Homework 5: Manifold Optimization and Optimal Transport

Due May 13, 2021

This is the fifth 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 as** `6838-hw5-<yourkerberos>.zip` **and writeup as** `6838-hw5-<yourkerberos>.pdf`, **where** `<yourkerberos>` **is replaced with your MIT Kerberos ID.**

Problem 1 (**Karcher Means on the Sphere** (20 points))**.** Computing averages is a simple and common operation in Euclidean space. The Karcher mean generalizes averages to manifolds. In this problem you will compute Karcher means on the 2D sphere $\mathbb{S}^2 = \{x \in \mathbb{R}^3 : \|x\| = 1\}$.

The Karcher Mean on $\mathbb{S}^2$ of points $z_i \in \mathbb{S}^2$ is defined as the following

$$x^* = \underset{x \in \mathbb{S}^2}{\operatorname{argmin}} \quad \sum_i d(x, z_i)^2 \tag{1}$$

where $d(x, y) = \cos^{-1}(x \cdot y)$ is the distance between two points on $\mathbb{S}^2$.

(a) Compute the Euclidean gradient and Euclidean Hessian of the objective in (**??**).

(b) Let $\Pi_{T_p} : \mathbb{R}^3 \to T_p\mathbb{S}^2$ be the projection of a vector onto $T_p\mathbb{S}^2$, the tangent space of $\mathbb{S}^2$ at $p \in \mathbb{S}^2$. The Riemannian gradient of f is then $\nabla^R f = \Pi_{T_p}(\nabla f)^T$. Note that by convention gradients are row vectors but the transpose makes $\nabla^R f$ a column vector. The Riemannian Hessian is then $H^R f = \Pi_{T_p} \nabla(\nabla^R f)$. Compute the Riemannian gradient and the Riemannian Hessian of the objective in (**??**).

(c) Implement the Riemannian gradient and Riemannian Hessian in `'SphereKarcherMeans'`. (Note: In Matlab, you can use `'checkgradient'` and `'checkhessian'` to verify your answer. `'checkhessian'` can be unstable, so you may have to try a few times.)

(d) Riemannian Trust Regions (RTR) is a second order method for manifold optimization which makes use of the Riemannian Hessian. Steepest descent is a first order method and does not use the Hessian. Solve the Karcher Means problem using `'steepestdescent'` and `'trustregions'`. On average, which is faster and by how much? Is the effort of computing a Hessian in this case worth it?

Problem 2 (**Rotation Synchronization** (20 points))**.** Consider the following generative model. We have N nodes, and to each node i is associated a ground truth rotation matrix $R_i \in \mathrm{SO}(3)$. For each pair (i, j), we observe a relative rotation corrupted by Gaussian noise:

$$O_{ij} = R_{ij} + \sigma\epsilon_{ij} = R_i^\top R_j + \sigma\epsilon_{ij}, \tag{2}$$

where the entries of ϵ_{ij} are i.i.d standard Gaussian for each (i, j).

(a) Given observations O_{ij}, formulate the maximum likelihood estimation problem for estimates of the rotations $\hat{R}_i \in \mathrm{SO}(3)$, the manifold of rotations.

(b) Compute the *Euclidean* gradient and Hessian of your objective function in (a).

(c) Implement your manifold optimization problem in `RotationSynchronization` using MANOPT. Use `GenerateObservations` to get input data. Plot the least squares error of your estimated $\hat{R}_{ij} = \hat{R}_i^\top \hat{R}_j$ against the true relative rotations R_{ij} over various values of σ. How does the method perform? Does it seem to be finding global optima, or is the result dependent on initialization.

Note: in MATLAB, you do not need to calculate the Riemannian gradient or Hessian by hand. MANOPT will do it for you. In Julia, you will need to compute at least the Riemannian gradient.

(d) Show that your problem in (a) can be transformed into a semidefinite program of the following form, with an additional rank constraint:

$$\begin{aligned} \text{maximize} \quad & \langle O, \hat{R} \rangle \\ \text{subject to} \quad & [\hat{R}]_{ii} = I_{3\times 3} \quad \forall i \\ & \hat{R} \succeq 0, \end{aligned} \tag{SDP}$$

where $[\hat{R}]_{ij}$ denotes the (i,j)th 3×3 block of the matrix $\hat{R} \in \mathbb{R}^{3N\times 3N}$, and O is the block matrix of observations, such that $[O]_{ij} = O_{ij}$.

(e) Suppose that (**??**) has a solution $\hat{R}^*$ of rank 3 with each block having positive determinant. Explain how to extract estimated rotations $\hat{R}_i$ from $\hat{R}^*$. Explain why the determinant assumption is reasonable if the noise σ is small.

Problem 3 (**Burer-Monteiro** (20 points))**.** Suppose we instead look for solutions $\hat{R}$ to (**??**) with rank $\hat{R} = r$, where $r \geq 3$. This is equivalent to the following optimization problem (known as the Burer-Monteiro problem):

$$\begin{aligned} \text{maximize} \quad & \operatorname{tr} Y^\top R Y \\ \text{subject to} \quad & [Y]_i [Y]_i^\top = I_{3\times 3} \quad \forall i, \end{aligned} \tag{BM}$$

where $Y \in \mathbb{R}^{3N\times r}$ and $[Y]_i$ denotes the ith $3 \times r$ block of Y.

(a) The *Stiefel manifold* $V_k(\mathbb{R}^d)$, where $k \leq d$, is the manifold of matrices $Y \in \mathbb{R}^{d\times k}$ with orthonormal columns. Explain how you can make a small change to your code from 2(c) to implement problem (**??**).

(b) Copy your code from 2(c) into a new file `RotationSynchronizationBM{.m,.jl}` and modify it to implement the Burer-Monteiro method. For various choices of error standard deviation σ and target rank $r \geq 3$, plot histograms of the rank of the solution Y. *Hint:* you can compute the (numerical) rank of Y using SVD. Under what conditions do you recover a rank-3 solution?

(c) Plot the estimation error as in 2(c). How does the Burer-Monteiro method compare to plain optimization over rotations?

(d) **Extra Credit** (10 points): We say a critical point Y^* of the nonconvex problem (**??**) is *rank-deficient* if rank $Y^* < r$. Suppose Y^* is a rank-deficient local maximum of the quadratic objective function in (**??**) on the constraint manifold. Show that $(Y^*)^\top Y^*$ is a global optimum of (**??**).

Hint: write down the primal-dual optimality conditions for (**??**) and the critical point conditions for (**??**) and compare.

Problem 4 (**Optimal Transport** (40 points))**.** In this problem you will implement the Sinkhorn method for approximating the "earth mover's distance" (EMD) between two probability distributions on a triangle mesh, a.k.a. optimal transport distance.

(a) Suppose we are given a pairwise squared distance matrix $C \in \mathbb{R}^{n\times n}$. C_{ij} measures the distance between bins i and j of a histogram with n bins. For example, $C_{ij} = \|\mathbf{x}_i - \mathbf{y}_j\|_2^2$ for given point sets $\mathbf{x}_1, \ldots, \mathbf{x}_n$ and $\mathbf{y}_1, \ldots, \mathbf{y}_n$. The EMD between histograms $\mathbf{p}$ and $\mathbf{q}$ is defined as

$$W(\mathbf{p},\mathbf{q}) = \begin{cases} \min_{T\in\mathbb{R}^{n\times n}} & \sum_{i=1}^n \sum_{j=1}^n T_{ij}C_{ij} \\ \text{subject to} & T_{ij} \geq 0, \quad \forall i,j \in \{1,\ldots,n\} \\ & \sum_j T_{ij} = p_i, \quad \forall i \in \{1,\ldots,n\} \\ & \sum_i T_{ij} = q_j, \quad \forall j \in \{1,\ldots,n\}. \end{cases} \tag{3}$$

Explain what $W(\mathbf{p},\mathbf{q})$ measures about the difference between $\mathbf{p}$ and $\mathbf{q}$. Compare it to other discrepancy measures between probability distributions such as the Kullback-Leibler divergence.

(b) EMD is difficult to compute when n is large. An alternative is the *entropy-regularized EMD* introduced by Marco Cuturi in **Sinkhorn Distances: Lightspeed Computation of Optimal Transport Distances**. The Sinkhorn distance between $\mathbf{p}$ and $\mathbf{q}$ is given by

$$W_\alpha(\mathbf{p},\mathbf{q}) = \begin{cases} \min_{T\in\mathbb{R}^{n\times n}} & \sum_{i=1}^n \sum_{j=1}^n T_{ij}C_{ij} + \alpha\left(\sum_{ij} T_{ij}\ln T_{ij} - 1\right) \\ \text{subject to} & T_{ij} \geq 0, \quad \forall i,j \in \{1,\ldots,n\} \\ & \sum_j T_{ij} = p_i, \quad \forall i \in \{1,\ldots,n\} \\ & \sum_i T_{ij} = q_j, \quad \forall j \in \{1,\ldots,n\}. \end{cases} \tag{4}$$

Define a matrix K_α in terms of C and α so that the objective for computing $W_\alpha(\mathbf{p},\mathbf{q})$ can be written as $\alpha \cdot \mathrm{KL}(T\|K_\alpha)$, where the KL divergence between $A, B \in \mathbb{R}_+^{n\times n}$ is

$$\mathrm{KL}(A\|B) = \sum_{ij} A_{ij} \ln \frac{A_{ij}}{B_{ij}}$$

(c) Show that the optimal matrix T in the minimization for W_α can be written as $T = \mathrm{diag}(\mathbf{v})K_\alpha\mathrm{diag}(\mathbf{w})$ for some $\mathbf{v}, \mathbf{w} \in \mathbb{R}^n$.

Hint: Use Lagrange multipliers; it may be useful to argue that the $T_{ij} \geq 0$ constraint is no longer necessary after entropic regularization.

(d) So far, we have assumed that we have a pairwise squared distance matrix C. Let's specialized to a triangle mesh, and define $C_{ij} = d(x_i, x_j)^2$ where x_i and x_j are vertices of the mesh and d denotes geodesic distance. Computing the full pairwise squared geodesic distance matrix C is very expensive. In **Convolutional Wasserstein Distances: Efficient Optimal Transportation on Geometric Domains**, Solomon et al. propose an alternative solution.

The heat kernel $\mathcal{H}_t(x, y)$ gives the amount of heat diffusion between $x, y \in M$ after time $t > 0$. In particular $\mathcal{H}_t(x, y)$ solves $\partial_t f_t = \Delta f_t$ with initial condition f_0 through the map

$$f_t(x) = \int_M f_0(y) \mathcal{H}_t(x, y) \mathrm{d}y. \tag{5}$$

We have provided an implementation of heat diffusion in the function `heatDiffusion` which you will use in the final part of this problem.

Varadhan's formula states that the distance on the manifold $d(x, y)$ can be recovered by transferring heat from x to y over a short time interval:

$$d(x, y)^2 = \lim_{t \to 0} [-2t \ln \mathcal{H}_t(x, y)]. \tag{6}$$

Argue that K_α can be approximated as $\mathcal{H}_{\alpha/2}$.

(e) The Sinkhorn algorithm for computing W_α proceeds as follows:

1 Initialize $T^0 \equiv H_{\alpha/2}$.

2 For $i = 1, 2, 3, \ldots$

i. If i is odd, compute

$$T^{(i)} \equiv \begin{cases} \arg\min_{T \in \mathbb{R}^{n \times n}} & \mathrm{KL}(T \| T^{(i-1)}) \\ \text{subject to} & \sum_j T_{ij} = p_i \quad \forall i \in \{1, \ldots, n\}. \end{cases} \tag{7}$$

ii. If i is even, compute

$$T^{(i)} \equiv \begin{cases} \arg\min_{T \in \mathbb{R}^{n \times n}} & \mathrm{KL}(T \| T^{(i-1)}) \\ \text{subject to} & \sum_i T_{ij} = q_j \quad \forall j \in \{1, \ldots, n\}. \end{cases} \tag{8}$$

Show that each $T^{(i)}$ can be written $T^{(i)} = \mathrm{diag}(\mathbf{v}^{(i)}) H_{\alpha/2} \mathrm{diag}(\mathbf{w}^{(i)})$ for some vectors $\mathbf{v}^{(i)}, \mathbf{w}^{(i)} \in \mathbb{R}^n$. Write the steps of the Sinkhorn algorithm in terms of these vectors. Your algorithm should involve only matrix-vector multiplication and per-element operations on vectors (multiplication/division).

(f) Implement the Sinkhorn algorithm in `emd{.m,.jl}` including reasonable stopping criteria. Try several probability distributions on multiple meshes. The example in the starter code should compute geodesic distances from point 1 to all other points on the mesh assuming you have coded everything correctly.