Homework 1: Discrete and Smooth Curves

Due March 3, 2021

> This is the first homework assignment for 6.838. Check the course website for additional materials and the late policy. You may work on assignments in groups, but every student must submit their own write up; please note your collaborators, if any, on your write up. **Submit your code and writeup in a zip file named `6838-hw1-<yourkerberos>.zip`, where `<yourkerberos>` is replaced with your MIT Kerberos ID.**
> Some of the notation used in this homework may be unfamiliar or rusty for computer science students, but undergrad calculus should be sufficient to answer all the problems. Get started early, and reach out for help during office hours and/or on Piazza—we will be generous!

Problem 1 (30 points)**.** In this problem, we introduce continuous and discrete methods in variational calculus, one of the main tools of the differential geometry toolbox.

(a) Suppose you are given a regular plane curve $\gamma : [0,1] \to \mathbb{R}^2$, and take $\mathbf{v} : [0,1] \to \mathbb{R}^2$ to be a vector field along γ. Recall that the arc length of γ is given by

$$s[\gamma] = \int_0^1 \|\gamma'(t)\|_2 \, dt.$$

We can think of $\gamma_h(t) := \gamma(t) + h\mathbf{v}(t)$ to be a displacement of γ along $\mathbf{v}$. Differentiate $f(h) := s[\gamma + h\mathbf{v}]$ with respect to h at $h = 0$ to yield an expression for $\frac{d}{dh} s[\gamma + h\mathbf{v}]|_{h=0}$.

Hint: The "differentiation under the integral sign" rule shows $\frac{d}{dh}\int_a^b g(t,h)\, dt = \int_a^b \frac{\partial g}{\partial h}(t,h)\, dt$.

(b) You can think of each $\mathbf{v}$ as an infinitesimal displacement (a "variation") of the entire curve γ at once. Explain how the derivative you took in (a) can be thought of as a directional derivative of arc length in the $\mathbf{v}$ "direction." In variational calculus, this derivative is known as the Gâteaux or variational derivative of $s[\cdot]$.

(c) Suppose $\mathbf{v}(0) = \mathbf{v}(1) = \mathbf{0}$. Define a vector-valued function $\mathbf{w}(s)$ so that

$$\left.\frac{d}{dh} s[\gamma + h\mathbf{v}]\right|_{h=0} = \int_0^{s(1)} \mathbf{v}(s^{-1}(\bar{s})) \cdot \mathbf{w}(\bar{s})\, d\bar{s},$$

where $s(t) = \int_0^t \|\gamma'(\bar{t})\|_2 \, d\bar{t}$ on the right-hand side is the arc length function and $\mathbf{w}$ can be written in terms of the curvature and Frenet frame of γ.

Note: Your formula for $\mathbf{w}$ should *not* include any term involving $\mathbf{v}$.

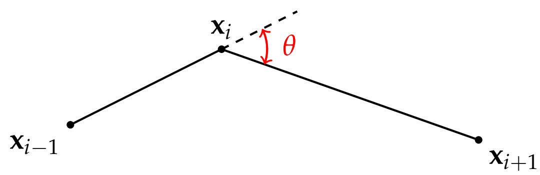
Hint: Simplify the left-hand side using your answer to (a). Use integration by parts.

For coding assignments, you can use either Julia or MATLAB as your programming language. The starter code handles visualization/problem setup and indicates where you should fill in your solutions. If you are using Julia, we recommend the VS Code extension for development. Plots will then appear inside VS Code. If you run your code directly from the terminal, change `WGLMakie` to `GLMakie` so that plots will appear in a separate window.

Problem 2 (35 points)**.** In this problem, you will develop a notion of discrete curvature of a plane curve.

(a) Suppose we have a discrete curve given by a series of points $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^2$. You can think of the vertex positions as parameterized by a vector $\mathbf{x} \in \mathbb{R}^{2n}$. Define an arc length functional $s(\mathbf{x}) : \mathbb{R}^{2n} \to \mathbb{R}_+$; for convenience, it is acceptable to notate $s(\mathbf{x}) = s(\mathbf{x}_1, \ldots, \mathbf{x}_n)$.

(b) Suppose $1 < i < n$. Write an expression for the gradient $\nabla_{\mathbf{x}_i} s$ of $s(\cdot)$ with respect to $\mathbf{x}_i$ and show that its norm is $2 \sin \frac{\theta}{2}$, where θ is the turning angle between the two segments adjacent to $\mathbf{x}_i$ (see figure).



(c) Take a look at `discreteCurve.m` (or `discreteCurve.jl`). The curve generates an $n \times 2$ array representing n points on a discrete two-dimensional curve. Modify the code to plot the derivative you computed in part (b).

(d) Propose a measure of discrete (unsigned) per-vertex curvature of a 2D discrete curve based on your answers to 1(c) and 2(b), and draw the curve colored by this value. For this, fill in code to compute `kappa` in `discreteCurve.m` (`discreteCurve.jl`).

Note: Make sure that your notion of curvature does not go to zero as you increase the number n of samples. Multiple answers are possible.

(e) For sufficiently small $h > 0$, one simple way to decrease the length of the curve would be to replace each point $\mathbf{x}_i$ with a new point $\mathbf{x}_i' := \mathbf{x}_i - (\nabla_{\mathbf{x}_i} s)h$, where $\nabla_{\mathbf{x}_i} s$ is the derivative of s with respect to $\mathbf{x}_i$ (make sure you understand why!). Implement this forward integration scheme with the endpoints fixed, and make sure that if you iterate enough times the curve approximates a straight line. What happens if h is too large?

Problem 3 (35 points)**.** In this problem, you will implement a special case of the "Discrete Elastic Rods" paper discussed in class—a closed loop formed from naturally straight, isotropic rod. Take a look at `elasticRods.m` (or `elasticRods.jl`) for starter code.

(a) Add code to compute the $3 \times 3 \times n$ array `parallelTransport`. Each 3x3 page of this array is a rotation matrix aligning the Bishop frames between successive edges (denoted P_i in the paper).

(b) The bending energy has the form

$$E_{\text{bend}} = \alpha \sum_i \frac{|(\boldsymbol{\kappa b})_i|^2}{\bar{l}_i},$$

where $\bar{l}_i$ is the dual length of vertex i (see §4.2.1 of the paper). The bending force at vertex i is thus

$$-\nabla_i E_{\text{bend}} = \sum_j \frac{-2\alpha}{\bar{l}_j} (\nabla_i (\boldsymbol{\kappa b})_j)^\top (\boldsymbol{\kappa b})_j.$$

Show that

$$\nabla_i (\boldsymbol{\kappa b})_j = \frac{1}{|\bar{\boldsymbol{e}}^{j-1}||\bar{\boldsymbol{e}}^j| + \boldsymbol{e}^{j-1} \cdot \boldsymbol{e}^j} \begin{cases} 2[\boldsymbol{e}^j] + (\boldsymbol{\kappa b})_j (\boldsymbol{e}^j)^\top & i = j-1 \\ 2[\boldsymbol{e}^{j-1}] - (\boldsymbol{\kappa b})_j (\boldsymbol{e}^{j-1})^\top & i = j+1 \\ -(2[\boldsymbol{e}^j] + (\boldsymbol{\kappa b})_j (\boldsymbol{e}^j)^\top + 2[\boldsymbol{e}^{j-1}] - (\boldsymbol{\kappa b})_j (\boldsymbol{e}^{j-1})^\top) & i = j \\ 0 & \text{otherwise,} \end{cases}$$

where $[\boldsymbol{x}]$ denotes a 3×3 matrix defined so that $[\boldsymbol{x}]\boldsymbol{y} = \boldsymbol{x} \times \boldsymbol{y}$ (see §7.1 of the paper).

(c) Implement the functions `computeBendForce` and `computeTwistForce`. Make sure to read the relevant sections in the paper carefully—in particular, note that $\bar{l}_i$ is *twice* the dual cell length $|\mathcal{D}_i|$ (see §4.2.1 in the paper), and pay attention to the transposes in the force equations. To test your work, run the `elasticRods` function.

(d) **Challenge problem** (5 points of the 35). Fixing the material parameters α and β (`bendModulus` and `twistModulus` in the code), plot the maximum displacement over time for a variety of total twist values. Try to determine the critical twist at which the *Michell instability* phase transition occurs. Does it correspond to the theoretical value $\theta = 2\pi\sqrt{3}\alpha/\beta$? Why or why not?