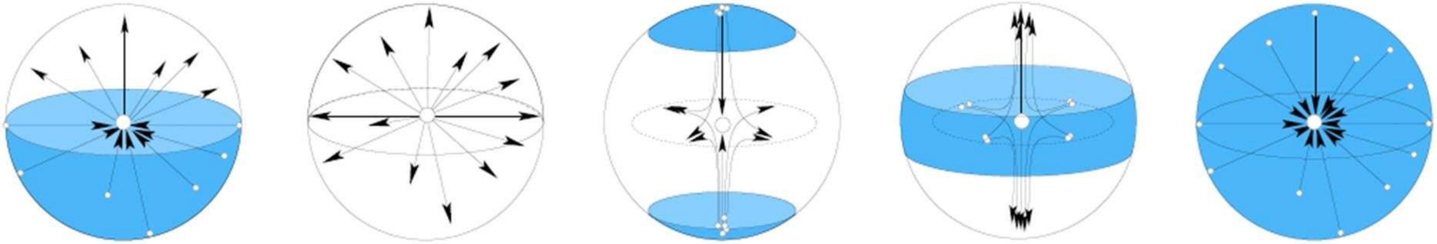
Lacking the geometric connectivity of the features

Additionally, for higher dimensional manifolds (>2), the saddle- saddle connections are not represented in the Reeb graph.

Morse Complex

- Instead of partitioning a manifold according to the behavior of level sets, it is more general to partition the manifold based on the behavior of the **gradient**.

- The gradient of a function defines a smooth **vector field** on M with zeroes at critical points.



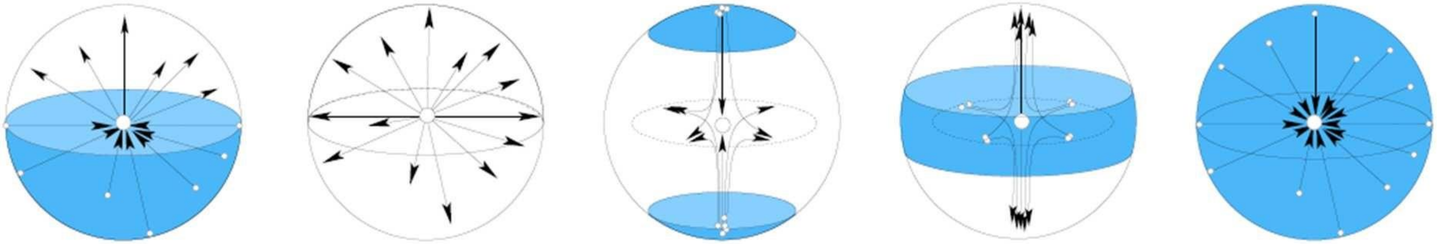
Morse Complex

- Integral line:

$$\frac{\partial}{\partial t} l(t) = \nabla f\, l(t()) \quad \text{for all } t \in R$$

  – Integral lines represent the flow along the gradient between critical points.

- Origin: $org(l) = \lim_{t \to -\infty} l(t)$
- Destination: $dest(l) = \lim_{t \to \infty} l(t)$



Morse Complex

Integral lines have the following properties

– Two integral lines are either disjoint or the same, i.e. uniqueness of each integral line

– Integral lines cover all of M

– The origin and destination of an integral line are critical points of f (except at boundary)

– In gradient vector field, integral lines are monotonic, i.e. $\mathrm{org}(l) \neq \mathrm{dest}(l)$

Morse Complex

Ascending/descending Manifolds

- Let p be a critical point of $f\colon M \to R$.
- The **ascending manifold** of p is the set of points belonging to integral lines whose origin is p.
- The descending manifold of p is the set of points belonging to integral lines whose destination is p.

Note that ascending and descending manifolds are also referred to as **unstable and stable** manifolds, **lower and upper disks**, and **right-hand and left-hand disks**.

Morse Complex

Morse Complex

– Let f: $M^d \rightarrow R$ be a Morse function. The complex of descending manifold of f is called the Morse complex

Morse Complex

Morse Complex

- Let f: $M^d \rightarrow R$ be a Morse function. The complex of descending manifold of f is called the Morse complex

CW-complexes

- Built on top of cells (0-cells, 1-cells, …., d-cells) via topologically gluing.
- The *C* stands for "closure-finite", and the *W* for "weak topology".
- Triangular mesh is one simple example of CW-complexes.

Morse-Smale Complex

Morse-Smale Function

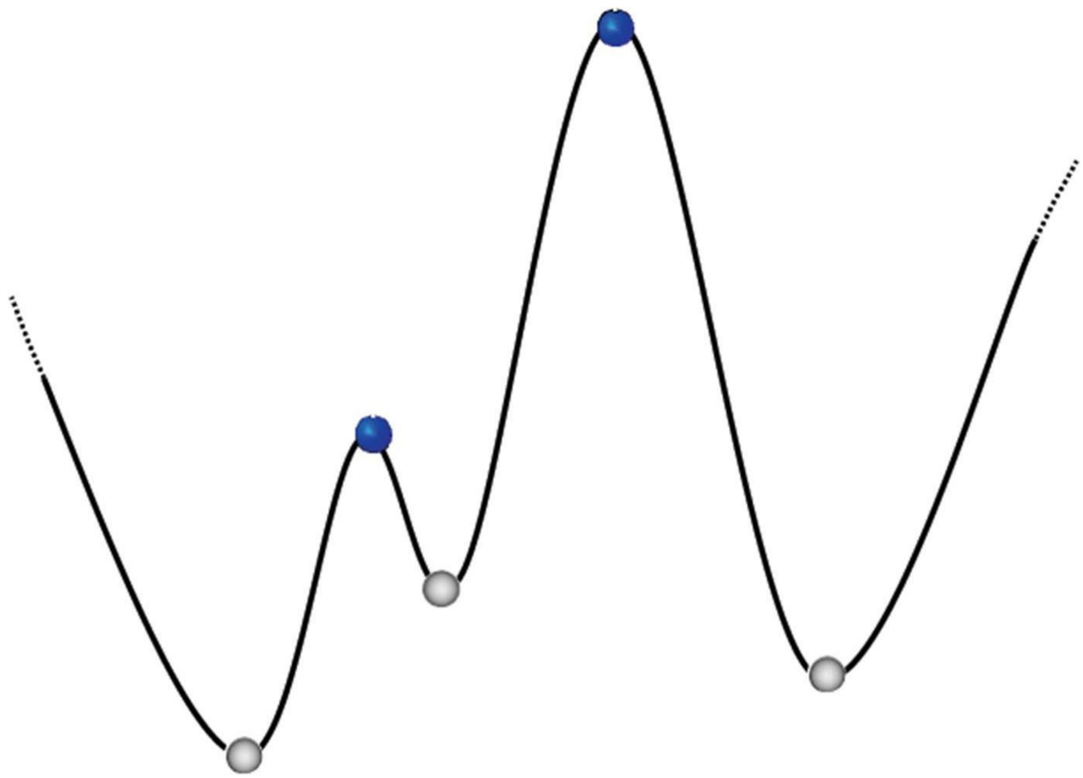
- A Morse function f is Morse-Smale if the ascending and descending manifolds intersect only transversally.
  - Intuitively, an intersection of two manifolds as transversal when they are not “parallel” at their intersection.

- A pair of critical points that are the origin and destination of an integral line in the Morse-Smale function cannot have the same index!
- Furthermore, the index of the critical point at the origin is less than the index of the critical point at the destination.

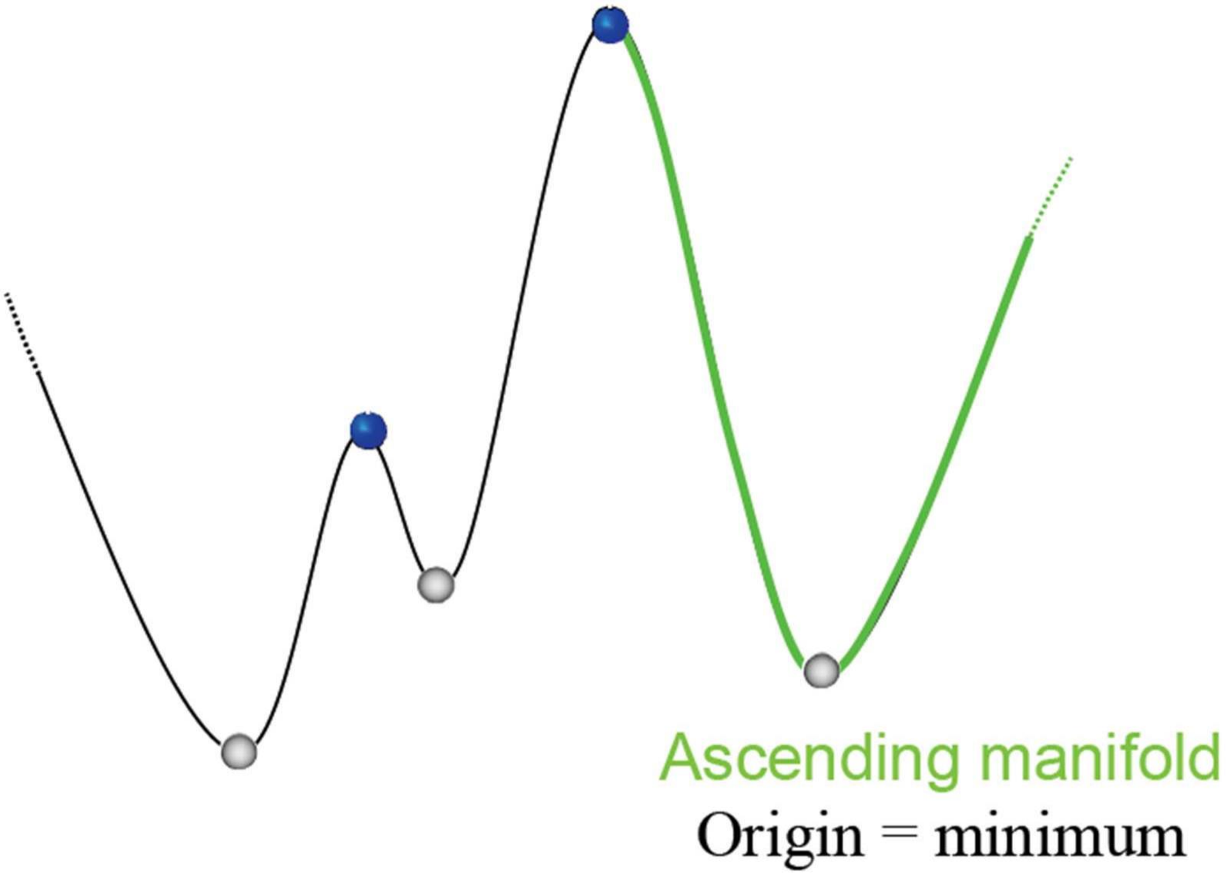
Morse-Smale Complex

Given a Morse-Smale function f, the Morse-Smale complex of f is the complex formed by the intersection of the Morse complex of f and the Morse complex of $-f$.

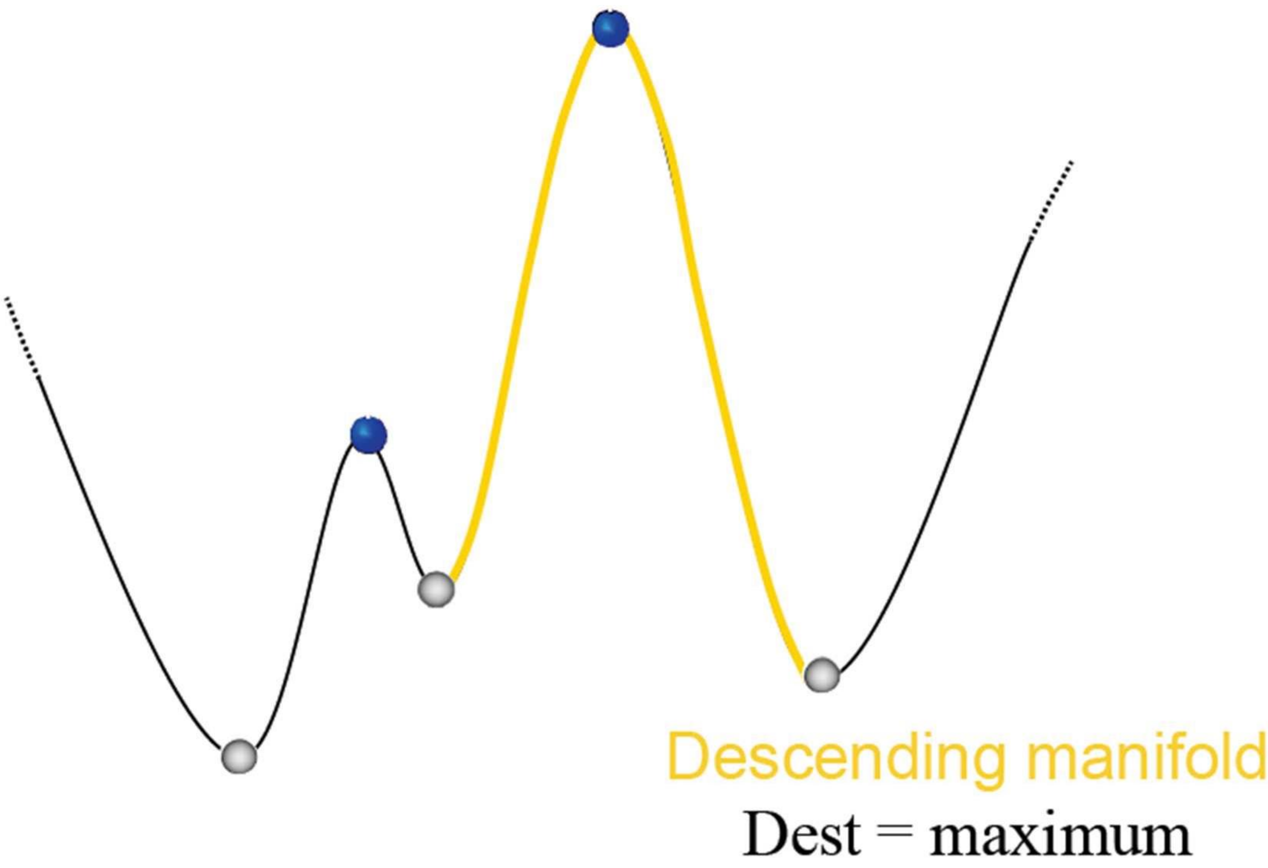
Morse-Smale Complex-1D



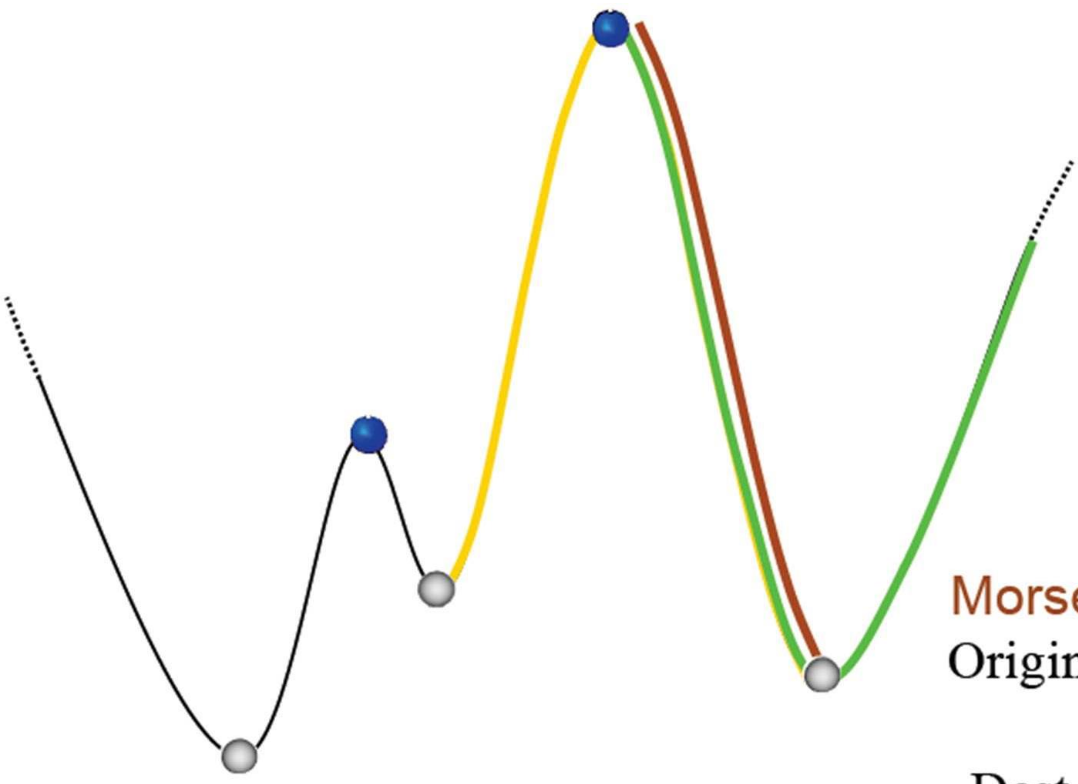
Morse-Smale Complex-1D



Morse-Smale Complex-1D

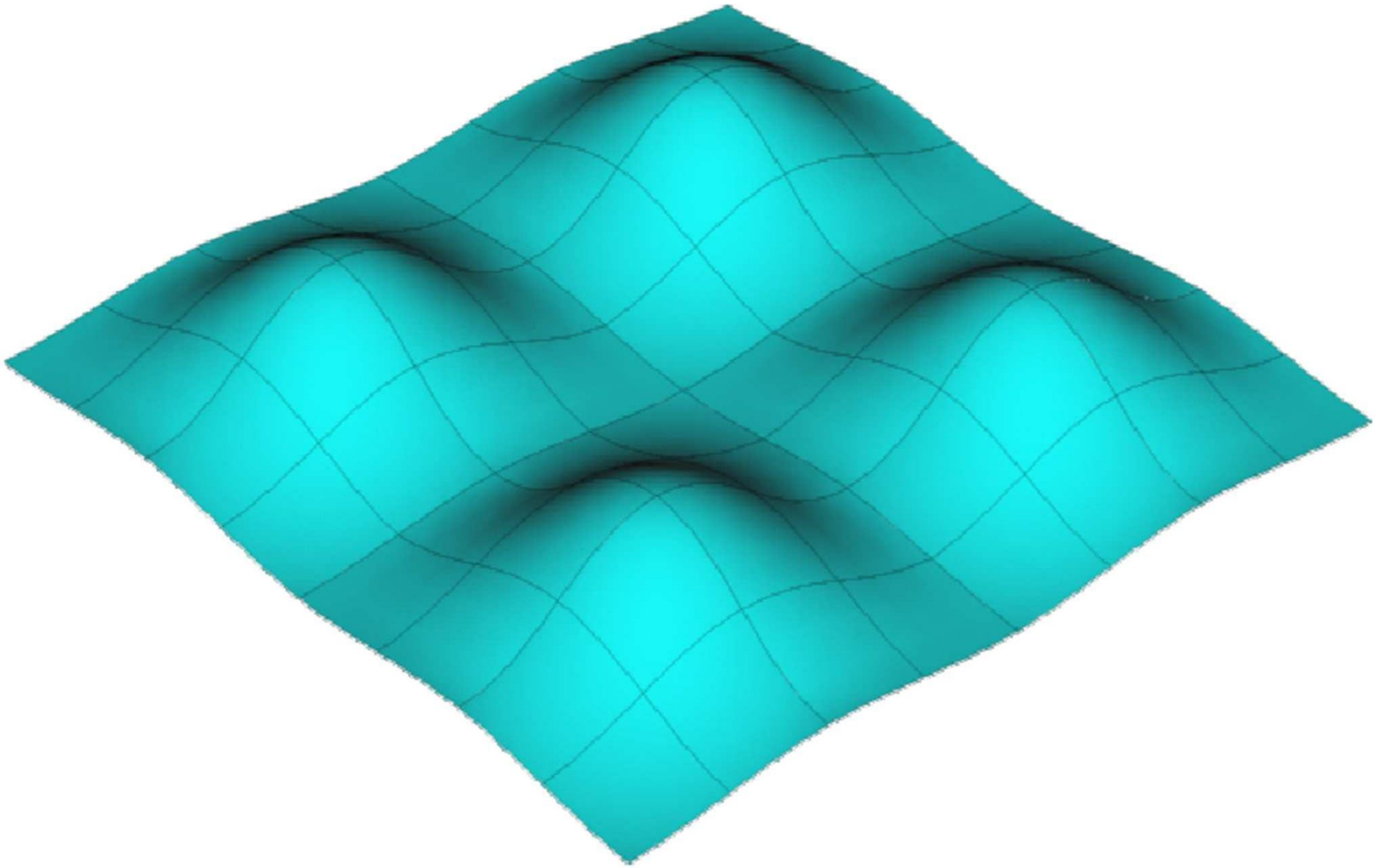


Morse-Smale Complex-1D

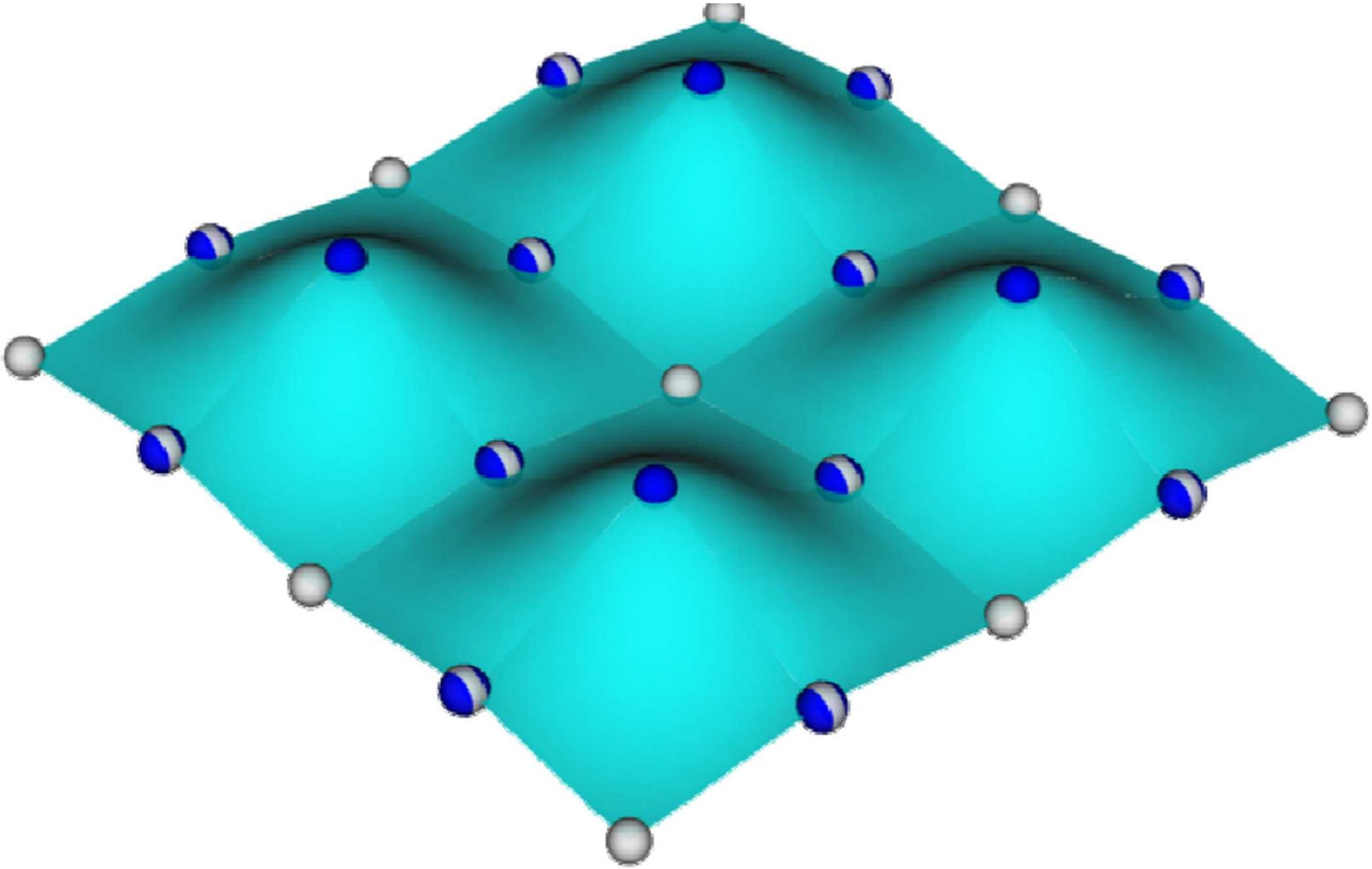


Morse-Smale cell
Origin = minimum
and
Dest = maximum

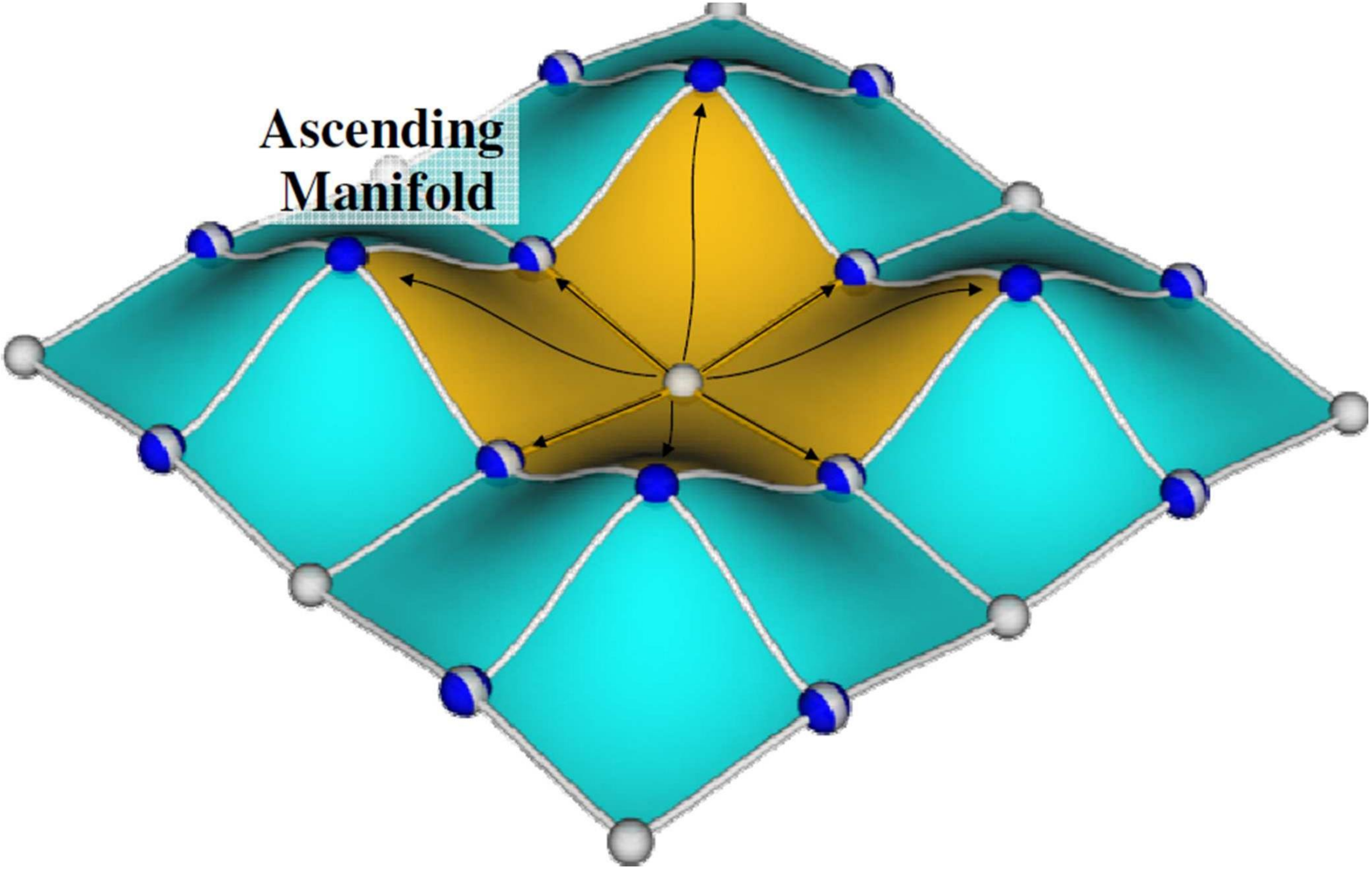
Morse-Smale Complex-2D



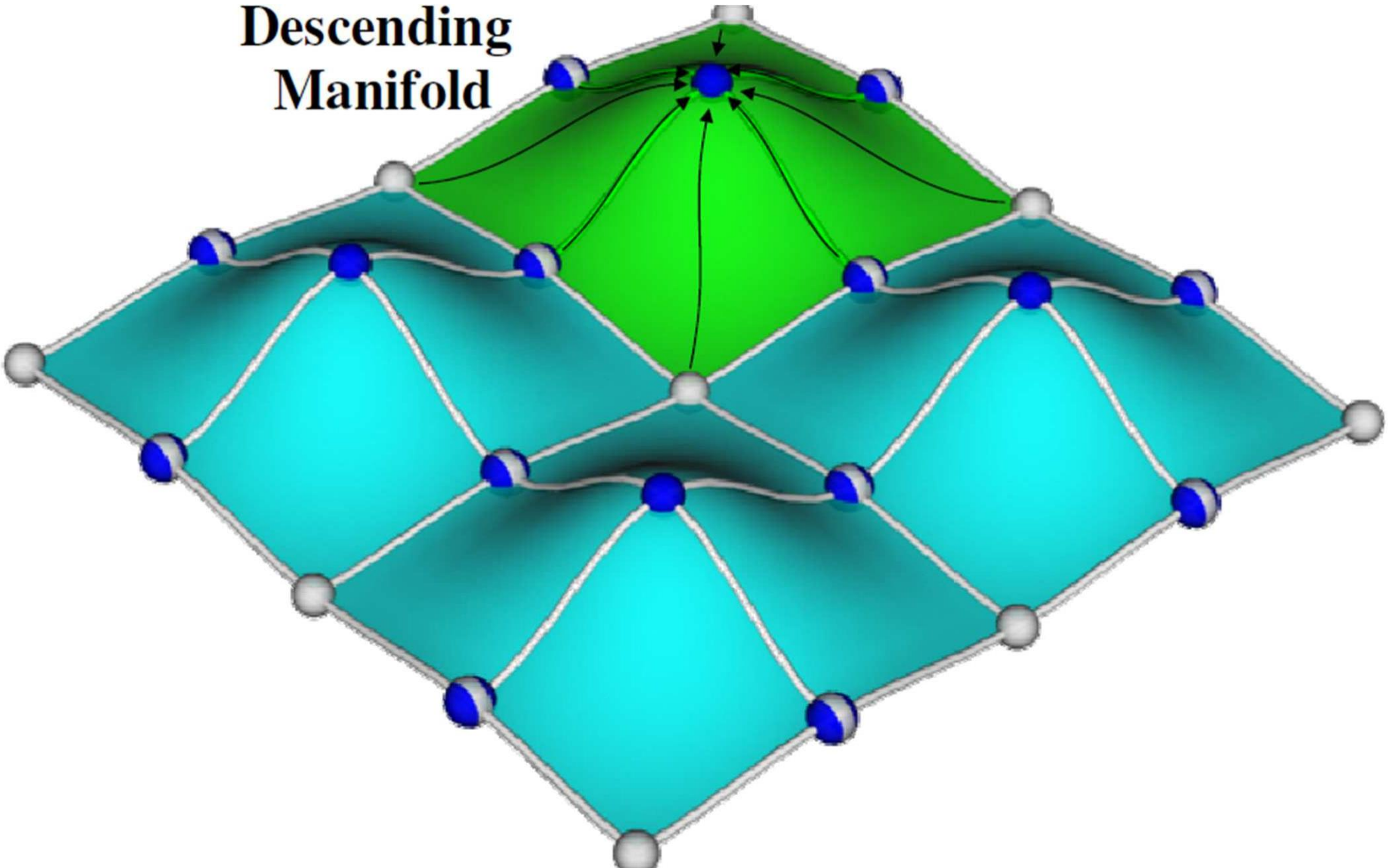
Morse-Smale Complex-2D



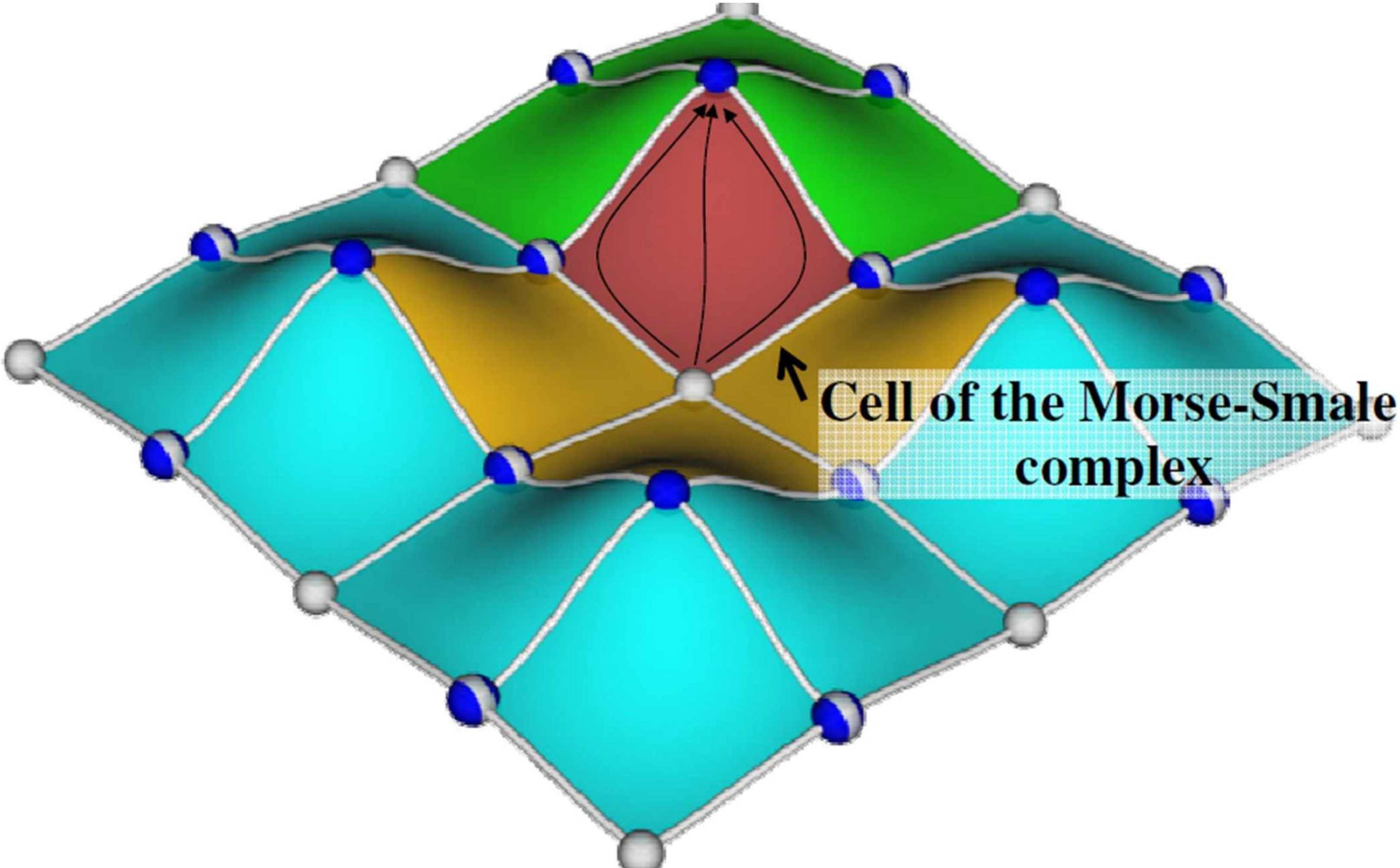
Morse-Smale Complex-2D



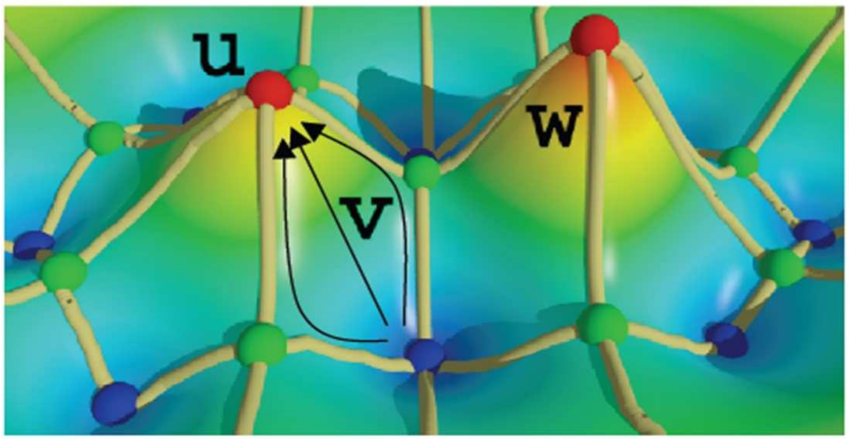
Morse-Smale Complex-2D

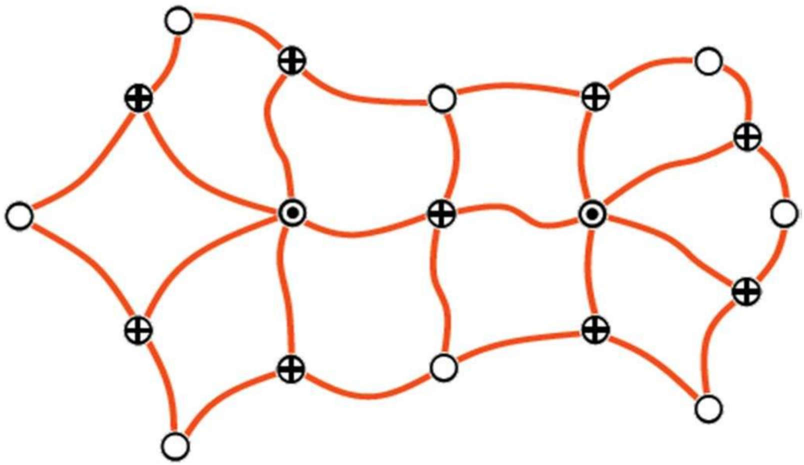


Morse-Smale Complex-2D



Morse-Smale Complex-2D

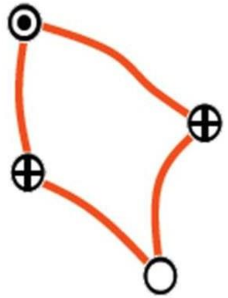




Decomposition into monotonic regions

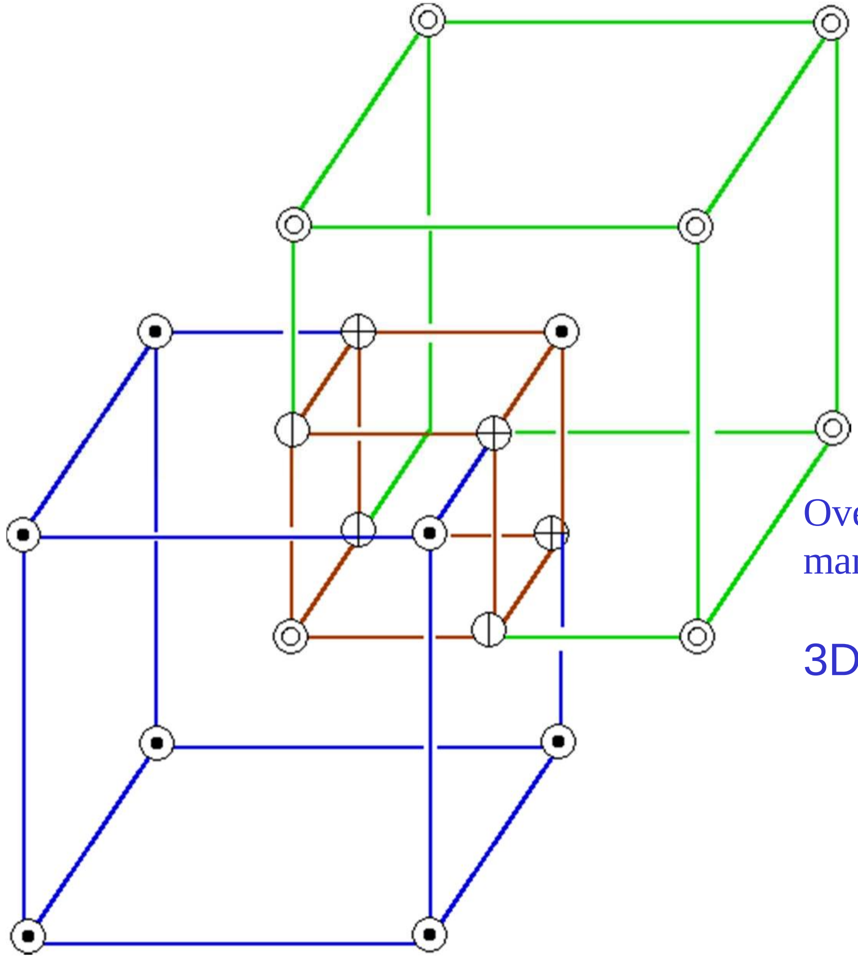
Combinatorial Structure 2D

- Nodes of the MS complex are exactly the critical points of the Morse function
- Saddles have exactly four arcs incident on them



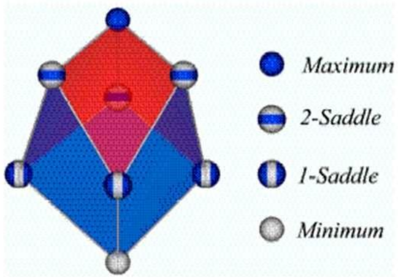
- All regions are quads
- Boundary of a region alternates between saddle- extremum
- 2k minima and maxima

Morse-Smale Complex in 3D



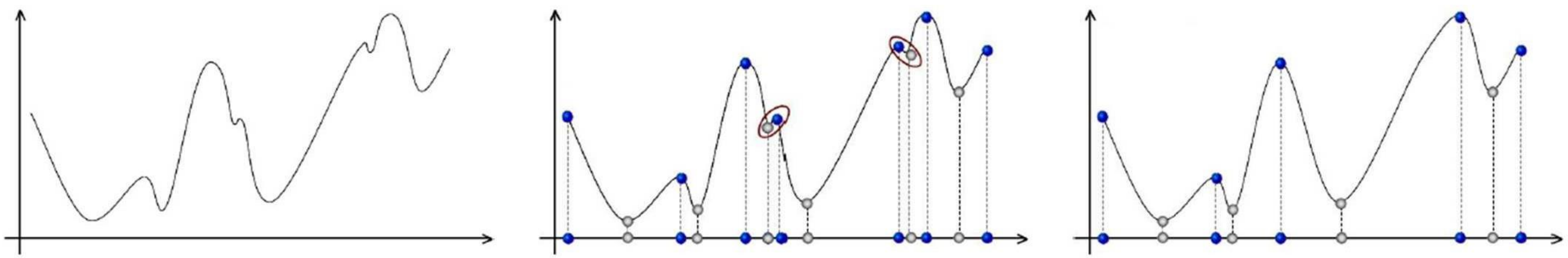
Overlay of Asc and Desc manifolds

3D MS Complex cell



Topological Simplification

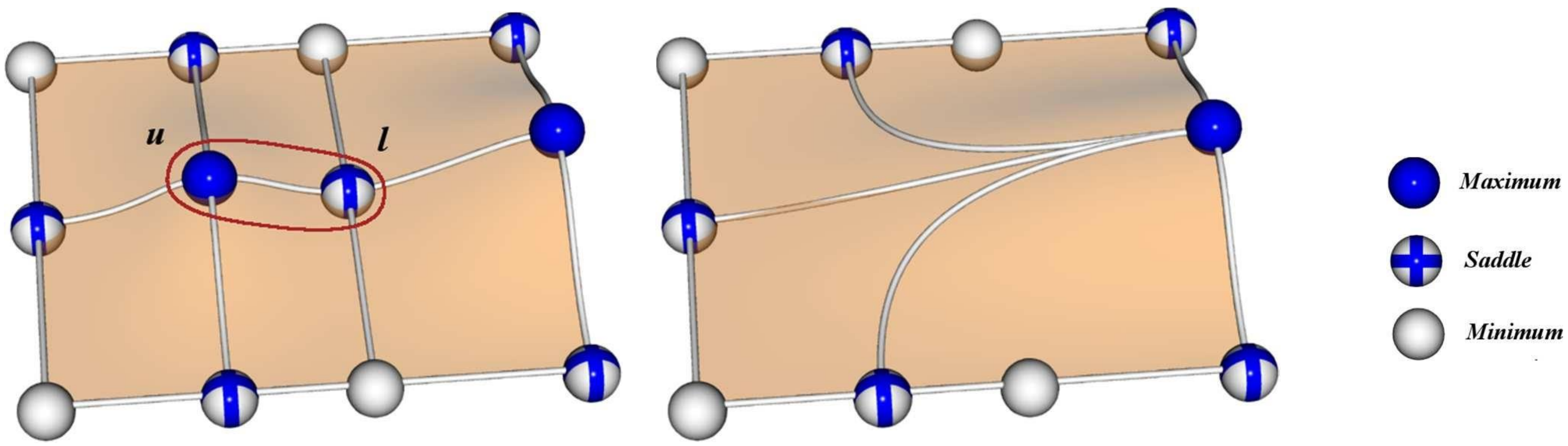
(Persistence) Let p_a be the critical point creating a boundary component B and p_b the critical point destroying B, then the pair (p_a, p_b) is a persistence pair. The difference is function value $|f(p_a) - f(p_b)|$ is called the persistence of the topological feature (p_a, p_b)



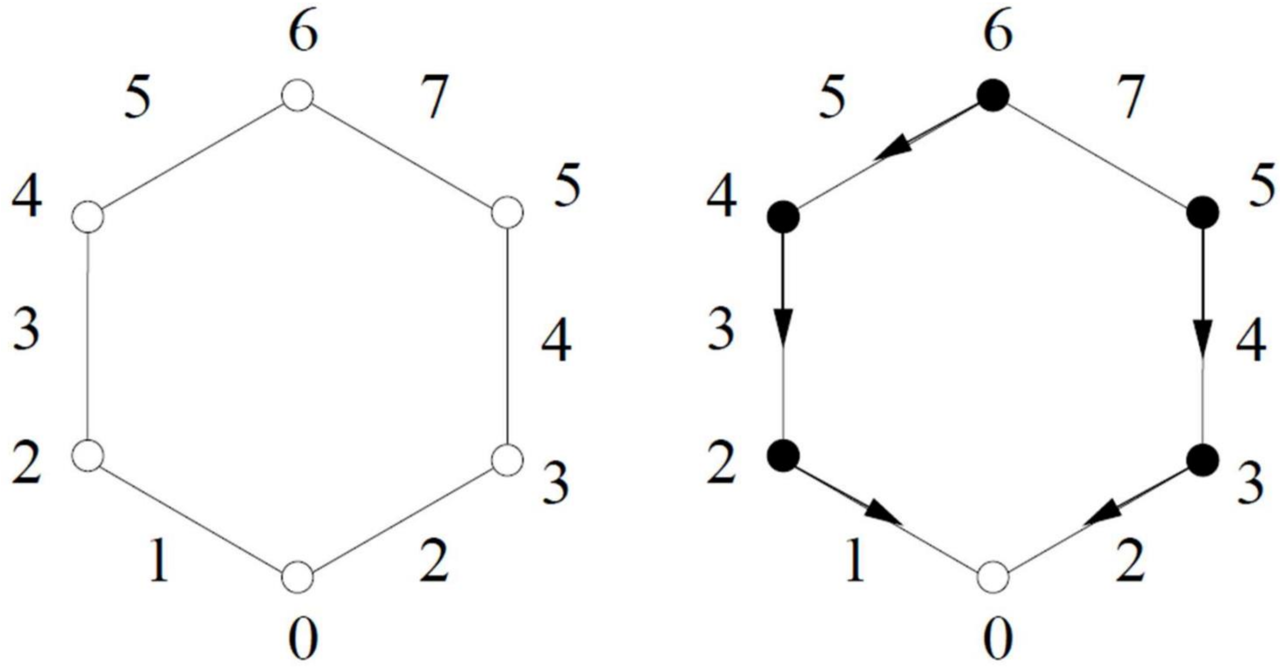
(a) (b) (c)

Topological Simplification

(Persistence) Let p_a be the critical point creating a boundary component B and p_b the critical point destroying B, then the pair (p_a, p_b) is a persistence pair. The difference is function value $|f(p_a) - f(p_b)|$ is called the persistence of the topological feature (p_a, p_b)



Discrete Morse-Smale Complex

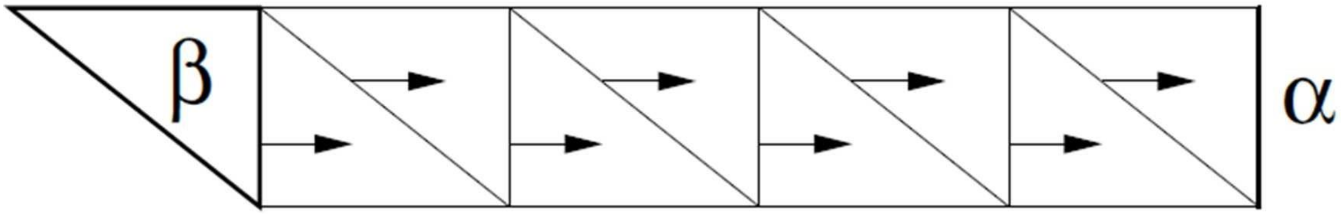


The gradient directions (the arrows) are always pointing from lower-dimensional cells to their neighboring cells that are exactly one-dimension higher.

Discrete Morse-Smale Complex

V-path:

A V-path is the discrete equivalent of a streamline in a smooth vector field. A discrete vector field in which all V-paths are monotonic and do not contain any loops is a discrete gradient field.

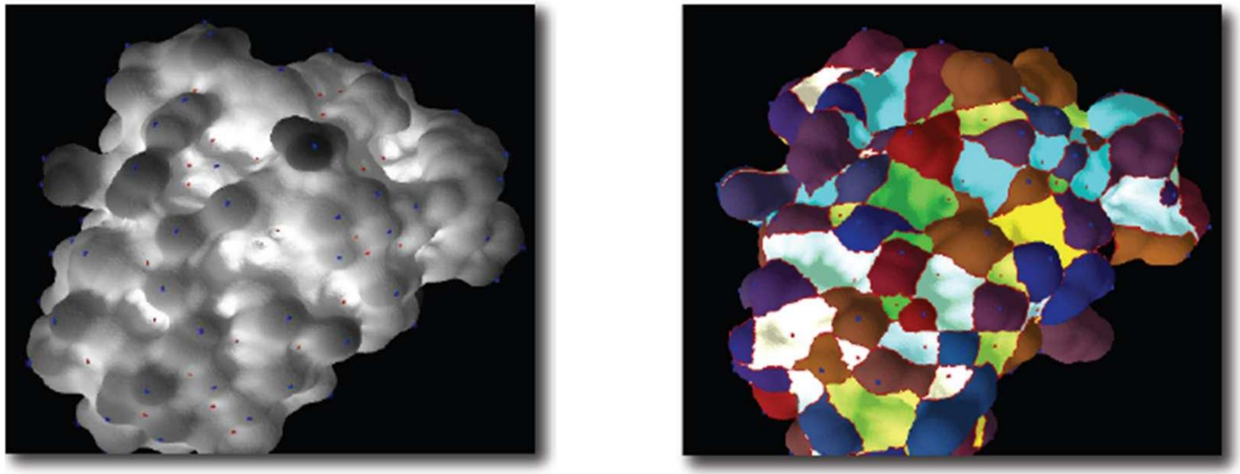


The discrete version allows for more efficient computations in terms of memory efficiency and parallelization.

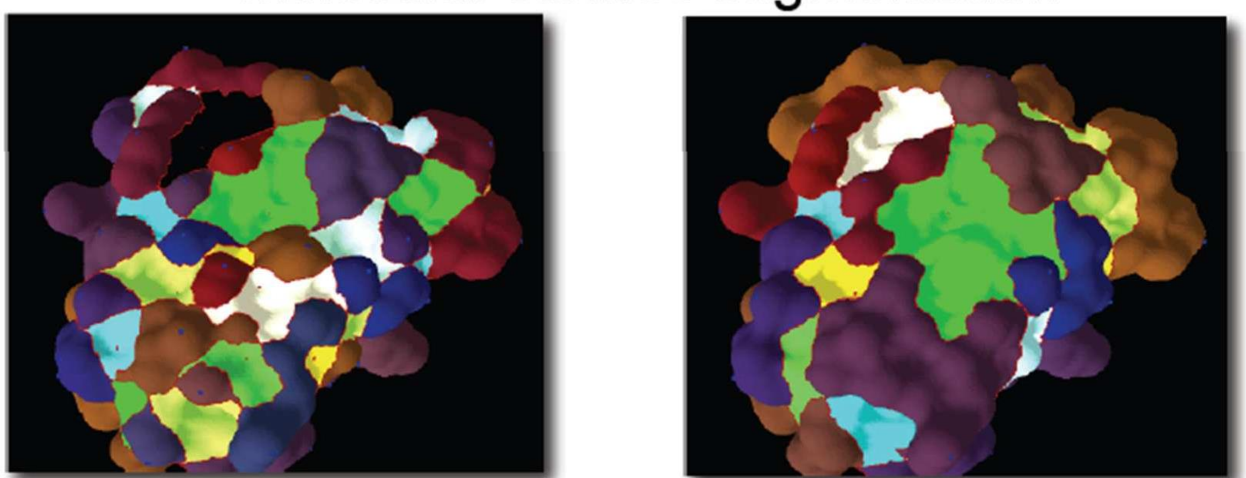
Application: Surface Segmentation

- Why segmentation?
  - Reduce the information overloaded
  - Identify unique features and properties
- There have been many proposed surface segmentation strategies to encode the structure of a function on a surface.
  - Surface networks ideally segment terrain-type data into the cells of the two-dimensional Morse-Smale complex, i.e., into regions of uniform gradient flow behavior. Such a segmentation of a surface would identify the features of a terrain such as peaks, saddles, dips, and the lines connecting them.
  - Image processing: watershed / distance field transform

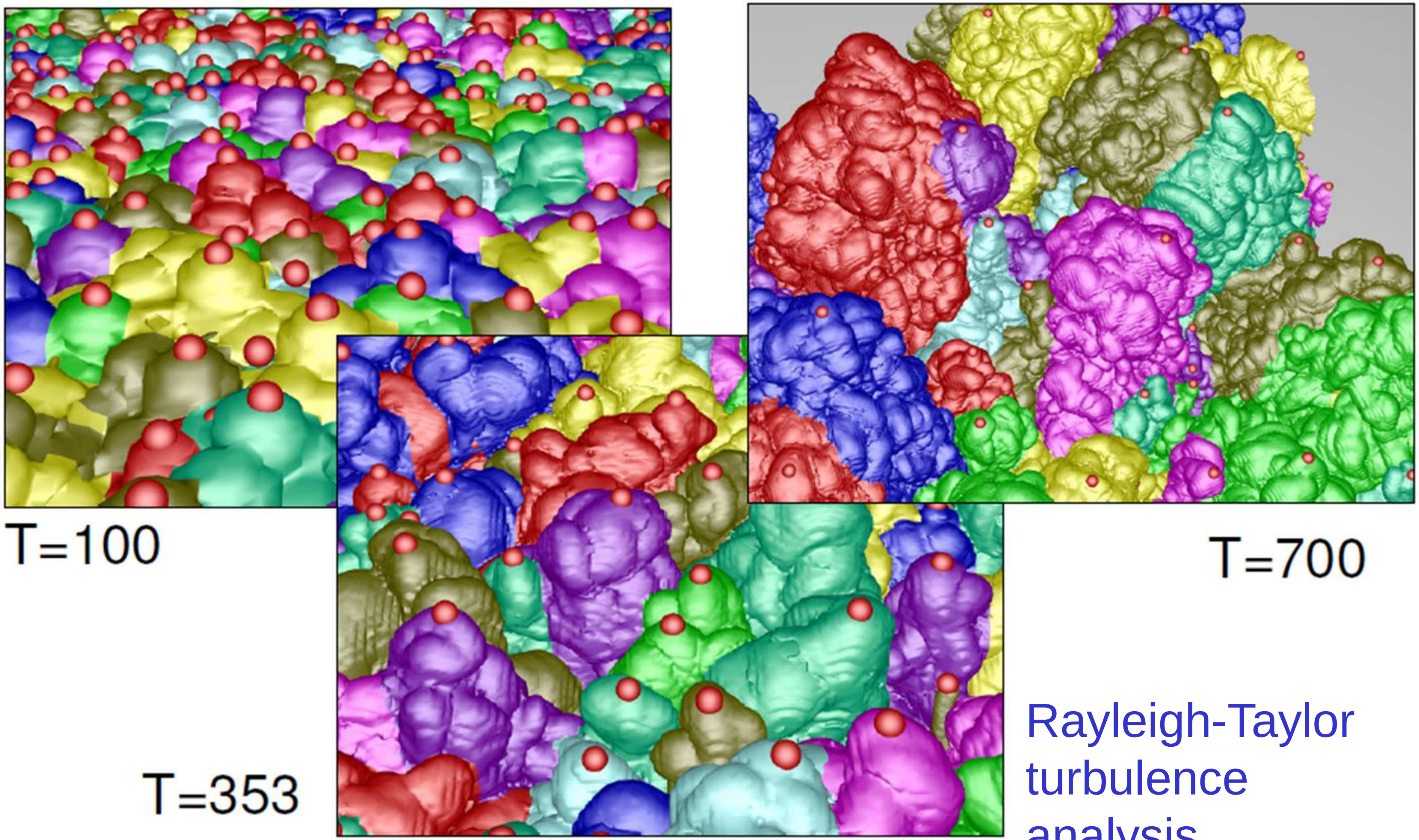
Applications



Molecular surface segmentation

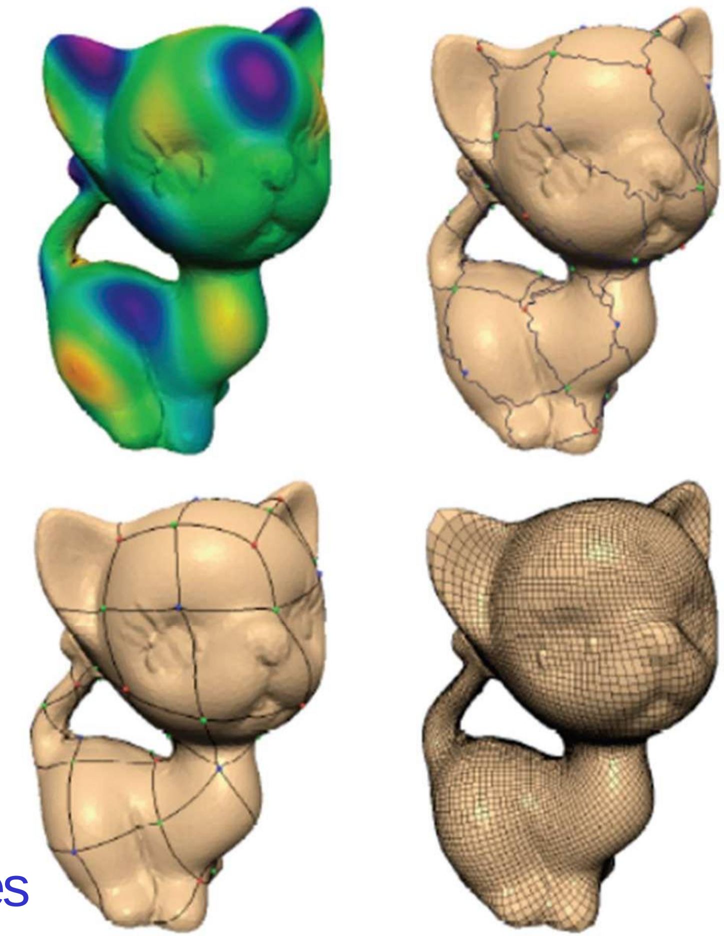


Applications



Rayleigh-Taylor turbulence analysis

Applications



Quadrangulation of surfaces