Untangling Knots Through Curve Repulsion

Candidate Number: (Redacted)
Honour School of Mathematics (Part C)
University of Oxford
Trinity Term 2023

Untangling Knots Through Curve Repulsion

Abstract

Curves are one of the fundamental objects in geometry and engineering, yet most analysis of curves often disregard their physical characteristics such as their spatial volume or uncrossability. One common situation that such physical characteristics become significant is when one attempts to untangle a knot. An approach to achieve this is to assign an "energy" to a curve such that this energy would increase when two points on "different sides" of a curve are closer, then one continuously deforms the curve to reduce this energy, the expectation being that the curve that achieves minimal energy must be the untangled knot. This dissertation explores numerical methods of achieving this.

Contents

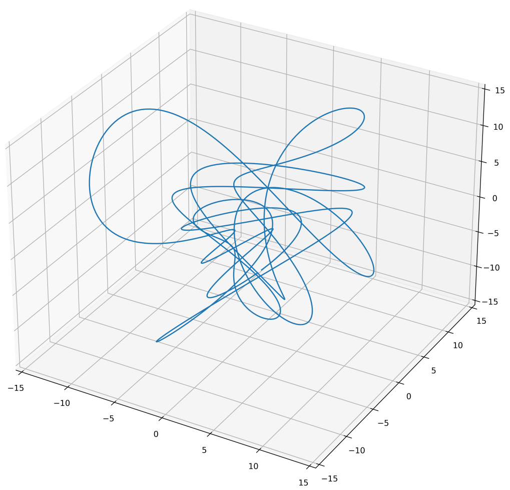


Figure 1.1: A tangled curve in $\mathbb{R}^3$

1 Introduction

Shape optimisation is an important idea in engineering, relevant in anywhere from aircraft designs to packaging ramen noodle. One of the simplest and most fundamental shapes to consider is a curve. While curves are simple objects in theory, they prove to be quite difficult to analyse in practice with realistic physics. Even in absence of other objects, one must consider resiliance to bending, stretching, and especially, impenetrability against itself. With these physical factors in mind, untangling a knot like the one shown (Figure 1.1) becomes a very complicated process, especially in a computer simulation. In this dissertation, we explore numerical methods to achieve this.

The main idea is to *assign energy that penalises "physical entanglement"*. Given a parameterised curve $\boldsymbol{\gamma} : M \to \mathbb{R}^3$ (M being the domain of the parameter, often an interval), one defines some *curve energy* $\mathcal{E}$ of the form:

$$\mathcal{E}(\boldsymbol{\gamma}) \coloneqq \iint_{M^2} k\left(\boldsymbol{\gamma}_x, \boldsymbol{\gamma}_y\right) \mathrm{d}\gamma_x \, \mathrm{d}\gamma_y \tag{1.1}$$

where $k(\boldsymbol{\gamma}_1, \boldsymbol{\gamma}_2) \geq 0$ is the *curve energy kernel* such that $k \to +\infty$ as $|\boldsymbol{\gamma}_1 - \boldsymbol{\gamma}_2| \to 0^+$.

A naïve choice of k satisfying this condition is $k_S(\boldsymbol{\gamma}_1, \boldsymbol{\gamma}_2) \coloneqq \frac{1}{|\boldsymbol{\gamma}_1 - \boldsymbol{\gamma}_2|}$. However, it turns out that $k_S \sim O\left(\frac{1}{|\boldsymbol{\gamma}_1 - \boldsymbol{\gamma}_2|}\right)$ as $|\boldsymbol{\gamma}_1 - \boldsymbol{\gamma}_2| \to 0$ (consider neighbouring points), meaning the $\mathcal{E}$ diverges all continuous curves $\boldsymbol{\gamma}$ of nonzero measure.

A more analytically sensible choice of k would be the tangent-point kernel introduced in a paper by Buck and Orloff[2] and later generalised by Yu, Schumacher, and Crane[5].

The next part of the idea is to *reduce $\mathcal{E}$ by continuously deforming the curve* based on a descent method until it reaches a stationary curve, at which, we expect it to be the "unknot" of the original curve. Note that by construction of $\mathcal{E}$, if the curve is to self-intersect, $\mathcal{E}$ increases, and the descent method encourages the curve to repel, preventing the self-intersection.

2 Gradient Flow Equation

Since we pose the problem as continuous reduction of some functional, we need an applicable framework which fits our intention. In our case, **gradient flow equation** seems to be appropriate.

2.1 Motivation of Gradient Flow Equation

For minimising a differentiable function $f : E \subset \mathbb{R}^n \to \mathbb{R}$, there is a well-known method known as **steepest descent method** (SDM)[1].

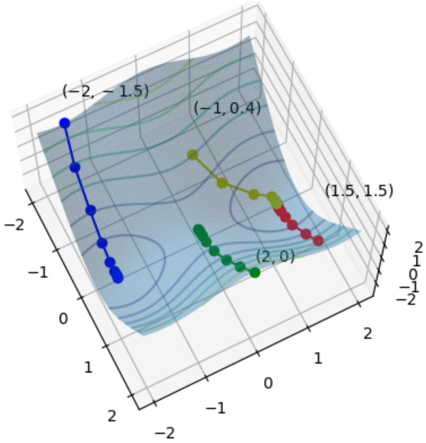
$$\mathbf{x}^{k+1} = \mathbf{x}^k - \alpha_k \nabla f\left(\mathbf{x}^k\right) \tag{2.1}$$

Starting from the initial input point $\mathbf{x}^0$, at each iteration, input points $\left\{\mathbf{x}^k\right\}$ move in the direction of "steepest" decrease, with specified step size $\alpha_k > 0$, reducing the value at evaluation of f. Note that in general, this method is not guaranteed to find the minimiser as shown in Figure 2.1(a). On the other hand, convergence is guaranteed under certain assumptions, for example, convexity and L-smoothness[1] with a certain choice of step size α_k.

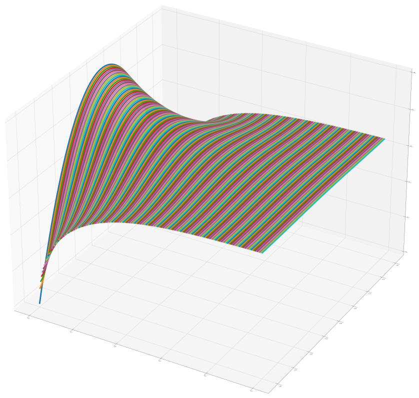
Analogously, differential equation known describing the reduction process of a functional $F : \mathcal{X} \to \mathbb{R}$ (where $\mathcal{X}$ is an inner product function space) can be motivated. Starting from (2.1), replacing $\mathbf{x}^k$ by f_k and $\nabla f\left(\mathbf{x}^k\right)$ by $\operatorname{grad}_{\mathcal{X}} F(f_k)$

$$f_{k+1} = f_k - \alpha_k \operatorname{grad}_{\mathcal{X}} F(f_k) \tag{2.2}$$

[1] $||\nabla f(\mathbf{x}) - \nabla f(\mathbf{y})|| \leq L||\mathbf{x} - \mathbf{y}||$ for some constant $L > 0$



(a) SDM applied to $f(x, y) = -3\cos x + \cos^2 y$ at different initial points.



(b) Solution to L^2 gradient flow equation with 1D energy functional $\mathcal{E}(f) \coloneqq \int_{-1}^{1} |\nabla f(x)|^2 \, dx$, which penealises variation in function. Note that the solution converges to a function with no variation.

Figure 2.1: Gradient flow can be understood as a continuous analogue of steepest descent.

Now think of f_k as "snapshots" at certain time $t = t_k$. Without loss of generality, let $\alpha_k \equiv 1$.[2] Dividing (2.2) by time step $\Delta t \coloneqq t_{k+1} - t_k$, and taking the limit as $\Delta t \to 0$, we acquire the **gradient flow equation**[5].

$$\frac{\partial f}{\partial t} = -\operatorname{grad}_{\mathcal{X}} F(f) \tag{2.3}$$

where index k transforms to "time" variable t.

Note that grad of a functional is not defined yet. This depends on the inner product function space $\mathcal{X}$ (eg. L^2, H^1, $\ldots$) of interest.

2.2 Gradient of Functional

In order to understand the gradient of a functional, it helps to recall the gradient of a function. Gradient of a function $f : \mathbb{R}^n \to \mathbb{R}$ characterises the first-order variation in a certain direction, that is,

$$\forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^n \quad \nabla f(\mathbf{x}) \cdot \mathbf{y} = \frac{\partial}{\partial \epsilon} f(\mathbf{x} + \epsilon \mathbf{y}) |_{\epsilon=0} \tag{2.4}$$

While this is not a conventional definition of function gradient, it is still an equivalent definition.

One could analogously construct the definition of gradient of a functional.

[2] This is justified by taking a differrent time scale; essentially nondimensionalisation. Also $\alpha_k \sim O(1)$ as $k \to \infty$ (eg. $\alpha = L^{-1}$ for L-smooth optimisation) is in fact a realistic choice.

Definition (Gradient of Functional)**.** For a functional $F : \mathcal{X} \to \mathbb{R}$, define functional gradient $\operatorname{grad}_{\mathcal{X}} F(f)$ as:

$$\forall f, g \in \mathcal{X} \quad \langle \operatorname{grad}_{\mathcal{X}} F(f), g \rangle_{\mathcal{X}} = \left. \frac{\partial}{\partial \epsilon} F\left(f + \epsilon g\right) \right|_{\epsilon=0} \tag{2.5}$$

where $\langle \cdot, \cdot \rangle_{\mathcal{X}}$ is the inner product defined over the inner product function space $\mathcal{X}$.

Common inner product spaces include L^2, H^1, which are defined in Appendix A.

Remark. By assuming time-independence of f in the gradient flow equation (2.3), we get a stationary state equation $\operatorname{grad}_{\mathcal{X}} F(f) = 0$, which implies $\frac{\partial}{\partial \epsilon} F\left(f + \epsilon g\right)|_{\epsilon=0} = 0$ for all $g \in \mathcal{X}$. This shows that the stationary state of the gradient flow equation is precisely the solution to the Euler-Lagrange equation which is expected.

2.2.1 Gradient on Integer Sobolev Spaces

Assume Ω is a boundary-free domain. Given $L^2(\Omega) = H^0(\Omega)$ gradient (Ω will be omitted in this section when unambiguous), one could express Sobolev gradients of other orders. For functional F, write $h \coloneqq \operatorname{grad}_{L^2} F$ for gradient of F in L^2, that is, by definition

$$\left. \frac{\partial}{\partial \epsilon} F\left(f + \epsilon g\right) \right|_{\epsilon=0} = \langle h, g \rangle_{L^2} \eqqcolon \mathcal{P} \tag{2.6}$$

By lemma B.1 and lemma B.2 on boundary-free domain Ω,[3]

$$\begin{aligned} \mathcal{P} &= \langle \Delta \Delta^{-1} h, g \rangle_{L^2} \\ &= \langle -\nabla \Delta^{-1} h, \nabla g \rangle_{L^2} \\ &= \langle -\Delta^{-1} h, g \rangle_{H^1} \end{aligned}$$

So one could spot that

$$\operatorname{grad}_{H^1} F = -\Delta^{-1} h \tag{2.7}$$

Remark. However, there is a subtlety here. Because inverting a Laplacian requires additional information (often about the boundary, even though we assume it was a boundary-free domain), Δ^{-1} can be problematic. One could resolve this by replacing Δ^{-1} by $(\Delta + c\,\mathrm{Id})^{-1}$ where $c \neq 0$ is a constant and Id is the identity operator. One justifies this replacement by noting the fact that H^1 gradient is only defined up to addition of null space of the Laplacian[4], and hence addition of $c\,\mathrm{Id}$ can be absorbed. Note that $(\Delta + c\,\mathrm{Id})$ is an invertible operator now.

Similarly, one could acquire gradient in H^2 by

$$\begin{aligned} \mathcal{P} &= \langle \Delta^{-1} h, \Delta g \rangle_{L^2} \\ &= \langle \Delta \left(\Delta^{-2} h\right), \Delta g \rangle_{L^2} \\ &= \langle \Delta^{-2} h, g \rangle_{H^2} \end{aligned}$$

[3] Integration by parts to "shift" operators to other function.
[4] "Harmonic functions"

Hence,

$$\operatorname{grad}_{H^2} F = \Delta^{-2} h \tag{2.8}$$

One could take it even further and define gradient in H^{-1},

$$\begin{aligned}\mathcal{P} &= \langle h, \Delta\Delta^{-1} g\rangle_{L^2} \\ &= \langle -\nabla h, \nabla\left(\Delta^{-1} g\right)\rangle_{L^2} \\ &= \langle -\Delta h, g\rangle_{H^{-1}}\end{aligned}$$

So,

$$\operatorname{grad}_{H^{-1}} F = -\Delta h \tag{2.9}$$

Remark. From (2.7), (2.8), and (2.9), one may deduce that choosing the right Sobolev space for a functional may make the gradient flow equation much easier to solve.

Example (Dirichlet Energy)**.** For example, given functional $\mathcal{E}(f) \coloneqq \int_\Omega |\nabla f(\mathbf{x})|^2 \,\mathrm{d}V$ over an open set $\Omega \subset \mathbb{R}^n$, one can explicitly deduce L^2 gradient from definition (2.5):

$$\begin{aligned}\langle \operatorname{grad}_{L^2} \mathcal{E}(f), g\rangle_{L^2} &= \frac{\partial}{\partial \epsilon} \mathcal{E}\left(f + \epsilon g\right)\bigg|_{\epsilon=0} \\ &= \frac{\partial}{\partial \epsilon} \int_\Omega \left(\nabla f + \epsilon \nabla g\right) \cdot \left(\nabla f + \epsilon \nabla g\right)\,\mathrm{d}V\bigg|_{\epsilon=0} \\ &= \int_\Omega \nabla f \cdot \nabla g \,\mathrm{d}V \\ &\overset{\text{IBP}}{=} \underbrace{\oint_{\partial\Omega} g \frac{\partial f}{\partial n}\,\mathrm{d}S}_{\text{Boundary Term}} - \int_\Omega g \Delta f \,\mathrm{d}V \\ &= \underbrace{\oint_{\partial\Omega} g \frac{\partial f}{\partial n}\,\mathrm{d}S}_{\text{Boundary Term}} + \langle -\Delta f, g\rangle_{L^2}\end{aligned}$$

Assuming natural boundary condition $\frac{\partial f}{\partial n} = 0$, or restricting our interest to periodic functions such that there is no boundary, we may set the boundary term to zero. We can then read off L^2 gradient to be $\operatorname{grad}_{L^2} \mathcal{E}(f) = h = -\Delta f$. The gradient flow equation turns out to be the heat equation:

$$\frac{\partial f}{\partial t} = \Delta f \tag{2.10}$$

(Refer to Figure 2.1(b) for the case $\Omega = (-1, 1)$.) However, note that one could "reduce the order" or the RHS by considering the gradient flow in H^1. Observe from (2.7) that $\operatorname{grad}_{H^1} \mathcal{E}(f) = \mathrm{Id}_\Omega$, where Id_Ω is the identity function over Ω. Solving H^1 gradient flow equation

$$\frac{\partial f}{\partial t} = -f \tag{2.11}$$

is much easier as it boils down to solving an ordinary differential equation, which the solution decays (or rather converges) exponentially fast.

On the other hand, what if one reduces the order too much by taking H^2? Since $\operatorname{grad}_{H^2} \mathcal{E}(f) = -\Delta^{-1} f$, the H^2 gradient flow equation becomes:

$$\frac{\partial f}{\partial t} = \Delta^{-1} f \tag{2.12}$$

This is solved by solving a Poisson's equation $\Delta g = f$ for g, then $g = \frac{\partial f}{\partial t}$.

2.3 Derivative Operators for Curves

For application of functional gradients on curves $\boldsymbol{\gamma} : M \to \mathbb{R}^3$, the following derivative operators replace the traditional derivative operators,

Definition (First Derivative Operator "Curve Gradient")**.** Define first derivative operator[5] $\tilde{\nabla}_{\boldsymbol{\gamma}}$ in a way that for $u\left(\boldsymbol{\gamma}(\cdot)\right) : M \to \mathbb{R}$,

$$\tilde{\nabla}_{\boldsymbol{\gamma}} u\left(\boldsymbol{\gamma}\left(s\right)\right) = \frac{\mathrm{d}u\left(\boldsymbol{\gamma}\left(s\right)\right)}{||\,\mathrm{d}\boldsymbol{\gamma}||} \frac{\mathrm{d}\boldsymbol{\gamma}}{||\,\mathrm{d}\boldsymbol{\gamma}||} = \frac{\mathrm{d}u\left(\boldsymbol{\gamma}\left(s\right)\right)}{||\,\mathrm{d}\boldsymbol{\gamma}||} \underbrace{\mathbf{T}(s)}_{\text{Unit tangent at } s} \tag{2.13}$$

This encapsulates the tangential derivative with the actual direction of the tangent vector.

Similarly, for Laplacian,

Definition (Second Derivative Laplacian Operator "Curve Laplacian")**.** Define second derivative operator $\tilde{\Delta}$ in a way that for $f : M \to \mathbb{R}$,

$$\tilde{\Delta}_{\boldsymbol{\gamma}} u\left(\boldsymbol{\gamma}(s)\right) = \tilde{\nabla}_{\boldsymbol{\gamma}}^T \tilde{\nabla}_{\boldsymbol{\gamma}} u\left(\boldsymbol{\gamma}(s)\right) \tag{2.14}$$

3 Finite Difference Method

To numerically solve a differential equation, one of the simplest method is the **finite difference method**.[4]

3.1 Motivating Example: Heat Equation

Suppose one needs to solve the heat equation $\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2}$ for $u = u(x,t)$ over $[-L, L] \times [0, T_f]$ with initial data $u\left(x, 0\right) = u_0(x)$ and Neumann boundary condition $f'(-L) = f'(L) = 0$.

One could diescretise the problem by constructing a mesh; taking equispaced points on the spatial domain $[-L, L]$ and the time domain $[0, T_f]$.

$$\begin{aligned} X_i &= -L + i\left(\Delta X\right) && \text{for } i = 0, 1, \cdots, N \\ T_j &= j\left(\Delta T\right) && \text{for } j = 0, 1, \cdots, M \end{aligned}$$

where N, M are the resolution of respective domains, and $\Delta X := \frac{2L}{N}$ and $\Delta T := \frac{T_f}{M}$.

Now the idea is to approximate the actual solution u at $(x,t) = (X_i, T_j)$ of the heat equation by the following algebraic (difference) equations.

$$\mathcal{D}_t U_i^j = \mathcal{D}_x^+ \mathcal{D}_x^- U_i^j \tag{3.1}$$
$$U_i^0 = u\left(X_i\right) \tag{3.2}$$
$$U_0^j - U_{-1}^j = 0 \tag{3.3}$$
$$U_{N+1}^j - U_N^j = 0 \tag{3.4}$$

for $i = 0, 1, \cdots, N$ and $j = 0, 1, \cdots, M$, where $\mathcal{D}_z$ is a finite difference operator that approximates a derivative in z variable (t for time, x for spatial variable). Note that "virtual points" X_{-1} and X_{N+1} are added for discretised Neumann boundary condition (3.3) and (3.4). One interprets U_i^j as an approximation of $u\left(X_i, T_j\right)$.

3.1.1 First-Order Finite Difference Operator

For utilising finite difference method for PDEs, one must choose an operator $\mathcal{D}_z$ to approximate the derivative with respect to variable z.

Consider the **forward difference operator** to get an approximation of $f'(z)$ at $z = Z_i$:

$$\mathcal{D}_z^+ f\left(Z_i\right) = \frac{f\left(Z_{i+1}\right) - f\left(Z_i\right)}{\Delta Z} = f'\left(Z_i\right) + \underbrace{O\left(\Delta Z\right)}_{\text{Consistency Error}} \tag{3.5}$$

Consider the **backward difference operator**, another approximation of $f'(z)$ at $z = Z_i$:

$$\mathcal{D}_z^- f\left(Z_i\right) = \frac{f\left(Z_i\right) - f\left(Z_{i-1}\right)}{\Delta Z} = f'\left(Z_i\right) + \underbrace{O\left(\Delta Z\right)}_{\text{Consistency Error}} \tag{3.6}$$

One can combine the idea of forward difference and backward difference to motivate the **central difference operator**, again for approximation of $f'(z)$ at $z = Z_i$:

$$\mathcal{D}_z^\circ f\left(Z_i\right) = \frac{f\left(Z_{i+1}\right) - f\left(Z_{i-1}\right)}{2\Delta Z} = f'\left(Z_i\right) + \underbrace{O\left(\left(\Delta Z\right)^2\right)}_{\text{Consistency Error}} \tag{3.7}$$

Note that one could write $\mathcal{D}_z^\circ f\left(Z_i\right) = \frac{1}{2}\left(D_z^+ + D_z^-\right) f\left(Z_i\right)$

While central difference operator may seem like the best choice in terms of consistency error, different choice of operators may result in much easier problem to solve albeit the cost of increased consistency error.

3.1.2 Second-Order Finite Difference Operator

For PDEs involving second derivatives (such as the Laplace's equation), one must also be able to approximate the second derivatives with respect to variable z. A sensible choice is to use both the forward difference operator and and the backward difference operator, known as the **second divided difference operator**:

$$\mathcal{D}_z^+\mathcal{D}_z^- f(Z_i) = \frac{f(Z_{i+1}) - 2f(Z_i) + f(Z_{i-1})}{(\Delta Z)^2} = f''(Z_i) + \underbrace{O\left((\Delta Z)^2\right)}_{\text{Consistency Error}} \tag{3.8}$$

One can easily check that $\mathcal{D}_z^+\mathcal{D}_z^- = \mathcal{D}_z^-\mathcal{D}_z^+$.

3.2 Euler Schemes for Heat Equation

Different choices for the first-order operator in (3.1) for the time variable leads to different equations to solve.

Taking $\mathcal{D}_t$ to be the forward difference operator, one acquires the **explicit Euler scheme**:

$$\frac{U_i^{j+1} - U_i^j}{\Delta T} = \frac{U_{i+1}^j - 2U_i^j + U_{i-1}^j}{(\Delta X)^2} \tag{3.9}$$

Note that figure 2.1(b) is the numerical solution acquired using this scheme. One advantage of this scheme is that it is easy to compute the "next time step" by observing that this can be written as $U_i^{j+1} = U_i^j + \frac{\Delta T}{(\Delta X)^2}\left(U_{i+1}^j - 2U_i^j + U_{i-1}^j\right)$, where the RHS is known. A drawback, however, is that a formal analysis using discrete Fourier transform shows that this scheme is stable if and only if $\mu := \frac{\Delta T}{(\Delta X)^2} \leq \frac{1}{2}$. For more detail, see [4].

4 Tangent-Point Energy

Given a curve $\boldsymbol{\gamma} : M \to \mathbb{R}^3$, a sensible choice of curve energy is the tangent-point energy[5].

Definition (Tangent-Point Energy)**.** For a continuously differentiable parameterised curve $\boldsymbol{\gamma} : M \to \mathbb{R}^3$, define **tangent-point energy** as:

$$\mathcal{E}_\beta^\alpha(\boldsymbol{\gamma}) := \iint_{M^2} k_\beta^\alpha(\boldsymbol{\gamma}_x, \boldsymbol{\gamma}_y, \mathbf{T}_x)\, \mathrm{d}\gamma_x\, \mathrm{d}\gamma_y \tag{4.1}$$

where $\mathbf{T}_x := \frac{\mathrm{d}\gamma_x}{\mathrm{d}t} \Big/ \left|\frac{\mathrm{d}\gamma_x}{\mathrm{d}t}\right|$ is the unit tangent vector at $\boldsymbol{\gamma}_x$ along the curve, and **tangent-point kernel** is given as:

$$k_\beta^\alpha(\mathbf{p}, \mathbf{q}, \mathbf{T}) := \frac{|\mathbf{T} \wedge (\mathbf{p} - \mathbf{q})|^\alpha}{|\mathbf{p} - \mathbf{q}|^\beta} \tag{4.2}$$

α and β are parameters one could choose, but for tangent-point energy to be well-defined, one may choose them to satisfy $\alpha > 1$ and $\beta \in [\alpha+2, 2\alpha+1)$.

Note that choosing $\alpha = 2$ and $\beta = 4$ results in the scaled version of the original tangent-point energy by Buck and Orloff[2].

The geometric intuition of $k_4^2(\boldsymbol{\gamma}_x, \boldsymbol{\gamma}_y, \mathbf{T}_x)$ (kernel of "Buck-Orloff tangent-point energy") is that the kernel evaluates to $\frac{1}{4r^2}$ where r is the radius of the smallest circle drawn that is tangent at $\boldsymbol{\gamma}_x$ and crosses through $\boldsymbol{\gamma}_y$ as shown in Figure 4.1.

The choice of parameters α and β changes the behaviour of the tangent-point energy as stated in the following lemma.

Lemma 4.1. *Tangent-point energy $\mathcal{E}_\beta^\alpha$ defined as (4.1) is scale invariant with respect to the curve if and only if $\beta = \alpha+2$. Moreover, if $\beta > \alpha+2$, then $\mathcal{E}_\beta^\alpha$ scales inversely with the curve.*

Proof. Take a parameterised curve $\boldsymbol{\gamma} : M \to \mathbb{R}^3$ and $\boldsymbol{\Gamma} \coloneqq c\boldsymbol{\gamma}$, a curve scaled by factor $c > 0$ of $\boldsymbol{\gamma}$. Note that the unit tangent vector is identical for $\boldsymbol{\gamma}$ and $\boldsymbol{\Gamma}$, that is, $\mathbf{T}_x \coloneqq \frac{\mathrm{d}\boldsymbol{\gamma}_x}{\mathrm{d}t} \Big/ |\frac{\mathrm{d}\boldsymbol{\gamma}_x}{\mathrm{d}t}| = \frac{\mathrm{d}\boldsymbol{\Gamma}_x}{\mathrm{d}t} / |\frac{\mathrm{d}\boldsymbol{\Gamma}_x}{\mathrm{d}t}|$

Then,

$$\frac{k_\beta^\alpha\left(\boldsymbol{\gamma}_x, \boldsymbol{\gamma}_y, \mathbf{T}_x\right)}{k_\beta^\alpha\left(\boldsymbol{\Gamma}_x, \boldsymbol{\Gamma}_y, \mathbf{T}_x\right)} = \frac{|\boldsymbol{\gamma}_x - \boldsymbol{\gamma}_y|^\alpha}{|\boldsymbol{\Gamma}_x - \boldsymbol{\Gamma}_y|^\beta} = c^{\beta-\alpha} \tag{4.3}$$

Also note that

$$\mathrm{d}\Gamma = c\,\mathrm{d}\gamma \tag{4.4}$$

So, we deduce from (4.1),

$$\mathcal{E}_\beta^\alpha\left(\boldsymbol{\Gamma}\right) = c^{\alpha-\beta+2}\mathcal{E}_\beta^\alpha\left(\boldsymbol{\gamma}\right) \tag{4.5}$$

Hence, $\mathcal{E}_\beta^\alpha$ is scale invariant with respect to the curve if and only if $\alpha - \beta + 2 = 0$, and if $\beta > \alpha+2$, $\mathcal{E}_\beta^\alpha$ scales as $O\left(\frac{1}{c^{\beta-(\alpha+2)}}\right)$ as $c \to \infty$. ■

Remark. If $\beta < \alpha + 2$, then the energy scales with the size of the curve, meaning that the energy is trivially minimised by scaling down the curve to a singularity, which is not desirable in our context.

One could also justify the condition on α and β by the following lemma.

Lemma 4.2. *Given α and β, the singularity of the kernel at a point and another point of which is arc-length $\epsilon > 0$ away from it is of order $O\left(\epsilon^{2\alpha-\beta}\right)$, that is, $k_\beta^\alpha\left(\boldsymbol{\gamma}(s), \boldsymbol{\gamma}(s+\epsilon), \mathbf{T}(s)\right) = O\left(\epsilon^{2\alpha-\beta}\right)$ as $\epsilon \to 0$. Moreover, if $2\alpha = \beta$, then the kernel converges to $\left(\frac{\kappa}{2}\right)^\alpha$ as the two points get closer, where κ is the curvature of the curve at the point.*

Proof. For $\boldsymbol{\gamma}(s) = (x(s), y(s), z(s))$ parameterised by arc-length, one recognises that the tangent vector at this point is $\mathbf{T} = \boldsymbol{\gamma}'(s)$. Note that $||\boldsymbol{\gamma}'(s)|| = ||\mathbf{T}|| = 1$.

By Taylor expansion:

$$\boldsymbol{\gamma}\left(s+\epsilon\right) = \boldsymbol{\gamma}(s) + \epsilon\boldsymbol{\gamma}'(s) + \frac{1}{2}\epsilon^2\boldsymbol{\gamma}''(s) + O\left(\epsilon^3\right) \tag{4.6}$$

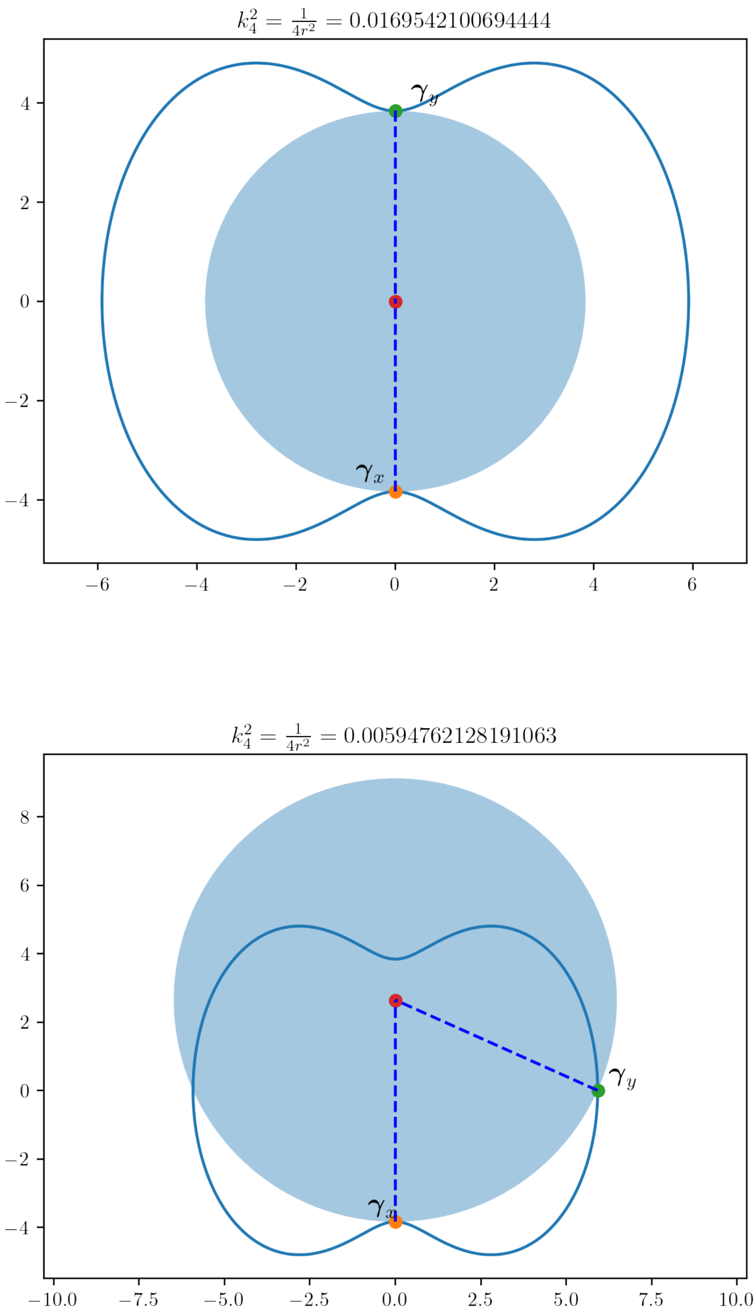


Figure 4.1: Intuition behind k_4^2

Then around $\epsilon = 0$,

$$
\begin{aligned}
& k_\beta^\alpha \left(\boldsymbol{\gamma}(s), \boldsymbol{\gamma}(s+\epsilon), \boldsymbol{\gamma}'(s)\right) \\
&= \frac{||\boldsymbol{\gamma}'(s) \wedge (\boldsymbol{\gamma}(s+\epsilon) - \boldsymbol{\gamma}(s))||^\alpha}{||\boldsymbol{\gamma}(s+\epsilon) - \boldsymbol{\gamma}(s)||^\beta} \\
&= \frac{||\boldsymbol{\gamma}'(s) \wedge \left(\epsilon\boldsymbol{\gamma}'(s) + \frac{1}{2}\epsilon^2\boldsymbol{\gamma}''(s) + O\left(\epsilon^3\right)\right)||^\alpha}{||\epsilon\boldsymbol{\gamma}'(s) + O\left(\epsilon^2\right)||^\beta} \\
&= \frac{\epsilon^{2\alpha-\beta}}{2^\alpha} \frac{||\boldsymbol{\gamma}'(s) \wedge \boldsymbol{\gamma}''(s) + O\left(\epsilon\right)||^\alpha}{||\boldsymbol{\gamma}'(s) + O\left(\epsilon\right)||^\beta} && \because \boldsymbol{\gamma}'(s) \wedge \boldsymbol{\gamma}'(s) = \mathbf{0} \\
&= \frac{\epsilon^{2\alpha-\beta}}{2^\alpha} \left(\frac{||\boldsymbol{\gamma}'(s)||^\alpha ||\boldsymbol{\gamma}''(s)||^\alpha}{||\boldsymbol{\gamma}'(s)||^\beta} + O\left(\epsilon\right) \right) && \because \boldsymbol{\gamma}'(s) \perp \boldsymbol{\gamma}''(s) \\
&= \frac{\epsilon^{2\alpha-\beta}}{2^\alpha} \left(\kappa^\alpha + O\left(\epsilon\right)\right) && \because ||\boldsymbol{\gamma}'(s)|| = ||\mathbf{T}|| = 1 \\
& && \text{and } ||\boldsymbol{\gamma}''(s)|| = \kappa
\end{aligned}
$$

So with arc-length perturbation of ϵ, the order of singularity of the kernel is $O\left(\epsilon^{2\alpha-\beta}\right)$. In particular, if $2\alpha - \beta = 0$, the kernel converges to $\left(\frac{\kappa}{2}\right)^\alpha$, as demonstrated in Figure 4.2(a). ■

From lemma 4.1 and lemma 4.2, the well-definedness condition of tangent-point energy immediately follows from arguing by considering integrability of functions with isolated poles of order lower than 1. (See Figure 4.3)

Corollary 4.3. *If $\alpha > 1$ and $\beta \in [\alpha + 2, 2\alpha + 1)$, tangent-point energy $\mathcal{E}_\beta^\alpha$ is well-defined.*

5 Unknotting Curves via Gradient Flow Equation

Now that gradient flow equation and tangent-point energy are introduced, one can formalise the process of untangling a tangled curve:

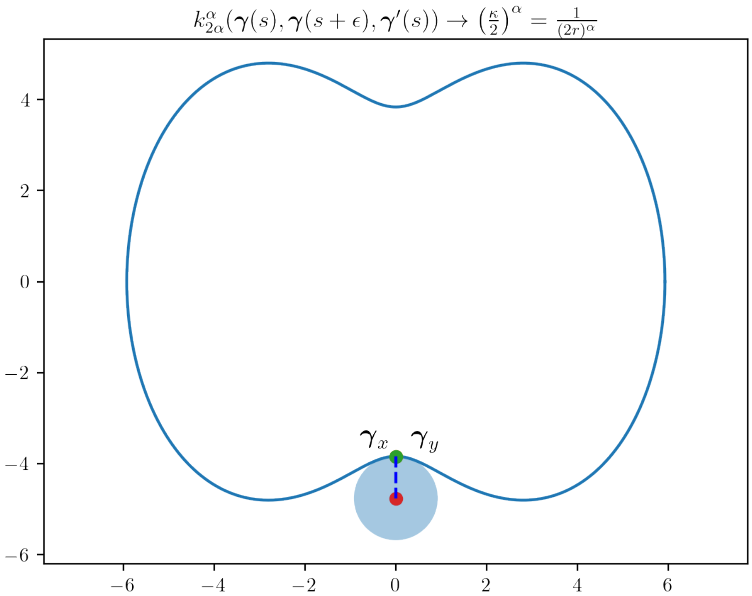
Definition (Curve Untangling Process)**.** Given a parameterised curve $\boldsymbol{\gamma} : M \times T \to \mathbb{R}^3$ over an interval M and time domain T, denote the following initial value problem as **curve untangling process**:

$$\frac{\partial \boldsymbol{\gamma}}{\partial t} = -\operatorname{grad}_{\mathcal{X}} \mathcal{E}_\beta^\alpha(\boldsymbol{\gamma}) - \operatorname{grad}_{\mathcal{X}} \mathcal{C}(\boldsymbol{\gamma}) \tag{5.1}$$

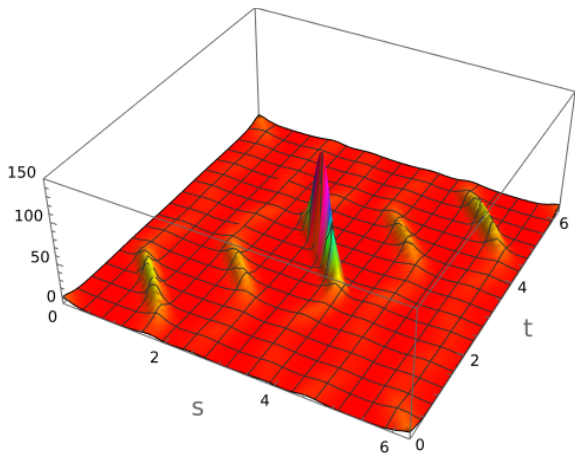
$$\boldsymbol{\gamma}(s; 0) = \boldsymbol{\gamma}_0(s) \tag{5.2}$$

where

- $\boldsymbol{\gamma}_0(s)$ is the parameterisation of the initial (tangled) curve (prescribed at $t = 0$)
- $\mathcal{E}_\beta^\alpha$ is the tangent-point energy (See (4.1))
- $\mathcal{C}$ is additional constraint energy to control behaviour of curve untangling process.



(a) $k_{2\alpha}^{\alpha}\left(\boldsymbol{\gamma}(s), \boldsymbol{\gamma}(s+\epsilon), \boldsymbol{\gamma}'(s)\right)$ converges to $\left(\frac{\kappa}{2}\right)^{\alpha}$.



(b) $k_{2\alpha}^{\alpha}$ heat map

Figure 4.2: Behaviours of $\beta = 2\alpha$ kernel

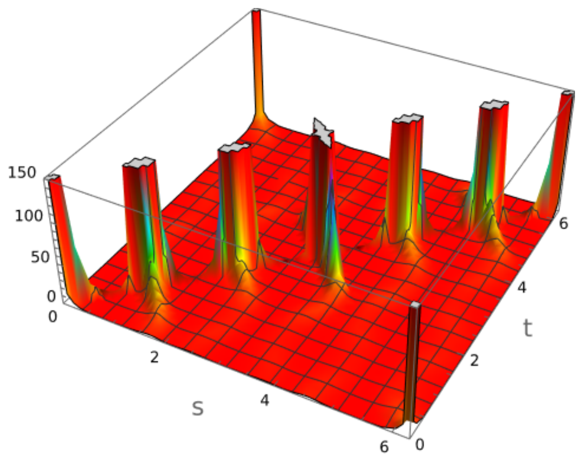


Figure 4.3: Heat map of $k^2_{4.5}$: Even with singularities, because their order is less than 1, the kernel is integrable, hence $\mathcal{E}^2_{4.5}$ is well-defined.

5.1 Discretisation for Numerical Computation

Solving (5.1), (5.2) analytically is challenging. Rather, we aim to acquire a numerical solution. Assume for simplicity that the curve of interest is simple closed.[5]

5.1.1 Discretisation of Curve

We start by discretising the initial curve $\boldsymbol{\gamma}_0$ by taking N points on a curve as shown in Figure 5.1. Represent the initially discretised curve in the tensor[6] form: $\boldsymbol{\Gamma}^0 = \left(\mathbf{x}_0^0, \mathbf{x}_1^0, \cdots, \mathbf{x}_{N-1}^0\right) \in \mathbb{R}^{3\times N}$, and the discretised curve at subsequent time step $k \in \mathbb{N} \cup \{0\}$ as $\boldsymbol{\Gamma}^k = \left(\mathbf{x}_0^k, \mathbf{x}_1^k, \cdots, \mathbf{x}_{N-1}^k\right) \in \mathbb{R}^{3\times N}$. Since we restrict our attention to a simple closed curve, it is convenient to extend the indexing rule by:

$$\mathbf{x}_i^k = \mathbf{x}_{r(i,N)}^k \qquad \text{where } r\,(i, N) = (\text{remainder of } i \div N) \tag{5.3}$$

so that $\mathbf{x}_N^k = \mathbf{x}_0^k$, $\mathbf{x}_{N+1}^k = \mathbf{x}_1^k$, etc.

Define (right) operator $[\cdot] : \mathbb{R}^{3\times N} \to \mathbb{R}$ such that for tensor $T \in \mathbb{R}^{3\times N}$,

$$T[i] := T\mathbf{e}_{r(i,N)}$$

where $\mathbf{e}_i$ is the i^{th} canonical vector. With this operator, one could write

$$\boldsymbol{\Gamma}^k[i] = \mathbf{x}_{r(i,N)}^k = \mathbf{x}_i^k \tag{5.4}$$

analogous to $\boldsymbol{\gamma} = \boldsymbol{\gamma}(s;t)$ being a parameterised curve, which is a vector-valued function.

Finally, denote by e_i^k for the (undirected) edge with vertex pair $\left(\mathbf{x}_i^k, \mathbf{x}_{i+1}^k\right)$.

[5] We may assume that the function definition of $\boldsymbol{\gamma} : M \to \mathbb{R}^3$ extends to $\boldsymbol{\gamma} : \mathbb{R} \to \mathbb{R}^3$ by periodicity.

[6] "Tensors are multidimensional arrays of numerical values and therefore generalize matrices to multiple dimensions."[3] Note that we use the definition from machine learning, rather than mathematics, merely as a convenient data type to capture a discretised curve.

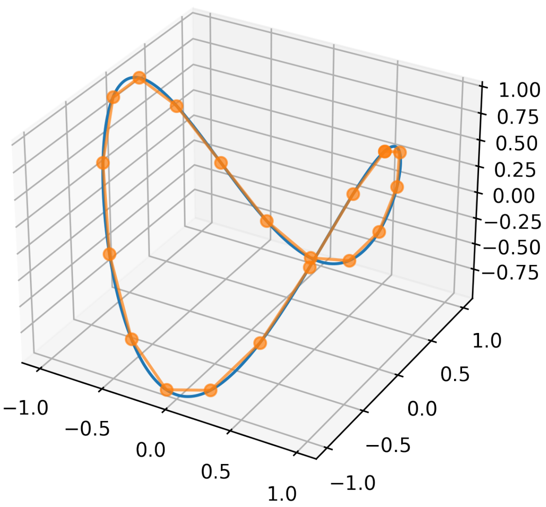


Figure 5.1: Discretisation of a simple closed curve by sampling the points along the curve.

5.1.2 Discretisation of Tangent-Point Energy

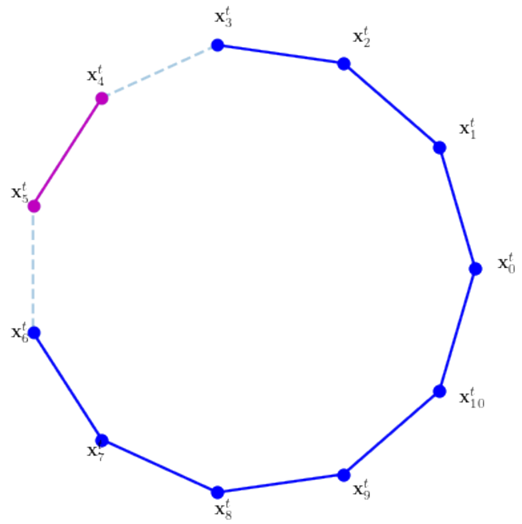
In order to acquire numerical solution, one must also be able to numerically compute the tangent-point energy. For this, we pose the energy quadrature $E^{\alpha}_{\beta}\left(\mathbf{\Gamma}^k\right)$ of the following form:

$$\mathcal{E}^{\alpha}_{\beta}\left(\boldsymbol{\gamma}\left(\cdot,t\right)\right) \coloneqq \iint_{M^2} k^{\alpha}_{\beta}(\boldsymbol{\gamma}_x,\boldsymbol{\gamma}_y)\,\mathrm{d}\gamma_x\,\mathrm{d}\gamma_y \approx E^{\alpha}_{\beta}\left(\mathbf{\Gamma}^k\right) \coloneqq \sum_{i,j\in\{0,\cdots,N-1\}} K^{\alpha}_{\beta}\left(i,j\right)||e^k_i||\,||e^k_j|| \tag{5.5}$$

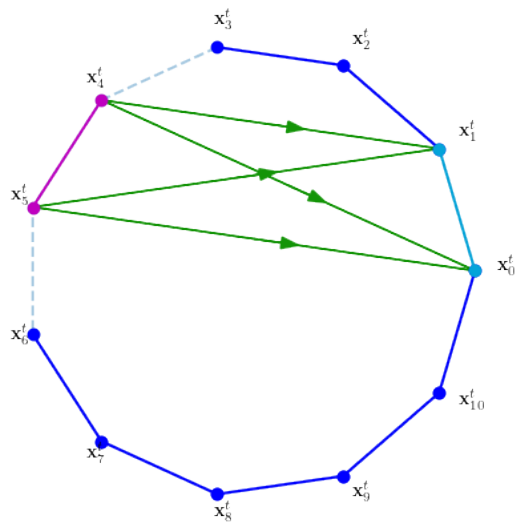
where K^{α}_{β} is an approximation of tangent-point kernel k^{α}_{β} (which is specified at (5.6)), and $||e^k_i||$ is the length of edge e^k_i, that is, $||e^k_i|| = ||\mathbf{x}^{k+1}_i - \mathbf{x}^k_i||$. Note that $\mathbf{\Gamma}^k$ is a polygonal curve, for which tangent-point energy (4.1) is not well-defined due to locally non-integrable contributions from vertices:

One way to resolve the issue is to "ignore" the adjacent edge contribution[5] in the energy quadrature E^{α}_{β} as shown in Figure 5.2(a). The justification is that as we take a finer mesh (N sufficiently large), the product of edge lengths ($||e_i||||e_j||$) should tend to zero sufficiently fast, resulting in approximation of the energy for the smooth curve, which did not have vertices resulting in local non-integrability in the first place.

It still remains to sensibly approximate the kernel $k^{\alpha}_{\beta}(\boldsymbol{\gamma}_x,\boldsymbol{\gamma}_y) \approx K^{\alpha}_{\beta}(i,j)$. One



(a) For a chosen edge e_i^k, ignore the two adjacent edges e_{i-1}^k, e_{i+1}^k. In the limit as $N \to 0$, because the edge lengths tend to zero, the discrepancy between the quadrature and the analytical value of the energy is expected to tend to zero.



(b) Tangent-point kernel is approximated by 4-point quadrature defined as (5.6).

Figure 5.2: Quadrature for approximation of tangent-point energy.

sensible approximation is to use the following 4-point quadrature[5].

$$K^\alpha_\beta(i,j) := \frac{1}{4}\Bigg(k^\alpha_\beta\left(\mathbf{x}^k_i, \mathbf{x}^k_j, \mathbf{T}^k_i\right) + k^\alpha_\beta\left(\mathbf{x}^k_i, \mathbf{x}^k_{j+1}, \mathbf{T}^k_i\right) + k^\alpha_\beta\left(\mathbf{x}^k_{i+1}, \mathbf{x}^k_j, \mathbf{T}^k_i\right) + k^\alpha_\beta\left(\mathbf{x}^k_{i+1}, \mathbf{x}^k_{j+1}, \mathbf{T}^k_i\right)\Bigg) \quad (5.6)$$

where $\mathbf{T}^k_i := \frac{\mathbf{x}^k_{i+1}-\mathbf{x}^k_i}{||\mathbf{x}^k_{i+1}-\mathbf{x}^k_i||}$ approximates the tangent vector to the curve at $\boldsymbol{\gamma}_x$. (See Figure 5.2(b).)

Putting (5.5) and (5.6) together, one can write the **tangent-point energy quadrature** as:

$$E^\alpha_\beta\left(\mathbf{\Gamma}^k\right) := \sum_{\substack{i,j\in\{0,\cdots,N-1\}\\ r(i-j,N)>1}} K^\alpha_\beta(i,j)||e^k_i||\,||e^k_j|| \quad (5.7)$$

where $r\,(i-j,N)$ is the geodesic distance between i and j in modulo N, characterising the avoidance of adjacent edges on simple closed polygonal curve. For discussion of quadratures for nonclosed curves, see appendix C.

5.1.3 Derivative Operators Discretised Curve

For discrete curve, the (forward and backward) first derivative operators (2.13) transforms to operator for $U\left(\mathbf{\Gamma}^k\left[\cdot\right]\right) : \{0,1,\cdots,N-1\} \to \mathbb{R}$ characterised by:

$$\tilde{\nabla}^+_{\mathbf{\Gamma}^k} U\left(\mathbf{\Gamma}^k\left[i\right]\right) = \frac{U\left(\mathbf{\Gamma}^k\left[i+1\right]\right) - U\left(\mathbf{\Gamma}^k\left[i\right]\right)}{||\mathbf{\Gamma}\left[i+1\right] - \mathbf{\Gamma}\left[i\right]||}\mathbf{T}^k_i \quad (5.8)$$

$$\tilde{\nabla}^-_{\mathbf{\Gamma}^k} U\left(\mathbf{\Gamma}^k\left[i\right]\right) = \frac{U\left(\mathbf{\Gamma}^k\left[i\right]\right) - U\left(\mathbf{\Gamma}^k\left[i-1\right]\right)}{||\mathbf{\Gamma}\left[i\right] - \mathbf{\Gamma}\left[i-1\right]||}\mathbf{T}^k_{i-1} \quad (5.9)$$

and analogously with second derivative operator:

$$\tilde{\Delta}_{\mathbf{\Gamma}^k} U\left(\mathbf{\Gamma}[i]\right) = \frac{\frac{U\left(\mathbf{\Gamma}^k[i+1]\right)-U\left(\mathbf{\Gamma}^k[i]\right)}{||\mathbf{\Gamma}^k[i+1]-\mathbf{\Gamma}^k[i]||} - \frac{U\left(\mathbf{\Gamma}^k[i]\right)-U\left(\mathbf{\Gamma}^k[i-1]\right)}{||\mathbf{\Gamma}^k[i]-\mathbf{\Gamma}^k[i-1]||}}{\frac{1}{2}||\mathbf{\Gamma}^k\left[i+1\right] - \mathbf{\Gamma}^k\left[i-1\right]||} \quad (5.10)$$

$$= \frac{||e_i||U_{i-1} - \left(||e_{i-1}|| + ||e_i||\right)U_i + ||e_{i-1}||U_{i+1}}{||e_{i-1}||\,||e_i||\,||f_i||} \quad (5.11)$$

$$= \frac{1}{||e_{i-1}||\,||e_i||\,||f_i||}\begin{pmatrix} ||e_i|| & -\left(||e_{i-1}|| + ||e_i||\right) & ||e_{i-1}||\end{pmatrix}\begin{pmatrix} U_{i-1} \\ U_i \\ U_{i+1}\end{pmatrix} \quad (5.12)$$

where one defines $f_i := \frac{1}{2}\left(\mathbf{\Gamma}^k\left[i+1\right] - \mathbf{\Gamma}^k\left[i-1\right]\right)$ and $U_i := U\left(\mathbf{\Gamma}^k\left[i\right]\right)$.

However, inverting the Laplacian is troublesome, as it is equivalent to inverting a singular matrix.

Lemma 5.1. *The matrix capturing curve Laplacian $\tilde{\Delta}_{\mathbf{\Gamma}^k}$ for closed discrete curve $\mathbf{\Gamma}^k$ is singular.*

Proof. The matrix L for $\tilde{\Delta}_{\mathbf{\Gamma}^k}$ for closed discrete curve $\mathbf{\Gamma}^k$ has the form:

$$L := \begin{pmatrix} * & * & & & & & * \\ * & * & * & & & & \\ & * & * & * & & & \\ & & \ddots & \ddots & \ddots & & \\ & & & * & * & * & \\ & & & & * & * & * \\ * & & & & & * & * \end{pmatrix}$$

To show that L is singular, it is sufficient to show that there is a nontrivial solution $\mathbf{x}$ to $L\mathbf{x} = \mathbf{0}$. Since rows follow from (5.12), observe that the rows sum to zero. Hence, one can spot such nontrivial solution $\mathbf{x} = (1, 1, \cdots, 1)^T$. ■

5.1.4 Finite Difference Scheme of Curve Untangling Process

Based on (5.1), one writes the following finite difference scheme:

$$\mathcal{D}_t \mathbf{\Gamma}^k = -\operatorname{Grad}_X E_\beta^\alpha(\mathbf{\Gamma}^k) - \operatorname{Grad}_X C\left(\mathbf{\Gamma}^k\right) \qquad \text{for } k = 0, 1, \cdots \tag{5.13}$$

where $\mathcal{D}_t$ is the finite difference operator over time, Grad_X is discrete equivalent[7] of $\operatorname{grad}_{\mathcal{X}}$ on discrete inner product space X (which may be omitted in the notation), E_β^α is the tangent-point energy quadrature defined as (5.7), and C is the discretised version of the constraint energy $\mathcal{C}$ (See section 5.2.1). Often times, however, it is easier to construct finite difference scheme directly from (5.1) after some simplification, rather than attempting to use (5.13) directly. Note that because similar operations are done for each point vector, this is a parallelisable task. For the simplest scheme, one could take the forward difference operator characterised as $\mathcal{D}_t \mathbf{\Gamma}^k[i] := \frac{\mathbf{\Gamma}^{k+1}[i] - \mathbf{\Gamma}^k[i]}{\Delta T}$.

5.2 Example: L^2 Explicit Euler Scheme

Now we visit the simplest concrete numerical scheme for curve untangling process. Assume for now scale-invariance by taking parameters α and β to satisfy $\beta = \alpha + 2$ (See lemma 4.1).

In L^2, $\operatorname{grad}_{L^2} \mathcal{E}_\beta^\alpha(\boldsymbol{\gamma})$ is simply the "first-order perturbation" as in (2.6). Discrete equivalent is the ℓ^2 space, where one may justify this by noting that first-order perturbation in a curve is analogous to perturbing each of the point on its discretisation. Taking $\mathcal{D}_t$ from (5.13) to be forward difference operator,

$$\frac{\mathbf{\Gamma}^{k+1} - \mathbf{\Gamma}^k}{\Delta T} = -\operatorname{Grad}_{\ell^2} E_\beta^\alpha\left(\mathbf{\Gamma}^k\right) \tag{5.14}$$

[7] Worth noting that $\operatorname{Grad}_X : \mathbb{T} \to \mathbb{T}$ where $\mathbb{T}$ is a set of tensors of certain shape; Grad_X maps tensors to tensors of the same shape.

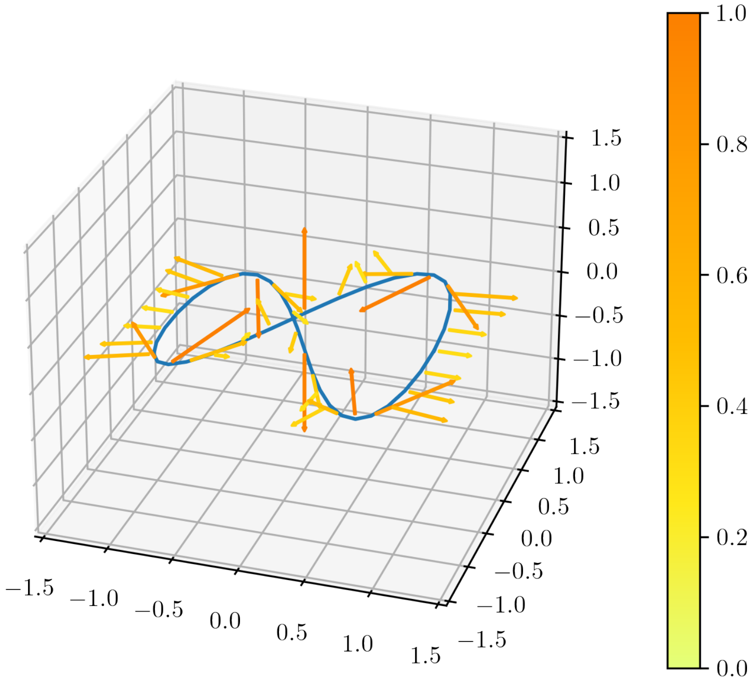


Figure 5.3: $-\operatorname{Grad}_{L^2}\left(\mathbf{\Gamma}^k\right)$ which is the direction of flow of a discretised curve $\mathbf{\Gamma}^k$ at each point, represented by the arrows. The magnitude is represented by both color and the length of the arrows.

where one could explicitly write $\operatorname{Grad}_{\ell^2} E_\beta^\alpha\left(\mathbf{\Gamma}^k\right)$ as

$$\operatorname{Grad}_{\ell^2} E_\beta^\alpha\left(\mathbf{\Gamma}^k\right) = \underbrace{\begin{pmatrix} \frac{\partial}{\partial x_{1,1}} & \frac{\partial}{\partial x_{1,2}} & \cdots & \frac{\partial}{\partial x_{1,N-1}} \\ \frac{\partial}{\partial x_{2,1}} & \frac{\partial}{\partial x_{2,2}} & \cdots & \frac{\partial}{\partial x_{1,N-1}} \\ \frac{\partial}{\partial x_{3,1}} & \frac{\partial}{\partial x_{3,2}} & \cdots & \frac{\partial}{\partial x_{3,N-1}} \end{pmatrix}}_{\nabla_{\mathbf{\Gamma}^k}} E_\beta^\alpha\left(\mathbf{\Gamma}^k\right) \tag{5.15}$$

where $x_{j,i}$ refers to the (j,i) coordinate variable for $3 \times N$ tensor. Note that $\operatorname{Grad}_{\ell^2} E_\beta^\alpha\left(\mathbf{\Gamma}^k\right) \in \mathbb{R}^{3\times N}$ is also a tensor of the same shape as $\mathbf{\Gamma}^k$, so naturally, most arithmetics[8] needed for (5.14) is well-defined.

Remark. $\operatorname{Grad}_{\ell^2}$ is in fact a linear operator with respect to its input, energy.

One could use this exact form of $\operatorname{Grad}_{\ell^2} E_\beta^\alpha\left(\mathbf{\Gamma}^k\right)$ as given in appendix D, but it could be considered cumbersome (even though there are benefits to implementing this as stated in section 5.2.2). One could alternatively approximate $\operatorname{Grad}_{\ell^2} E_\beta^\alpha\left(\mathbf{\Gamma}^k\right)$

[8]Such as addition, subtraction, and scalar multiplication.

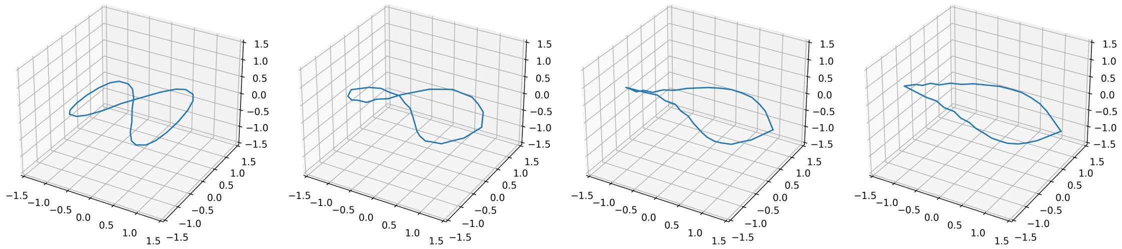


Figure 5.4: L^2 gradient flow on a figure-eight curve. Note that there are "sharp" points, along with "high frequency oscillation" (which are undesirable or unexpected characteristics); this may be due to the fact that L^2 gradient flow does not take into account of the smoothness of the curve.

by central difference scheme, for example: for $i = 0, 1, \cdots, N-1$ and $j = 1, 2, 3$,

$$\mathbf{e}_j \cdot \left(\mathrm{Grad}_{\ell^2} E_\beta^\alpha \left(\mathbf{\Gamma}^k\right)[i]\right) = \frac{\partial E_\beta^\alpha(\mathbf{\Gamma}^k)}{\partial x_{j,i}} \approx \frac{1}{2\Delta X}\left(\bar{E}_\beta^\alpha(i) - \underline{E}_\beta^\alpha(i)\right) \tag{5.16}$$

where

$$\bar{E}_\beta^\alpha(i) \coloneqq E_\beta^\alpha\left((\mathbf{x}_0, \cdots, \mathbf{x}_{i-1}, \mathbf{x}_i + \Delta X \mathbf{e}_j, \mathbf{x}_{i+1}, \cdots, \mathbf{x}_{N-1})\right)$$
$$\underline{E}_\beta^\alpha(i) \coloneqq E_\beta^\alpha\left((\mathbf{x}_0, \cdots, \mathbf{x}_{i-1}, \mathbf{x}_i - \Delta X \mathbf{e}_j, \mathbf{x}_{i+1}, \cdots, \mathbf{x}_{N-1})\right)$$

Remark. Notice that (5.14) can be interpreted as SDM over all the coordinates on the curve! This is consistent with the motivation of gradient flow equation given in section 2.1.

We now attempt the finite difference scheme with a *scale-variant energy*, that is, when $\beta > \alpha + 2$. If one attempts the same finite difference scheme as (5.14) one may observe that the curve "grows" in size, indefinitely. This is due to the fact that with $\beta > \alpha + 2$, by the virtue of lemma 4.1, energy scales inversely proportional to its scale factor. (See Figure 5.5) To use scale-variant energy, one may consider having constraint energy as part of the curve untagling process.

5.2.1 Constraint Energy

From lemma 4.1 and as demonstrated in Figure 5.5, in the case that $\beta > \alpha + 2$, one could trivially minimise tangent-point energy $\mathcal{E}_\beta^\alpha$ of the curve (and by the same logic, E_β^α of the discretised curve, albeit due to increases in edge lengths, would cease to be a "valid" quadrature) by scaling the curve to infinity. In order to avoid this phenomenon, or to change the behaviour of the flow, one may add additional energy which penalises unwanted behaviours.

Here are some examples of constraint one could take. (Since we are interested in numerical schemes, constraints are expressed for C rather than $\mathcal{C}$.)

- Taking $C\left(\mathbf{\Gamma}^k\right) \coloneqq \lambda \sum_{i=0}^{N-1} ||\mathbf{\Gamma}^k[i]||^p$ adds the motivation for the curve not to stray away from the origin.

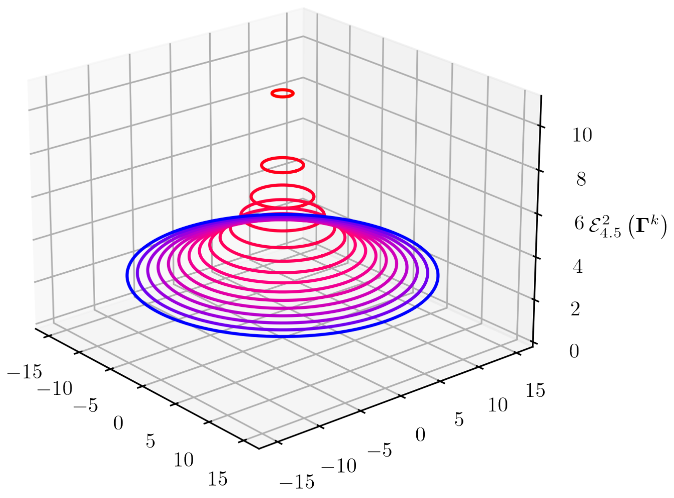


Figure 5.5: The height represents $\mathcal{E}^2_{4.5}$ for circles of different radius. Note that the energy decreases trivially by taking a larger circle.

- Taking $C\left(\mathbf{\Gamma}^k\right) := \lambda \left| \sum_{i=0}^{N-1} ||e_i|| - L \right|^p$ adds the motivation for the curve to stay close to some constant arc-length L.
- Taking $C\left(\mathbf{\Gamma}^k\right) := \lambda \sum_{i=0}^{N-1} \left| ||e_i|| - L_i \right|^p$ adds the motivation for the edge lengths on the discretised curve not to deviate from prescribed edge lengths L_i.

$\lambda \geq 0$ is the strength of the overall constraint, and $p > 0$ is the order of "thresholding", where higher p would lead to harsher "thresholding".

5.2.2 Time Complexity

Since we expect to take N to be very large, it is reasonable to consider the computational work.

If we *use a difference scheme* to approximate $\operatorname{Grad}_{\ell^2} E^\alpha_\beta \left(\mathbf{\Gamma}^k\right)$, for every time step, one needs to be able to evaluate E^α_β for a given $\mathbf{\Gamma}^k \in \mathbb{R}^{3\times N}$. From (5.7), it takes $O\left(N^2\right)$ evaluations of the kernel. Since the approximation for $\operatorname{Grad}_{L^2} E^\alpha_\beta$ requires evaluation of the energy for each point perturbed, it takes $O\left(N^3\right)$ evaluations of the kernel for each time step.

On the other hand, this is where doing the hard work of implementing *exact gradient computation* is beneficial, For exact gradient implementation as in appendix D, it takes $O\left(N\right)$ evaluations[9] of the derivative of the kernel. This implies the overall

[9] $4\left(N-3\right)$ to be exact.

computation needed for a single step is $O\left(N^2\right)$.

5.2.3 Discussion of Implicit L^2 Euler

A problem with explicit schemes is that, there often exists some sort of condition for stability. This tends to be an issue particularly when using approximation of derivatives in both time and space variables.

For unconditional stability, one may attempt (fully) implicit Euler scheme, that is, taking $\mathcal{D}_t$ in (5.13) to be the backward difference operator,

$$\frac{\boldsymbol{\Gamma}^k - \boldsymbol{\Gamma}^{k-1}}{\Delta T} = -\operatorname{Grad}_{\ell^2} E_\beta^\alpha\left(\boldsymbol{\Gamma}^k\right) \tag{5.17}$$

This scheme is, however, nonlinear. One needs an efficient rootfinding algorithm (such as Newton's method) to compute each time step.

Another potential scheme is the Crank-Nicolson scheme, where one combines both explicit and implicit scheme to achieve higher order of error.

5.3 Example: Euler Scheme in Other Spaces

Consider curve untangling process in H^{-1}. By (2.9), we may write $\operatorname{grad}_{H^{-1}} \mathcal{E}_\beta^\alpha\left(\boldsymbol{\gamma}\right) = -\tilde{\Delta}_{\boldsymbol{\gamma}}\left(\operatorname{grad}_{L^2} \mathcal{E}_\beta^\alpha\left(\boldsymbol{\gamma}\right)\right)$, where we do not necessarily assume scale-invariance. For the discrete equivalent, take

$$\operatorname{Grad} E_\beta^\alpha = -\tilde{\Delta}_{\boldsymbol{\Gamma}^k}\left(\operatorname{Grad}_{l^2} E_\beta^\alpha\left(\boldsymbol{\Gamma}^k\right)\right) = -\tilde{\Delta}_{\boldsymbol{\Gamma}^k}\left(\nabla_{\boldsymbol{\Gamma}^k} E_\beta^\alpha\left(\boldsymbol{\Gamma}^k\right)\right) \tag{5.18}$$

Note that computing "third derivative operator" could increase complexity of the scheme. In fact, if one approximates the Laplacian as (5.10), for each time step, it takes $O\left(N^4\right)$ evaluations of the kernel, so the scheme becomes impractical.

On the other hand, consider curve untangling process in H^1. By (2.7), we write $\operatorname{grad}_{H^1} \mathcal{E}_\beta^\alpha\left(\boldsymbol{\gamma}\right) = -\tilde{\Delta}_{\boldsymbol{\gamma}}^{-1} \operatorname{grad}_{L^2} \mathcal{E}_\beta^\alpha$, and (5.1) without constraint term can be written as:

$$\frac{\partial \boldsymbol{\gamma}}{\partial t} = \tilde{\Delta}_{\boldsymbol{\gamma}}^{-1} \operatorname{grad}_{L^2} \mathcal{E}_\beta^\alpha\left(\boldsymbol{\gamma}\right) \tag{5.19}$$

Discretising this with forward difference for time, one gets H^1 explicit Euler scheme:

$$\left(\frac{\boldsymbol{\Gamma}^{k+1} - \boldsymbol{\Gamma}^k}{\Delta T}\right) = \tilde{\Delta}_{\boldsymbol{\Gamma}^k}^{-1} \nabla_{\boldsymbol{\Gamma}^k} E_\beta^\alpha\left(\boldsymbol{\Gamma}^k\right) \tag{5.20}$$

where $\tilde{\Delta}_{\boldsymbol{\Gamma}^k}$ is the Laplacian as given in (5.10), and $\Delta_{\boldsymbol{\Gamma}^k}$ is still the conventional gradient operator over all the coordinates of $\boldsymbol{\Gamma}^k$. Note that by lemma 5.1, one needs to "regularise" the curve Laplacian; we do not attempt to solve for an exact H^1 gradient, but rather a close one. To compute a step, one performs three steps:

1. Compute $\nabla_{\boldsymbol{\Gamma}^k} E_\beta^\alpha\left(\boldsymbol{\Gamma}^k\right)$.

2. Solve linear system $\mathcal{L}\mathcal{G}^T = \left(\nabla_{\boldsymbol{\Gamma}^k} E_\beta^\alpha\left(\boldsymbol{\Gamma}^k\right)\right)^T$ for $\mathcal{G} \in \mathbb{R}^{3\times N}$

- $\mathcal{L} = \tilde{\Delta}_{\mathbf{\Gamma}^k} + cI$ is the (almost tridiagonal by (5.12)) matrix capturing the "regularised" discrete curve Laplacian $\tilde{\Delta}_{\mathbf{\Gamma}^k}$; because curve Laplacian matrix alone is singular, the linear system would not have a unique solution, and the regularisation is to pick a sensible solution to the singular linear system.
- $c > 0$ is a constant such that c closer to zero characterises "more H^1-like flow", at the cost of higher conditioning number.

3. Evolve by $\mathbf{\Gamma}^{k+1} = (\Delta T)\,\mathcal{G} + \mathbf{\Gamma}^k$.

Note that the linear system $\mathcal{L}\mathcal{G}^T = \left(\nabla_{\mathbf{\Gamma}^k} E^\alpha_\beta\left(\mathbf{\Gamma}^k\right)\right)^T$ has the following structure:

$$\underbrace{\begin{pmatrix} * & * & & & & & * \\ * & * & * & & & & \\ & * & * & * & & & \\ & & \ddots & \ddots & \ddots & & \\ & & & * & * & * & \\ & & & & * & * & * \\ * & & & & & * & * \end{pmatrix}}_{\mathcal{L}\in\mathbb{R}^{N\times N}} \mathcal{G}^T = \underbrace{\begin{pmatrix} * & * & * \\ * & * & * \\ * & * & * \\ \vdots & \vdots & \vdots \\ * & * & * \\ * & * & * \\ * & * & * \end{pmatrix}}_{\left(\nabla_{\mathbf{\Gamma}^k} E^\alpha_\beta\left(\mathbf{\Gamma}^k\right)\right)^T\in\mathbb{R}^{N\times 3}}$$

Appendices

A Definitions of Important Inner Product Spaces

(Adapted based on definitions given in lecture note by Süli[4]. For our purposes, we focus our attention to interval $I \subset \mathbb{R}$.) Here are some of the notable inner product spaces.

A.1 L^2 Space

Definition (L^2 Space)**.** For interval $I \subset \mathbb{R}$, L^2 space over is defined as

$$L^2 = \left\{ f : I \to \mathbb{R} \,\middle|\, f \text{ measurable}, \left(\int_I |f|^2 \,\mathrm{d}x \right)^{1/2} < \infty \right\} \tag{A.1}$$

L^2 inner product is defined as

$$\forall f, g \in L^2 \;\; \langle f, g \rangle_{L^2} = \int_I fg \,\mathrm{d}x \tag{A.2}$$

A.2 H^k Space

To define Sobolev (inner product) spaces (denoted H^k where $k \in \mathbb{N} \cup \{0\}$) one must define weak derivative operator D:

Definition (Weak Derivative)**.** For u locally integrable on I, if there exists w such that for all infinitely smooth $v : I \to \mathbb{R}$ with compact support,

$$\int_I wv \,\mathrm{d}x = (-1)^\alpha \int_I u \frac{\mathrm{d}^\alpha v}{\mathrm{d}x^\alpha} \,\mathrm{d}x \tag{A.3}$$

then w is said to be **weak derivative** of order α of u, and one writes $D^\alpha u = w$.

Weak derivative extends the definition of conventional derivative, and is equivalent to the conventional derivative for smooth functions. With weak derivatives introduced, one may now define Sobolev inner product spaces.

Definition (H^k Space)**.** Sobolev inner product space of order $k \in \mathbb{N} \cup \{0\}$ (denoted H^k) is defined as

$$H^k = \left\{ f \in L^2 \mid D^\alpha f \in L^2, \alpha \leq k \right\} \tag{A.4}$$

H^k inner product is defined as:

$$\forall f, g \in H^k \;\; \langle f, g \rangle_{H^2} = \sum_{\alpha \leq k} \langle D^\alpha f, D^\alpha g \rangle_{L^2} \tag{A.5}$$

Remark. Note that $H^0 = L^2$ by definition. One could say that Sobolev inner product spaces extend L^2 space. It also turns out that H^k are Hilbert spaces.

B Gradient Operator in Sobolev Spaces

For acquiring gradient on integer Sobolev spaces from L^2 gradient over boundary-free Ω, there are two main "rules" one could use.

Lemma B.1 (Shifting Gradient Operator)**.**

$$\langle \nabla f, \nabla g \rangle_{L^2} = -\langle \Delta f, g \rangle_{L^2} = -\langle f, \Delta g \rangle_{L^2} \tag{B.1}$$

Proof.

$$\begin{aligned}
\langle \nabla f, \nabla g \rangle_{L^2} &= \int_\Omega \nabla f \cdot \nabla g \, \mathrm{d}V \\
&= \int_\Omega \left(\nabla \cdot (g \nabla f) - g \Delta f \right) \mathrm{d}V \\
&\overset{\text{IBP}}{=} \underbrace{\oint_{\partial\Omega} g \nabla f \cdot \mathbf{n} \, \mathrm{d}s}_{\text{Boundary Term}} - \int_\Omega g \Delta f \, \mathrm{d}V \\
&= -\langle \Delta f, g \rangle_{L^2} + \underbrace{\oint_{\partial\Omega} g \nabla f \cdot \mathbf{n} \, \mathrm{d}s}_{\text{Boundary Term}}
\end{aligned}$$

For boundary-free Ω, boundary terms can be taken to be zero. ■

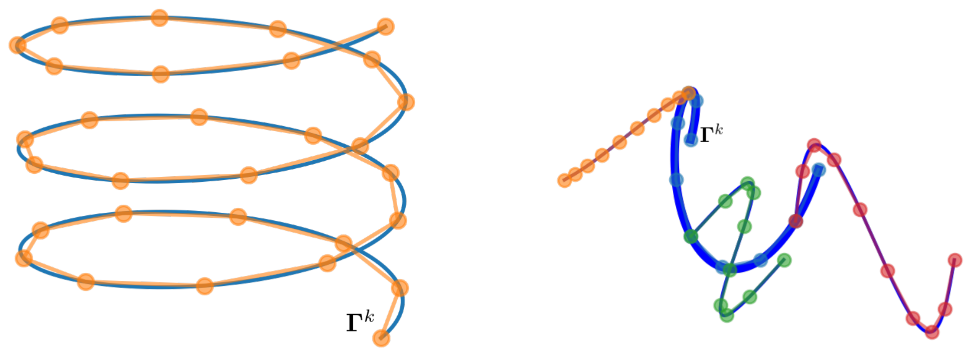
Lemma B.2 (Shifting Laplacian Operator)**.**

$$\langle \Delta f, g \rangle_{L^2} = \langle f, \Delta g \rangle \tag{B.2}$$

Proof.

$$\begin{aligned}
\langle \Delta f, g \rangle_{L^2} &= \int_\Omega g \Delta f \, \mathrm{d}V \\
&= \int_\Omega \left(\nabla \cdot (g \nabla f) - \nabla f \cdot \nabla g \right) \mathrm{d}V \\
&\overset{\text{IBP}}{=} \underbrace{\oint_{\partial\Omega} g \nabla f \cdot \mathbf{n} \, \mathrm{d}s}_{\text{Boundary Term}} - \int_\Omega \nabla f \cdot \nabla g \, \mathrm{d}V \\
&= \underbrace{\oint_{\partial\Omega} g \nabla f \cdot \mathbf{n} \, \mathrm{d}s}_{\text{Boundary Term}} - \langle \nabla f, \nabla g \rangle_{L^2}
\end{aligned}$$

Now, use lemma B.1 and take boundary terms to be zero. ■



(a) A helix is an example of a curve homeomorphic to a line segment, so is in $\mathbb{L}$.

(b) A curve with 3 branches from a main "stem". This curve is homeomorphic to some tree, so is in $\overline{\mathbb{L}}$.

Figure C.1: Curves in different homeomorphism classes.

C Tangent-Point Energy Quadrature: Other Homeomorphism Classes

Because homeomorphism relation is an equivalence relation, one could define equivalence classes. We will call them homeomorphism classes. *We restrict our discussion to bounded curves.*

To make notation more concise, we introduce some labels:

- Write $\mathbb{S}$ for the equivalence class of curves that are simple and closed; in other words, curves homeomorphic to a unit circle.
- Write $\mathbb{L}$ for the equivalence class of curves that are homeomorphic to a line segment.
- Write $\overline{\mathbb{L}}$ for the union of equivalence classes for curves that are homeomorphic to a tree graph, or a "bus network".

We have defined tangent-point energy quadrature for $\boldsymbol{\gamma} \in \mathbb{S}$ at (5.7).

For curves in different homeomorphism classes, one may make minor changes to (5.7) to get a sensible quadrature.

Note that because curves in $\mathbb{L}$ and $\overline{\mathbb{L}}$ may have boundaries unlike curves in $\mathbb{S}$. For the gradient operators derived in section 2.2.1 to be applicable, one assumes natural boundary conditions; where the boundary integrals in lemma B.1 and B.2 to be zero for any g appropriate.

C.1 Curves in $\mathbb{L}$

Given a curve $\boldsymbol{\gamma} \in \mathbb{L}$ such as the helix (Figure C.1(a)), one could take the following as its tangent-point energy quadrature for its discretisation $\boldsymbol{\Gamma}^k$:

$$E^{\alpha}_{\beta}\left(\boldsymbol{\Gamma}^k\right) := \sum_{\substack{i,j\in\{1,\cdots,N-2\}\\ r(i-j,N)>1}} K^{\alpha}_{\beta}(i,j)||e_i||\,||e_j|| \tag{C.1}$$

where the contribution from each end is neglected. Note that unlike a curve from $\mathbb{S}$ one need not generalise the indexing rule to be "cyclic".

C.2 Curves in $\overline{\mathbb{L}}$

For a curve $\boldsymbol{\gamma} \in \overline{\mathbb{L}}$ that has m "branches" from a single "stem", one needs to take into account of at what points on the discretisation branching happens. Suppose for all branches protrude from the stem. On Figure C.1(b), the stem is represented by the thick blue curve.

One can capture this curve by a tensor of the following form.

$$\boldsymbol{\Gamma}^k = \left(\underbrace{\mathbf{x}^k_{0,0},\cdots,\mathbf{x}^k_{0,n_0-1}}_{\text{Stem}}\middle|\underbrace{\mathbf{x}^k_{1,0},\cdots,\mathbf{x}^k_{1,n_1-1}}_{\text{Branch 1}}\middle|\cdots\middle|\underbrace{\mathbf{x}^k_{m,0},\cdots,\mathbf{x}^k_{m,n_m-1}}_{\text{Branch } m}\right) \tag{C.2}$$

where the "linkage points" for branches to the stem are $\mathbf{x}^k_{1,0},\cdots,\mathbf{x}^k_{m,0}$, which can be identified by the fact that $\mathbf{x}^k_{1,0},\cdots,\mathbf{x}^k_{m,0} \in \left\{\mathbf{x}^k_{0,0},\cdots,\mathbf{x}^k_{0,n_0-1}\right\}$, that is, all duplicate points are to be understood as linkage points. Denote the multiset[10] of linkage points as $L\left(\boldsymbol{\Gamma}^k\right)$. Then one may note the relation $N = \sum_{i=0}^{m} n_i - |L\left(\boldsymbol{\Gamma}^k\right)| = \sum_{i=0}^{m} n_i - m$. Note that this is a natural generalisation for curves in $\mathbb{L}$.

Now, one may write a tangent-point energy quadrature as:

$$E^{\alpha}_{\beta}\left(\boldsymbol{\Gamma}^k\right) := \sum_{p,q\in\{0,1,\cdots,m\}} \sum_{\substack{i\in\{1,\cdots,n_p-2\}\\ j\in\{1,\cdots,n_q-2\}\\ \sigma\left(e_{p_i},e_{q_j}\right)=1}} K^{\alpha}_{\beta}(p_i,q_j)||e_{p_i}||\,||e_{q_j}|| \tag{C.3}$$

where p_i (similarly with q_j) refers to the index of vector $\mathbf{x}^k_{p,i}$ in (C.2), that is,

$$p_i = \sum_{j=0}^{p-1} n_j + i$$

and

$$\sigma\left(e_1,e_2\right) = \begin{cases} 0 & \exists \text{ shared vertex between edges } e_1, e_2 \\ 1 & \text{Otherwise} \end{cases}$$

Remark. Representation (C.2) is **not unique**; for example, one could take different curve to be the "stem". The quadrature given by (C.3) is, however, invariant under different representation, hence well-defined.

[10] "a set that allows multiplicity of same elements"

C.3 General Curves: Loops and Trees

One may now attempt to generalise to curves consisting of both closed loops (as in $\mathbb{S}$), and trees (as in $\mathbb{L}$).

While it is possible to write down the generalised quadrature, but it may be more practical to summarise the strategy to build one.

1. Identify the "vertices".
2. Take an (ordered) pair of points on the discretised curve except for the points at boundary (that is, end of the branches or stems).
3. Take two separate edges that involve the two points, but only one each. If the two edges share a vertex, disregard.
4. Sum $K||e_i||\,||e_j||$ over all the pair of points chosen.

D Exact ℓ^2 Gradient of E_β^α

Recall the discretisation of tangent-point energy for simple closed curve $\mathbf{\Gamma}^k$ from (5.7) with K_β^α defined by (5.6).

To compute $\mathrm{Grad}_{\ell^2} E_\beta^\alpha\left(\mathbf{\Gamma}^k\right)$, one can consider subproblem of computing $\nabla_{\mathbf{x}_k} E_\beta^\alpha\left(\mathbf{\Gamma}\right)$ for each $k = 0, 1, \cdots, N-1$.

Fix k. There are $4\,(N-3)$ pairs of indices (i,j) where the summand $K_\beta^\alpha(i,j)||\mathbf{e}_i||\,||\mathbf{e}_j||$ involves $\mathbf{x}_k$ are:

- $(k,0), \cdots, (k,k-2), (k,k+2), \cdots, (k,N-1)$
- $(k-1,0), \cdots, (k-1,k-3), (k-1,k+1), \cdots, (k-1,N-1)$
- $(0,k), \cdots, (k-2,k), (k+2,k), \cdots, (N-1,k)$
- $(0,k-1), \cdots, (k-3,k-1), (k+1,k-1), \cdots, (N-1,k-1)$

We now attempt to construct explicit derivative in a "modular fashion". If we take gradient of the summand directly,[11]

$$\nabla_{\mathbf{x}_k}\left(K_\beta^\alpha(i,j)||\mathbf{e}_i||\,||\mathbf{e}_j||\right) = ||\mathbf{e}_i||\,||\mathbf{x}_j||\nabla_{\mathbf{x}_k} K_\beta^\alpha(i,j) + K_\beta^\alpha\,(i,j)\,\nabla_{\mathbf{x}_k}\left(||\mathbf{e}_i||\,||\mathbf{e}_j||\right) \quad \text{(D.1)}$$

Due to restriction $r\,(i-j,N) > 1$, at most one of $||\mathbf{e}_i||$ and $||\mathbf{x}_j||$ may involve $\mathbf{x}_k$ at a time.

First note that, if $m \neq k$,

$$\nabla_{\mathbf{x}_k}||\mathbf{x}_k - \mathbf{x}_m|| = \frac{\mathbf{x}_k - \mathbf{x}_m}{||\mathbf{x}_k - \mathbf{x}_m||} \quad \text{(D.2)}$$

[11] recall $||\mathbf{e}_i|| = ||\mathbf{x}_i - \mathbf{x}_{i+1}||$ and $||\mathbf{e}_j|| = ||\mathbf{x}_j - \mathbf{x}_{j+1}||$.

Now, the demanding part is to compute $\nabla_{\mathbf{x}_k} k_\beta^\alpha$, since it is needed for computing $\nabla_{\mathbf{x}_k} K_\beta^\alpha$. Note that

$$\begin{aligned}
k_\beta^\alpha\left(\mathbf{x}_p, \mathbf{x}_q, \mathbf{T}_r\right) &= k_\beta^\alpha\left(\mathbf{x}_p, \mathbf{x}_q, \frac{\mathbf{x}_{r+1}-\mathbf{x}_r}{||\mathbf{x}_{r+1}-\mathbf{x}_r||}\right) \\
&= \frac{\sqrt{||\mathbf{x}_{r+1}-\mathbf{x}_r||^2||\mathbf{x}_p-\mathbf{x}_q||^2-\left(\left(\mathbf{x}_{r+1}-\mathbf{x}_r\right)\cdot\left(\mathbf{x}_p-\mathbf{x}_q\right)\right)^2}}{||\mathbf{x}_p-\mathbf{x}_q||^\beta||\mathbf{x}_{r+1}-\mathbf{x}_r||^\alpha} \\
&= \frac{\left(\xi_{p,q,r}\right)^{\alpha/2}}{\eta_{p,q,r}}
\end{aligned} \tag{D.3}$$

where we have defined

$$\begin{aligned}
\xi_{p,q,r} &= ||\mathbf{x}_{r+1}-\mathbf{x}_r||^2||\mathbf{x}_p-\mathbf{x}_q||^2-\left(\left(\mathbf{x}_{r+1}-\mathbf{x}_r\right)\cdot\left(\mathbf{x}_p-\mathbf{x}_q\right)\right)^2 \\
\eta_{p,q,r} &= ||\mathbf{x}_p-\mathbf{x}_q||^\beta||\mathbf{x}_{r+1}-\mathbf{x}_r||^\alpha
\end{aligned}$$

Then, we may express $\nabla_{\mathbf{x}_k} k_\beta^\alpha\left(\mathbf{x}_p, \mathbf{x}_q, \mathbf{T}_r\right)$ as:

$$\begin{aligned}
\nabla_{\mathbf{x}_k} k_\beta^\alpha\left(\mathbf{x}_p, \mathbf{x}_q, \mathbf{T}_r\right) &= \nabla_{\mathbf{x}_k}\left(\frac{\left(\xi_{p,q,r}\right)^{\alpha/2}}{\eta_{p,q,r}}\right) \\
&= \frac{1}{\left(\eta_{p,q,r}\right)^2}\left(\frac{\alpha}{2}\left(\xi_{p,q,r}\right)^{\alpha/2-1}\eta_{p,q,r}\nabla_{\mathbf{x}_k}\xi_{p,q,r}-\left(\xi_{p,q,r}\right)^{\alpha/2}\nabla_{\mathbf{x}_k}\eta_{p,q,r}\right)
\end{aligned} \tag{D.4}$$

It now remains to compute $\nabla_{\mathbf{x}_k}\xi_{p,q,r}$ and $\nabla_{\mathbf{x}_k}\eta_{p,q,r}$ for relevant (p,q,r) tuples.

There are five classes of relevant (p,q,r) tuples.

If $(p,q,r)=(k,j,k)$,

$$\begin{aligned}
\xi_{k,j,k} &= ||\mathbf{x}_{k+1}-\mathbf{x}_k||^2||\mathbf{x}_k-\mathbf{x}_j||^2-\left(\left(\mathbf{x}_{k+1}-\mathbf{x}_k\right)\cdot\left(\mathbf{x}_k-\mathbf{x}_j\right)\right)^2 \\
\eta_{k,j,k} &= ||\mathbf{x}_k-\mathbf{x}_j||^\beta||\mathbf{x}_k-\mathbf{x}_{k+1}||^\alpha \\
\nabla_{\mathbf{x}_k}\xi_{k,j,k} &= 2\left(\mathbf{x}_k-\mathbf{x}_{k+1}\right)||\mathbf{x}_k-\mathbf{x}_j||^2+2||\mathbf{x}_{k+1}-\mathbf{x}_k||^2\left(\mathbf{x}_k-\mathbf{x}_j\right) \\
&\quad -2\left(\left(\mathbf{x}_{k+1}-\mathbf{x}_k\right)\cdot\left(\mathbf{x}_k-\mathbf{x}_j\right)\right)\left(\mathbf{x}_j+\mathbf{x}_{k+1}-2\mathbf{x}_k\right) \\
\nabla_{\mathbf{x}_k}\eta_{k,j,k} &= \beta||\mathbf{x}_k-\mathbf{x}_j||^{\beta-2}||\mathbf{x}_k-\mathbf{x}_{k+1}||^\alpha\left(\mathbf{x}_k-\mathbf{x}_j\right) \\
&\quad +\alpha||\mathbf{x}_k-\mathbf{x}_j||^\beta||\mathbf{x}_k-\mathbf{x}_{k+1}||^{\alpha-2}\left(\mathbf{x}_k-\mathbf{x}_{k+1}\right)
\end{aligned}$$

If $(p,q,r)=(i,j,k)$ where $i\neq k$ and $i\neq k-1$

$$\begin{aligned}
\xi_{i,j,k} &= ||\mathbf{x}_{k+1}-\mathbf{x}_k||^2||\mathbf{x}_i-\mathbf{x}_j||^2-\left(\left(\mathbf{x}_{k+1}-\mathbf{x}_k\right)\cdot\left(\mathbf{x}_i-\mathbf{x}_j\right)\right)^2 \\
\eta_{i,j,k} &= ||\mathbf{x}_i-\mathbf{x}_j||^\beta||\mathbf{x}_k-\mathbf{x}_{k+1}||^\alpha \\
\nabla_{\mathbf{x}_k}\xi_{i,j,k} &= 2||\mathbf{x}_i-\mathbf{x}_j||^2\left(\mathbf{x}_k-\mathbf{x}_{k+1}\right) \\
&\quad -2\left(\left(\mathbf{x}_{k+1}-\mathbf{x}_k\right)\cdot\left(\mathbf{x}_i-\mathbf{x}_j\right)\right)\left(\mathbf{x}_j-\mathbf{x}_i\right) \\
\nabla_{\mathbf{x}_k}\eta_{i,j,k} &= \alpha||\mathbf{x}_i-\mathbf{x}_j||^\beta||\mathbf{x}_k-\mathbf{x}_{k+1}||^{\alpha-2}\left(\mathbf{x}_k-\mathbf{x}_{k+1}\right)
\end{aligned}$$

If $(p, q, r) = (k-1, j, k-1)$,

$$
\begin{aligned}
\xi_{k-1,j,k-1} &= ||\mathbf{x}_k - \mathbf{x}_{k-1}||^2 ||\mathbf{x}_{k-1} - \mathbf{x}_j||^2 - ((\mathbf{x}_k - \mathbf{x}_{k-1}) \cdot (\mathbf{x}_{k-1} - \mathbf{x}_j))^2 \\
\eta_{k-1,j,k-1} &= ||\mathbf{x}_{k-1} - \mathbf{x}_j||^\beta ||\mathbf{x}_{k-1} - \mathbf{x}_k||^\alpha \\
\nabla_{\mathbf{x}_k} \xi_{k-1,j,k-1} &= 2||\mathbf{x}_{k-1} - \mathbf{x}_j||^2 (\mathbf{x}_k - \mathbf{x}_{k-1}) \\
&\quad - 2 ((\mathbf{x}_k - \mathbf{x}_{k-1}) \cdot (\mathbf{x}_{k-1} - \mathbf{x}_j)) (\mathbf{x}_{k-1} - \mathbf{x}_j) \\
\nabla_{\mathbf{x}_k} \eta_{k-1,j,k-1} &= \alpha ||\mathbf{x}_{k-1} - \mathbf{x}_j||^\beta ||\mathbf{x}_k - \mathbf{x}_{k-1}||^{\alpha-2} (\mathbf{x}_k - \mathbf{x}_{k-1})
\end{aligned}
$$

If $(p, q, r) = (k, j, k-1)$,

$$
\begin{aligned}
\xi_{k,j,k-1} &= ||\mathbf{x}_k - \mathbf{x}_{k-1}||^2 ||\mathbf{x}_k - \mathbf{x}_j||^2 - ((\mathbf{x}_k - \mathbf{x}_{k-1}) \cdot (\mathbf{x}_k - \mathbf{x}_j))^2 \\
\eta_{k,j,k-1} &= ||\mathbf{x}_k - \mathbf{x}_j||^\beta ||\mathbf{x}_{k-1} - \mathbf{x}_k||^\alpha \\
\nabla_{\mathbf{x}_k} \xi_{k,j,k-1} &= 2||\mathbf{x}_k - \mathbf{x}_j||^2 (\mathbf{x}_k - \mathbf{x}_{k-1}) \\
&\quad + 2||\mathbf{x}_k - \mathbf{x}_{k-1}||^2 (\mathbf{x}_k - \mathbf{x}_j) \\
&\quad - 2 ((\mathbf{x}_k - \mathbf{x}_{k-1}) \cdot (\mathbf{x}_k - \mathbf{x}_j)) (2\mathbf{x}_k - \mathbf{x}_j - \mathbf{x}_{k-1}) \\
\nabla_{\mathbf{x}_k} \eta_{k,j,k-1} &= \beta ||\mathbf{x}_{k-1} - \mathbf{x}_k||^\alpha ||\mathbf{x}_k - \mathbf{x}_j||^{\beta-2} (\mathbf{x}_k - \mathbf{x}_j) \\
&\quad + \alpha ||\mathbf{x}_k - \mathbf{x}_j||^\beta ||\mathbf{x}_k - \mathbf{x}_{k-1}||^{\alpha-2} (\mathbf{x}_k - \mathbf{x}_{k-1})
\end{aligned}
$$

If $(p, q, r) = (i, k, j)$,

$$
\begin{aligned}
\xi_{i,k,j} &= ||\mathbf{x}_{j+1} - \mathbf{x}_j||^2 ||\mathbf{x}_k - \mathbf{x}_i||^2 - ((\mathbf{x}_{j+1} - \mathbf{x}_j) \cdot (\mathbf{x}_k - \mathbf{x}_i))^2 \\
\eta_{i,k,j} &= ||\mathbf{x}_k - \mathbf{x}_i||^\beta ||\mathbf{x}_j - \mathbf{x}_{j+1}||^\alpha \\
\nabla_{\mathbf{x}_k} \xi_{i,k,j} &= 2||\mathbf{x}_{j+1} - \mathbf{x}_j||^2 (\mathbf{x}_k - \mathbf{x}_i) \\
&\quad - 2 ((\mathbf{x}_{j+1} - \mathbf{x}_j) \cdot (\mathbf{x}_k - \mathbf{x}_i)) (\mathbf{x}_{j+1} - \mathbf{x}_j) \\
\nabla_{\mathbf{x}_k} \eta_{i,k,j} &= \beta ||\mathbf{x}_j - \mathbf{x}_{j+1}||^\alpha ||\mathbf{x}_k - \mathbf{x}_i||^{\beta-2} (\mathbf{x}_k - \mathbf{x}_i)
\end{aligned}
$$

With all these cases covered, one may back substitute to (D.1) $\sim$ (D.4) to acquire the exact gradient by summing over the $4(N-3)$ pairs of indices.

References

[1] Amir Beck. "Chapter 8: Primal and Dual Projected Subgradient Methods". In: *First-Order Methods in Optimization*, pp. 195–245. DOI: 10.1137/1.9781611974997.ch8. eprint: https://epubs.siam.org/doi/pdf/10.1137/1.9781611974997.ch8. URL: https://epubs.siam.org/doi/abs/10.1137/1.9781611974997.ch8.

[2] Gregory Buck and Jeremey Orloff. "A simple energy function for knots". In: *Topology and its Applications* 61.3 (Feb. 1995), pp. 205–214. DOI: 10.1016/0166-8641(94)00024-w.

[3] Stephan Rabanser, Oleksandr Shchur, and Stephan Günnemann. *Introduction to Tensor Decompositions and their Applications in Machine Learning*. 2017. DOI: 10.48550/ARXIV.1711.10781. URL: https://arxiv.org/abs/1711.10781.

[4] Endre Süli. *B6.1 Numerical Solution of Partial Differential Equations*. Lecture Note. 2021.

[5] Chris Yu, Henrik Schumacher, and Keenan Crane. "Repulsive Curves". In: *ACM Transactions on Graphics* 40.2 (Apr. 2021), pp. 1–21. DOI: 10.1145/3439429.