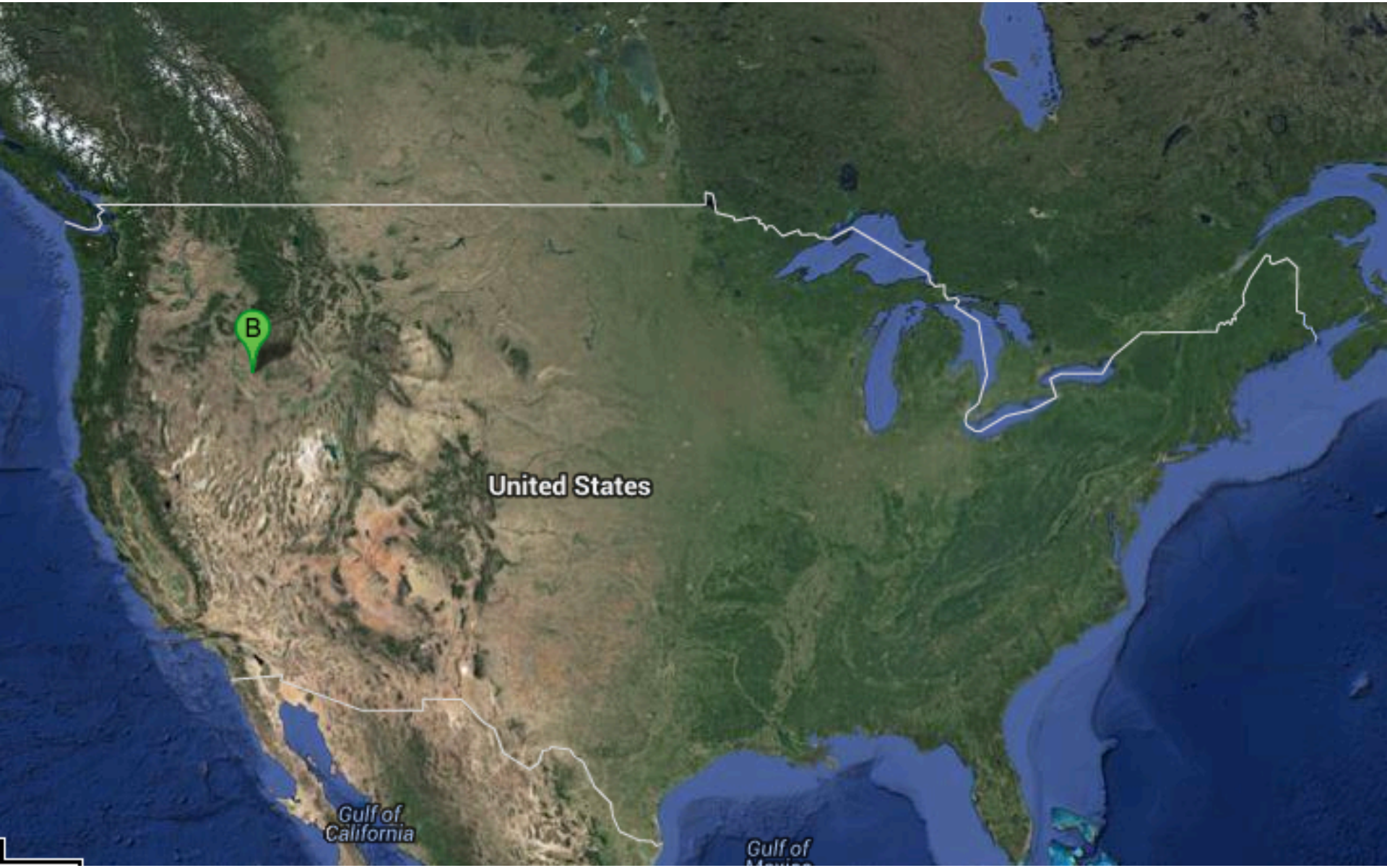
Approximation on surfaces with radial basis functions: from global to local methods

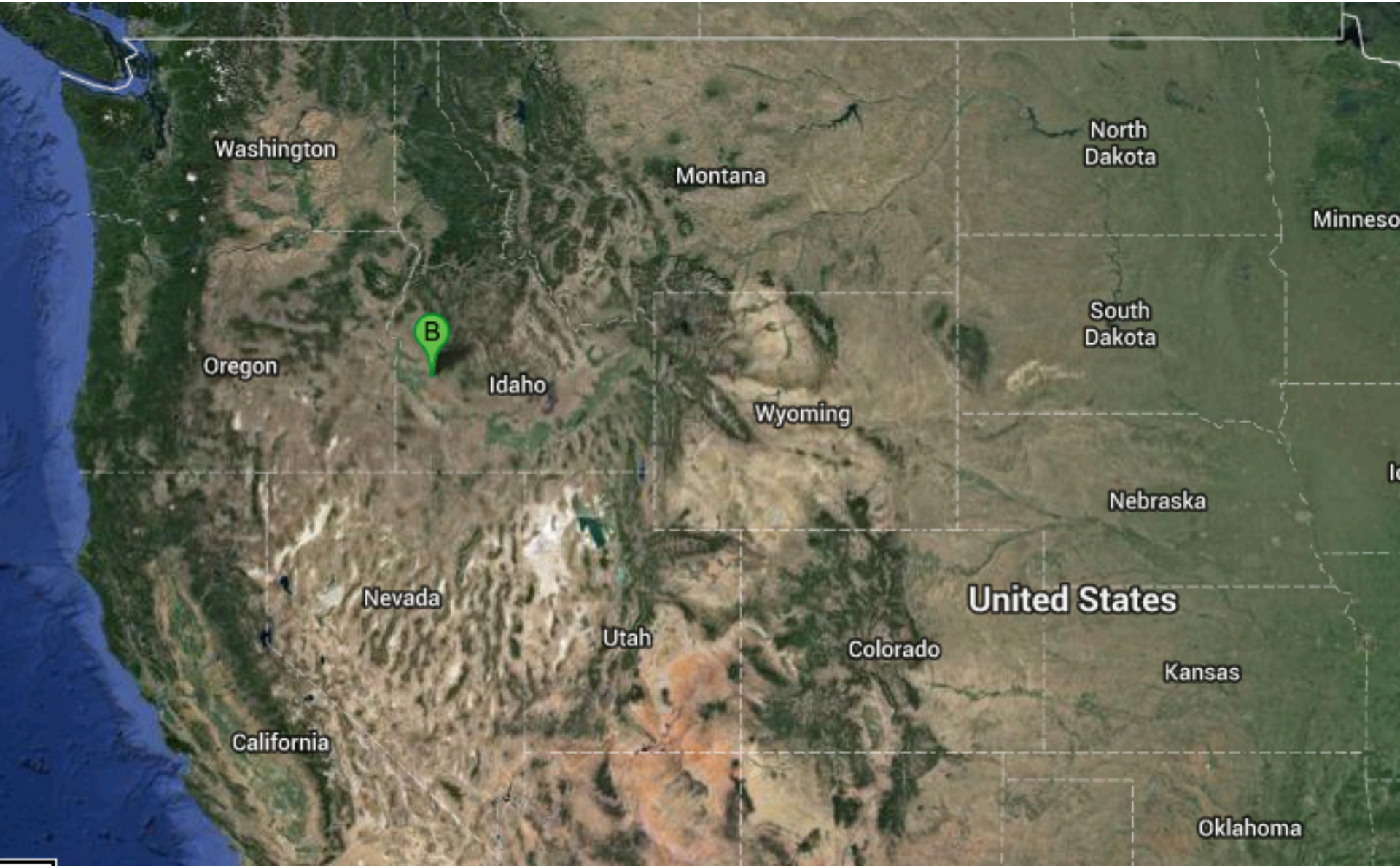
Grady B. Wright
Boise State University

Collaborators: Edward Fuselier (High Point University), Aaron Fogelson, Mike Kirby, and Varun Shankar (University of Utah).

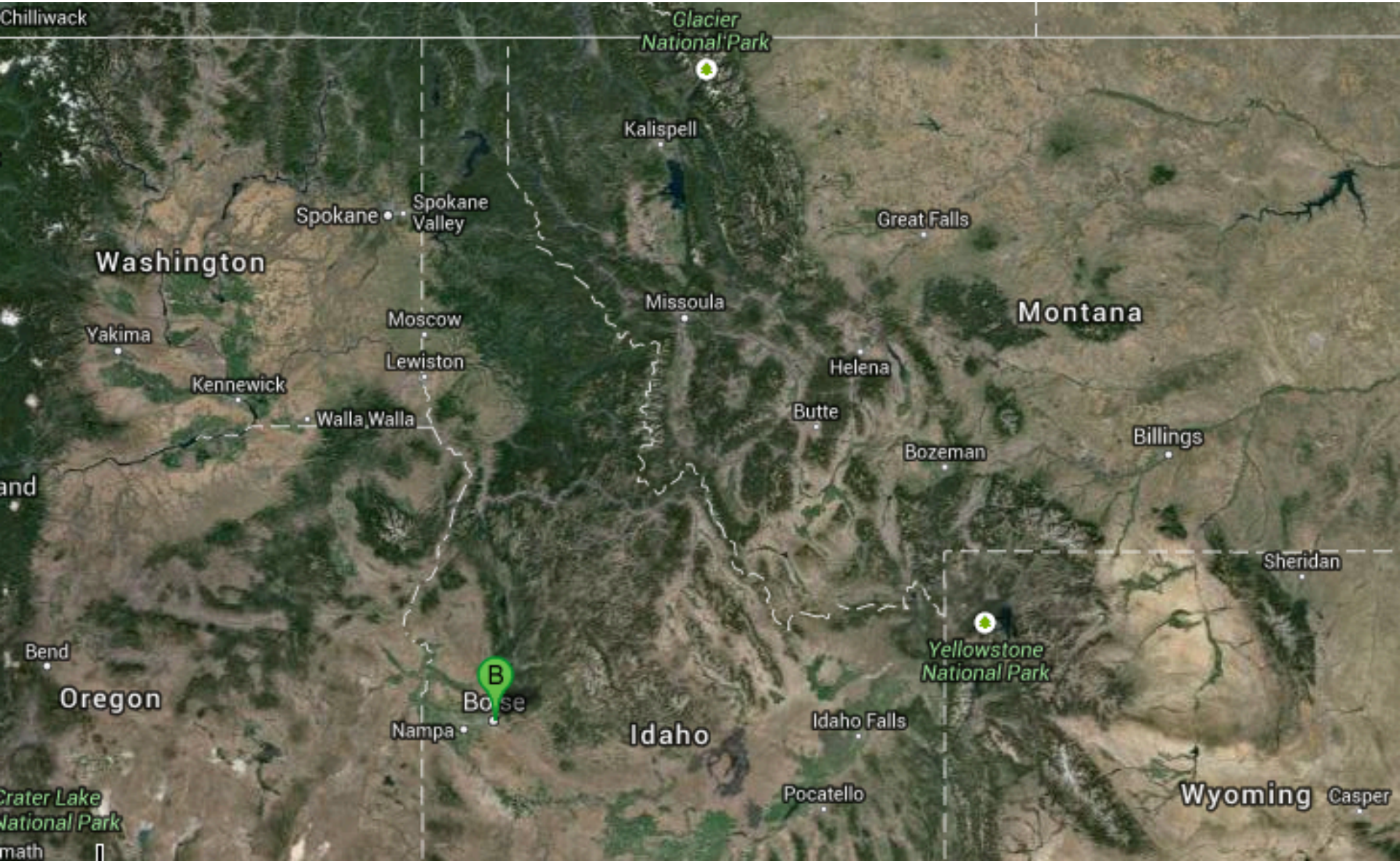
Where is Boise (boy-see)?



Where is Boise (boy-see)?



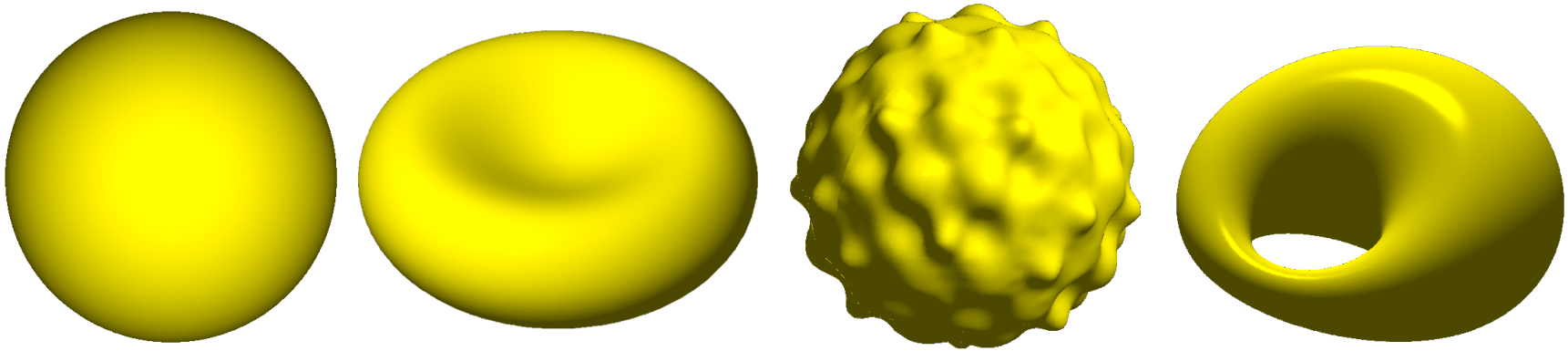
Where is Boise (boy-see)?



Outline

- Introduction to reaction diffusion equations on surfaces
- Overview of some currently used methods
- Review radial basis function (RBF) interpolation
- Global RBF method for approximating surface derivatives
- RBF method for reaction diffusion equations
- Numerical experiments and applications
- Approximating surface derivatives using radial basis function generated finite differences (RBF-FD)
- Numerical experiments and applications
- Concluding remarks

Focus: Reaction diffusion equations on surfaces



- Prototypical model: 2 interacting species

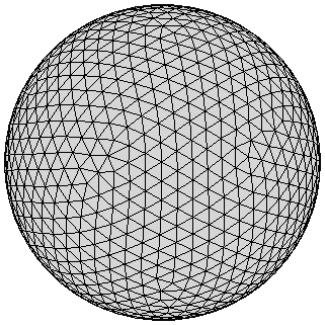
$$\frac{\partial u}{\partial t} = \delta_u \Delta_{\mathbb{M}} u + f_u(t,u,v)$$
$$\frac{\partial v}{\partial t} = \delta_v \Delta_{\mathbb{M}} v + f_v(t,u,v)$$

$\Delta_{\mathbb{M}}$ is the Laplace-Beltrami operator for the surface

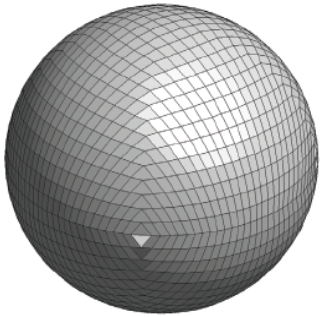
- Applications
  - Biology: diffusive transport on a membrane, pattern formation on animal coats, and tumor growth.
  - Chemistry: waves in excitable media (cardiac arrhythmia, electrical signals in the brain).
  - Computer graphics: texture mapping and synthesis and image processing.

Current methods and RBF approach

1. Surface-based: approximate the PDE *on the surface* using *intrinsic* coordinates.

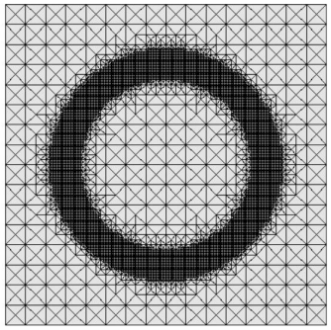


Triangulated Mesh
Dziuk (1988)
Stam (2003)
Xu (2004)
Dziuk & Elliot (2007)

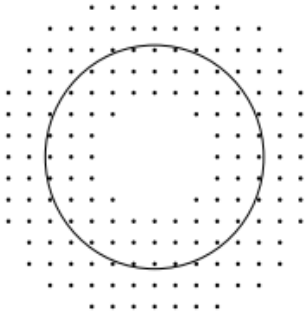


Logically rectangular grid
Calhoun & Helzel (2009)

2. Embedded: approximate the PDE in the *embedding space*, restrict solution to surface.

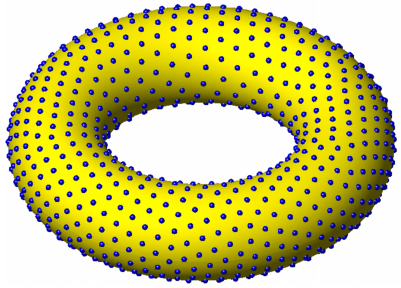


Level Set
Bertalmio *et al.* (2001)
Schwartz *et al.* (2005)
Greer (2006)
Sbalzarini *et al.* (2006)
Dziuk & Elliot (2010)



Closest point:
Ruuth, Merriman (2008)
Macdonald, Ruuth (2008,2009)
März, Macdonald (2012)
Piret (2012)
von Glehn, März, Macdonald (2013)
Macdonald, Merriman, Ruuth (2013)

- RBF methods: Fuselier & W (2013) and Shankar, W, Kirby, & Fogelson (2014)

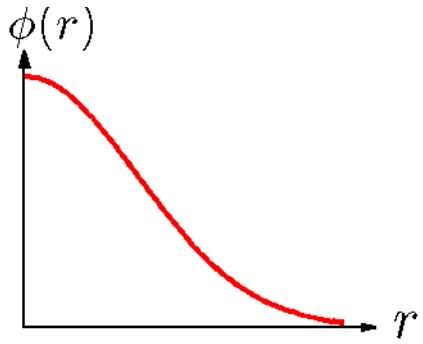


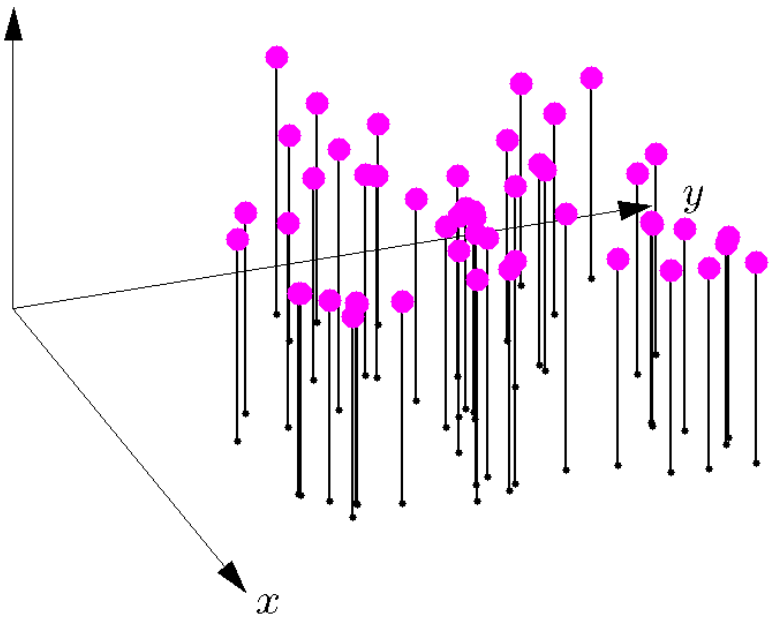
- Similarity to 1: approximate the PDE *on the surface*.
- Similarity to 2: use *extrinsic* coordinates.
- Differences: method is mesh-free;
  computations done in same dimension as the surface.

Radial basis function (RBF) interpolation

Key idea: linear combination of translates and rotations of a single radial kernel:

f $\qquad X = \{\mathbf{x}_j\}_{j=1}^N \subset \Omega, \quad f\big|_X = \{f_j\}_{j=1}^N$





Basic RBF Interpolant for $\Omega \subseteq \mathbb{R}^2$

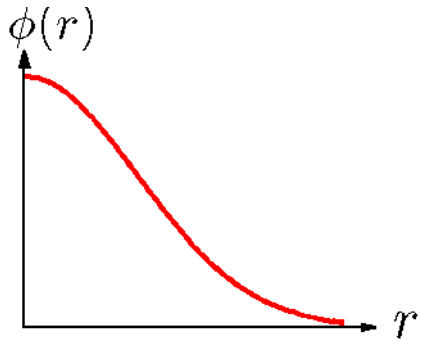
$$I_X f(\mathbf{x}) = \sum_{j=1}^{N} c_j \phi(\|\mathbf{x} - \mathbf{x}_j\|)$$

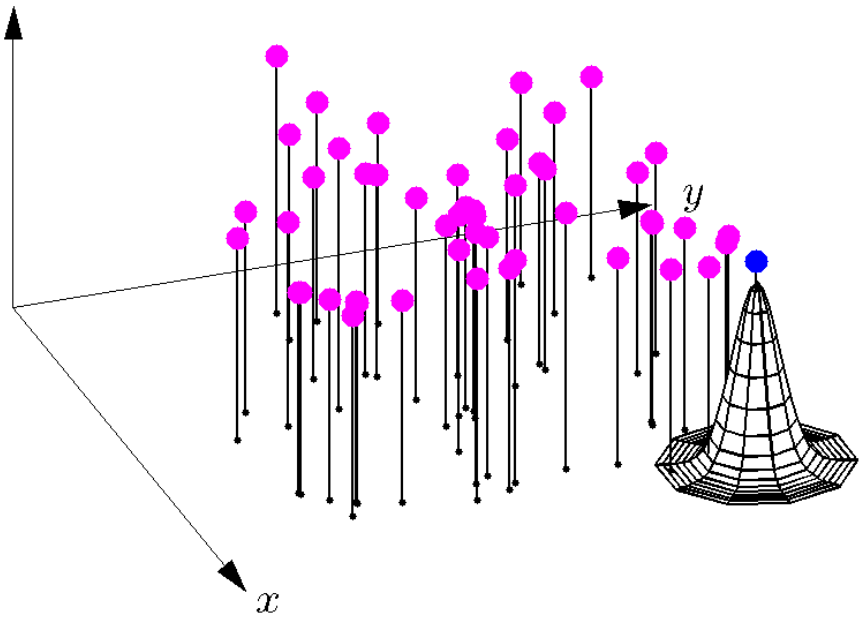
where $\|\mathbf{x} - \mathbf{x}_j\| = \sqrt{(x - x_j)^2 + (y - y_j)^2}$

Radial basis function (RBF) interpolation

Key idea: linear combination of translates and rotations of a single radial kernel:

$f \qquad X = \{\mathbf{x}_j\}_{j=1}^N \subset \Omega, \quad f\big|_X = \{f_j\}_{j=1}^N$





Basic RBF Interpolant for $\Omega \subseteq \mathbb{R}^2$

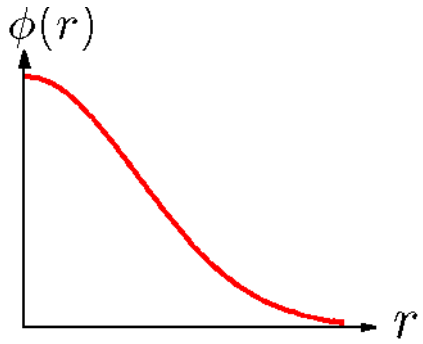
$$I_X f(\mathbf{x}) = \sum_{j=1}^{N} c_j \phi(\|\mathbf{x} - \mathbf{x}_j\|)$$

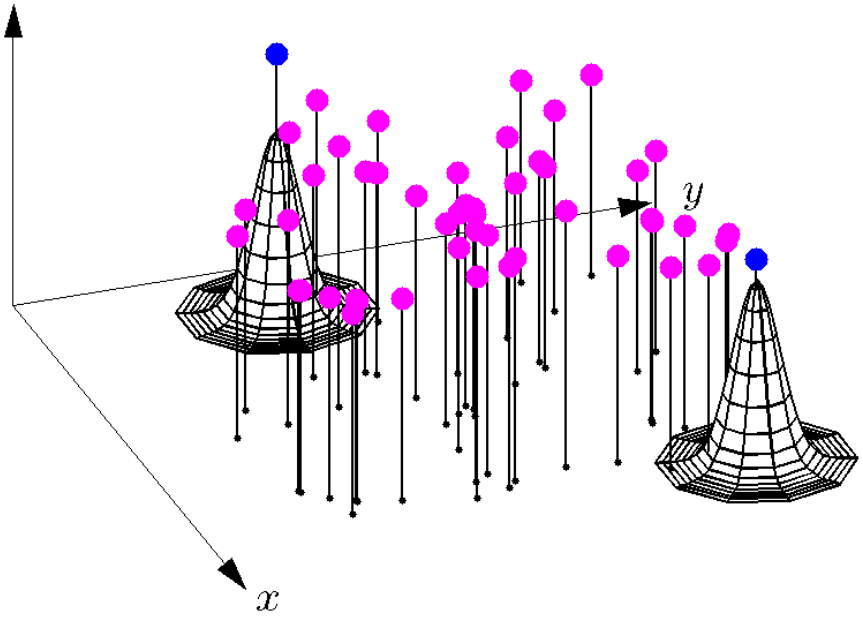
where $\|\mathbf{x} - \mathbf{x}_j\| = \sqrt{(x - x_j)^2 + (y - y_j)^2}$

Radial basis function (RBF) interpolation

Key idea: linear combination of translates and rotations of a single radial kernel:

$f \qquad X = \{\mathbf{x}_j\}_{j=1}^N \subset \Omega, \quad f\big|_X = \{f_j\}_{j=1}^N$





Basic RBF Interpolant for $\Omega \subseteq \mathbb{R}^2$

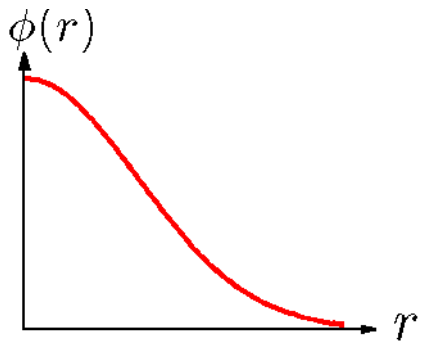
$$I_X f(\mathbf{x}) = \sum_{j=1}^{N} c_j \phi(\|\mathbf{x} - \mathbf{x}_j\|)$$

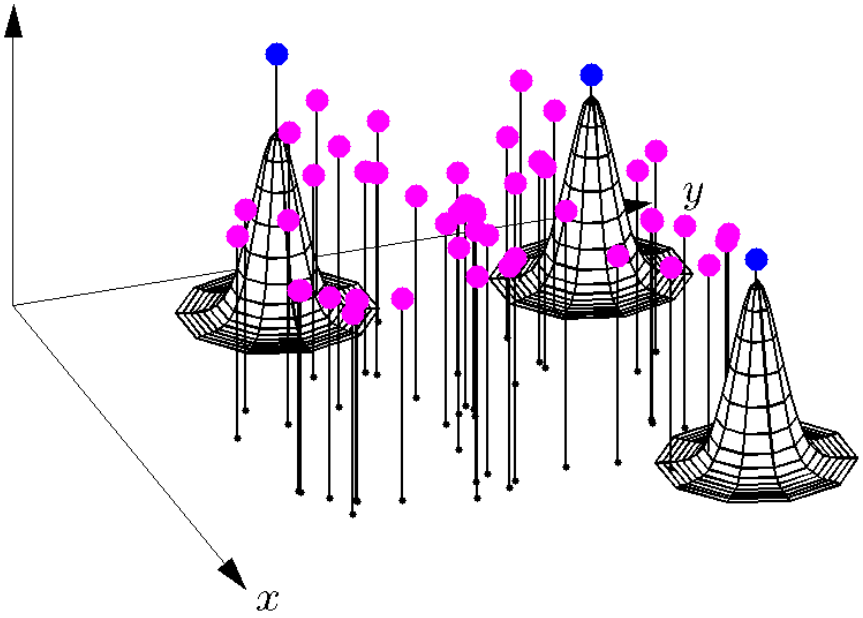
where $\|\mathbf{x} - \mathbf{x}_j\| = \sqrt{(x - x_j)^2 + (y - y_j)^2}$

Radial basis function (RBF) interpolation

Key idea: linear combination of translates and rotations of a single radial kernel:

f $\qquad X = \{\mathbf{x}_j\}_{j=1}^N \subset \Omega, \quad f\big|_X = \{f_j\}_{j=1}^N$





Basic RBF Interpolant for $\Omega \subseteq \mathbb{R}^2$

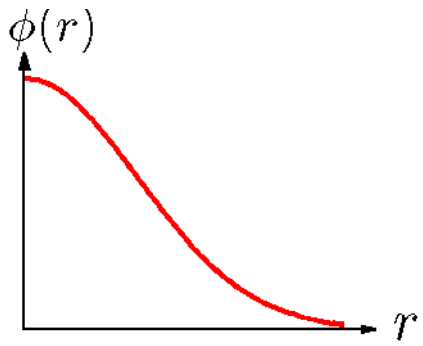
$$I_X f(\mathbf{x}) = \sum_{j=1}^{N} c_j \phi(\|\mathbf{x} - \mathbf{x}_j\|)$$

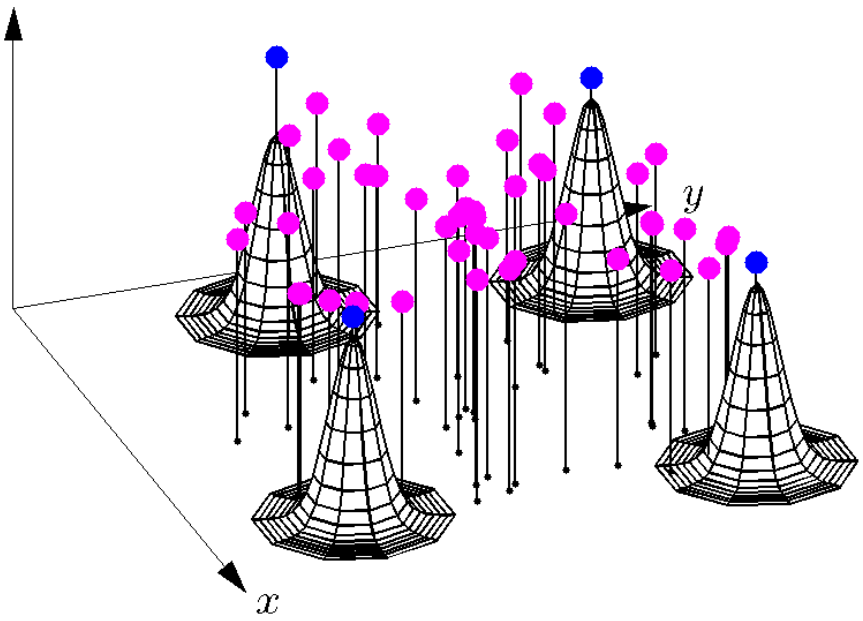
where $\|\mathbf{x} - \mathbf{x}_j\| = \sqrt{(x - x_j)^2 + (y - y_j)^2}$

Radial basis function (RBF) interpolation

Key idea: linear combination of translates and rotations of a single radial kernel:

f $\qquad X = \{\mathbf{x}_j\}_{j=1}^N \subset \Omega, \quad f\big|_X = \{f_j\}_{j=1}^N$





Basic RBF Interpolant for $\Omega \subseteq \mathbb{R}^2$

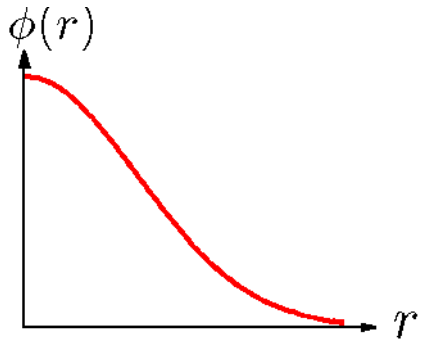
$$I_X f(\mathbf{x}) = \sum_{j=1}^{N} c_j \phi(\|\mathbf{x} - \mathbf{x}_j\|)$$

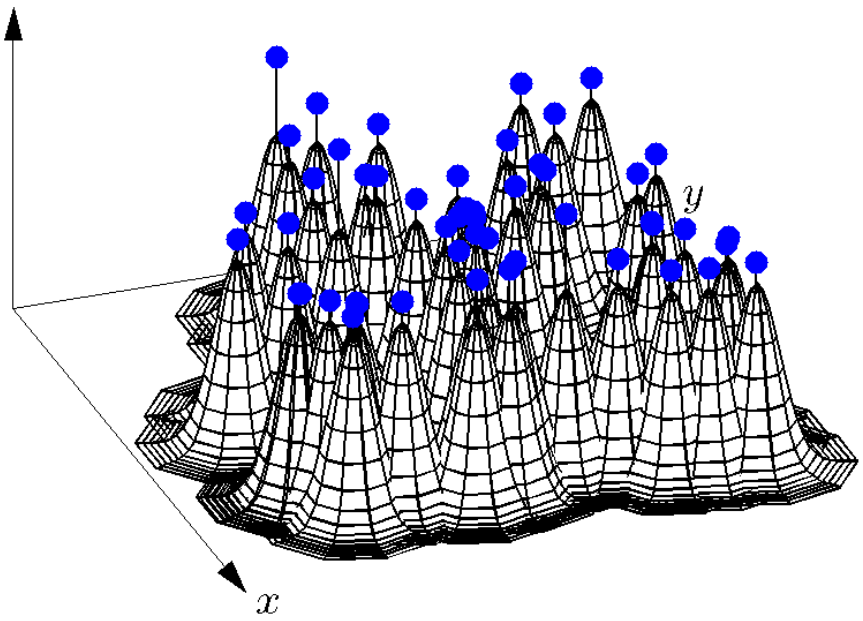
where $\|\mathbf{x} - \mathbf{x}_j\| = \sqrt{(x - x_j)^2 + (y - y_j)^2}$

Radial basis function (RBF) interpolation

Key idea: linear combination of translates and rotations of a single radial kernel:

f $\qquad X = \{\mathbf{x}_j\}_{j=1}^N \subset \Omega, \quad f\Big|_X = \{f_j\}_{j=1}^N$





Basic RBF Interpolant for $\Omega \subseteq \mathbb{R}^2$

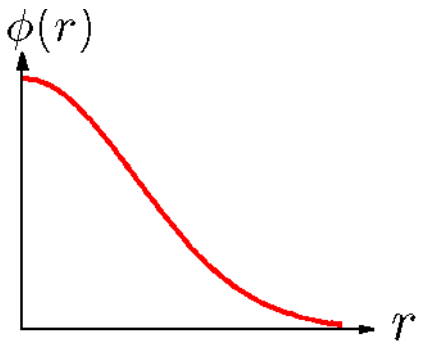
$$I_X f(\mathbf{x}) = \sum_{j=1}^{N} c_j \phi(\|\mathbf{x} - \mathbf{x}_j\|)$$

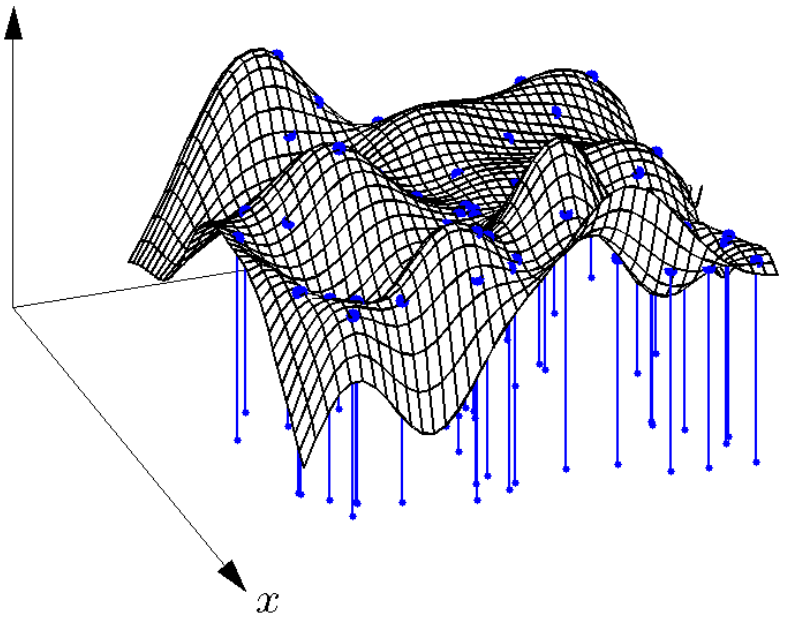
where $\|\mathbf{x} - \mathbf{x}_j\| = \sqrt{(x - x_j)^2 + (y - y_j)^2}$

Radial basis function (RBF) interpolation

Key idea: linear combination of translates and rotations of a single radial kernel:

$f \qquad X = \{\mathbf{x}_j\}_{j=1}^N \subset \Omega, \quad f\Big|_X = \{f_j\}_{j=1}^N$





Basic RBF Interpolant for $\Omega \subseteq \mathbb{R}^2$

$$I_X f(\mathbf{x}) = \sum_{j=1}^{N} c_j \phi(\|\mathbf{x} - \mathbf{x}_j\|)$$

Linear system for determining the interpolation coefficients

$$\underbrace{\begin{bmatrix} \phi(\|\mathbf{x}_1 - \mathbf{x}_1\|) & \phi(\|\mathbf{x}_1 - \mathbf{x}_2\|) \cdots \phi(\|\mathbf{x}_1 - \mathbf{x}_N\|) \\ \phi(\|\mathbf{x}_2 - \mathbf{x}_1\|) & \phi(\|\mathbf{x}_2 - \mathbf{x}_2\|) \cdots \phi(\|\mathbf{x}_2 - \mathbf{x}_N\|) \\ \vdots & \vdots \quad \ddots \quad \vdots \\ \phi(\|\mathbf{x}_N - \mathbf{x}_1\|) & \phi(\|\mathbf{x}_N - \mathbf{x}_2\|) \cdots \phi(\|\mathbf{x}_N - \mathbf{x}_N\|) \end{bmatrix}}_{A_X} \underbrace{\begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_N \end{bmatrix}}_{\underline{c}} = \underbrace{\begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_N \end{bmatrix}}_{\underline{f}}$$

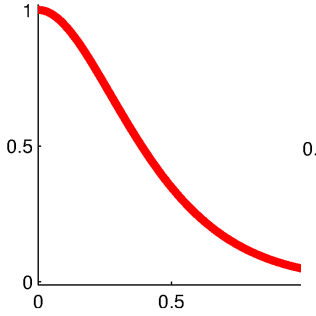
A_X is guaranteed to be positive definite if ϕ is positive definite.

Radial basis function (RBF) interpolation

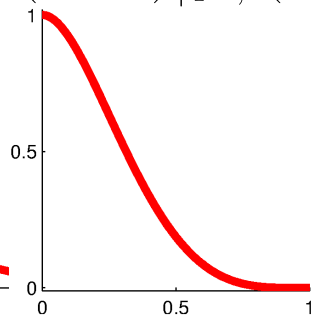
- Classes and examples of positive definite radial kernels:

Piecewise smooth $\phi(r)$ | Infinitely smooth $\phi(r)$

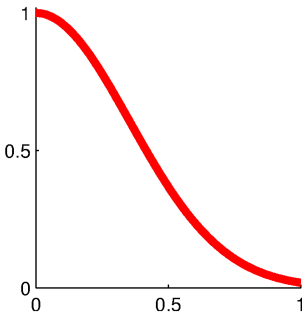
$(\varepsilon r)^\nu K_\nu(\varepsilon r)$ | $(1-\varepsilon r)_+^k p_{d,k}(\varepsilon r)$ | $\exp(-(\varepsilon r)^2)$ | $(1+(\varepsilon r)^2)^{-1/2}$ | $(\varepsilon r)^{-\nu} J_\nu(\varepsilon r)$ | $(\varphi * \varphi)(r)$



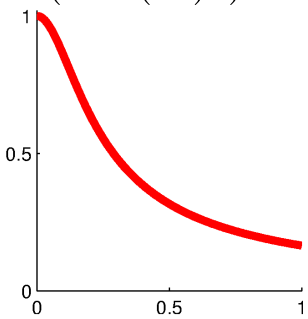
Matérn



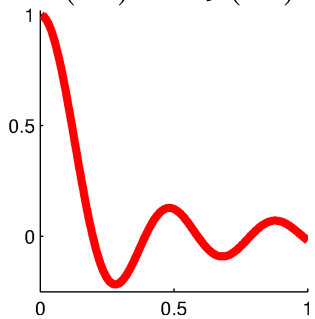
Wendland



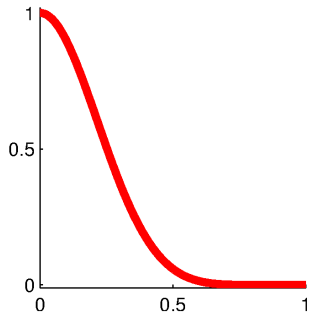
Gaussian



Inverse multiquadric



J-Bessel



Platte

$\widehat{\phi}(\boldsymbol{\omega}) \sim (1+\|\boldsymbol{\omega}\|_2^2)^{-\tau}$

$\widehat{\phi}(\boldsymbol{\omega})$ decays faster than any polynomial power

- Error estimates for interpolant depend on

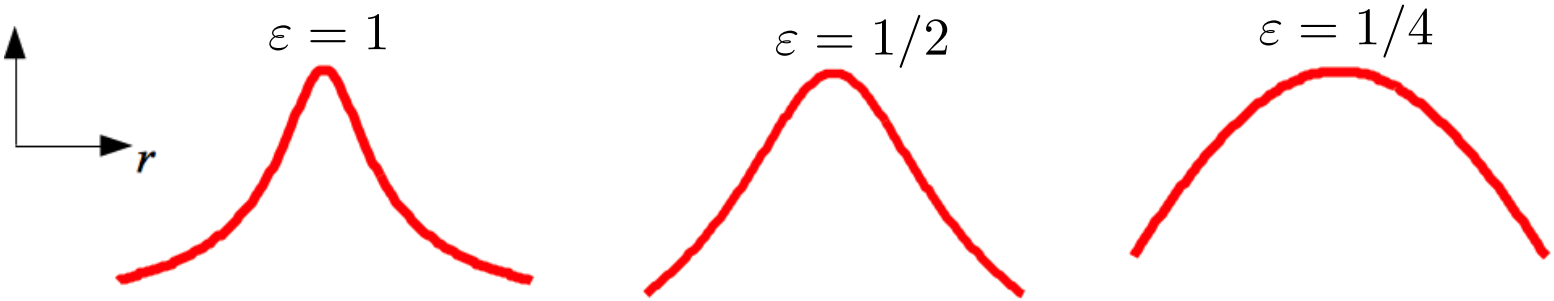
1. Decay rate of Fourier transform $\widehat{\phi}(\boldsymbol{\omega})$
2. Distribution of nodes X
3. Smoothness of target function f
4. Shape parameter ε

Radial basis function (RBF) interpolation

- Smooth kernels with a shape parameter.

Ex: $\phi(r) = \exp(-(\varepsilon r)^2)$ $\phi(r) = \dfrac{1}{\sqrt{1+(\varepsilon r)^2}}$ $\phi(r) = \sqrt{1+(\varepsilon r)^2}$

Issue: Effect of decreasing ε leads to severe ill-conditioning of interp. matrices

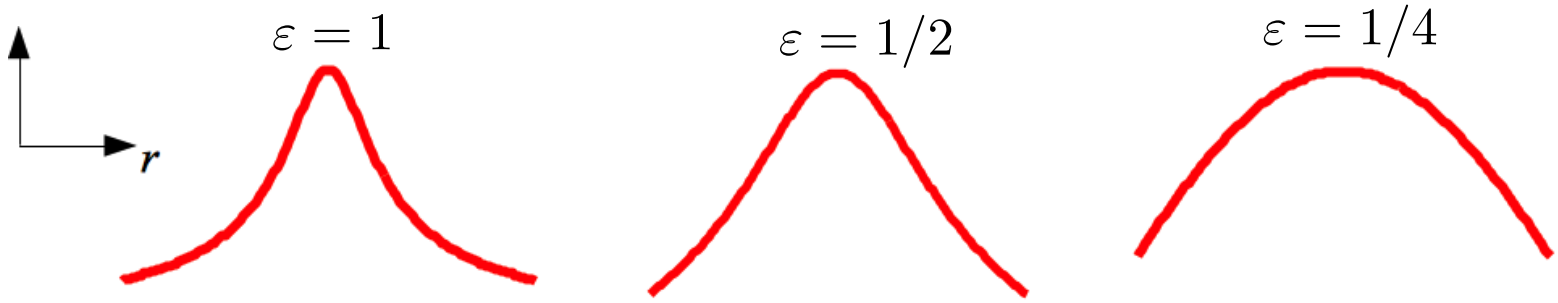


Radial basis function (RBF) interpolation

- Smooth kernels with a shape parameter.

Ex: $\phi(r) = \exp(-(\varepsilon r)^2)$ $\phi(r) = \dfrac{1}{\sqrt{1+(\varepsilon r)^2}}$ $\phi(r) = \sqrt{1+(\varepsilon r)^2}$

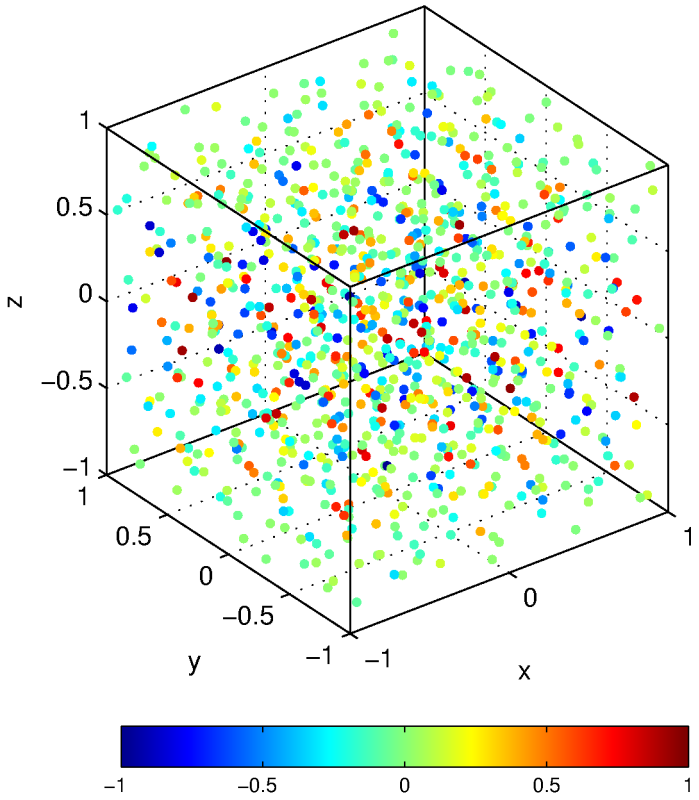
Issue: Effect of decreasing ε leads to severe ill-conditioning of interp. matrices



- Although the interpolation matrices are ill-conditioned, the RBF interpolants remain well-behaved even as $\varepsilon \to 0$.
- Driscoll & Fornberg (2002) first to show that the RBF interpolants converge to polynomial interpolants as $\varepsilon \to 0$.
- Active research area is the development of algorithms to bypass the ill-conditioning issues.
  - Often called "stable algorithms"
  - Fornberg's group has been the most active in this area.

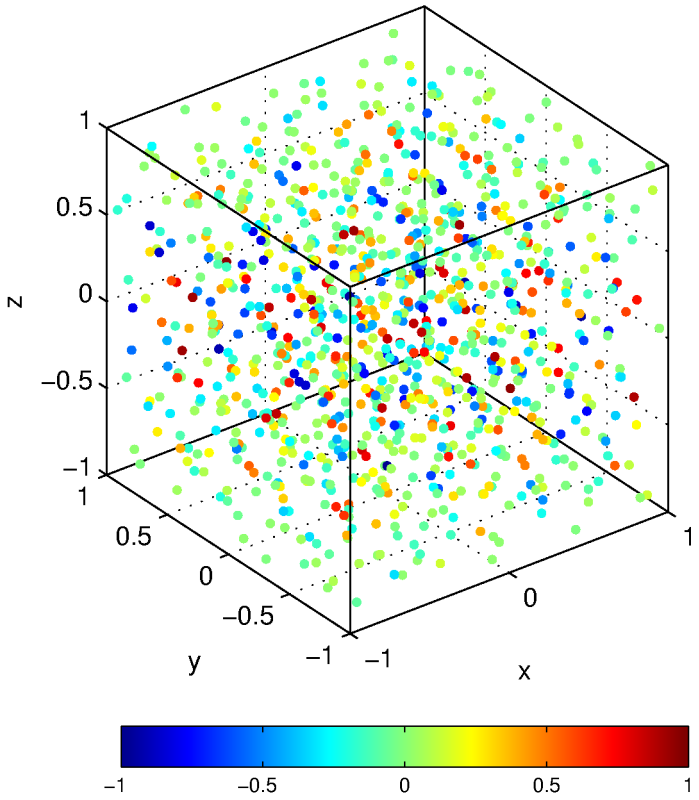
RBF interpolation on surfaces

- Let $\Omega \subset \mathbb{R}^d$ and $X = \{\mathbf{x}_j\}_{j=1}^N$ a set of "scattered" nodes on Ω.
- Suppose a continuous target function $f : \Omega \to \mathbb{R}$ is sampled at X.
- How can we interpolate $f\Big|_X$?
- Example: $\Omega = [-1, 1]^3$



RBF interpolation on surfaces

- Let $\Omega \subset \mathbb{R}^d$ and $X = \{\mathbf{x}_j\}_{j=1}^N$ a set of "scattered" nodes on Ω.
- Suppose a continuous target function $f : \Omega \to \mathbb{R}$ is sampled at X.
- How can we interpolate $f\Big|_X$?
- Example: $\Omega = [-1, 1]^3$



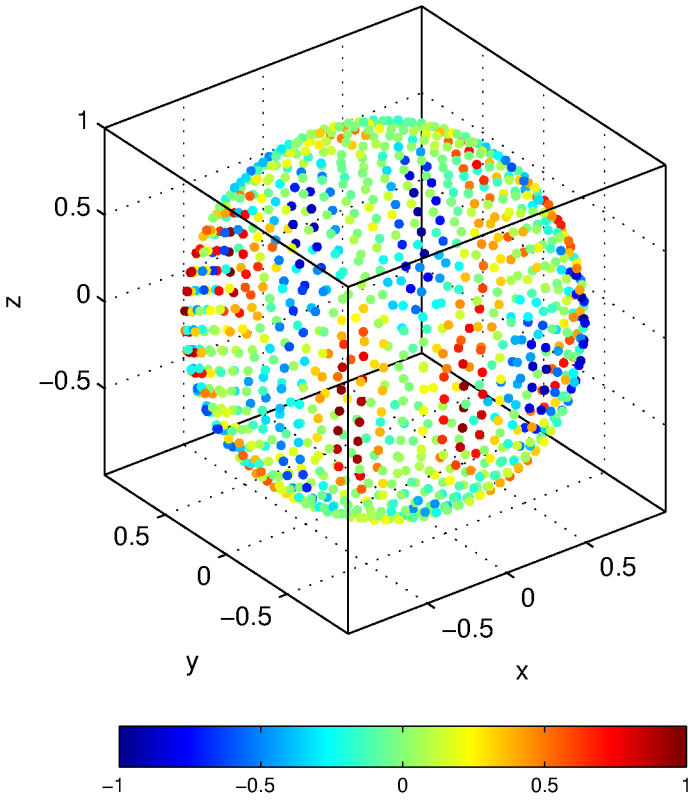
Use standard RBF interpolation:

$$I_X f(\mathbf{x}) = \sum_{j=1}^{N} c_j \phi(\|\mathbf{x} - \mathbf{x}_j\|), \text{ with}$$

$$\|\mathbf{x} - \mathbf{x}_j\|^2 = (x - x_j)^2 + (y - y_j)^2 + (z - z_j)^2$$

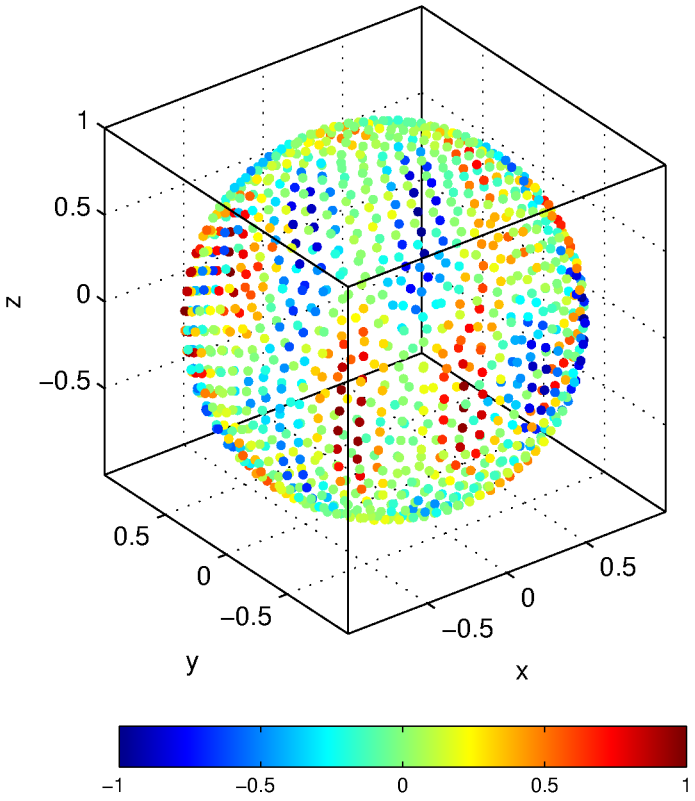
RBF interpolation on surfaces

- Let $\Omega \subset \mathbb{R}^d$ and $X = \{\mathbf{x}_j\}_{j=1}^N$ a set of "scattered" nodes on Ω.
- Suppose a continuous target function $f : \Omega \to \mathbb{R}$ is sampled at X.
- How can we interpolate $f\Big|_X$?
- Example: $\Omega = \mathbb{S}^2$



RBF interpolation on surfaces

- Let $\Omega \subset \mathbb{R}^d$ and $X = \{\mathbf{x}_j\}_{j=1}^N$ a set of "scattered" nodes on Ω.
- Suppose a continuous target function $f : \Omega \to \mathbb{R}$ is sampled at X.
- How can we interpolate $f\Big|_X$?
- Example: $\Omega = \mathbb{S}^2$



Use standard RBF interpolation:

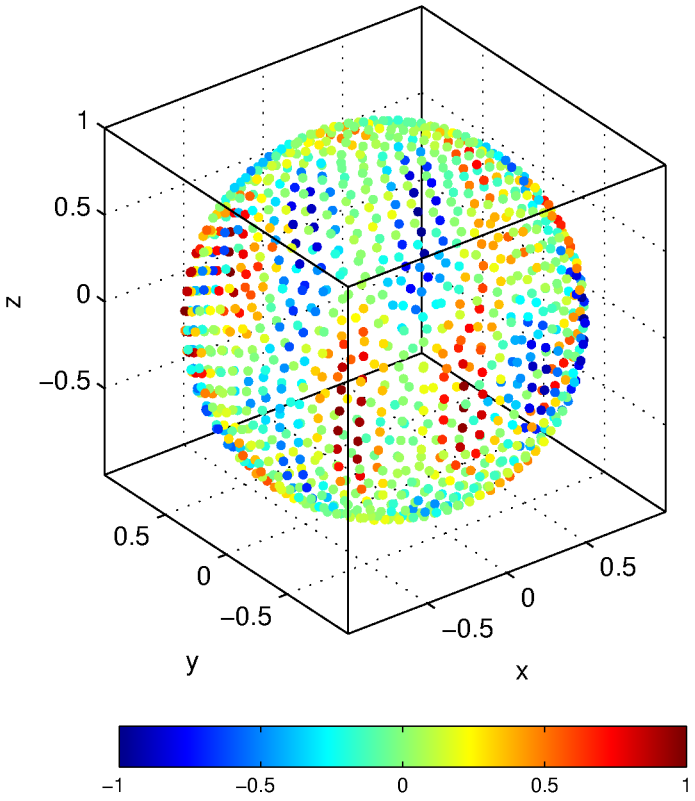
$$I_X f(\mathbf{x}) = \sum_{j=1}^{N} c_j \phi(\|\mathbf{x} - \mathbf{x}_j\|), \text{ with}$$

$$\|\mathbf{x} - \mathbf{x}_j\|^2 = (x - x_j)^2 + (y - y_j)^2 + (z - z_j)^2$$

- First RBF study by Cheney (1995)
- Many, many studies since then.

RBF interpolation on surfaces

- Let $\Omega \subset \mathbb{R}^d$ and $X = \{\mathbf{x}_j\}_{j=1}^N$ a set of "scattered" nodes on Ω.
- Suppose a continuous target function $f : \Omega \to \mathbb{R}$ is sampled at X.
- How can we interpolate $f\Big|_X$?
- Example: $\Omega = \mathbb{S}^2$



Use standard RBF interpolation:

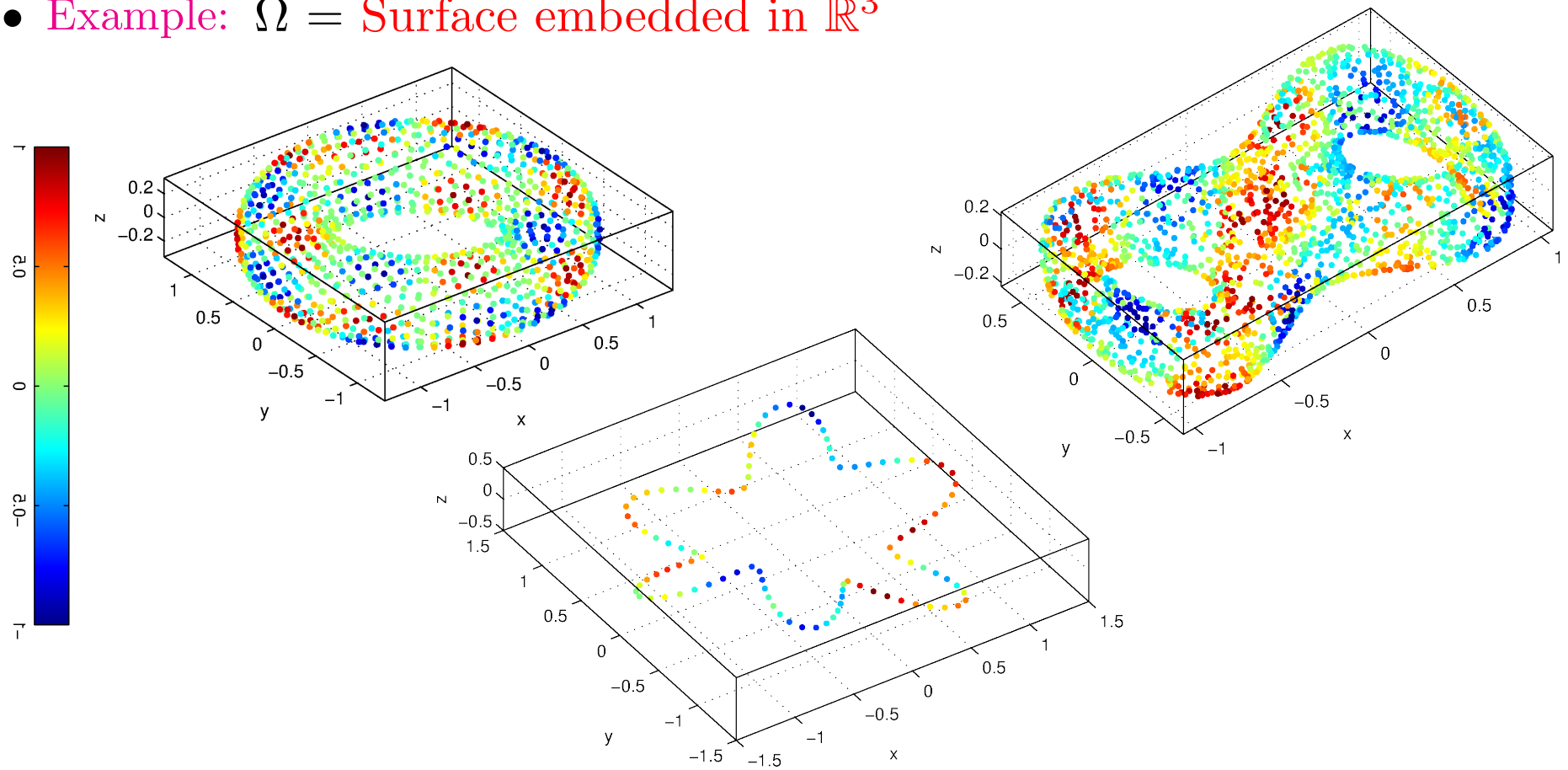
$$I_X f(\mathbf{x}) = \sum_{j=1}^{N} c_j \psi(\mathbf{x}^T \mathbf{x}_j), \text{ with}$$

$$\|\mathbf{x} - \mathbf{x}_j\|^2 = 2(1 - \mathbf{x}^T \mathbf{x}_j)$$

- First RBF study by Cheney (1995)
- Many, many studies since then.
- History in work of Schoenberg (1940)

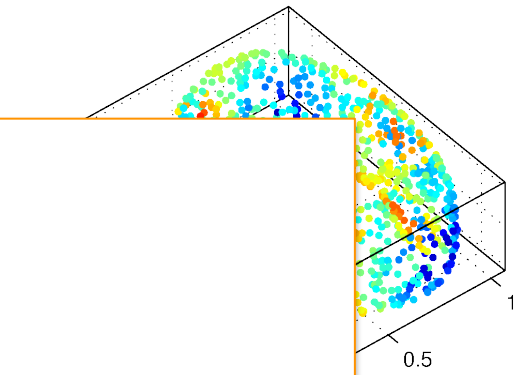
RBF interpolation on surfaces

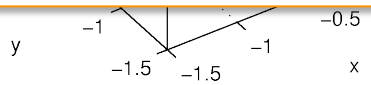
- Let $\Omega \subset \mathbb{R}^d$ and $X = \{\mathbf{x}_j\}_{j=1}^N$ a set of "scattered" nodes on Ω.
- Suppose a continuous target function $f : \Omega \to \mathbb{R}$ is sampled at X.
- How can we interpolate $f\Big|_X$?
- Example: $\Omega =$ Surface embedded in $\mathbb{R}^3$



RBF interpolation on surfaces

- Let $\Omega \subset \mathbb{R}^d$ and $X = \{\mathbf{x}_j\}_{j=1}^N$ a set of "scattered" nodes on Ω.
- Suppose a continuous target function $f : \Omega \to \mathbb{R}$ is sampled at X.
- How can we interpolate $f\Big|_X$?
- Example: $\Omega =$ Surface embedded in $\mathbb{R}^3$





Use standard RBF interpolation!

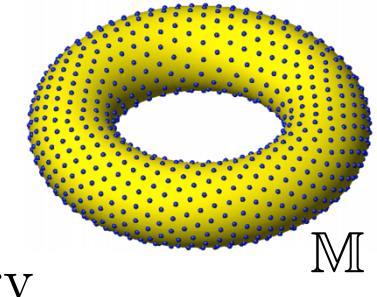
$$I_X f(\mathbf{x}) = \sum_{j=1}^{N} c_j \phi(\|\mathbf{x} - \mathbf{x}_j\|), \text{ with}$$

$$\|\mathbf{x} - \mathbf{x}_j\|^2 = (x - x_j)^2 + (y - y_j)^2 + (z - z_j)^2$$

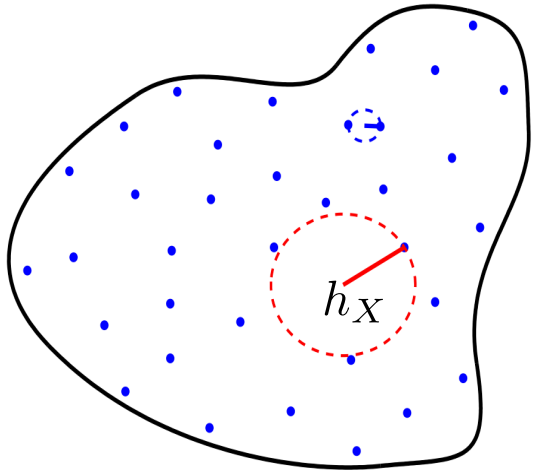
- M.J.D. Powell, *DAMTP Technical Report*, 2001
- G. Fasshauer, *Meshless Approximation Methods with Matlab*, 2007
- Fuselier & W, *SINUM*, 2012

A flavor of the type of interpolation error estimates

- Specific error estimate results from Fuselier & W (2012).
  - More general results are given in the paper.

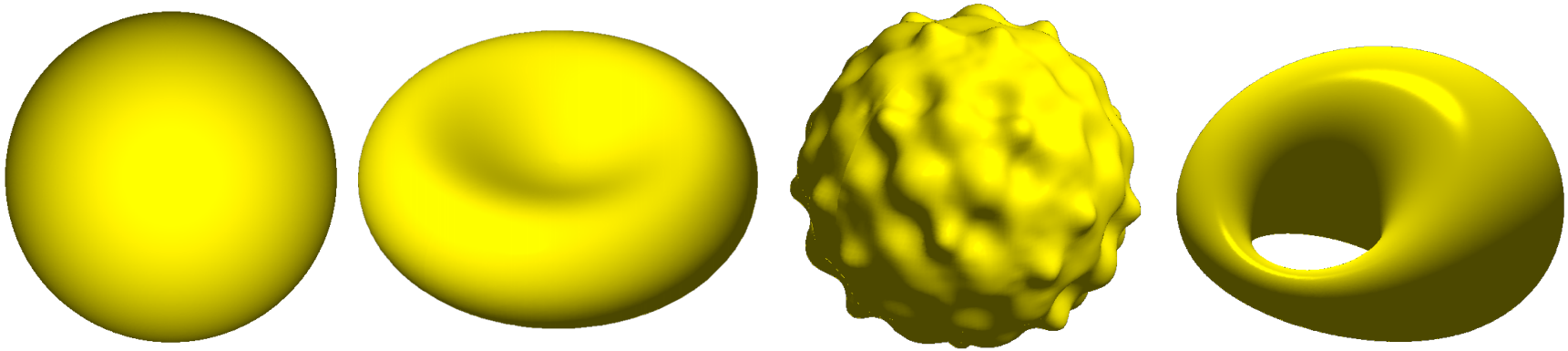


- Assumptions and notation
  - $\mathbb{M}$ compact smooth surface in $\mathbb{R}^3$ without a boundary
  - $\dim(\mathbb{M}) = 2$
  - $X = \{\mathbf{x}_j\}_{j=1}^N \subset \mathbb{M}$
  - $h_X = \sup_{\mathbf{x}\in\mathbb{M}} \text{dist}_{\mathbb{M}}(\mathbf{x}, X)$
  - $\hat{\phi}(\boldsymbol{\omega}) \sim (1 + \|\boldsymbol{\omega}\|_2^2)^{-\tau}$, $\tau > 3/2$
  - $s = \tau - 1/2$



Theorem: If $f \in H^s(\mathbb{M})$ then $\|f - I_X f\|_{L_2(\mathbb{M})} \leq C_{f,\mathbb{M}} h_X^s$

Return: Reaction diffusion equations on surfaces



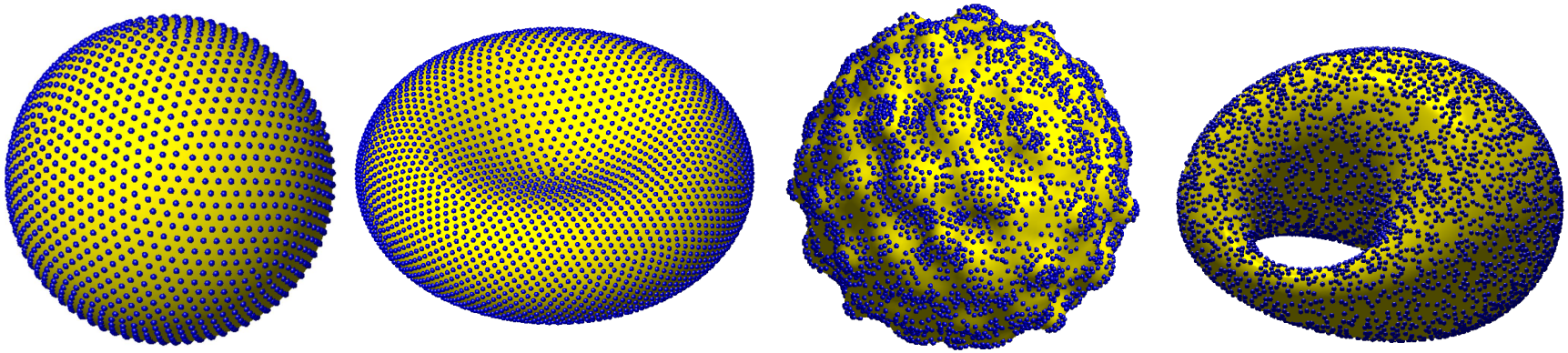
- Prototypical model: 2 interacting species

$$\frac{\partial u}{\partial t} = \delta_u \Delta_{\mathbb{M}} u + f_u(t,u,v)$$
$$\frac{\partial v}{\partial t} = \delta_v \Delta_{\mathbb{M}} v + f_v(t,u,v)$$

$\Delta_{\mathbb{M}}$ is the Laplace-Beltrami operator for the surface

- Applications
  - Biology: diffusive transport on a membrane, pattern formation on animal coats, and tumor growth.
  - Chemistry: waves in excitable media (cardiac arrhythmia, electrical signals in the brain).
  - Computer graphics: texture mapping and synthesis and image processing.

Return: Reaction diffusion equations on surfaces



- Prototypical model: 2 interacting species

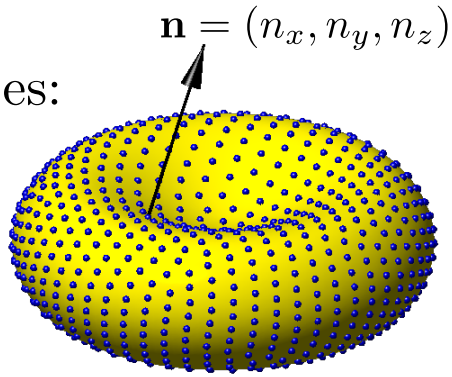
$$\frac{\partial u}{\partial t} = \delta_u \Delta_{\mathbb{M}} u + f_u(t,u,v)$$
$$\frac{\partial v}{\partial t} = \delta_v \Delta_{\mathbb{M}} v + f_v(t,u,v)$$

$\Delta_{\mathbb{M}}$ is the Laplace-Beltrami operator for the surface

- Applications
  - Biology: diffusive transport on a membrane, pattern formation on animal coats, and tumor growth.
  - Chemistry: waves in excitable media (cardiac arrhythmia, electrical signals in the brain).
  - Computer graphics: texture mapping and synthesis and image processing.

Differential operators on surfaces

- Let $\mathbb{M} \subset \mathbb{R}^3$ be a smooth, compact, two dimensional surface.
- Surface gradient on $\mathbb{M}$ in *extrinsic* (or Cartesian) coordinates:



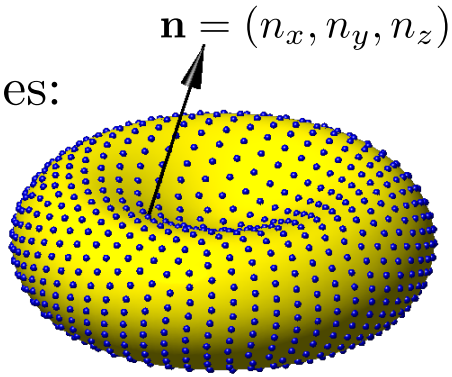
$$\nabla_{\mathbb{M}} := \mathbf{P}\nabla = (\mathbf{I} - \mathbf{n} \otimes \mathbf{n})\,\nabla$$

- After some manipulations

$$\nabla_{\mathbb{M}} := \begin{bmatrix} (\mathbf{e}_x \cdot \mathbf{P})\,\nabla \\ (\mathbf{e}_y \cdot \mathbf{P})\,\nabla \\ (\mathbf{e}_z \cdot \mathbf{P})\,\nabla \end{bmatrix} = \begin{bmatrix} (\mathbf{e}_x - n_x\mathbf{n}) \cdot \nabla \\ (\mathbf{e}_y - n_y\mathbf{n}) \cdot \nabla \\ (\mathbf{e}_z - n_z\mathbf{n}) \cdot \nabla \end{bmatrix} = \begin{bmatrix} \mathbf{p}_x \cdot \nabla \\ \mathbf{p}_y \cdot \nabla \\ \mathbf{p}_z \cdot \nabla \end{bmatrix} = \begin{bmatrix} \mathcal{G}^x \\ \mathcal{G}^y \\ \mathcal{G}^z \end{bmatrix}$$

Differential operators on surfaces

- Let $\mathbb{M} \subset \mathbb{R}^3$ be a smooth, compact, two dimensional surface.
- Surface gradient on $\mathbb{M}$ in *extrinsic* (or Cartesian) coordinates:



$$\nabla_{\mathbb{M}} := \mathbf{P}\nabla = (\mathbf{I} - \mathbf{n} \otimes \mathbf{n})\,\nabla$$

- After some manipulations

$$\nabla_{\mathbb{M}} := \begin{bmatrix} (\mathbf{e}_x \cdot \mathbf{P})\,\nabla \\ (\mathbf{e}_y \cdot \mathbf{P})\,\nabla \\ (\mathbf{e}_z \cdot \mathbf{P})\,\nabla \end{bmatrix} = \begin{bmatrix} (\mathbf{e}_x - n_x\mathbf{n}) \cdot \nabla \\ (\mathbf{e}_y - n_y\mathbf{n}) \cdot \nabla \\ (\mathbf{e}_z - n_z\mathbf{n}) \cdot \nabla \end{bmatrix} = \begin{bmatrix} \mathbf{p}_x \cdot \nabla \\ \mathbf{p}_y \cdot \nabla \\ \mathbf{p}_z \cdot \nabla \end{bmatrix} = \begin{bmatrix} \mathcal{G}^x \\ \mathcal{G}^y \\ \mathcal{G}^z \end{bmatrix}$$

- Surface divergence of smooth vector field $\mathbf{f} : \mathbb{R}^3 \to \mathbb{R}^3$ ($\mathbf{f} = (f_x, f_y, f_z)$):

$$\nabla_{\mathbb{M}} \cdot \mathbf{f} := (\mathbf{P}\nabla) \cdot \mathbf{f} = \mathcal{G}^x f_x + \mathcal{G}^y f_y + \mathcal{G}^z f_z$$

- Laplace-Beltrami operator on $\mathbb{M}$ in *extrinsic* coordinates:

$$\Delta_{\mathbb{M}} := (\mathbf{P}\nabla)\cdot(\mathbf{P}\nabla) = \mathcal{G}^x\mathcal{G}^x + \mathcal{G}^y\mathcal{G}^y + \mathcal{G}^z\mathcal{G}^z = \mathcal{D}_{xx} + \mathcal{D}_{yy} + \mathcal{D}_{zz}$$

RBF approximation of surface derivative operators

Idea from Fuselier & W (2013):

- Let $X = \{\mathbf{x}_j\}_{j=1}^N \subset \mathbb{M}$ and some smooth target $f : \mathbb{M} \to \mathbb{R}$.
- Interpolate $\underline{f} := f\big|_X$, using RBF interpolant:

$$I_X f(\mathbf{x}) = \sum_{j=1}^{N} c_j \phi(\|\mathbf{x} - \mathbf{x}_j\|)$$

RBF approximation of surface derivative operators

Idea from Fuselier & W (2013):

- Let $X = \{\mathbf{x}_j\}_{j=1}^N \subset \mathbb{M}$ and some smooth target $f : \mathbb{M} \to \mathbb{R}$.
- Interpolate $\underline{f} := f\big|_X$, using RBF interpolant:

$$I_X f(\mathbf{x}) = \sum_{j=1}^{N} c_j \phi(\|\mathbf{x} - \mathbf{x}_j\|), \quad A_X \underline{c} = \underline{f}$$

- Apply $\mathcal{G}^x$, $\mathcal{G}^y$, $\mathcal{G}^z$ to $I_X f$ and evaluate at X.

Example:

$$(\mathcal{G}^x[I_X f])\big|_X = \underbrace{B_X^x A_X^{-1}}_{G_X^x} \underline{f}, \quad \text{where } (B_X^x)_{i,j} = (\mathcal{G}^x \phi(\|\mathbf{x} - \mathbf{x}_j\|))\big|_{\mathbf{x}=\mathbf{x}_i}$$

RBF approximation of surface derivative operators

Idea from Fuselier & W (2013):

- Let $X = \{\mathbf{x}_j\}_{j=1}^N \subset \mathbb{M}$ and some smooth target $f : \mathbb{M} \to \mathbb{R}$.
- Interpolate $\underline{f} := f\big|_X$, using RBF interpolant:

$$I_X f(\mathbf{x}) = \sum_{j=1}^{N} c_j \phi(\|\mathbf{x} - \mathbf{x}_j\|), \quad A_X \underline{c} = \underline{f}$$

- Apply $\mathcal{G}^x$, $\mathcal{G}^y$, $\mathcal{G}^z$ to $I_X f$ and evaluate at X:

$$(\mathcal{G}^x[I_X f])\big|_X = G_X^x \underline{f}, \quad (\mathcal{G}^y[I_X f])\big|_X = G_X^y \underline{f}, \quad (\mathcal{G}^z[I_X f])\big|_X = G_X^z \underline{f}$$

RBF approximation of surface derivative operators

Idea from Fuselier & W (2013):

- Let $X = \{\mathbf{x}_j\}_{j=1}^N \subset \mathbb{M}$ and some smooth target $f : \mathbb{M} \to \mathbb{R}$.
- Interpolate $\underline{f} := f\big|_X$, using RBF interpolant:

$$I_X f(\mathbf{x}) = \sum_{j=1}^{N} c_j \phi(\|\mathbf{x} - \mathbf{x}_j\|), \quad A_X \underline{c} = \underline{f}$$

- Apply $\mathcal{G}^x$, $\mathcal{G}^y$, $\mathcal{G}^z$ to $I_X f$ and evaluate at X:

$$(\mathcal{G}^x[I_X f])\big|_X = G_X^x \underline{f}, \quad (\mathcal{G}^y[I_X f])\big|_X = G_X^y \underline{f}, \quad (\mathcal{G}^z[I_X f])\big|_X = G_X^z \underline{f}$$

- Approximate $(\Delta_{\mathbb{M}} f)\big|_X$ using G_X^x, G_X^y, G_X^z:

$$(\Delta_{\mathbb{M}} f)\big|_X = ([\mathcal{G}^x\mathcal{G}^x + \mathcal{G}^y\mathcal{G}^y + \mathcal{G}^z\mathcal{G}^z]f)\big|_X \approx \underbrace{[G_X^x G_X^x + G_X^y G_X^y + G_X^z G_X^z]}_{L_X} \underline{f}$$

- L_X is an $N \times N$ differentiation matrix.

Comments on surface Laplacian approximation

$$(\Delta_{\mathbb{M}} f)\big|_X = ([\mathcal{G}^x\mathcal{G}^x + \mathcal{G}^y\mathcal{G}^y + \mathcal{G}^z\mathcal{G}^z]f)\big|_X \approx \underbrace{[G_X^x G_X^x + G_X^y G_X^y + G_X^z G_X^z]}_{L_X}\underline{f}$$

- Constructed by repeated application of "interpolate then differentiate".
  - Known in the 1-D spline literature as iterated derivative approximation
- Do not have to analytically differentiate the surface normal vectors.
- Error estimates given in Fuselier & W (2013).
- Approximation may exhibit a type of super convergence when measuring the errors at the nodes (Fuselier & W (2014)).
- We use L_X in the method-of-lines (MOL) to numerically solve surface PDEs:

$$\frac{\partial u}{\partial t} = \delta\Delta_{\mathbb{M}} u + g(t,u) \xrightarrow{\text{MOL}} \frac{d}{dt}\underline{u} = \delta L_X \underline{u} + g(t,\underline{u})$$

We use a SBDF scheme to advance the system in time.

Eigenvalue stability

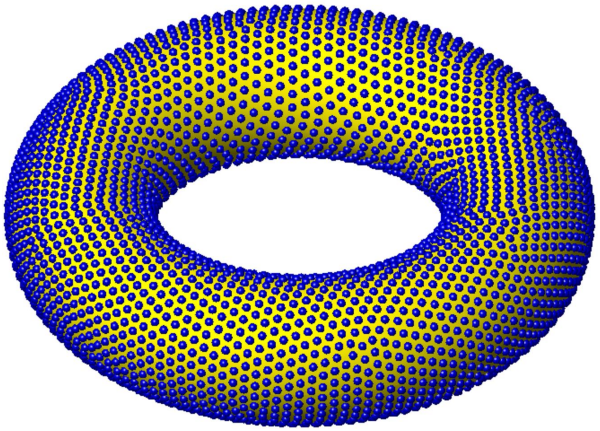
- Necessary condition for time-stability of the numerical scheme:

  All eigenvalues of L_X need to be in the left-half plane

- Our discrete approximation does not guarantee this will be true.
- No proof yet of sufficient conditions for this property.
- Experimental evidence suggests they are, provided the node set X is "sufficiently" dense.

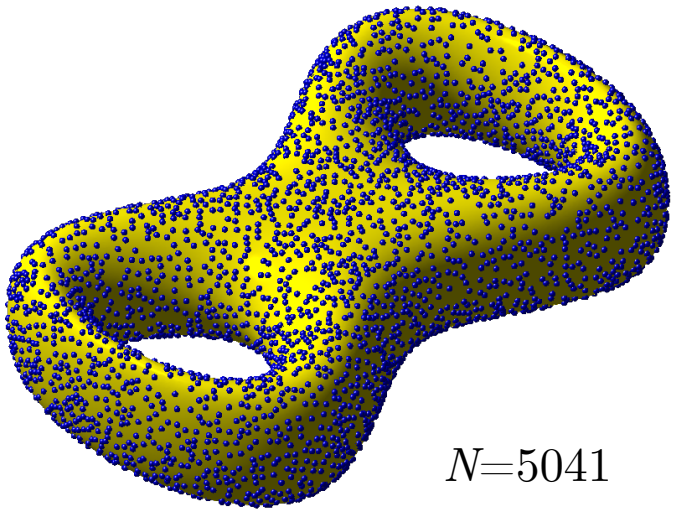
Examples: IMQ radial kernel, $\phi(r) = \dfrac{1}{\sqrt{1+(\varepsilon r)^2}}$

Torus



N=4000

Bretzel



N=5041

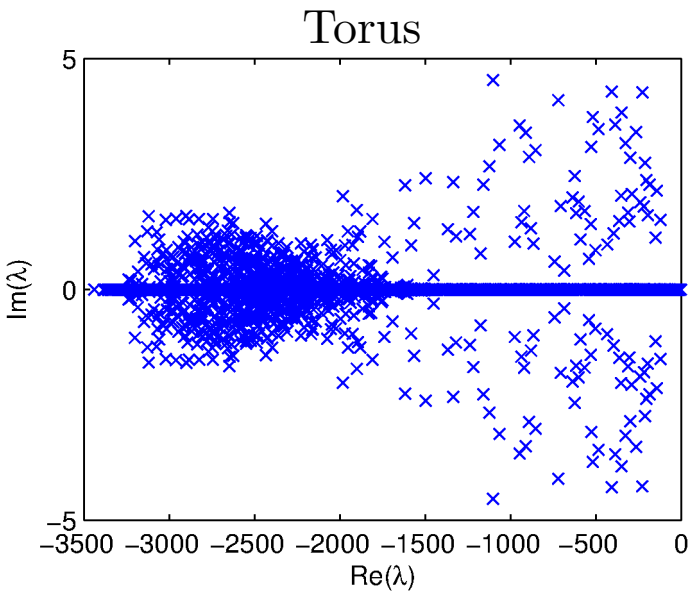
Eigenvalue stability

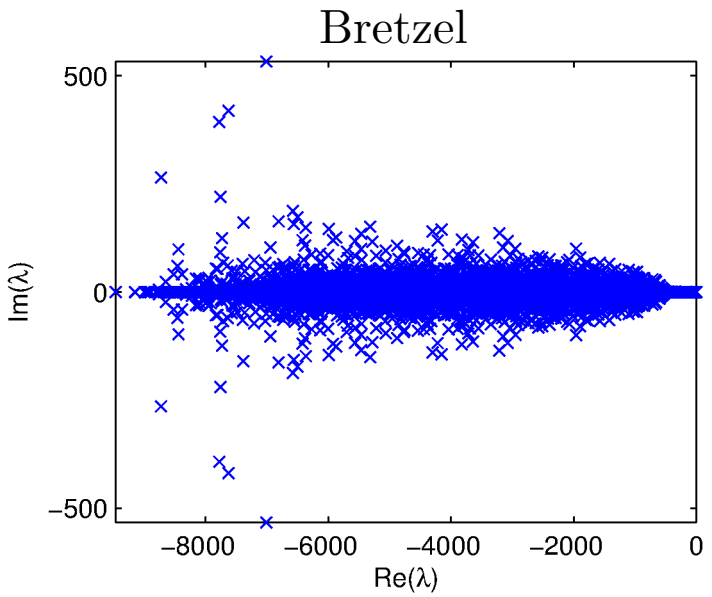
- Necessary condition for time-stability of the numerical scheme:

  All eigenvalues of L_X need to be in the left-half plane

- Our discrete approximation does not guarantee this will be true.
- No proof yet of sufficient conditions for this property.
- Experimental evidence suggests they are, provided the node set X is "sufficiently" dense.

Examples: Plot of the eigenvalues





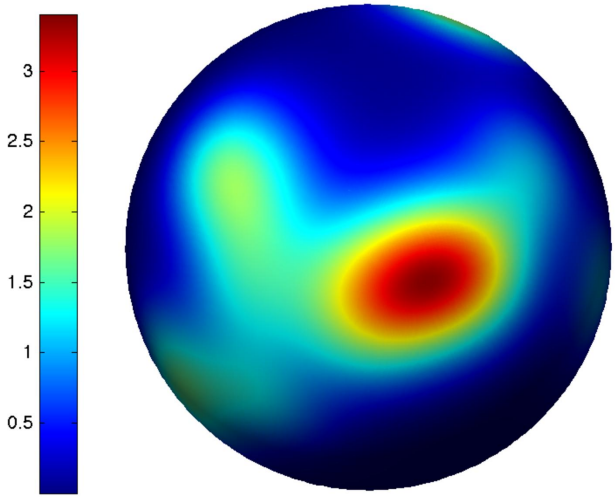
Convergence tests: setup

- Radial kernel: inverse multiquadric (IMQ) $\phi(r) = \dfrac{1}{\sqrt{1+(\varepsilon r)^2}}$
- Time-integrator: BDF4 with Δt=0.02 (chosen for accuracy)
- PDE: Forced diffusion equation with forcing chosen to give known solution:

$$\frac{\partial u}{\partial t} = \Delta_{\mathbb{M}} u + g(t,u)$$

Sphere exact solution:

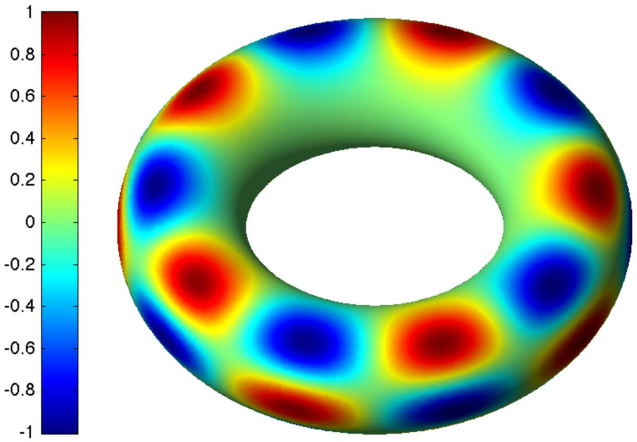
$$u(t,\mathbf{x}) = \exp(-5t) \sum_{k=1}^{23} \exp\left(-10 \arccos(\boldsymbol{\xi}_k \cdot \mathbf{x})\right)$$



Solution at t=0

Torus exact solution:

$$u(t,\mathbf{x}) = \frac{1}{8} e^{-2t} \left(x^5 - 10x^3y^2 + 5xy^4\right)\left(x^2+y^2-60z^2\right)$$



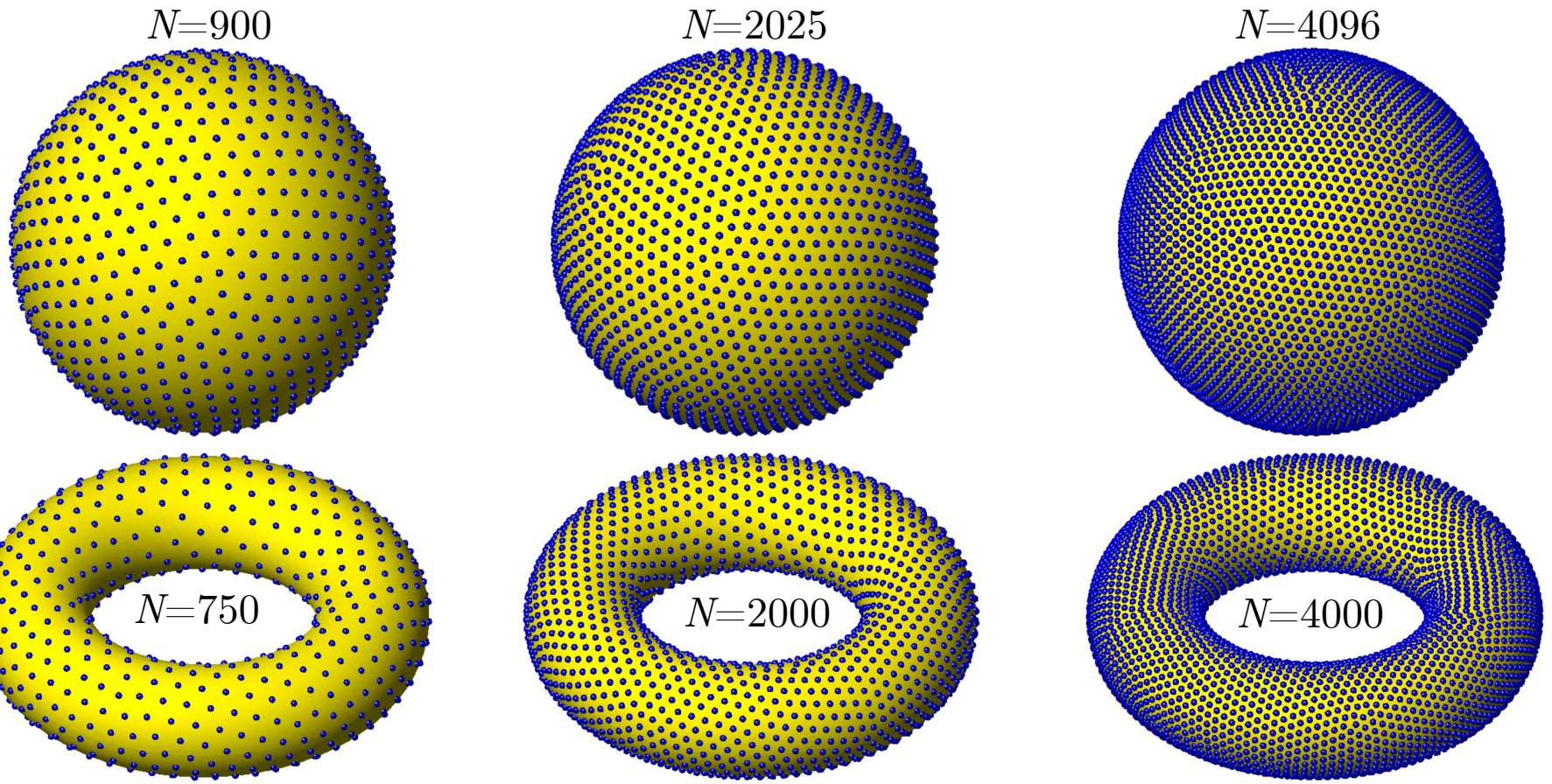
Solution at t=0

Convergence tests: node sets

- We use minimal energy (ME) node sets in the convergence studies.
- These are quasi-uniform distributions of nodes on the surfaces, satisfying:

Mesh norm satisfies: $h_X \sim \dfrac{1}{\sqrt{N}}$

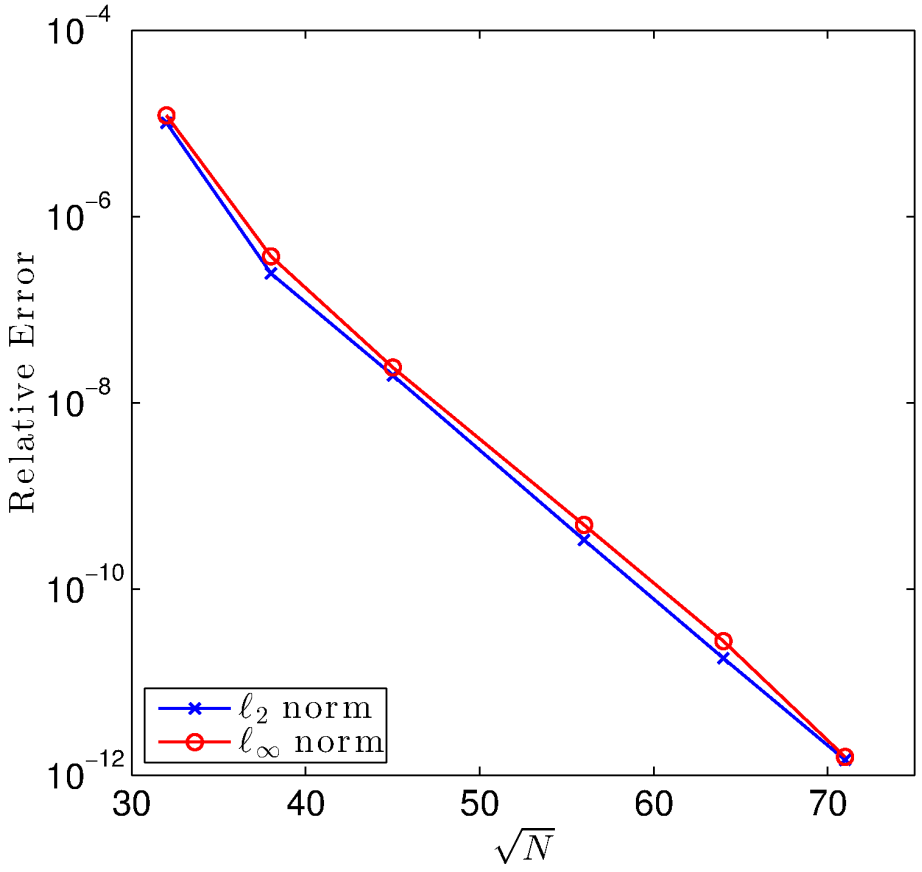
Examples:



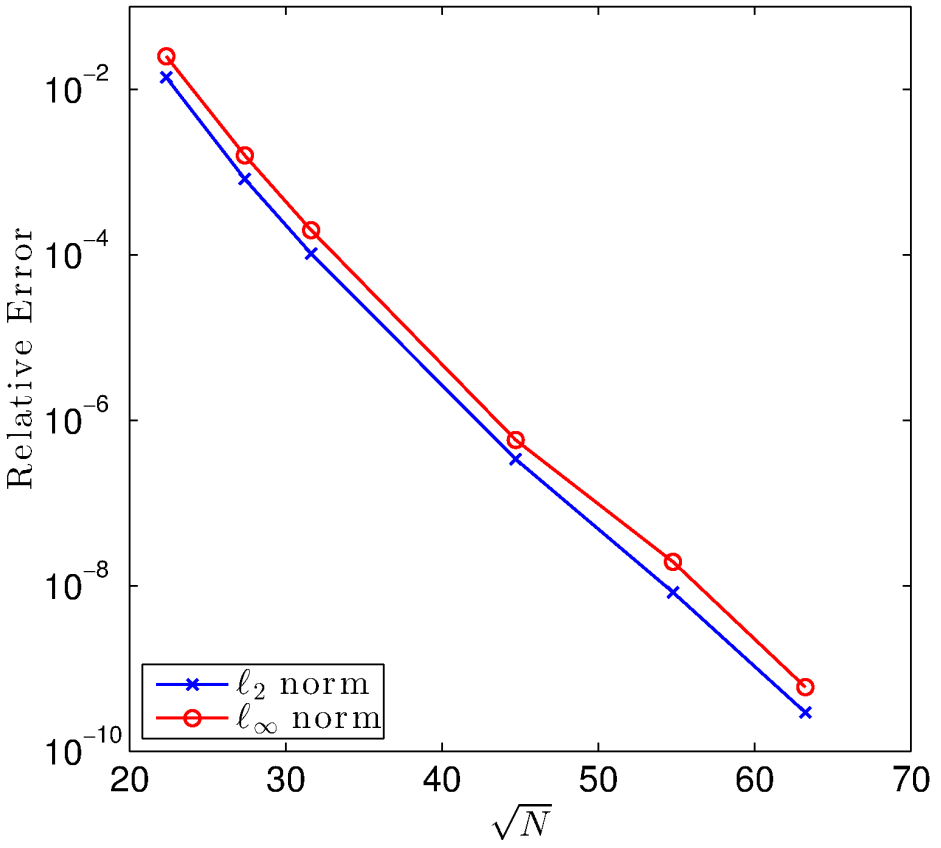
Convergence tests: results

- Semilogy plots of the errors (two-norm and max-norm) vs. $\sqrt{N}$.

Sphere test:



Torus test:



Application: Turing patterns

- Pattern formation via non-linear reaction-diffusion systems; Turing (1952)

  Possible mechanism for animal coat formation (and other morphogenesis phenomena)



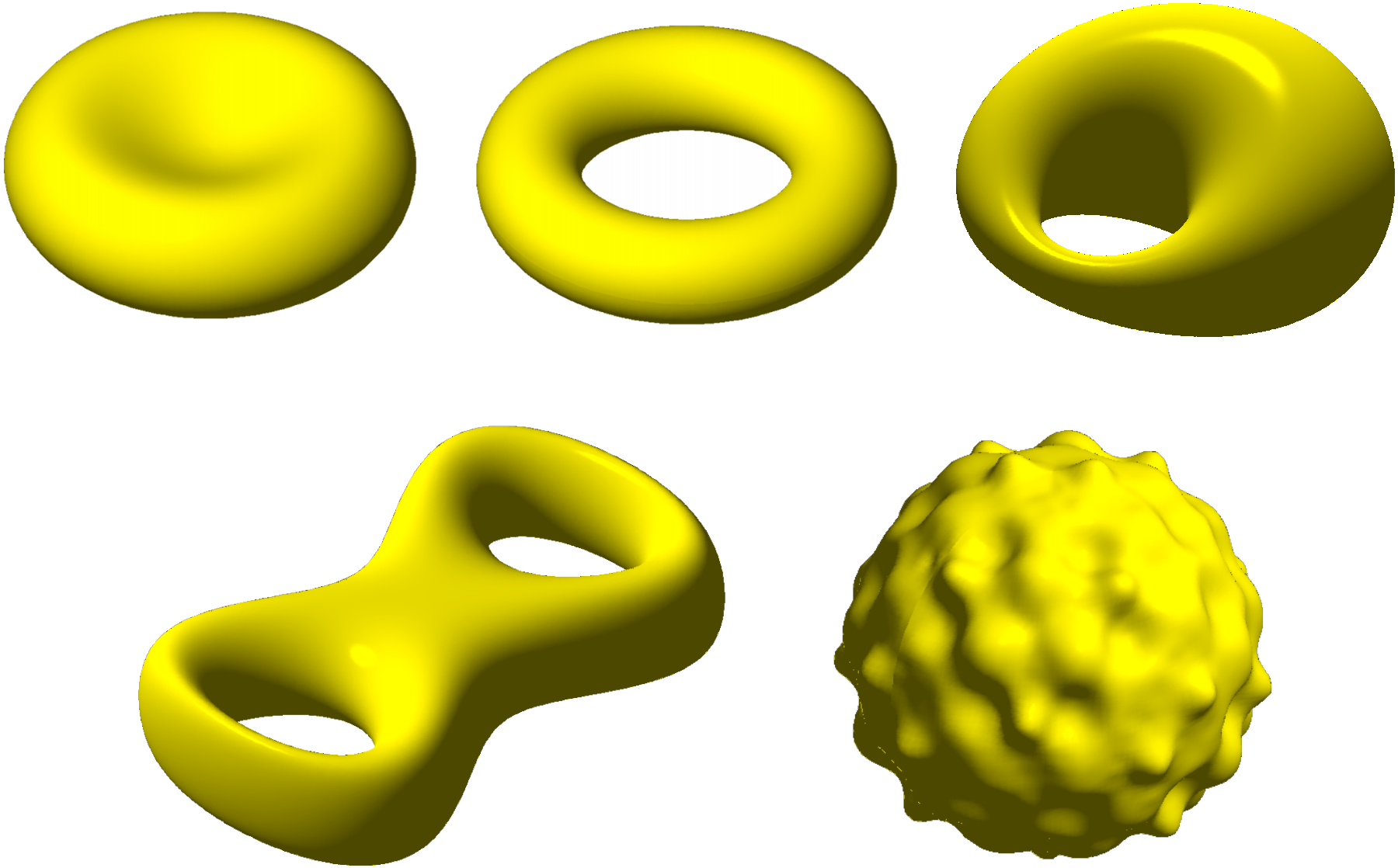
- Example system: Barrio *et al.* (1999)

$$\frac{\partial u}{\partial t} = \delta_u \Delta_{\mathbb{M}} u + \alpha u(1 - \tau_1 v^2) + v(1 - \tau_2 u)$$
$$\frac{\partial v}{\partial t} = \delta_v \Delta_{\mathbb{M}} v + \beta v \left(1 + \frac{\alpha\tau_1}{\beta} uv\right) + u(\gamma + \tau_2 v)$$

- These types of systems have been studied extensively in planar domains.
- Recent studies have focused on the sphere.
- Growing interest in studying these on more general surfaces.

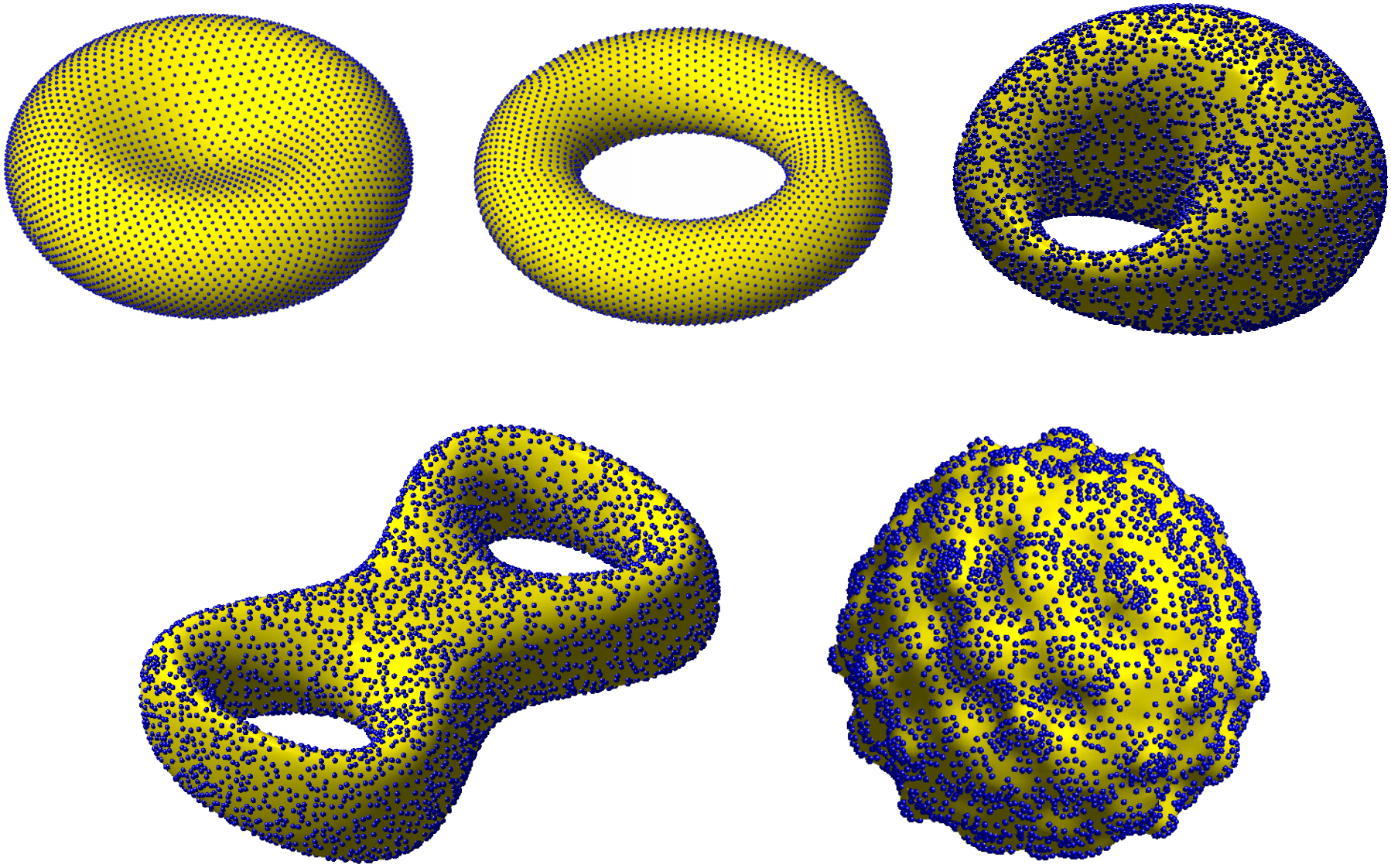
Application: Turing patterns

- Surfaces used in the numerical experiments:



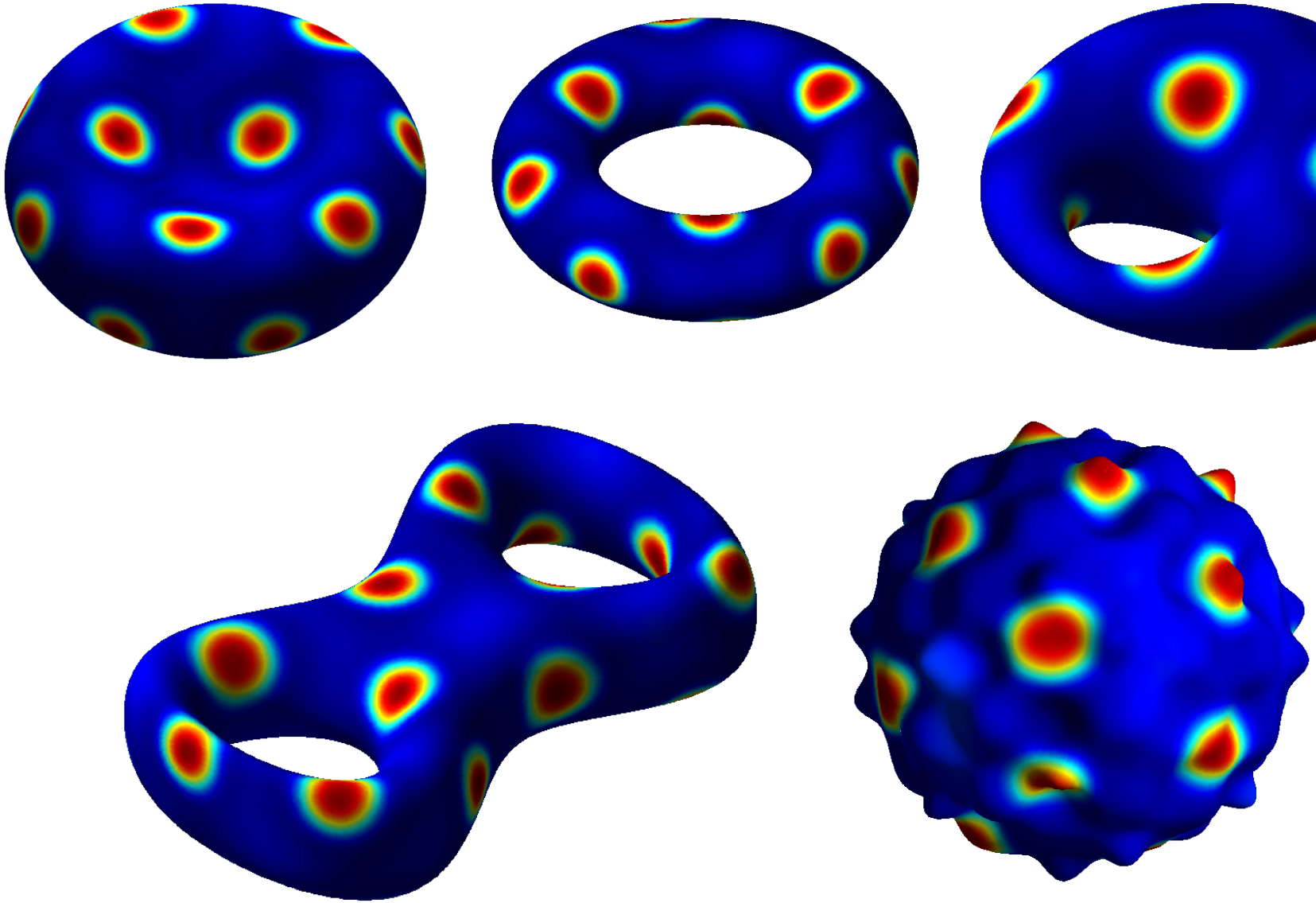
Application: Turing patterns

- Node sets X used in the numerical experiments:



Application: Turing patterns

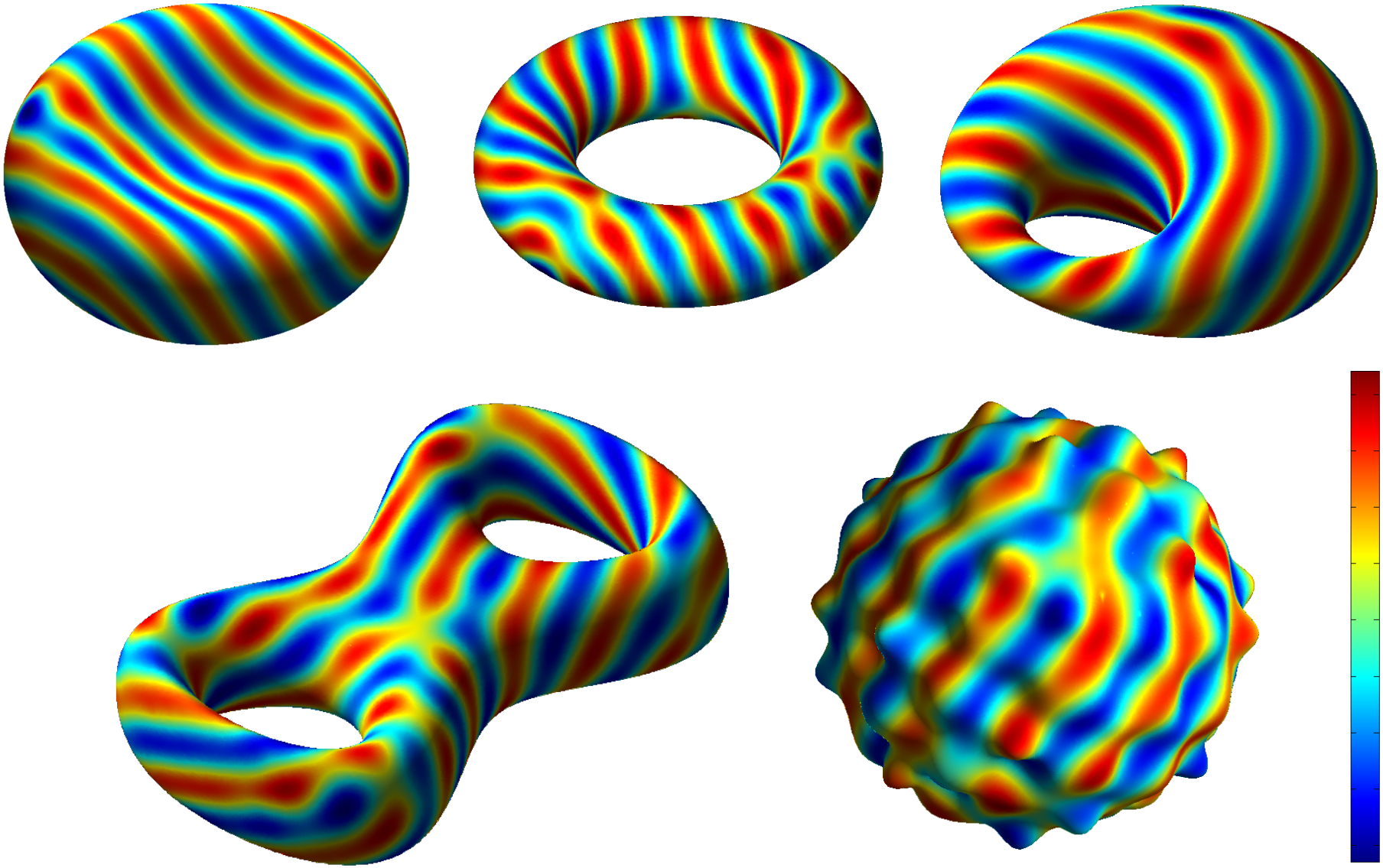
- Numerical solutions: *steady* spot patterns (visualization of u component)



Initial condition: u and v set to random values between +/- 0.5

Application: Turing patterns

- Numerical solutions: *steady* stripe patterns (visualization of u component)



Initial condition: u and v set to random values between +/- 0.5

Application: spiral waves in excitable media

- Example system: Barkley (1991)

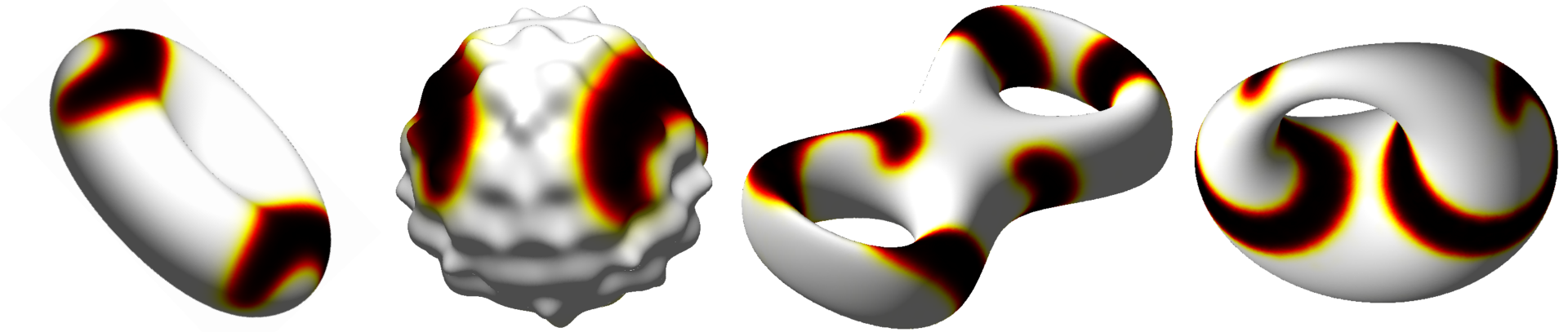
$$\frac{\partial u}{\partial t} = \delta_u \Delta_{\mathbb{M}} u + \frac{1}{\epsilon} u\,(1-u)\left(u - \frac{v+b}{a}\right)$$

$$\frac{\partial v}{\partial t} = \delta_v \Delta_{\mathbb{M}} v + u - v$$

u=activator species
v=inhibitor species

Simplification of FitzHugh-Nagumo model for a spiking neuron.

- Studied extensively on planar regions and somewhat on the sphere.
- Growing interest more physically relevant domains like surfaces.
- Snapshots from different numerical simulations with our method:



visualization of the u (activator) component

Comments on the global RBF method

- Method can be used to approximate surface derivatives in a relatively straightforward manner.
  - These approximations can provide high rates of approximation.
  - Can be used to also solve PDEs to high accuracy.
- Computational cost is quite high:
  - Setup: $\mathcal{O}(N^3)$ operations
  - Each time-step: $\mathcal{O}(N^2)$ operations

Comments on the global RBF method

- Method can be used to approximate surface derivatives in a relatively straightforward manner.
  - These approximations can provide high rates of approximation.
  - Can be used to also solve PDEs to high accuracy.
- Computational cost is quite high:
  - Setup: $\mathcal{O}(N^3)$ operations
  - Each time-step: $\mathcal{O}(N^2)$ operations
- Future: Biological Applications
  - Models on evolving surfaces.
  - Models coupling processes on surfaces to processes in the bulk medium.
  - Models involving feedback between shape of the surface and chemical processes on the surface.
- For these we need to improve the computational cost!
  - Radial basis finite difference formulas (RBF-FD)

RBF generated finite differences (RBF-FD)

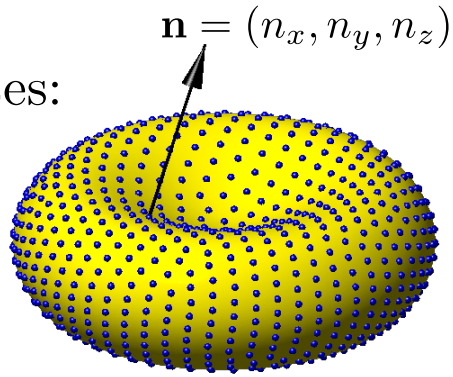
- Generalization of finite-difference (FD) method to scattered nodes using RBFs to compute the FD weights.
  - Tolstykch & Shirobokov (2003)
  - Shu, Ding, & Yeo (2003)
  - W (2003)

- References for RBF-FD on sphere:
  - Fornberg & Lehto (2011)
  - Flyer, Lehto, Blaise, W & St-Cyr (2012)
  - Bollig, Flyer & Erlebacher (2012)
  - Flyer, W & Fornberg (2014)

- References for RBF-FD on more general surfaces:
  - Shankar, W, Kirby, & Fogelson (2014)

Recall: Differential operators on surfaces

- Let $\mathbb{M} \subset \mathbb{R}^3$ be a smooth, compact, two dimensional manifold.
- Surface gradient on $\mathbb{M}$ in *extrinsic* (or Cartesian) coordinates:



$$\nabla_{\mathbb{M}} := \mathbf{P}\nabla = (\mathbf{I} - \mathbf{n} \otimes \mathbf{n})\,\nabla$$

- After some manipulations

$$\nabla_{\mathbb{M}} := \begin{bmatrix} (\mathbf{e}_x \cdot \mathbf{P})\,\nabla \\ (\mathbf{e}_y \cdot \mathbf{P})\,\nabla \\ (\mathbf{e}_z \cdot \mathbf{P})\,\nabla \end{bmatrix} = \begin{bmatrix} (\mathbf{e}_x - n_x\mathbf{n}) \cdot \nabla \\ (\mathbf{e}_y - n_y\mathbf{n}) \cdot \nabla \\ (\mathbf{e}_z - n_z\mathbf{n}) \cdot \nabla \end{bmatrix} = \begin{bmatrix} \mathbf{p}_x \cdot \nabla \\ \mathbf{p}_y \cdot \nabla \\ \mathbf{p}_z \cdot \nabla \end{bmatrix} = \begin{bmatrix} \mathcal{G}^x \\ \mathcal{G}^y \\ \mathcal{G}^z \end{bmatrix}$$

- Surface divergence of smooth vector field $\mathbf{f} : \mathbb{R}^3 \to \mathbb{R}^3$ ($\mathbf{f} = (f_x, f_y, f_z)$):

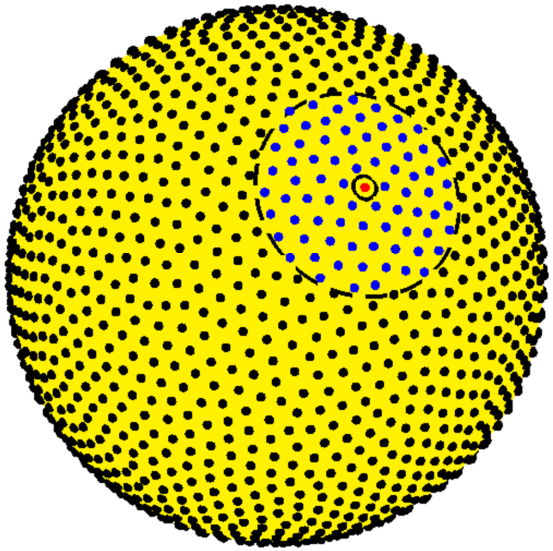
$$\nabla_{\mathbb{M}} \cdot \mathbf{f} := (\mathbf{P}\nabla) \cdot \mathbf{f} = \mathcal{G}^x f_x + \mathcal{G}^y f_y + \mathcal{G}^z f_z$$

- Laplace-Beltrami operator on $\mathbb{M}$ in *extrinsic* coordinates:

$$\Delta_{\mathbb{M}} := (\mathbf{P}\nabla)\cdot(\mathbf{P}\nabla) = \mathcal{G}^x\mathcal{G}^x + \mathcal{G}^y\mathcal{G}^y + \mathcal{G}^z\mathcal{G}^z = \mathcal{D}_{xx} + \mathcal{D}_{yy} + \mathcal{D}_{zz}$$

RBF generated finite differences (RBF-FD)

- Consider $X = \{\mathbf{x}_j\}_{j=1}^{N} \subset \mathbb{M}$, where $\mathbf{x}_j = (x_j, y_j, z_j)$:



Key Steps:

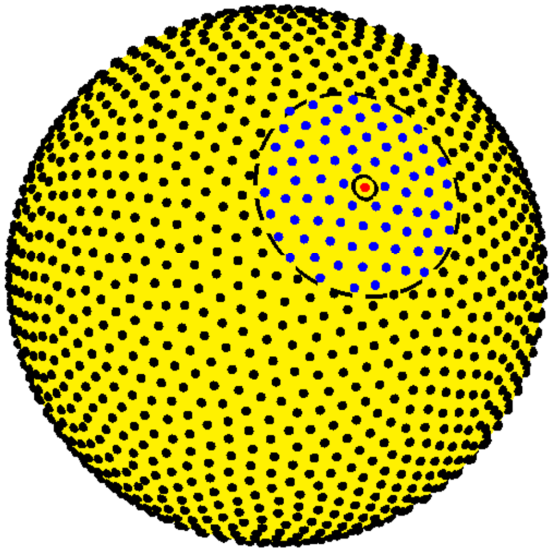
1. For each node $\mathbf{x}_j$, choose $n-1$ of it's nearest neighbors: $X_j = \{\mathbf{x}_i\}_{i=1}^{n}$, with $\mathbf{x}_1 = \mathbf{x}_j$.

2. Approximate $(\Delta_{\mathbb{M}} f)\big|_{\mathbf{x}_j}$ using linear a combination of the values of f sampled at X_j:

$$(\Delta_{\mathbb{M}} f)\Big|_{\mathbf{x}_j} \approx \sum_{i=1}^{n} c_i^j f(\mathbf{x}_i)$$

RBF generated finite differences (RBF-FD)

- Consider $X = \{\mathbf{x}_j\}_{j=1}^{N} \subset \mathbb{M}$, where $\mathbf{x}_j = (x_j, y_j, z_j)$:



Key Steps:

2(a) Compute discrete approximations to $\mathcal{G}^x$, $\mathcal{G}^y$, $\mathcal{G}^z$ locally on X_j:
$G^x_{X_j}$, $G^y_{X_j}$, $G^z_{X_j}$

2(b) Approximate $(\Delta_{\mathbb{M}} f)\big|_{X_j}$ using $G^x_{X_j}$, $G^y_{X_j}$, $G^z_{X_j}$:

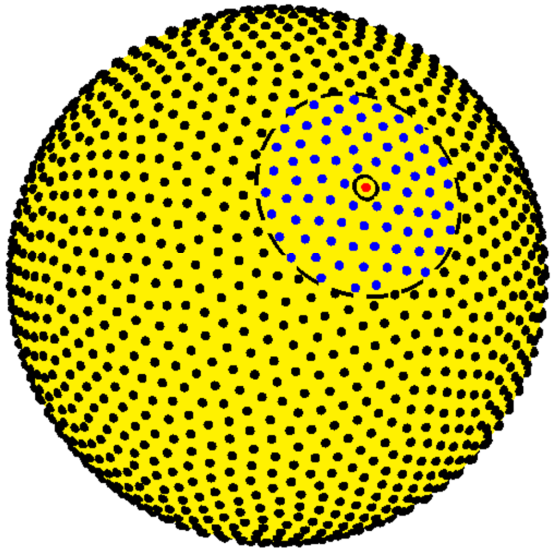
$$(\Delta_{\mathbb{M}} f)\big|_{X_j} \approx [G^x_{X_j} G^x_{X_j} + G^y_{X_j} G^y_{X_j} + G^z_{X_j} G^z_{X_j}] \underline{f}_j = L_{X_j} \underline{f}_j$$

2(c) First row of L_{X_j} contains weights for approximating $(\Delta_{\mathbb{M}} f)$ at $\mathbf{x} = \mathbf{x}_j$:

$$\begin{bmatrix} c_1^j & \cdots & c_n^j \end{bmatrix} = L_{X_j}(1, 1:n)$$

RBF generated finite differences (RBF-FD)

- Consider $X = \{\mathbf{x}_j\}_{j=1}^N \subset \mathbb{M}$, where $\mathbf{x}_j = (x_j, y_j, z_j)$:



Key Steps:

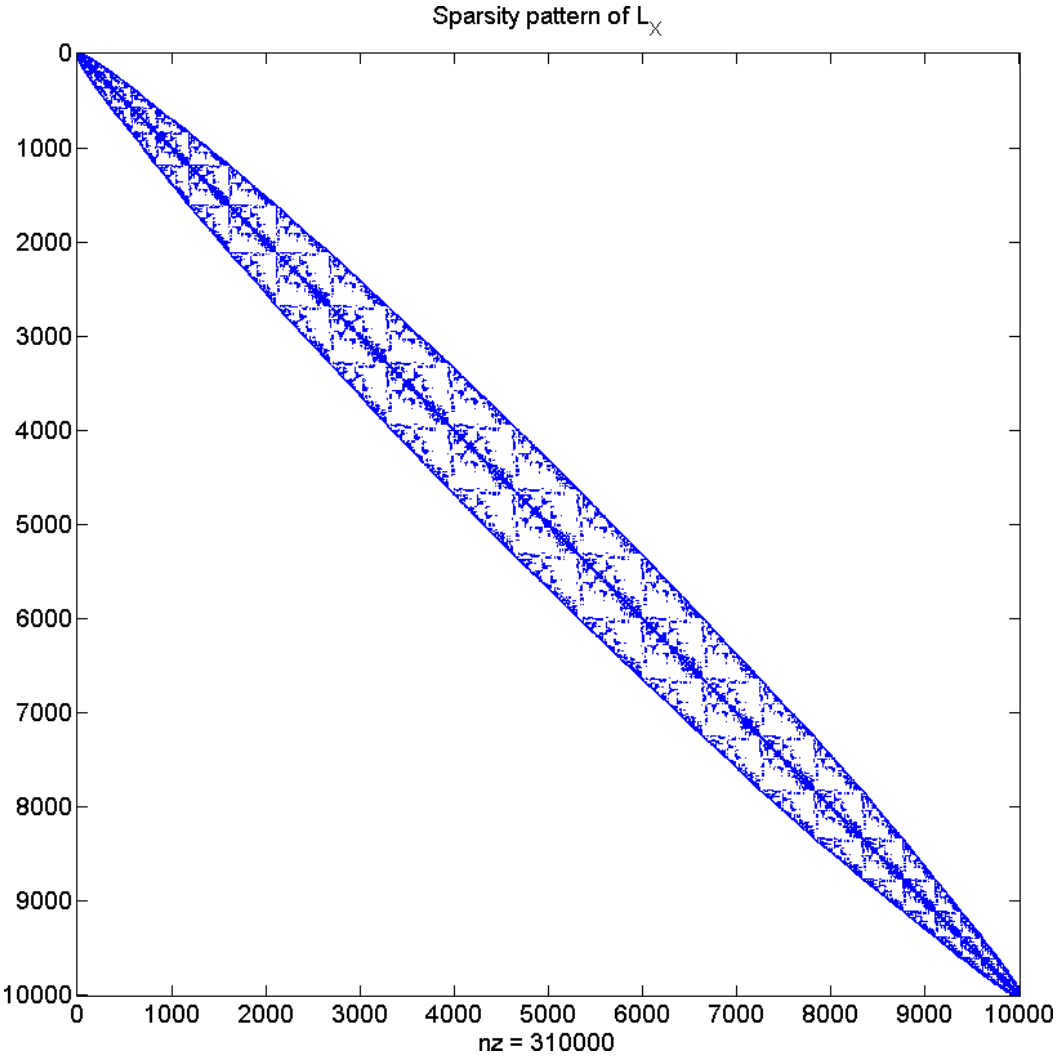
3. Assemble all the weights into a *sparse* N-by-N differentiation matrix L_X.

4. Combine with the method-of-lines for numerically solving time-dependent PDEs:

$$\frac{\partial u}{\partial t} = \delta \Delta_{\mathbb{M}} u + g(t,u) \xrightarrow{\text{MOL}} \frac{d}{dt}\underline{u} = \delta L_X \underline{u} + g(t,\underline{u})$$

RBF generated finite differences (RBF-FD)

- Example differentiation matrix for N=10,000, n=31:



0.31% of entries are non-zero

- Compare to the global RBF method, which results in a dense differentiation matrix.

Some details of the RBF-FD approach

- Comparison of computational cost associated with different approaches for approximating surface Laplacian

Differentiation Matrix L_x	Global RBF	RBF-FD*
Construction:	$O(N^3)$	$O(n^3 N) + O(N \log N)$
Evaluation:	$O(N^2)$	$O(nN)$

- Computing the RBF-FD weights is an embarrassingly parallel problem.

Some details of the RBF-FD approach

- Eigenvalue stability for RBF-FD
  - Non-uniform node sets can lead to eigenvalues of L_X with positive real-part.
  - Optimizing the RBF shape parameter ε per RBF-FD stencil appears to be a robust way to fix this problem:

$$\left(\mathcal{G}^x[I_X f]\right)\big|_{X_j} = \underbrace{B^x_{X_j} A^{-1}_{X_j}}_{G^x_{X_j}} \underline{f}, \quad \text{where } (B^x_{X_j})_{i,k} = \left(\mathcal{G}^x \phi(\|\mathbf{x} - \mathbf{x}_k\|)\right)\big|_{\mathbf{x}=\mathbf{x}_i}$$

  - Our strategy: choose ε so that all A_{X_j} have roughly the same condition number.
  - Requires solving small (non-linear) optimization problem for each stencil.

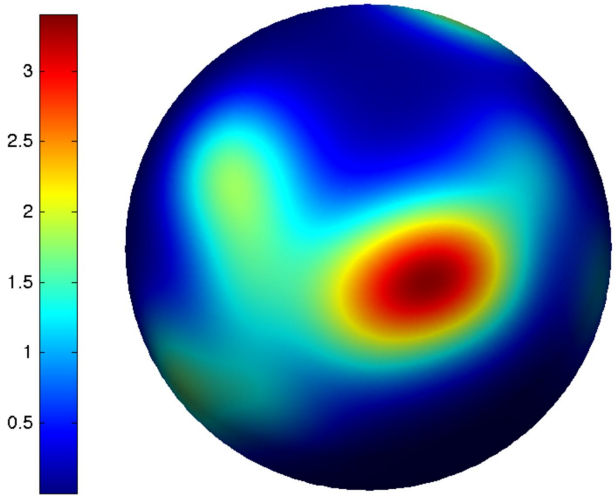
Convergence tests: setup

- Radial kernel: inverse multiquadric (IMQ) $\phi(r) = \dfrac{1}{\sqrt{1+(\varepsilon r)^2}}$
- Time-integrator: BDF4 with Δt=0.05 (chosen for accuracy)
- PDE: Forced diffusion equation with forcing chosen to give known solution:

$$\frac{\partial u}{\partial t} = \Delta_{\mathbb{M}} u + g(t,u)$$

Sphere exact solution:

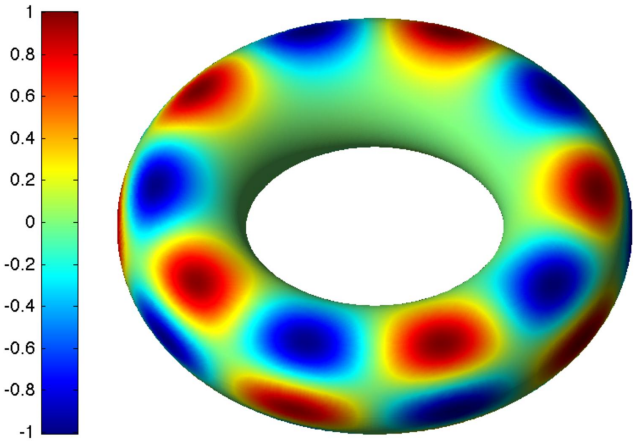
$$u(t,\mathbf{x}) = \exp(-5t) \sum_{k=1}^{23} \exp\left(-10 \arccos(\boldsymbol{\xi}_k \cdot \mathbf{x})\right)$$



Solution at t=0

Torus exact solution:

$$u(t,\mathbf{x}) = \frac{1}{8} e^{-2t} \left(x^5 - 10x^3y^2 + 5xy^4\right)\left(x^2+y^2-60z^2\right)$$

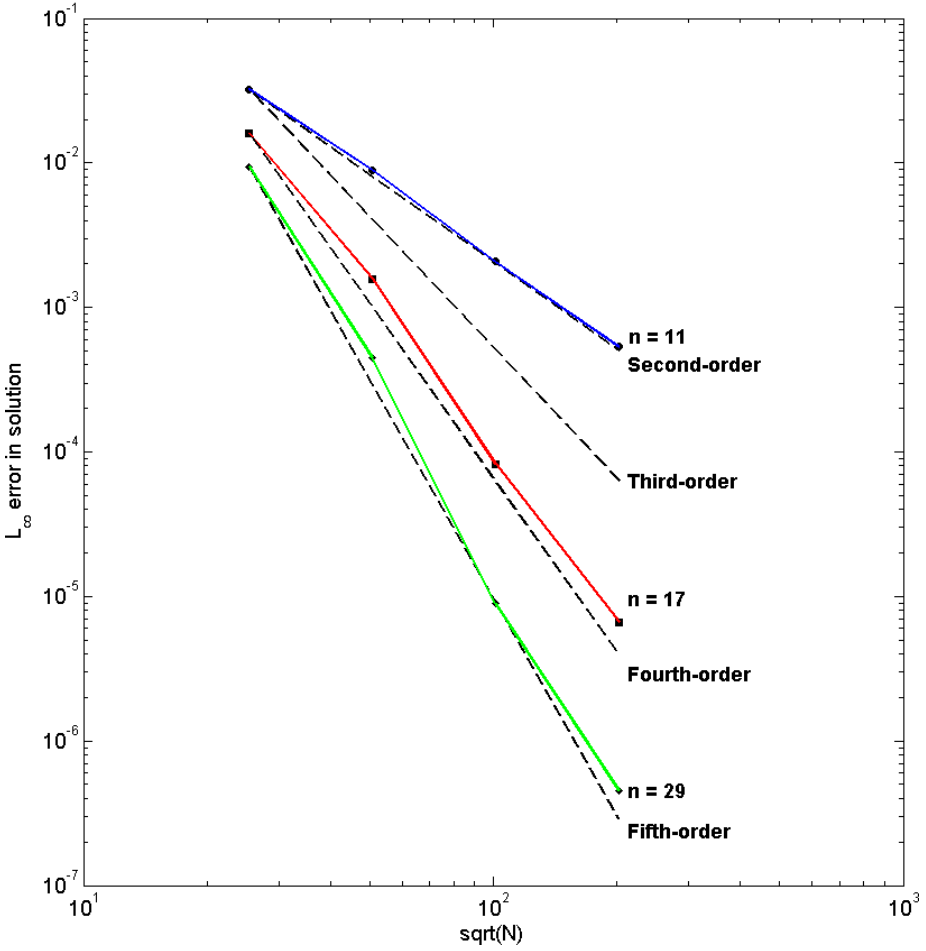


Solution at t=0

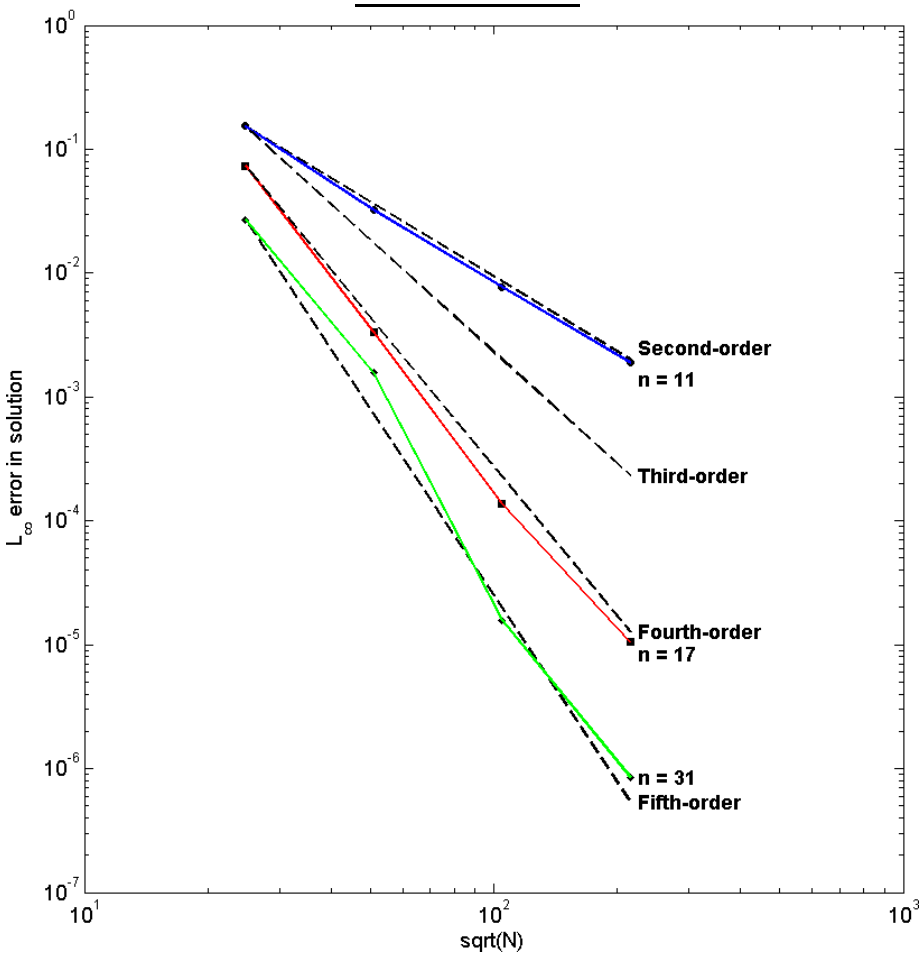
Convergence tests: results

- Log-log plots of the relative max-norm errors vs. $\sqrt{N}$.
- Colored curves correspond to different stencil sizes n.
- Dashed lines indicate various convergence rates: $C(\sqrt{N})^{-k}$

Sphere test:

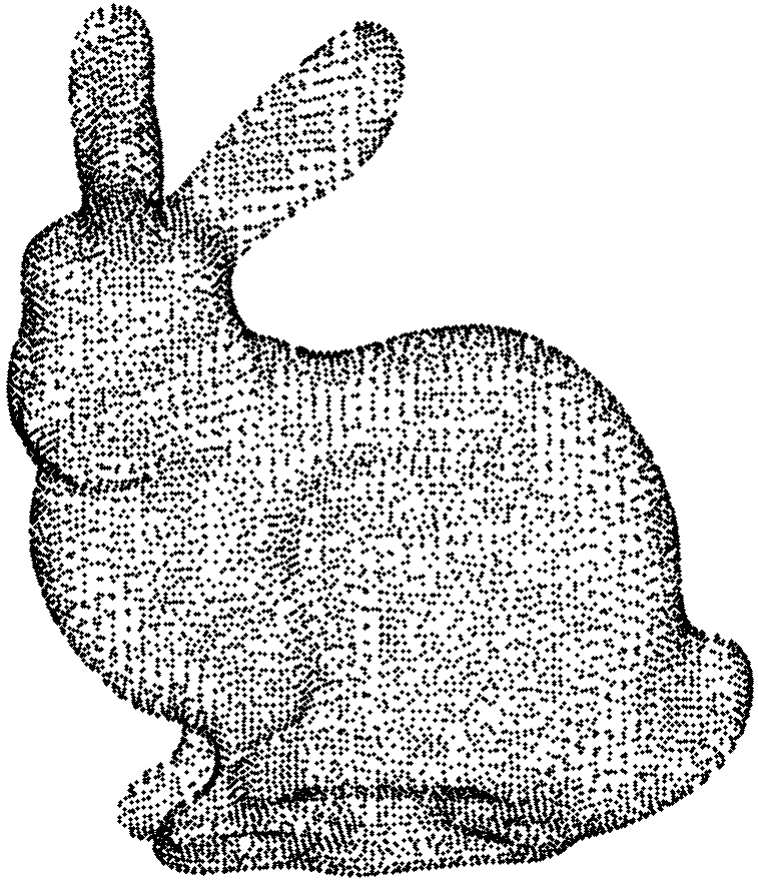


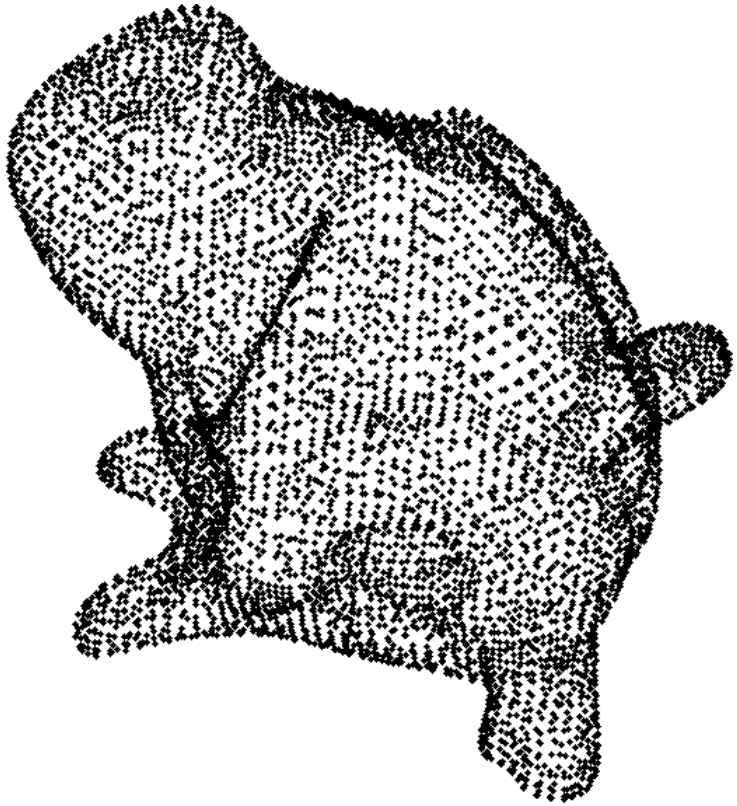
Torus test:



Application: Turing patterns

- Same model as the global RBF problem.
- Nodes given by point clouds on the surface.
- Surface normals computed from MeshLab software



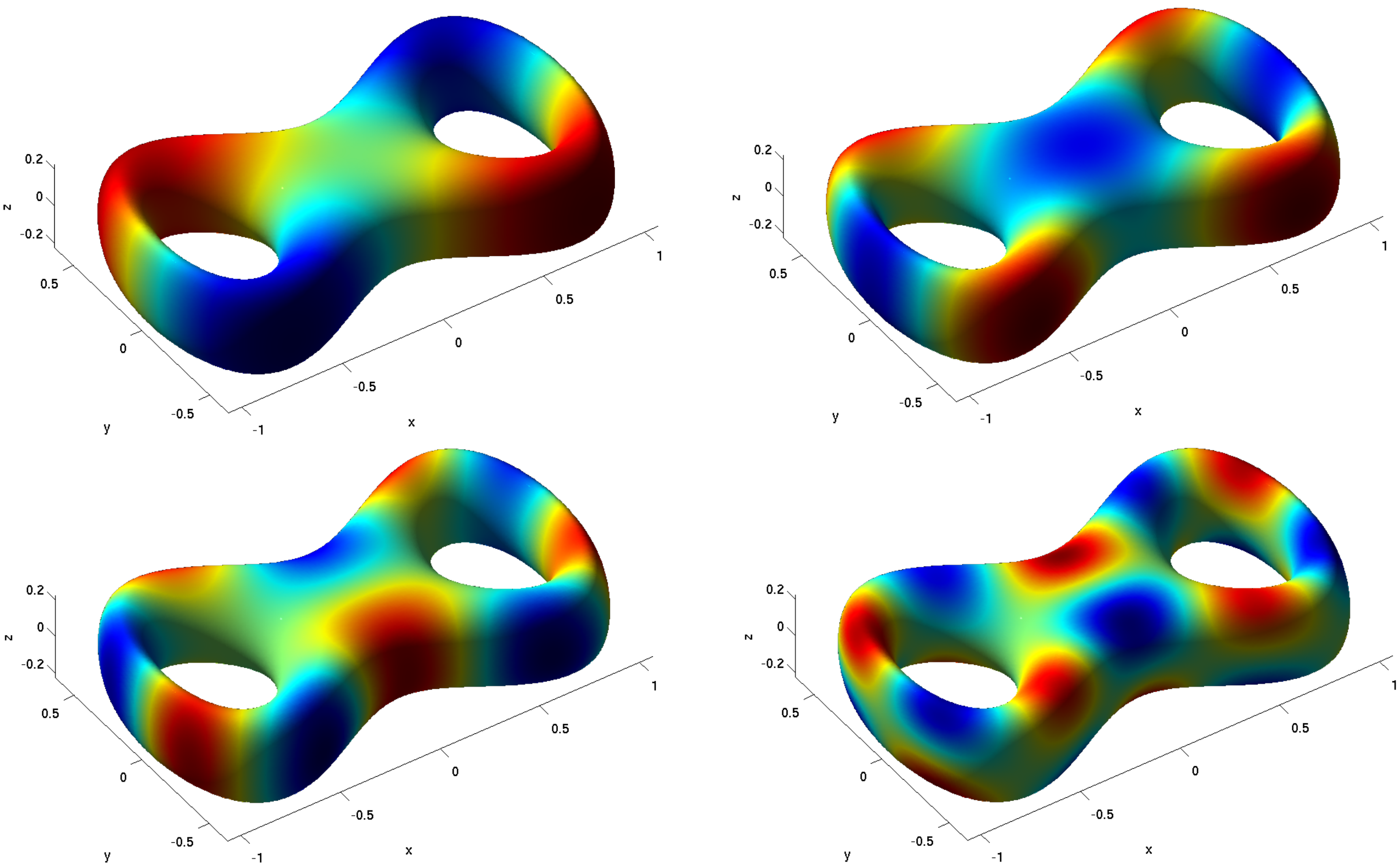


Application: Turing patterns

- Examples of spot and stripe patterns.

Application: surface harmonics

- Approximations to eigenfunctions of surface Laplacian.



Can one hear the shape of a bretzel?

Concluding remarks

- Computational cost of the RBF-FD is $\mathcal{O}(N)$.
- RBF-FD maintains the flexibility of the global method, albeit at the cost of reduced accuracy.
- Allows larger point clouds and more complicated surfaces can be handled.
- Future: algorithmic developments
  - Surfaces with boundaries
  - "Cheap" methods for approximating surface normal vectors
  - Anisotropic diffusion
  - Addition of advection
  - Stable algorithms for handling small shape parameters.
- Future: Biological Applications
  - Models on evolving surfaces.
  - Models coupling processes on surfaces to processes in the bulk medium.
  - Models involving feedback between shape of the surface and chemical processes on the surface.