

Stat 992: Lecture 08

Kernel smoothing in manifolds.

Moo K. Chung mchung@stat.wisc.edu

February 5, 2004

1. *Heat kernel in polygonal surface.* Assuming the Laplace-Beltrami operator Δ has eigenvalues $0 = \lambda_0 < \lambda_1 < \lambda_2 < \dots$ with corresponding eigenfunctions ϕ_i such that $\Delta\phi_i = -\lambda_i\phi_i$, heat kernel can be constructed as

$$K_\sigma(x, y) = \sum_{i=0}^{\infty} e^{-\lambda_i \sigma} \phi_i(x) \phi_i(y).$$

Then we can construct a kernel smoothing estimator $K_\sigma * Y$; however, this is numerically complicated. Following *parametrix expansion* (Lafferty and Lebanon, 2004; Schoen and Yau, 1994; Rosenberg, 1997; Ekholm, T., 2004). We can approximate the heat kernel locally

$$K_\sigma(x, y) = \frac{1}{(2\pi\sigma)^{1/2}} \exp\left[-\frac{d^2(x, y)}{2\sigma^2}\right] \times [u_0(x, y) + u_1(x, y)\sigma^2 + O(\sigma^4)].$$

Note that $u_0(x, x) = 1$ and $u_1(x, x) = \frac{s(x)}{12}$ where $s(x)$ is the scalar curvature. Now when x and y are close and making σ very small, we can approximate the heat kernel locally

$$K_\sigma(x, y) \approx \frac{1}{(2\pi\sigma)^{1/2}} \exp\left[-\frac{d^2(x, y)}{2\sigma^2}\right].$$

Since this kernel is not normalized in manifolds, we normalize and truncate it

$$\tilde{K}_\sigma(x, y) = \frac{\exp\left[-\frac{d^2(x, y)}{2\sigma^2}\right] \mathbf{1}_B}{\int_B \exp\left[-\frac{d^2(x, y)}{2\sigma^2}\right] dx dy}$$

for some region B . This is basically a continuous version of the Nadaraya-Watson kernel estimator.

2. *Numerical implementation in polygonal surface.* In polygonal surfaces, it is easy to construct isotropic kernel by iteratively constructing it in adjacent polygons. The usual polygonal surface looks like Figure 1. Let $M_{poly} = (V, E, T)$ be the discrete representation of of manifolds M where V, E, T

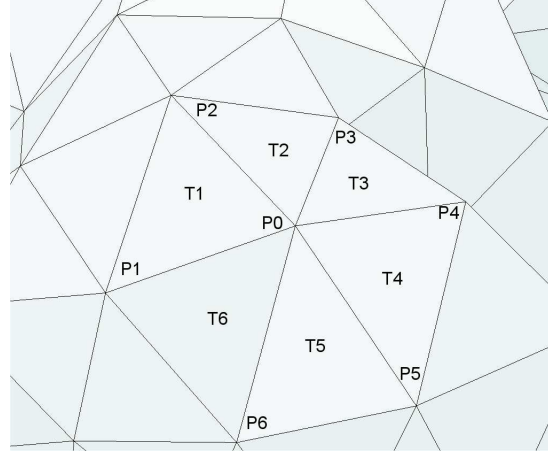


Figure 1: The usual mesh in the neighborhood of vertex p_0 .

are the set of vertices, edges and triangles that forms the polygonal surface respectively. Let n_T be the number of triangles and n_V be the number of vertices in M_{poly} . Consider a stochastic model $Y(p_0) = \mu(p_0) + \epsilon(p_0), p_0 \in V$. Let $p_1, \dots, p_m$ be the adjacent vertices to p_0 . In polygonal surface, the geodesic curve between two adjacent vertices is the edge connecting them so

$$d(p_i, p_0) = (p_i - p_0)'(p_i - p_0)$$

if p_i are represented as coordinate vectors. The kernel estimator is then defined as

$$\hat{\mu}(p_0) = K_\sigma * Y(p_0) = \sum_{i=0}^m K_\sigma(p_i, p_0) Y(p_i),$$

where the weights are

$$K_\sigma(p_i, p_0) = \frac{\exp\left(-\frac{(p_i - p_0)'(p_i - p_0)}{2\sigma^2}\right)}{\sum_{i=1}^m \exp\left(-\frac{(p_i - p_0)'(p_i - p_0)}{2\sigma^2}\right)}.$$

Afterward we smooth via iterated kernel smoothing scheme. For simulation, we used small bandwidth 0.5 mm with 200 iteration which gives effective bandwidth of 7 mm. This seems very inefficient way of smoothing but this method avoid

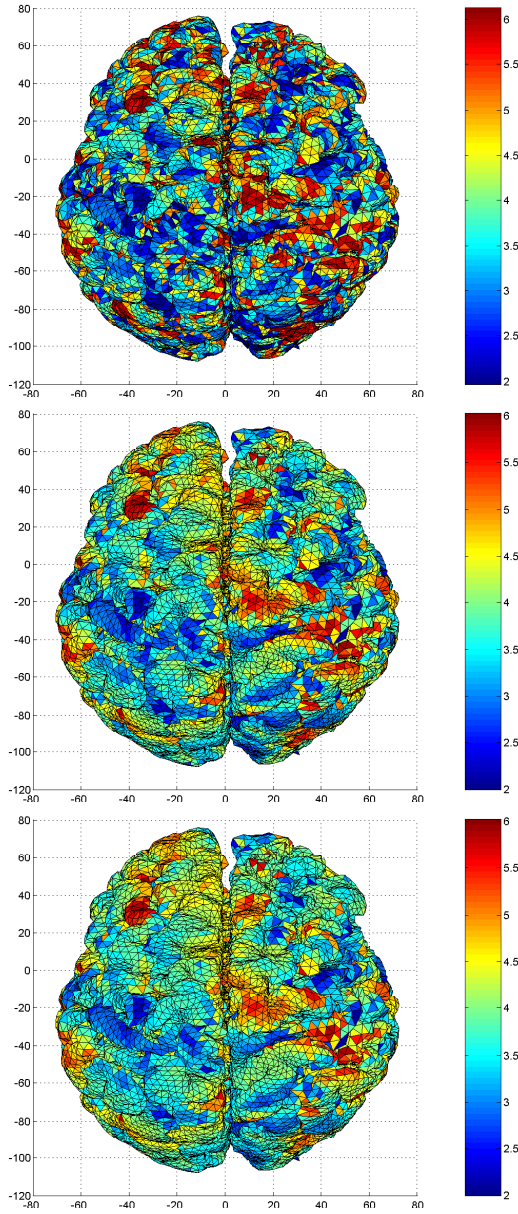


Figure 2: Top: original thickness measure. Middle: 100 iterations with $\sigma = 0.5$. Bottom: 200 iterations with $\sigma = 0.5$.

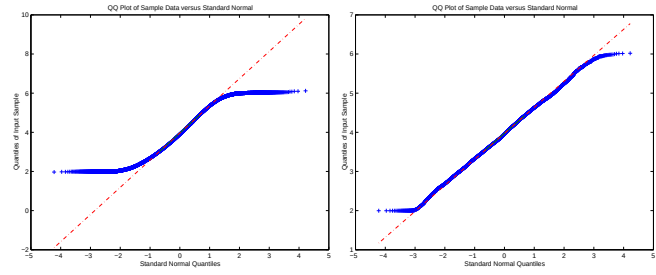


Figure 3: Left: QQ-plot of the original cortical thickness measure. Right: QQ-plot after iterated kernel smoothing with 100 iterations with $\sigma = 0.5$ mm that is equivalent to 5 mm filter.

finding the geodesics on mesh surfaces. To speed up computation, we compute kernel weights for every vertices and it took less than 18 seconds in average for 81920 triangle mesh in Pentium 1.5Ghz M-processor. For the iterated kernel smoothing itself, it took less than 12 seconds in average for 200 iterations.

There are many reasons why we apply Gaussian kernel smoothing in imaging analysis. One reason has to do with the distributional assumptions of data. It is a quick way to convert non Gaussian data into normally distributed data.

Problem 17. Prove this fact. :)