Contents

1 Notes on theory

This document serves to have an idea of the theory behind the project and why certain algorithms have been developed.

The project is mainly based on the paper "Gromov-Wasserstein Averaging of Kernel and Distance Matrices", by G. Peyré, M. Cuturi and J. Solomon.

1.1 Gromov-Wasserstein Averaging of Kernel and Distance Matrices

First of all, let us fix some notation: in the problems presented in this paper, we don't really see the structure of a graph, indeed we will work with: i) a vector μ of length $n \in N$, where n could be thougth as the number of nodes and the i-th element of μ is the importance of the i-th node. We will always assume μ to be non-negative and sum 1, and it is initialized to be the uniform distribution if not specified ii) a square matrix C of dimensione n, whose element C_{ij} represents how far the i-th node is from the j-th. It is improperly called **distance matrix**, because it is not really a distance, in the sense that we don't require it to be symmetric, non-negative and to satisfy the traingular inequality. Most properly we will call it **dissimilarity matrix**. Anyway, in many cases it will be induced by what we will call an **embedding**: to understand this, imagine that the nodes of the graph are point $\{x_i\}$ of R^2 (or R^d more generally) and that $k:R^2 \text{ x } R^2 \to R$ is any function, then we define $C_{ij} = k(x_{i,xj})$.

So, the problems presented in this first part can be defined in the very general setting in which we have a couple (C,μ), as above, but mostly C will be defined using a function k as shown above.

We will call such a couple a **metric measure space**.

1.1.1 Gromov Wasserstein discrepancy

Given a couple of metric measure spaces (C, μ) and (C^*, μ^*), the authors define the discrepancy between them using the definition given by F. Memoli in "Gromov–Wasserstein Distances and the Metric Approach to Object Matching":

$$\mathrm{GW}(C, \mu; C^*, \mu^*) = \min_{T \in \Gamma(\mu, \mu^*)} \sum_{i,j,k,l} L(C_{ik}, C^*_{jl}) T_{ij} T_{kl},$$

where $T \in \Gamma(\mu, \mu^*)$ means that $\sum_j T_{ij} = \mu_i$ and $\sum_i T_{ij} = \mu^*_j$.

The idea of such definition is the following: find a multivariate matching T (also called transport plan,i.e. a function that can assigns more than a value to a single point) between the nodes, in such a way that the weigth μ and μ' are preserved by this matching and $C_{i,j}$ is not "much" different from $C_{T(i),T(j)}$.

In other words, as all the possible matchings vary, they want to minimize the sum of all $L(C_i, j; C_T(i), T(j))$, where L is a fixed loss function, usually initialized to be i) either the common square distance $L(x, y) = |x-y|^2$ ii) or the so called Kullback-Leibler divergence $L(x, y) = KL(a|b) = a \log(a/b) - a + b$

1. Entropic Gromov-Wasserstein discrepancy

   Computationally, the minimization problem above is not so efficient to solve, thus the authors consider a sligth modification of it, adding an **entropic convex**, that must be thought as a regularizing term:

   $$\mathrm{GW}_\varepsilon(C, \mu; C^*, \mu^*) = \min_{T \in \Gamma(\mu, \mu^*)} \sum_{i,j,k,l} L(C_{ik}, C^*_{jl}) T_{ij} T_{kl} - \varepsilon H(T), \quad (1)$$

   where $H(T) = -\sum_{i,j} T_{ij}(\log(T_{ij} - 1))$.

   so that the solution of this problem (and then of the original one) can be evaluated using the projected gradient descent algorithm, which in a particular choice of some constants (i.e. epsilon and tau, we will be in this hypothesis) it can be substituted by the Sinkhorn algorithm, which in time is very a efficient algorithm. [Here I'm referring to eqaution (8) and Proposition 2.]

1.1.2 Gromov-Wasserstein barycenters

Using the entropic Gromov-Wasserstein discrepancy, given a finite collection of metric measure spaces $(C_s, \mu_s)_{s=1}^S$ and a vector of weigths (λ_s), the authors define the barycenter problem as

$$\min_{C \in \mathbb{R}^{N \times N}} \sum_{s=1}^{S} \lambda_s \, \mathrm{GW}_\varepsilon(C, \mu; C_s, \mu_s), \tag{2}$$

where we are assuming that the weight μ of the barycenter is fixed in advance. This is the problem on which we are concentrating, writing in Julia the algorithm they proposed using the theory above. [I'm referring to Algorithm 1.]