Surface Computing: Solve Partial Differential Equations on Surfaces using the Closest Point Method

Yujia Chen,
joint work with Colin Macdonald (supervisor),
Ingrid von Glehn, and Tom März

Mathematical Institute, University of Oxford

Oriel CBL course, Oxford, Aug 2014

Outline of Lectures 1 & 2

- Lecture 1
  - Applications of Surface PDEs.
  - Surface differential operators and examples of surface PDEs.
  - Introduction to Closest Point Method (Ruuth–Merriman approach).
- Lecture 2
  - Matrix formulation of spatial discretizations.
  - Octave/MATLAB implementations: diffusion equation on a unit circle/sphere, reaction-diffusion equation on a unit sphere (if time permits).

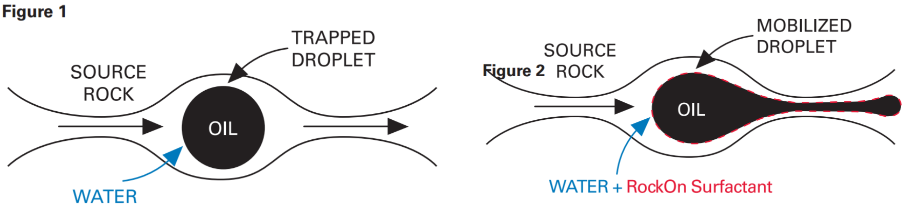
Applications where surface PDEs arise

Turing pattern formation on animals[1]

[1] http://www.simonsfoundation.org/quanta/20130325-biologists-home-in-on-turing-patterns/

Applications where surface PDEs arise

Surfactant diffusing and transported along the interface[2]



[2]http://www.halliburton.com/public/multichem/contents/Brochures/web/H010276-RockOn.pdf

Applications where surface PDEs arise

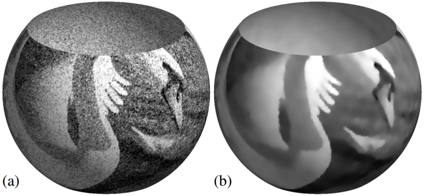
Forest fire front propagation on mountains[3]

See also level-set simulation on Colin's website:
https://people.maths.ox.ac.uk/macdonald/closestpoint/

[3] http://www.cerfacs.fr/globc/links/presentation/Sieste_delmotte_2011.pdf

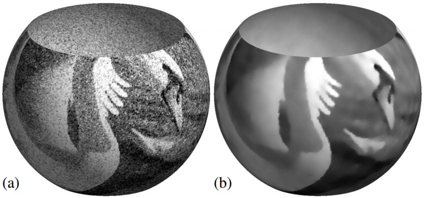
Applications where surface PDEs arise

Image denoising
(Biddle,von Glehn,
Macdonald, März, 2013)

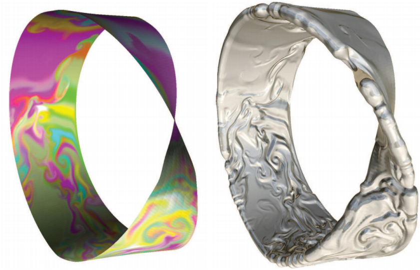


Applications where surface PDEs arise

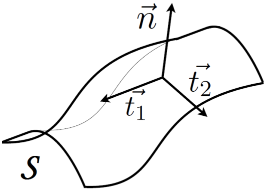
Image denoising
(Biddle,von Glehn,
Macdonald, März, 2013)



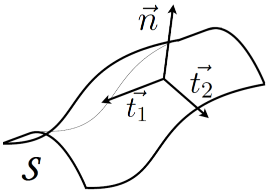
Fluid effects
(Auer, Macdonald, Treib,
Schneider, Westermann 2012)



Surface Differential Operators



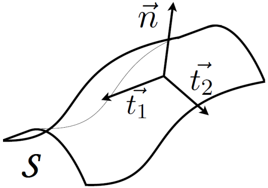
Surface Differential Operators



For a scalar function u, there exists a coordinate system (n, t_1, t_2) so that

$$\nabla u = \frac{\partial u}{\partial n}\vec{n} + \frac{\partial u}{\partial t_1}\vec{t_1} + \frac{\partial u}{\partial t_2}\vec{t_2} \quad (1)$$

Surface Differential Operators

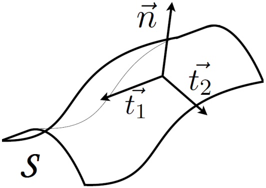


For a scalar function u, there exists a coordinate system (n, t_1, t_2) so that

$$\nabla u = \frac{\partial u}{\partial n}\vec{n} + \frac{\partial u}{\partial t_1}\vec{t_1} + \frac{\partial u}{\partial t_2}\vec{t_2} \quad (1)$$

$$\nabla_{\mathcal{S}} u = \frac{\partial u}{\partial t_1}\vec{t_1} + \frac{\partial u}{\partial t_2}\vec{t_2}$$

Surface Differential Operators



For a vector function $\vec{v} = v_1\vec{t_1} + v_2\vec{t_2} + v_3\vec{n}$:

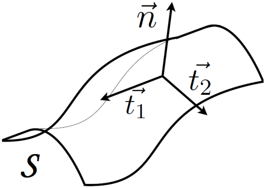
$$\nabla_{\mathcal{S}} \cdot \vec{v} = \frac{\partial v_1}{\partial t_1} + \frac{\partial v_2}{\partial t_2}$$

For a scalar function u, there exists a coordinate system (n, t_1, t_2) so that

$$\nabla u = \frac{\partial u}{\partial n}\vec{n} + \frac{\partial u}{\partial t_1}\vec{t_1} + \frac{\partial u}{\partial t_2}\vec{t_2} \qquad (1)$$

$$\nabla_{\mathcal{S}} u = \frac{\partial u}{\partial t_1}\vec{t_1} + \frac{\partial u}{\partial t_2}\vec{t_2}$$

Surface Differential Operators



For a scalar function u, there exists a coordinate system (n, t_1, t_2) so that

$$\nabla u = \frac{\partial u}{\partial n}\vec{n} + \frac{\partial u}{\partial t_1}\vec{t_1} + \frac{\partial u}{\partial t_2}\vec{t_2} \qquad (1)$$

$$\nabla_{\mathcal{S}} u = \frac{\partial u}{\partial t_1}\vec{t_1} + \frac{\partial u}{\partial t_2}\vec{t_2}$$

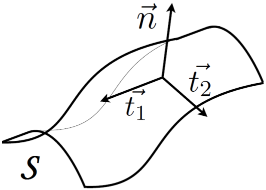
For a vector function $\vec{v} = v_1\vec{t_1} + v_2\vec{t_2} + v_3\vec{n}$:

$$\nabla_{\mathcal{S}} \cdot \vec{v} = \frac{\partial v_1}{\partial t_1} + \frac{\partial v_2}{\partial t_2}$$

Laplace–Beltrami Operator:

$$\Delta_{\mathcal{S}} u = \nabla_{\mathcal{S}} \cdot \nabla_{\mathcal{S}} u = \frac{\partial^2 u}{\partial t_1^2} + \frac{\partial^2 u}{\partial t_2^2}$$

Surface Differential Operators



For a scalar function u, there exists a coordinate system (n, t_1, t_2) so that

$$\nabla u = \frac{\partial u}{\partial n}\vec{n} + \frac{\partial u}{\partial t_1}\vec{t_1} + \frac{\partial u}{\partial t_2}\vec{t_2} \qquad (1)$$

$$\nabla_{\mathcal{S}} u = \frac{\partial u}{\partial t_1}\vec{t_1} + \frac{\partial u}{\partial t_2}\vec{t_2}$$

For a vector function $\vec{v} = v_1\vec{t_1} + v_2\vec{t_2} + v_3\vec{n}$:

$$\nabla_{\mathcal{S}} \cdot \vec{v} = \frac{\partial v_1}{\partial t_1} + \frac{\partial v_2}{\partial t_2}$$

Laplace–Beltrami Operator:

$$\Delta_{\mathcal{S}} u = \nabla_{\mathcal{S}} \cdot \nabla_{\mathcal{S}} u = \frac{\partial^2 u}{\partial t_1^2} + \frac{\partial^2 u}{\partial t_2^2}$$

Notice: equation (1) only holds for a special coordinate system (n, t_1, t_2) (see Appendix A). Nevertheless, the spirit is the same for any coordinate system: Surface derivatives are just Cartesian derivatives getting rid of normal components.

Examples of surface PDEs

Turing pattern:
reaction-diffusion equation

$$u = \nu_u \Delta_{\mathcal{S}} u + f(u, v)$$
$$v = \nu_v \Delta_{\mathcal{S}} v + g(u, v)$$

Surfactant:
diffusion-advection equation

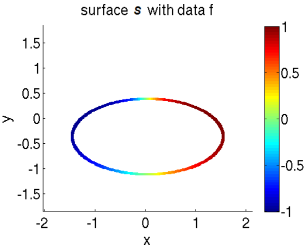
$$\frac{\partial c}{\partial t} + \mathbf{v} \cdot \nabla c + c \nabla_{\mathcal{S}} \cdot \mathbf{v} - \Delta_{\mathcal{S}} c = 0$$

Front propagation:
level set equation

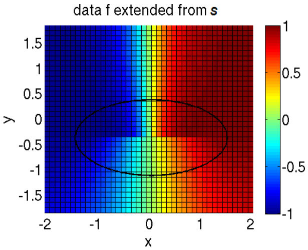
$$\frac{\partial \phi}{\partial t} + \mathbf{v} \cdot \nabla_{\mathcal{S}} \phi = 0$$

Image de-noising:
anisotropic diffusion

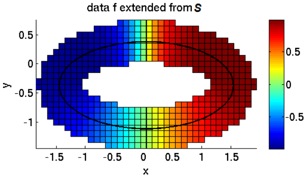
$$\frac{\partial u}{\partial t} = \nabla_{\mathcal{S}} \cdot (g(\|\nabla_{\mathcal{S}} u\|) \nabla_{\mathcal{S}} u)$$



A surface function.



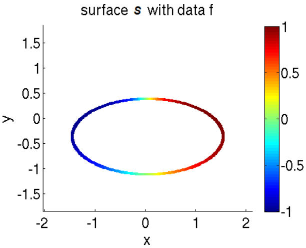
The function extended to $\mathbb{R}^2$.



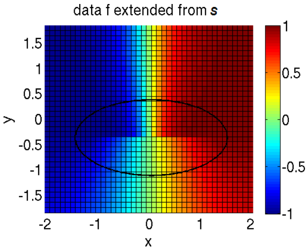
The function extended to a narrow band.

- Extend the surface PDE to an equation in the embedding space so that the solution of the embedding equation agree with the solution of the surface PDE.

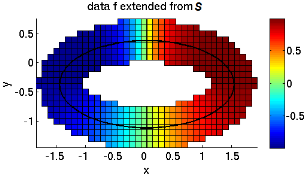
Embedding methods for solving surface PDEs



A surface function.



The function extended to $\mathbb{R}^2$.

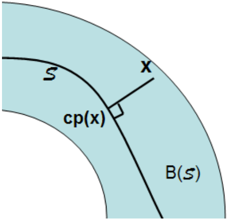


The function extended to a narrow band.

- Extend the surface PDE to an equation in the embedding space so that the solution of the embedding equation agree with the solution of the surface PDE.
- If we do not extend the surface quantity in a special way, then the Cartesian PDE would be complicated (see Appendix B). We will extend the surface function so that the embedding equation is simple.

Closest Point Extension and Closest Point Principle

$\mathrm{cp} : B(\mathcal{S}) \to \mathcal{S}$, and $\mathrm{cp}(\mathbf{x})$ is a point on the surface that is closest to $\mathbf{x}$.



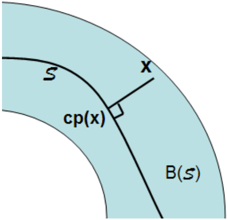
$$u(\mathbf{y}),\ \mathbf{y} \in \mathcal{S} \quad \xrightarrow{extend} \quad v(\mathbf{x}),\ \mathbf{x} \in B(\mathcal{S})$$

$$v(\mathbf{x}) = u(\mathrm{cp}(\mathbf{x})), \quad \mathbf{x} \in B(\mathcal{S}),$$

v is called a closest point extension of u.
v is constant along the normals to $\mathcal{S}$.
Denote $v = Eu$.

Closest Point Extension and Closest Point Principle

cp : $B(\mathcal{S}) \to \mathcal{S}$, and cp$(\mathbf{x})$ is a point on the surface that is closest to $\mathbf{x}$.



$$u(\mathbf{y}),\ \mathbf{y} \in \mathcal{S} \quad \overset{extend}{\longrightarrow} \quad v(\mathbf{x}),\ \mathbf{x} \in B(\mathcal{S})$$

$$v(\mathbf{x}) = u(\mathrm{cp}(\mathbf{x})), \quad \mathbf{x} \in B(\mathcal{S}),$$

v is called a closest point extension of u.
v is constant along the normals to $\mathcal{S}$.
Denote $v = Eu$.

Gradient Principle

$\nabla[(Eu)(y)] = \nabla_S u(y), \quad y \in S.$

Divergence Principle

$\nabla \cdot [(E\mathbf{A})(y)] = \nabla_S \cdot \mathbf{A}(y), \quad y \in S.$

Laplacian Principle

$\Delta[(Eu)(y)] = \Delta_{\mathcal{S}} u(y), \quad y \in \mathcal{S}.$

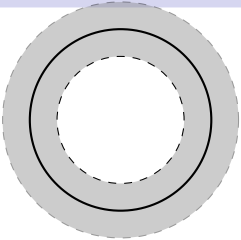
(Ruuth & Merriman 2008, März & Macdonald 2012)

Example: diffusion equation on a unit circle

Consider the diffusion equation on a unit circle:

$$\begin{cases} u_t(t, \mathbf{x}) = \Delta_{\mathcal{S}}(t, \mathbf{x}), \\ u(0, \mathbf{x}) = u_0(\mathbf{x}). \end{cases}$$

where $\mathbf{x} \in \mathcal{S}$, and $t \in [0, T]$.

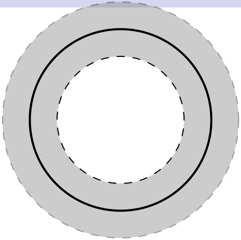


Example: diffusion equation on a unit circle

Consider the diffusion equation on a unit circle:

$$\begin{cases} u_t(t,\mathbf{x}) & = & \Delta_{\mathcal{S}}(t,\mathbf{x}), \\ u(0,\mathbf{x}) & = & u_0(\mathbf{x}). \end{cases}$$

where $\mathbf{x} \in \mathcal{S}$, and $t \in [0, T]$.



We use forward Euler method for time-stepping, and uniformly divide $[0, T]$ into $t^0 = 0, t^1 = \tau, \cdots, t^N = N\tau = T$. Ideally, we would like:

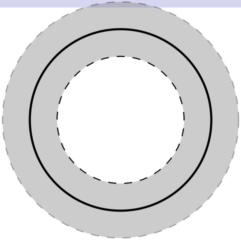
$$u(t^{n+1},\mathbf{x}) = u(t^n,\mathbf{x}) + \tau \Delta_{\mathcal{S}} u(t^n,\mathbf{x}), \quad n = 0, \cdots, N-1 \qquad (2)$$

Example: diffusion equation on a unit circle

Consider the diffusion equation on a unit circle:

$$\begin{cases} u_t(t, \mathbf{x}) &= \Delta_{\mathcal{S}}(t, \mathbf{x}), \\ u(0, \mathbf{x}) &= u_0(\mathbf{x}). \end{cases}$$

where $\mathbf{x} \in \mathcal{S}$, and $t \in [0, T]$.



We use forward Euler method for time-stepping, and uniformly divide $[0, T]$ into $t^0 = 0, t^1 = \tau, \cdots, t^N = N\tau = T$. Ideally, we would like:

$$u(t^{n+1}, \mathbf{x}) = u(t^n, \mathbf{x}) + \tau \Delta_{\mathcal{S}} u(t^n, \mathbf{x}), \quad n = 0, \cdots, N-1 \tag{2}$$

Initially, let $v(0, \mathbf{y}) = u_0(\mathrm{cp}(\mathbf{y}))$, for $\mathbf{y} \in B(\mathcal{S})$ (i.e., $v(0, \mathbf{y}) = Eu_0$). Then $\Delta v(0, \mathbf{x}) = \Delta_{\mathcal{S}} u_0(\mathrm{cp}(\mathbf{x}))$ for $\mathbf{x} \in \mathcal{S}$.

If we define: $$\tilde{v}(\tau, \mathbf{y}) = v(0, \mathbf{y}) + \tau \Delta v(0, \mathbf{y}), \quad \mathbf{y} \in B(\mathcal{S}), \tag{3}$$

then $\tilde{v}(\tau, \mathbf{x}) = u(\tau, \mathbf{x})$ for $\mathbf{x} \in \mathcal{S}$, where $u(\tau, \mathbf{x}) = u(0, \mathbf{x}) + \tau \Delta_{\mathcal{S}} u(0, \mathbf{x})$ is defined by (2). Question: is $\tilde{v}(\tau, \mathbf{y})$ constant along the normals?

Example:diffusion equation on a unit circle

No, $\tilde{v}(\tau, \mathbf{y})$ is no longer constant along the normals.
In order to carry on evolvement with the Cartesian Laplacian Δ, we re-enforce v to be constant along the normals to $\mathcal{S}$ by:

$$v(\tau, \mathbf{y}) = \tilde{v}(\tau, \mathrm{cp}(\mathbf{y})), \quad \mathbf{y} \in B(\mathcal{S}).$$

Example:diffusion equation on a unit circle

No, $\tilde{v}(\tau, \mathbf{y})$ is no longer constant along the normals.
In order to carry on evolvement with the Cartesian Laplacian Δ, we re-enforce v to be constant along the normals to $\mathcal{S}$ by:

$$v(\tau, \mathbf{y}) = \tilde{v}(\tau, \mathrm{cp}(\mathbf{y})), \quad \mathbf{y} \in B(\mathcal{S}).$$

Suppose we've computed up to $v(t^n, \mathbf{y})$ s.t. $v(t^n, \mathbf{y})$ is constant along the normals to $\mathcal{S}$, and $v(t^n, \mathbf{x}) = u(t^n, \mathbf{x})$ for any $\mathbf{x} \in \mathcal{S}$. Then we can compute $v(t^{n+1}, \mathbf{y})$ by alternating the following two steps:

$$\begin{aligned} \tilde{v}(t^{n+1}, \mathbf{y}) &= v(t^n, \mathbf{y}) + \tau \Delta v(t^n, \mathbf{y}) \\ v(t^{n+1}, \mathbf{y}) &= \tilde{v}(t^{n+1}, \mathrm{cp}(\mathbf{y})) \end{aligned}$$

Obviously $v(t^{n+1}, \mathbf{y})$ is still constant along the normals to $\mathcal{S}$ and $v(t^{n+1}, \mathbf{x}) = u(t^{n+1}, \mathbf{x})$ for any $\mathbf{x} \in \mathcal{S}$.

Closest Point Method: Ruuth–Merriman approach 2008

To solve the surface diffusion equation $u_t = \Delta_{\mathcal{S}} u$ on $\mathcal{S}$, extend the surface function u on $\mathcal{S}$ to function v in $B(\mathcal{S})$: let $v(0, \mathbf{y}) = u_0(\mathrm{cp}(\mathbf{y}))$, $\mathbf{y} \in B(\mathcal{S})$.
Then for $n = 0, \cdots, N-1$, do:
$\tilde{v}(t^{n+1}, \mathbf{y}) = v(t^n, \mathbf{y}) + \tau\, \Delta v(t^n, \mathbf{y})$;
$v(t^{n+1}, \mathbf{y}) = \tilde{v}(t^{n+1}, \mathrm{cp}(\mathbf{y}))$.

Closest Point Method: Ruuth–Merriman approach 2008

To solve the surface diffusion equation $u_t = \Delta_{\mathcal{S}} u$ on $\mathcal{S}$, extend the surface function u on $\mathcal{S}$ to function v in $B(\mathcal{S})$: let $v(0, \mathbf{y}) = u_0(\mathrm{cp}(\mathbf{y}))$, $\mathbf{y} \in B(\mathcal{S})$.
Then for $n = 0, \cdots, N-1$, do:
$\tilde{v}(t^{n+1}, \mathbf{y}) = v(t^n, \mathbf{y}) + \tau \, \Delta v(t^n, \mathbf{y})$;
$v(t^{n+1}, \mathbf{y}) = \tilde{v}(t^{n+1}, \mathrm{cp}(\mathbf{y}))$.

Ruuth–Merriman Closest Point Iteration:
For any closed smooth surface $\mathcal{S}$ and corresponding tubular neighborhood $B(\mathcal{S})$, solve a general surface PDE $u_t = f(u, \nabla_{\mathcal{S}} u, \nabla_{\mathcal{S}} \cdot u, \Delta_{\mathcal{S}} u)$:
(1) Let $v(0, \mathbf{y}) = u_0(\mathrm{cp}(\mathbf{y}))$ for $\mathbf{y} \in B(\mathcal{S})$.
For $n = 0, \cdots, N-1$, do:
(2) $\tilde{v}(t^{n+1}, \mathbf{y}) = v(t^n, \mathbf{y}) + \tau \, f(v(t^n, \mathbf{y}), \nabla v(t^n, \mathbf{y}), \nabla \cdot v(t^n, \mathbf{y}), \Delta v(t^n, \mathbf{y}))$;
(3) $v(t^{n+1}, \mathbf{y}) = \tilde{v}(t^{n+1}, \mathrm{cp}(\mathbf{y}))$.

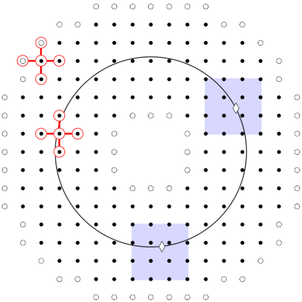
Full discretization: back to diffusion on circle example

$$\tilde{v}(t^{n+1}, \mathbf{y}) = v(t^n, \mathbf{y}) + \tau \Delta v(t^n, \mathbf{y})$$
$$v(t^{n+1}, \mathbf{y}) = \tilde{v}(t^{n+1}, \mathrm{cp}(\mathbf{y}))$$

Full discretization: back to diffusion on circle example

Macdonald & Ruuth, 2009

$\tilde{v}(t^{n+1}, \mathbf{y}) = v(t^n, \mathbf{y}) + \tau \Delta v(t^n, \mathbf{y})$
$v(t^{n+1}, \mathbf{y}) = \tilde{v}(t^{n+1}, \mathrm{cp}(\mathbf{y}))$
Let $v(t^n, x_j, y_k) \approx v^n_{j,k}$
$\tilde{v}^{n+1}_{j,k} = v^n_{j,k} + \tau \frac{v^n_{j-1,k} + v^n_{j+1,k} + v^n_{j,k-1} + v^n_{j,k+1} - 4v^n_{j,k}}{h^2}$
We hope $v^{n+1}_{j,k} = \tilde{v}^{n+1}(\mathrm{cp}(x_j, y_k))$, but $\mathrm{cp}(x_j, y_k)$ is not a grid point. We approximate $\tilde{v}^{n+1}(\mathrm{cp}(x_j, y_k))$ through Lagrange polynomial interpolation:
$\sum_{(x_l, y_m) \in \mathrm{Interp}(\mathrm{cp}(x_j, y_k))} \omega_{l,m} \tilde{v}^{n+1}_{l,m}$
(More details to derive on board..)



Laplacian stencil $\mathrm{Lap}\big((x_j, y_k)\big)$

Interpolation stencil $\mathrm{Interp}\big(\mathrm{cp}(x_j, y_k)\big)$

How do we choose the computational band?

$$band(E) = \bigcup_{\mathbf{x}\in\mathcal{S}} \{\text{Interp}(\mathbf{x})\}$$

$$band(L) = \bigcup_{\mathbf{x}_i \in band(E)} \{\text{Lap}(\mathbf{x}_i)\}$$

$$band(E) \subset band(L)$$

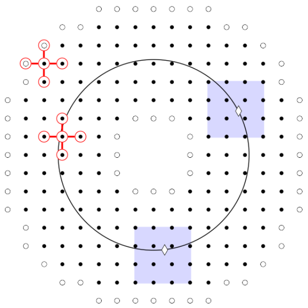
In practice, choose a band B so that $B \supset band(L)$.
An empirical formula (Ruuth–Merriman, 2008) is:

$$B = \{\mathbf{x}_i : \|\mathbf{x}_i - \text{cp}(\mathbf{x}_i)\|_2 \leq C\Delta x\},$$

where

$$C = \sqrt{(d-1)(\frac{p+1}{2})^2 + (order/2 + \frac{p+1}{2})^2}.$$

Macdonald & Ruuth, 2009



Laplacian stencil $\text{Lap}(\mathbf{x}_i)$

Interpolation stencil $\text{Interp}(\mathbf{x})$

For a chosen meshsize h, what time-step τ shall we choose?

For a chosen meshsize h, what time-step τ shall we choose?

$$\tau = O(h^2)$$

For a chosen meshsize h, what time-step τ shall we choose?

$$\tau = O(h^2)$$

One intuition: the continuous heat equation satisfies maximum principle.

$$\max_{\mathbf{x}} u(t_1, \mathbf{x}) \geq \max_{\mathbf{x}} u(t_2, \mathbf{x}), \quad t_1 \leq t_2.$$

Therefore we hope our numerical solution also satisfies this principle. If we forget about the surface, just consider the heat equation in 2D:

$$\begin{aligned} v_{j,k}^{n+1} &= v_{j,k}^n + \frac{\tau}{h^2}(v_{j-1,k}^n + v_{j+1,k}^n + v_{j,k-1}^n + v_{j,k+1}^n - 4v_{j,k}^n) \\ &= (1 - 4\frac{\tau}{h^2})v_{j,k}^n + \frac{\tau}{h^2}v_{j-1,k}^n + \frac{\tau}{h^2}v_{j+1,k}^n + \frac{\tau}{h^2}v_{j,k-1}^n + \frac{\tau}{h^2}v_{j,k+1}^n \end{aligned}$$

Sufficient condition for discrete maximum principle $max_{j,k} v^{n+1} \leq max_{j,k} v^n$ to hold is that all the coefficients are non-negative. We need $1 - 4\frac{\tau}{h^2} \geq 0$. Therefore $\tau \leq \frac{1}{4}h^2$.

For a chosen meshsize h, what time-step τ shall we choose?

$$\tau = O(h^2)$$

One intuition: the continuous heat equation satisfies maximum principle.

$$\max_{\mathbf{x}} u(t_1, \mathbf{x}) \geq \max_{\mathbf{x}} u(t_2, \mathbf{x}), \quad t_1 \leq t_2.$$

Therefore we hope our numerical solution also satisfies this principle. If we forget about the surface, just consider the heat equation in 2D:

$$\begin{aligned} v_{j,k}^{n+1} &= v_{j,k}^n + \frac{\tau}{h^2}(v_{j-1,k}^n + v_{j+1,k}^n + v_{j,k-1}^n + v_{j,k+1}^n - 4v_{j,k}^n) \\ &= (1 - 4\frac{\tau}{h^2})v_{j,k}^n + \frac{\tau}{h^2}v_{j-1,k}^n + \frac{\tau}{h^2}v_{j+1,k}^n + \frac{\tau}{h^2}v_{j,k-1}^n + \frac{\tau}{h^2}v_{j,k+1}^n \end{aligned}$$

Sufficient condition for discrete maximum principle $max_{j,k}v^{n+1} \leq max_{j,k}v^n$ to hold is that all the coefficients are non-negative. We need $1 - 4\frac{\tau}{h^2} \geq 0$. Therefore $\tau \leq \frac{1}{4}h^2$.

Rigorously speaking, discrete maximum principle does not hold in the Ruuth–Merriman algorithm. However for numerical stability, $\tau = O(h^2)$ is required.

Matrix formulation

$$\tilde{v}^{n+1} = v^n + \tau \Delta v^n$$
$$\tilde{v}^{n+1}_{j,k} = v^n_{j,k} + \tau \frac{v^n_{j-1,k} + v^n_{j+1,k} + v^n_{j,k-1} + v^n_{j,k+1} - 4v^n_{j,k}}{h^2}$$
$$\Longleftrightarrow \tilde{\mathbf{v}}^{n+1}_h = \mathbf{v}^n_h + \tau \mathbf{L}_h \mathbf{v}^n_h$$

Matrix formulation

$$\tilde{v}^{n+1} = v^n + \tau \Delta v^n$$
$$\tilde{v}_{j,k}^{n+1} = v_{j,k}^n + \tau \frac{v_{j-1,k}^n + v_{j+1,k}^n + v_{j,k-1}^n + v_{j,k+1}^n - 4v_{j,k}^n}{h^2}$$
$$\iff \tilde{\mathbf{v}}_h^{n+1} = \mathbf{v}_h^n + \tau \mathbf{L}_h \mathbf{v}_h^n$$

$$v^{n+1} = \tilde{v}^{n+1}(\mathrm{cp})$$
$$v_{j,k}^{n+1} = \sum_{(x_l, y_m) \in \mathrm{Interp}(\mathrm{cp}(x_j, y_k))} \omega_{l,m} \tilde{v}_{l,m}^{n+1}$$
$$\iff \mathbf{v}_h^{n+1} = \mathbf{E}_h \tilde{\mathbf{v}}_h^{n+1}$$

(More details to derive on board..)

Matrix formulation

$$\tilde{v}^{n+1} = v^n + \tau \Delta v^n$$
$$\tilde{v}_{j,k}^{n+1} = v_{j,k}^n + \tau \frac{v_{j-1,k}^n + v_{j+1,k}^n + v_{j,k-1}^n + v_{j,k+1}^n - 4v_{j,k}^n}{h^2}$$
$$\Longleftrightarrow \tilde{\mathbf{v}}_h^{n+1} = \mathbf{v}_h^n + \tau \mathbf{L}_h \mathbf{v}_h^n$$

$$v^{n+1} = \tilde{v}^{n+1}(\mathrm{cp})$$
$$v_{j,k}^{n+1} = \sum_{(x_l, y_m) \in \mathrm{Interp}(\mathrm{cp}(x_j, y_k))} \omega_{l,m} \tilde{v}_{l,m}^{n+1}$$
$$\Longleftrightarrow \mathbf{v}_h^{n+1} = \mathbf{E}_h \tilde{\mathbf{v}}_h^{n+1}$$

(More details to derive on board..)

Benefits of matrix formulation:

- Vectorized implementations.
- Eigenvalue analysis for numerical stability.

Appendix A: Orthogonal curvilinear coordinate systems[4]

Orthogonal coordinate system (u, v, w), unit vectors $(\hat{\mathbf{u}}, \hat{\mathbf{v}}, \hat{\mathbf{w}})$.
Suppose the coordinates change by infinitesimal amount
$(u, v, w) \to (u + du, v + dv, w + dw)$,
then the length vector $\mathbf{s}$ changes by

$$d\mathbf{s} = \frac{\partial \mathbf{s}}{\partial u} du + \frac{\partial \mathbf{s}}{\partial v} dv + \frac{\partial \mathbf{s}}{\partial w} dw = f du \hat{\mathbf{u}} + g dv \hat{\mathbf{v}} + h dw \hat{\mathbf{w}},$$

where $f = \|\frac{\partial s}{\partial u}\|$, $g = \|\frac{\partial s}{\partial v}\|$, $h = \|\frac{\partial s}{\partial w}\|$.

Cartesian (x, y, z), $(\hat{\mathbf{x}}, \hat{\mathbf{y}}, \hat{\mathbf{z}})$

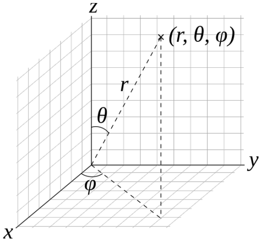
$$d\mathbf{s} = dx\hat{\mathbf{x}} + dy\hat{\mathbf{y}} + dz\hat{\mathbf{z}},$$

$$(f = 1, g = 1, h = 1)$$

Spherical (r, θ, ϕ), $(\hat{\mathbf{r}}, \hat{\boldsymbol{\theta}}, \hat{\boldsymbol{\phi}})$

$$d\mathbf{s} = dr\hat{\mathbf{r}} + rd\theta\hat{\boldsymbol{\theta}} + r\sin(\theta)d\phi\hat{\boldsymbol{\phi}},$$

$$(f = 1, g = r, h = r\sin(\theta))$$



[4] http://www.theory.caltech.edu/classes/ph125a/VCCC.pdf

Calculus in orthogonal curvilinear coordinate systems

Gradient: For a scalar function $F(u, v, w)$, we would like to calculate

$$\nabla F = (\nabla F)_u \hat{\mathbf{u}} + (\nabla F)_v \hat{\mathbf{v}} + (\nabla F)_w \hat{\mathbf{w}}.$$

If the coordinates change by $(u, v, w) \to (u + du, v + dv, w + dw)$ then scalar function F changes by

$$dF = \frac{\partial F}{\partial u} du + \frac{\partial F}{\partial v} dv + \frac{\partial F}{\partial w} dw \tag{4}$$

Also we have

$$dF = \nabla F \cdot d\mathbf{s} = (\nabla F)_u f du + (\nabla F)_v f dv + (\nabla F)_w f dw \tag{5}$$

Comparing (4) and (5) we have $(\nabla F)_u = \frac{1}{f} \frac{\partial F}{\partial u}$, $(\nabla F)_v = \frac{1}{g} \frac{\partial F}{\partial v}$, and $(\nabla F)_w = \frac{1}{h} \frac{\partial F}{\partial w}$. Therefore

$$\nabla F = \frac{1}{f} \frac{\partial F}{\partial u} \hat{\mathbf{u}} + \frac{1}{g} \frac{\partial F}{\partial v} \hat{\mathbf{v}} + \frac{1}{h} \frac{\partial F}{\partial w} \hat{\mathbf{w}}.$$

Calculus in orthogonal curvilinear coordinate systems

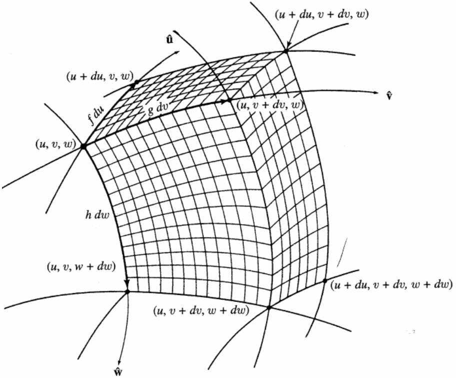
Divergence: Vector function
$\mathbf{A}(u,v,w) = \mathbf{A}_u\hat{\mathbf{u}} + \mathbf{A}_v\hat{\mathbf{v}} + \mathbf{A}_w\hat{\mathbf{w}}$.

$$\int_{\mathbf{S}} \mathbf{A}\cdot d\mathbf{S} = (\nabla\cdot\mathbf{A})dV$$

$$\int_{\mathbf{S}(\text{front}+\text{back})} \mathbf{A}\cdot d\mathbf{S}$$
$$= -(gh\mathbf{A}_u)dvdw|_{(u,v,w)} + (gh\mathbf{A}_u)dvdw|_{(u+du,v,w)}$$
$$= \frac{\partial}{\partial u}(gh\mathbf{A}_u)dudvdw$$

Left+Right$= \frac{\partial}{\partial v}(fh\mathbf{A}_v)dudvdw$

Top+Bottom$= \frac{\partial}{\partial w}(fg\mathbf{A}_w)dudvdw$



Infinitesimal volume

Using $dV = (fgh)dudvdw$, we have:

$$\nabla\cdot\mathbf{A} = \frac{1}{fgh}\Big(\frac{\partial}{\partial u}(gh\mathbf{A}_u) + \frac{\partial}{\partial v}(fh\mathbf{A}_v) + \frac{\partial}{\partial w}(fg\mathbf{A}_w)\Big)$$

Surface differential operators

Gradient: $\nabla F = \frac{1}{f}\frac{\partial F}{\partial u}\hat{\mathbf{u}} + \frac{1}{g}\frac{\partial F}{\partial v}\hat{\mathbf{v}} + \frac{1}{h}\frac{\partial F}{\partial w}\hat{\mathbf{w}}$.
Divergence: $\nabla \cdot \mathbf{A} = \frac{1}{fgh}\Big(\frac{\partial}{\partial u}(gh\mathbf{A}_u) + \frac{\partial}{\partial v}(fh\mathbf{A}_v) + \frac{\partial}{\partial w}(fg\mathbf{A}_w)\Big)$.
Laplacian: $\Delta F = \nabla \cdot \nabla F = \frac{1}{fgh}\Big(\frac{\partial}{\partial u}(\frac{gh}{f}\frac{\partial F}{\partial u}) + \frac{\partial}{\partial v}(\frac{fh}{g}\frac{\partial F}{\partial v}) + \frac{\partial}{\partial w}(\frac{fg}{h}\frac{\partial F}{\partial w})\Big)$.
Exercise: gradient, divergence, and Laplacian in spherical coordinates.

Surface differential operators

Gradient:$\nabla F = \frac{1}{f}\frac{\partial F}{\partial u}\hat{\mathbf{u}} + \frac{1}{g}\frac{\partial F}{\partial v}\hat{\mathbf{v}} + \frac{1}{h}\frac{\partial F}{\partial w}\hat{\mathbf{w}}$.
Divergence: $\nabla \cdot \mathbf{A} = \frac{1}{fgh}\Big(\frac{\partial}{\partial u}(gh\mathbf{A}_u) + \frac{\partial}{\partial v}(fh\mathbf{A}_v) + \frac{\partial}{\partial w}(fg\mathbf{A}_w)\Big)$.
Laplacian: $\Delta F = \nabla \cdot \nabla F = \frac{1}{fgh}\Big(\frac{\partial}{\partial u}(\frac{gh}{f}\frac{\partial F}{\partial u}) + \frac{\partial}{\partial v}(\frac{fh}{g}\frac{\partial F}{\partial v}) + \frac{\partial}{\partial w}(\frac{fg}{h}\frac{\partial F}{\partial w})\Big)$.
Exercise: gradient, divergence, and Laplacian in spherical coordinates.

For a smooth surface, consider (n, t_1, t_2), with unit vectors $(\hat{\mathbf{n}}, \hat{\mathbf{t_1}}, \hat{\mathbf{t_2}})$, where $\hat{\mathbf{n}}$ is the unit normal field, $\hat{\mathbf{t_1}}$, and $\hat{\mathbf{t_2}}$ are unit tangential fields.
Surface Gradient:$\nabla_{\mathcal{S}} F = \frac{1}{g}\frac{\partial F}{\partial t_1}\hat{\mathbf{t_1}} + \frac{1}{h}\frac{\partial F}{\partial t_1}\hat{\mathbf{t_2}}$.
Surface Divergence: $\nabla_{\mathcal{S}} \cdot \mathbf{A} = \frac{1}{fgh}\Big(\frac{\partial}{\partial t_1}(fh\mathbf{A}_{t_1}) + \frac{\partial}{\partial t_2}(fg\mathbf{A}_{t_2})\Big)$
Surface Laplacian (Laplace–Beltrami):
$\Delta_{\mathcal{S}} F = \nabla_{\mathcal{S}} \cdot \nabla_{\mathcal{S}} F = \frac{1}{fgh}\Big(\frac{\partial}{\partial t_1}(\frac{fh}{g}\frac{\partial F}{\partial t_1}) + \frac{\partial}{\partial t_2}(\frac{fg}{h}\frac{\partial F}{\partial t_2})\Big)$

Let u be a scalar function defined on a dimension-$(n-1)$ $\mathcal{S}$ embedded in $\mathbb{R}^n$, v defined in $\mathbb{R}^n$ be an extension of u: $v|_{\mathcal{S}} = u$, and $\hat{\mathbf{n}}$ be the unit normal vector of $\mathcal{S}$.

$$\begin{aligned} \nabla_{\mathcal{S}} u = \nabla_{\mathcal{S}} v &= \nabla v - (\nabla v \cdot \hat{\mathbf{n}})\hat{\mathbf{n}} = \nabla v - \hat{\mathbf{n}}(\hat{\mathbf{n}}^T \nabla v) = \nabla v - (\hat{\mathbf{n}}\mathbf{n}^T)\nabla v \\ &= (\mathbf{I} - \hat{\mathbf{n}}\hat{\mathbf{n}}^T)\nabla v = \mathbf{P}\nabla v, \end{aligned}$$

where $\mathbf{P} = \mathbf{I} - \hat{\mathbf{n}}\hat{\mathbf{n}}^T$ is the projection operator (matrix).

Let u be a scalar function defined on a dimension-$(n-1)$ $\mathcal{S}$ embedded in $\mathbb{R}^n$, v defined in $\mathbb{R}^n$ be an extension of u: $v|_{\mathcal{S}} = u$, and $\hat{\mathbf{n}}$ be the unit normal vector of $\mathcal{S}$.

$$\begin{aligned}\nabla_{\mathcal{S}} u = \nabla_{\mathcal{S}} v \quad &= \nabla v - (\nabla v \cdot \hat{\mathbf{n}})\hat{\mathbf{n}} = \nabla v - \hat{\mathbf{n}}(\hat{\mathbf{n}}^T \nabla v) = \nabla v - (\hat{\mathbf{n}}\mathbf{n}^T)\nabla v \\ &= (\mathbf{I} - \hat{\mathbf{n}}\hat{\mathbf{n}}^T)\nabla v = \mathbf{P}\nabla v,\end{aligned}$$

where $\mathbf{P} = \mathbf{I} - \hat{\mathbf{n}}\hat{\mathbf{n}}^T$ is the projection operator (matrix).

Let $\mathbf{A}$ be a vector function on $\mathcal{S}$, $\mathbf{B}$ in $\mathbb{R}^n$ be an extension of $\mathbf{A}$:

$$\begin{gathered}\nabla_{\mathcal{S}} \cdot \mathbf{A} = \nabla_{\mathcal{S}} \cdot \mathbf{B} = \mathbf{P}\nabla \cdot \mathbf{B} \\ = (\mathbf{I} - \hat{\mathbf{n}}\hat{\mathbf{n}}^T)\nabla \cdot \mathbf{B} = \nabla \cdot \mathbf{B} - \hat{\mathbf{n}}^T(\nabla \mathbf{B})\hat{\mathbf{n}},\end{gathered}$$

where $\nabla \mathbf{B}$ is the Jacobian matrix of $\mathbf{B}$.

Let u be a scalar function defined on a dimension-$(n-1)$ $\mathcal{S}$ embedded in $\mathbb{R}^n$, v defined in $\mathbb{R}^n$ be an extension of u: $v|_{\mathcal{S}} = u$, and $\hat{\mathbf{n}}$ be the unit normal vector of $\mathcal{S}$.

$$\begin{aligned}\nabla_{\mathcal{S}} u = \nabla_{\mathcal{S}} v \quad &= \nabla v - (\nabla v \cdot \hat{\mathbf{n}})\hat{\mathbf{n}} = \nabla v - \hat{\mathbf{n}}(\hat{\mathbf{n}}^T \nabla v) = \nabla v - (\hat{\mathbf{n}}\mathbf{n}^T)\nabla v \\ &= (\mathbf{I} - \hat{\mathbf{n}}\hat{\mathbf{n}}^T)\nabla v = \mathbf{P}\nabla v,\end{aligned}$$

where $\mathbf{P} = \mathbf{I} - \hat{\mathbf{n}}\hat{\mathbf{n}}^T$ is the projection operator (matrix).

Let $\mathbf{A}$ be a vector function on $\mathcal{S}$, $\mathbf{B}$ in $\mathbb{R}^n$ be an extension of $\mathbf{A}$:

$$\begin{gathered}\nabla_{\mathcal{S}} \cdot \mathbf{A} = \nabla_{\mathcal{S}} \cdot \mathbf{B} = \mathbf{P}\nabla \cdot \mathbf{B} \\ = (\mathbf{I} - \hat{\mathbf{n}}\hat{\mathbf{n}}^T)\nabla \cdot \mathbf{B} = \nabla \cdot \mathbf{B} - \hat{\mathbf{n}}^T(\nabla \mathbf{B})\hat{\mathbf{n}},\end{gathered}$$

where $\nabla \mathbf{B}$ is the Jacobian matrix of $\mathbf{B}$.

$$\begin{gathered}\Delta_{\mathcal{S}} u = \Delta_{\mathcal{S}} v = \nabla_{\mathcal{S}} \cdot \nabla_{\mathcal{S}} v = \mathbf{P}\nabla \cdot \mathbf{P}\nabla v \\ \text{(optional exercise)} = \nabla \cdot \nabla v - \kappa \frac{\partial v}{\partial \hat{\mathbf{n}}} - \hat{\mathbf{n}}^T \mathrm{Hess}(v)\hat{\mathbf{n}},\end{gathered}$$

where $\kappa = \nabla \cdot \hat{\mathbf{n}}$, $\frac{\partial v}{\partial \hat{\mathbf{n}}} = \nabla v \cdot \hat{\mathbf{n}}$, $\mathrm{Hess}(v)$ be the Hessian of v.