Curvature-aware Regularization on Riemannian Submanifolds

Kwang In Kim James Tompkin Christian Theobalt
Max-Planck-Institut für Informatik

Abstract

One fundamental assumption in object recognition as well as in other computer vision and pattern recognition problems is that the data generation process lies on a manifold and that it respects the intrinsic geometry *of the manifold. This assumption is held in several successful algorithms for diffusion and regularization, in particular, in graph-Laplacian-based algorithms. We claim that the performance of existing algorithms can be improved if we additionally account for how the manifold is embedded within the ambient space, i.e., if we consider the* extrinsic geometry *of the manifold. We present a procedure for characterizing the extrinsic (as well as intrinsic) curvature of a manifold M which is described by a sampled point cloud in a high-dimensional Euclidean space. Once estimated, we use this characterization in general diffusion and regularization on M, and form a new regularizer on a point cloud. The resulting* re-weighted graph Laplacian *demonstrates superior performance over classical graph Laplacian in semi-supervised learning and spectral clustering.*

1. Introduction

One of the fundamental assumptions in manifold-based data processing algorithms is that the intrinsic geometry of a manifold is relevant to the data which lie upon it. For instance, the graph Laplacian matrix is used to measure the pair-wise dissimilarities of the evaluation of a function f on a given point cloud $\mathcal{X}$, and subsequently this can be used for discretized diffusion and regularization of f on $\mathcal{X}$. One way of justifying the use of the graph Laplacian comes from its limit case behavior as $|\mathcal{X}| \to \infty$: When the data $\mathcal{X}$ is generated from an underlying manifold M, *i.e.*, when the corresponding probability distribution P has support in M, the graph Laplacian converges to the Laplace-Beltrami operator [2, 10] that respects only the intrinsic geometry of M. Accordingly, for a large $\mathcal{X}$, the graph Laplacian helps us measure the variation of functions along M and neglect any random perturbations normal to M that might be irrelevant noise. Graph Laplacian and other manifold-based approaches (*e.g.*, [7, 13]) are justified in exploiting intrinsic geometry by successes in semi-supervised learning, spectral clustering, and dimensionality reduction applications.

In this paper, we question a fundamental assumption of manifold-based algorithms. It is well known that the extrinsic geometry of M, that is, how M is embedded in an ambient space, is important for image and mesh surface processing. However, is the extrinsic geometry relevant at all for high-dimensional data processing? Our main contribution is to suggest that the answer might be *yes*.

The anisotropic diffusion process on manifolds motivates our question above, and connects the high-dimensional data processing problem with low-dimensional image and mesh surface processing (Sec. 2). The anisotropic diffusion process exploits the extrinsic (as well as intrinsic) geometry, and we discuss how this can be extended to any sub-manifold with arbitrary dimension and co-dimension. This presents a practical diffusion and regularization scheme which can be applied even when the manifold is not observed directly but is indirectly presented as a sampled point cloud (Sec. 3). This regularization leads to a *re-weighted graph Laplacian*, which we evaluate in the context of semi-supervised learning and spectral clustering, and discover that the new algorithm significantly improves the performance over classical graph Laplacian (Sec. 5).

2. Anisotropic diffusion on manifolds

The Laplace-Beltrami operator Δ on a manifold M is defined from the *divergence* and *gradient* operators:

$$\Delta f = -\operatorname{div}\operatorname{grad} f, \tag{1}$$

where f is a smooth function on M. This is one of the most important operators in differential geometry and is applied to describe physical phenomena on M. In particular, it is the generator of the isotropic diffusion process:

$$\frac{\partial f}{\partial t} = -\Delta f \tag{2}$$

which describes the evolution of f on M as how the values of f *spread* over time. This process is isotropic and homogeneous in the sense that the diffusivity is the same for

any location and any direction on M. When f represents an image as a two-dimensional manifold embedded in $\mathbb{R}^3$ $(x, y, f(x, y))$, it can be shown that evolving f according to Equ. 2 has an effect corresponding to convolution with Gaussian kernels [19, 17].

It is often desirable to non-uniformly distribute the diffusivity, as shown by many image processing applications. For instance, in image denoising, important structures such as edges should remain unchanged, so diffusion should be weak near edges. Furthermore, diffusion should be stronger in the direction along edges rather than across edges. This can be realized with *anisotropic* diffusion on $\mathbb{R}^2$ [19, 18]:

$$\frac{\partial f}{\partial t} = -\Delta_D f := \operatorname{div} D \operatorname{grad} f, \tag{3}$$

where D is a positive definite operator that controls the strength and direction of diffusion. For instance, Weickert *et al*. [19], construct D from the tensor product of grad$f$ with itself. In this case, D depends on f and Equ. 3 becomes a non-linear equation. A similar approach has also been taken for processing a two-dimensional surface embedded in $\mathbb{R}^3$. A typical application is surface processing where f represents the three-dimensional locations of sampled surface points in $\mathbb{R}^3$ [6, 5, 21]. In this context, D can be constructed based on how the surfaces are curved in $\mathbb{R}^3$. We wish the diffusivity to be strong for planar regions and weak across highly curved regions. For instance, Clarenz *et al*. [5] proposed constructing D based on the principal curvatures and the corresponding principal directions at each location on the surface. The resulting diffusion process smooths flat regions and enhances ridges on the surface. A similar effect can also be obtained by diffusing surface normal vectors using mean and Gaussian curvatures [21].

Anisotropic diffusion has been successful in processing two-dimensional objects embedded in $\mathbb{R}^3$ such as images and surfaces (in which the normal is uniquely defined up to the change of sign); however, its application to high-dimensional data has not yet been explored. The aim of our paper is to extend this framework to construct a generator of anisotropic diffusion processes (Δ_D) and, with it, to build a discretized anisotropic regularizer on $\mathcal{X}$.

We first note that the Laplace-Beltrami operator (Equ. 1) can also be used as a regularizer on a manifold. It can be used to measure the first-order variation of f on M:

$$\|f\|_\Delta^2 := \int_M f(x)\Delta f(x) dV(x) = \int_M \|\nabla f\|_g^2 dV(x), \tag{4}$$

where g and dV are the Riemannian metric and the *natural volume element* [15] of M, respectively. The connection between the two aspects of Δ as a regularizer and as a generator of isotropic diffusion processes on M is well established: Intuitively, from the regularization perspective, minimizing $\|f\|_\Delta^2$ corresponds to penalizing the variation of f isotropically. More rigorous discussion is available [19, 11]. Extending this connection to anisotropic diffusion (Equ. 3) is straightforward: with Δ_D as a regularizer, we emphasize the variation of f along the direction of high diffusivity.

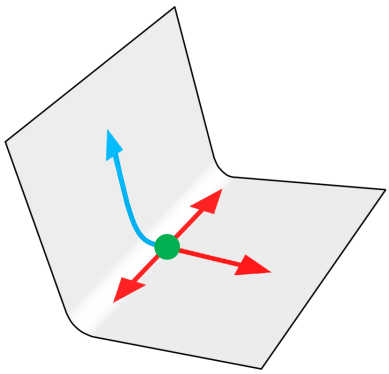

Figure 1. A manifold with high extrinsic curvature and zero intrinsic curvature at the green dot. Since it has zero intrinsic curvature here, intrinsically it is equivalent to $\mathbb{R}^2$.

The success of anisotropic diffusion on images and surfaces and the reinterpretation of the Laplace-Beltrami operator as a regularizer leads us to the conjecture that it would be desirable to perform *anisotropic regularization* on manifolds of any dimension and co-dimension: Regularization should be weak along paths in regions with high (extrinsic and/or intrinsic) curvature. Fig. 1 visualizes the underlying idea with an example of a two-dimensional surface embedded in $\mathbb{R}^3$. In this example, the red arrows pass through planar regions, and here diffusivity should be strong in the directions of the red arrows. Conversely, the blue arrow passes through a highly extrinsically curved region that corresponds to a boundary between two manifolds. Here, the diffusivity should be weak in the direction of the blue arrow. However, existing manifold-based data diffusion and regularization operators are not capable of this (*e.g*., the Laplace-Beltrami or the Hessian [7] operators). These operators respect only intrinsic geometry. Since the surface in Fig. 1 is intrinsically identical to $\mathbb{R}^2$, these operators do not distinguish between the two spaces. In particular, diffusivity is the same at every point in the surface and in $\mathbb{R}^2$.[1]

This constructed example intuition extends to real problems like pattern classification. Fig. 2 shows the results of our preliminary pattern classification experiment, where the directions of estimated high curvature are often perpendicular to the directions of class decision boundaries. This supports the idea of controlling diffusivity based on the direction and strength of both intrinsic and extrinsic curvature.

To build upon this, we next discuss a procedure which estimates the extrinsic and intrinsic curvature and, with it, develops a practical regularization operator on a manifold.

[1] In this extreme example, only extrinsic curvature exists. In general, sub-manifold curvature manifests both intrinsically and extrinsically.

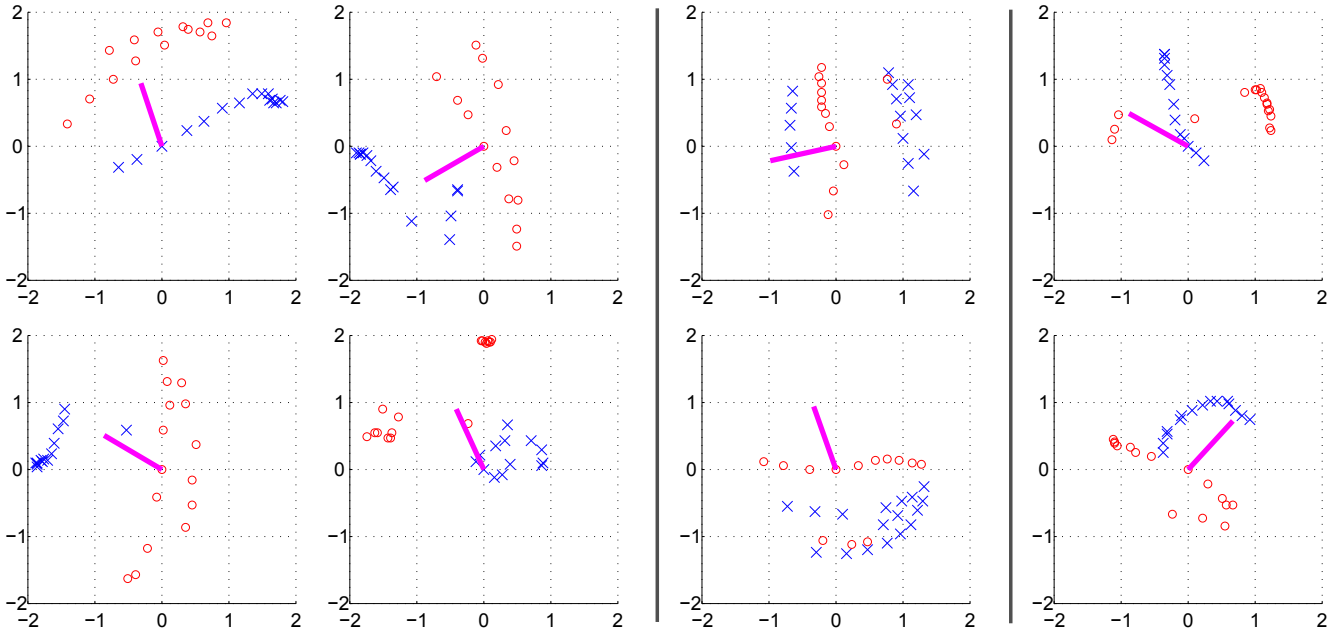

Figure 2. Directions of high curvature for COLT2 database (see Sec. 5): Each plot shows the nearest neighbors of a data point (at the origin) projected onto Riemannian normal coordinates. We fixed the tangent space dimensionality at 2; in practice, it must be determined, *e.g.*, based on cross-validation. The data points are sampled randomly from a set of *difficult* points, the neighborhoods of which include significant variation in ground truth labels. Circles and crosses represent two classes while magenta lines show the direction of highest curvature. This direction is the first eigenvector of the generalized shape operator. Estimated high curvature directions are often perpendicular to decision boundary directions: *First two columns*: the directions are strongly inversely correlated. *Third column*: the directions pass through multiple decision boundaries but are still perpendicular. *Last column*: the directions are less strongly correlated but still reasonable.

3. Curvature-aware regularization

In general, the curvature of a Riemannian manifold M is captured by a fourth-order tensor called the Riemann curvature tensor. Then, how the manifold M (of dimension m) is curved with respect to the ambient manifold $\widetilde{M}$ (of dimension n), is characterized by the difference of the corresponding curvature tensors. A theorem of Gauss [15] states that this quantity is completely determined by a third-order operator called the *second fundamental form*. Suppose ∇ and $\widetilde{\nabla}$ are the (Riemannian) *connections* in M and $\widetilde{M}$, respectively. The second fundamental form $II : \mathcal{T}(M) \times \mathcal{T}(M) \to \mathcal{N}(M)$, with $\mathcal{T}(M)$ and $\mathcal{N}(M)$ being the tangent and normal bundles of M in $\widetilde{M}$ respectively, quantifies how the ambient derivative $\widetilde{\nabla}$ deviates from the intrinsic derivative ∇: For $X, Y \in \mathcal{T}(M)$:

$$\widetilde{\nabla}_X Y = \nabla_X Y + II(X, Y). \tag{5}$$

At each point $p \in M$, evaluating $II(X, Y)$ corresponds to projecting $(\widetilde{\nabla}_X Y)$ onto normal space $N_p(M) \subset \mathcal{N}(M)$.

To facilitate the subsequent computation of II and to gain a deeper insight into its geometric characteristics, we represent II in a special coordinate frame. The analysis in the reminder of this section focuses entirely on a coordinate chart at a point $p \in M$ and, accordingly, without loss of generality we focus on II evaluated at p. For simplicity of notation, we omit the specifier for p, *e.g.*, $T(M)$ actually means $T_p(M) \subset \mathcal{T}(M)$.

First, we construct an *adapted orthonormal frame* [14, 15] $\{Y_1, \ldots, Y_n\}$ which specifies an orthonormal coordinate chart $\{y^1, \ldots, y^n\}$ centered at p in $\widetilde{M}$ such that $\{\frac{\partial}{\partial y^1}, \ldots, \frac{\partial}{\partial y^m}\} = \{Y_1, \ldots, Y_m\}$ spans the tangent space $T_p(M)$ of M_p. In particular, we use Riemannian normal coordinates in $\widetilde{M}$. Suppose that at an open neighborhood U_p (of p) in M, embedding $i : M \to \widetilde{M}$ is represented by:

$$y^i = y^i(x^1, \ldots, x^m) \text{ for } i = 1, \ldots, n, \tag{6}$$

where $\{x^1, \ldots, x^m\}$ is a coordinate chart at U_p. Then, in the combined coordinates $\{x^1, \ldots, x^m, y^{m+1}, \ldots, y^n\}$, the second fundamental form has a particularly simple form:

$$II = \sum_{r,s=1}^{m} \sum_{i=m+1}^{n} \left[\left(\frac{\partial^2 y^i}{\partial x^r \partial x^s} \right) dx^r dx^s \right] Y_i. \tag{7}$$

This representation not only facilitates the subsequent computation but also clearly manifests the geometrical significance of the second fundamental form: II corresponds to the sum of the Hessians $\left(\frac{\partial^2 y^i}{\partial x^r \partial x^s} \right)$ of coordinate values $\{y^i(x^1, \ldots, x^m), i = m+1, \ldots, n\}$ at p as hyper-surfaces,

each of which characterizes how the corresponding surface is bending, *i.e.*, the curvature.

This simplicity in the representation of II is due to the use of the Riemannian normal coordinate in $\widetilde{M}$, in which the manifold appears Euclidean up to second-order and, accordingly, the corresponding Riemannian metric $\widetilde{g}$ becomes Euclidean (at p). In general coordinates, the Christoffel symbols $\widetilde{\Gamma}^i_{jk}$ corresponding to $\widetilde{\nabla}$ appear in Equation 7.

Generalized shape operators. We have just seen how to characterize the curvature of any arbitrary Riemannian submanifold M with codimensionality higher than 1. Our next step is to build a generalization of the operator $D_p : T(M) \to T(M)$ in Equation 3 using the second fundamental form II. First, we raise the first index of II in M:

$$II^\sharp := \sum_{r,s,\delta=1}^{m} \sum_{i=m+1}^{n} \left[\left(\frac{\partial^2 y^i}{\partial x^r \partial x^s} \right) g^{r\delta} \partial_\delta dx^s \right] Y_i. \quad (8)$$

Then, the *generalized (absolute) shape operator* $s : T(M) \to T(M)$ is constructed by casting the individual Hessians positive definite[2] and removing the normal component by taking the inner product of the normal component of $II^\sharp$ with $\sum_{i=m+1}^{n} Y_i$. To cast, we take the absolute values of eigenvalues of each Hessian H^i in $\{x^i\}$:

$$s = \sum_{r,s,\delta=1}^{m} \sum_{i=m+1}^{n} \left[|H^i|_P \right]_{rs} g^{r\delta} \partial_\delta dx^s, \quad (9)$$

where $|A|_P$ is a positive definite version of a matrix A. The last step makes s depending on the choice of the normal frame $\{Y_i\}_{i=m+1}^n$ which we fix by exploiting the distribution of the data on M (see Sec. 4: *estimating the normal coordinate* paragraph).[3]

In informal terms, s receives a vector $Z_p \in T_pM$ and magnifies or reduces in x each of its components $\{Z^i\}$ depending on how the corresponding coordinate directions $\{\frac{\partial}{\partial x^i}\}$ are curved in $\{y^{m+1}, \dots, y^n\}$. In particular, when $\widetilde{M} = \mathbb{R}^n$ and we construct a geodesic $c : (-\epsilon, \epsilon) \to M$ of a unit vector Z_p (*i.e.*, $\|Z_p\| = 1$, $c(0) = p$, and $\dot{c}(p) = Z_p$), $\|sZ_p\|$ corresponds to the curvature of $c(0)$ where c is interpreted as a one-dimensional submanifold of $\mathbb{R}^n$.

[2] We do not use the sign of the curvature. Accordingly, the corresponding diffusivity depends only on the curvature direction and magnitude. This operation is geometric, see [20].

[3] Another way of constructing the orthonormal frame $\{Y_{m+1}, \dots, Y_n\} \subset \mathcal{N}(M)$ is to choose each normal vector Y_i incrementally by maximizing the squared norm of Y_i-component $\left\| \sum_{r,s=1}^{m} \left[\left(\frac{\partial^2 y^i}{\partial x^r \partial x^s} \right) dx^r dx^s \right] \right\|^2_{T_p^*(M) \otimes T_p^*(M)}$ with the orthogonality condition (*i.e.* $\tilde{g}(Y_i, Y_j) = 0$ for $j < i$ and $i, j \in m+1, \dots, n$), where $T_p^*(M)$ is the cotangent space of M at p. This choice makes the resulting shape operator s entirely geometric but it is computationally more demanding than our method.

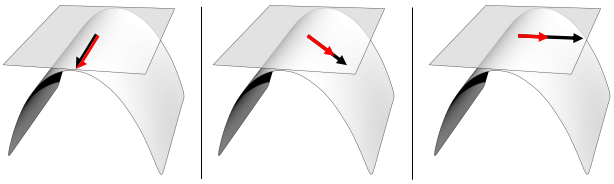

Figure 3. Examples of applying the diffusivity operator D_p to vectors in $T_p(M)$: Evaluation point p is at the origin of each vector arrow. A two-dimensional surface manifold M embedded in $\mathbb{R}^3$ is generated by bending a plane along a fixed direction. The tangent space $T_p(M)$ is shown as a transparent plane. (Left) When the black input vector is orthogonal to the bending direction along which M has no curvature, the resulting red output vector is identical to the input; (Right) If the input is parallel to the bending direction along which M is maximally curved, the output is maximally shrunken. In particular, when the curvature is infinite, the output vector is zero. (Middle) In general, the input vector is shrunk depending on how M is curved along the direction of input.

This operation is exactly opposite to what we would like to perform: it expands the vector into the direction of high curvature. Finally, our vector-valued diffusivity operator D_p is constructed as an (pseudo-) inverse of s:

$$D_p = (S_p + I)^{-1}, \quad (10)$$

where S_p is a matrix representation of s at p in x. From Eq. 9 and with the positive definiteness of g, then D_p is a positive definite operator. When the curvature is zero in any direction (*i.e.*, $S_p = 0$), the diffusivity is maximum ($D_p = I$). Otherwise, the input Z_p is shrunken down depending on the curvature of M along the direction of Z_p. For instance, at a point p lying on the surface generated by bending a plane in $\mathbb{R}^3$ along a specific direction (Fig. 3), $\|D_p(Z_p)\|$ is the maximum and the minimum when Z_p is orthogonal to and parallel with the direction of bending, respectively, and is smaller than $\|Z_p\|$, otherwise. Based on this observation, we construct a scalar-valued diffusivity operator d_p as:

$$d_p(Z_p) = \frac{\|D_p(Z_p)\|}{\|Z_p\|}. \quad (11)$$

Even though d_p is scalar-valued and so the corresponding vector-valued diffusivity operator D_p^d ($D_p^d(Z_p) := d_p(Z_p)Z_p$) does not change the direction of the input vector Z_p, it still leads to anisotropic diffusion since the corresponding output depends on the *direction* of Z_p.

In general, one could construct D_p and d_p as any nonlinear function of S_p. Depending on this non-linearity, the input vector Z_p can even be elongated resulting in, *e.g.*, edge-sharpening diffusion in the case of images [19].

4. Regularization on a point cloud in $\mathbb{R}^n$

In many practical applications, the manifold M is not directly observed but is indirectly observed as a sampled

point cloud $\mathcal{X} = \{X_i\}_{i=1}^{l} \subset M \subset \mathbb{R}^n$, and accordingly, M is isometrically embedded in $\widetilde{M} = \mathbb{R}^n$. As the manifold is no longer analytically observed, we denote a point in the point cloud as X instead of $p \in M$. This section presents a practical regularization scheme for this case.

Estimating the normal coordinates. To facilitate the evaluation of s_{X_α} at point $X_\alpha \in \mathcal{X}$, we introduce Riemannian normal coordinates for M and $\widetilde{M}$ at X_α. For notation convenience, X denotes a point in M and the corresponding embedded point $i(X)$ in $\mathbb{R}^n$. In $\mathbb{R}^n$, every orthonormal basis $\{Y_1, \ldots, Y_n\}$ leads to normal coordinates $\{y^1, \ldots, y^n\}$ based on the identification $\frac{\partial}{\partial y^i} = Y_i$ for $i = 1, \ldots, n$. For M, similarly to [13], we approximate the normal coordinates based on the distribution of the data in neighborhood structure: First, the tangent space $T_{X_\alpha}M$ is estimated by performing principal component analysis (PCA) on k nearest neighbors $N_k(X_\alpha)$ of X_α. The m leading eigenvectors $\{u_r\}_{r=1}^{m}$ span $N_k(X_\alpha)$. Then, the normal coordinates $\{x_r\}_{r=1}^{m}$ of a point X_j centered at X_α are given as:

$$x_r(X_j) = \langle u_r, X_j - X_\alpha \rangle . \tag{12}$$

In practice, we may not know the dimensionality m of M. In this case, we regard it as a hyper-parameter to be tuned for subsequent applications.

Since $\{u_r\}_{r=1}^{n}$ constitutes an orthonormal basis in $\mathbb{R}^n$, the corresponding normal coordinates $\{y_r\}_{r=1}^{n}$ are given as:

$$y_r(X_j) = \langle u_r, X_j - X_\alpha \rangle \tag{13}$$

which corresponds to fixing the open parameters (adapted orthonormal frames $\{Y_r\}_{r=m+1}^{n}$) by $\{u_r\}_{r=m+1}^{n}$ in constructing s (Eq. 9).

Estimating the Hessian. In normal coordinates (x), the metric g becomes Euclidean. Accordingly, the calculation of the shape operator (Equation 9) boils down to the estimation of the classical Hessian.

Similarly to [13], this can be estimated by fitting a second-order polynomial $q(x)$ to $\{y^i(X_j)\}_{j=1}^{k}, X_j \in N_k(X_\alpha)$:

$$q^{(i)}(x) = \sum_{r}^{m} \sum_{s=r}^{m} A_{rs}^{(i)} x_r x_s, \tag{14}$$

where the zeroth-order and the first-order terms are fixed at 0.[4] In the limit, as the neighborhood size tends to zero, $q^{(i)}(x)$ becomes the second-order Taylor expansion of y^i around X_α. In particular,

$$A_{rs}^{(i)} = \frac{1}{2} \frac{\partial^2 y^i}{\partial x^r \partial x^s}\Big|_{X_\alpha}. \tag{15}$$

[4]By construction, $\{\frac{\partial}{\partial x^j}\}_{j=1}^{m}$ are tangent to M at X_α. As such, the variation of $\{y^i\}_{i=m+1}^{n}$ with respect to $\{x^j\}_{j=1}^{m}$ is zero up to first order.

We use standard linear least squares to fit the polynomial:

$$q^{(i)}(x_j) = \arg\min_{w \in \mathbb{R}^P} \sum_{j=1}^{k} \left(y^i(X_j) - (\phi(X_j)w)_j \right)^2, \tag{16}$$

where $P = m(m+1)/2$ and the basis function ϕ extracts the monomials of the normal coordinates centered at X_α: $\phi(X_j) = [x_1x_1, x_1x_2, \ldots, x_mx_m]$ for $X_j \in N_k(X_\alpha)$. With $\Phi = [\phi(X_1), \ldots, \phi(X_k)]^\top$, and $\mathbf{f}_\alpha = [f(X_1), \ldots, f(X_k)]^\top$, the solution is obtained as:

$$w = \Phi^+ \mathbf{f}_\alpha, \tag{17}$$

where Φ^+ denotes the pseudo-inverse of Φ.

Convergence properties. In general, from the analysis of Hein et al. [10], given the exact tangent space $T_{X_\alpha}M$, the approximation of normal coordinate values based on PCA yields an error of $O(\epsilon^2)$ where ϵ^2 is the radius of $N_k(X_i)$. Further, the Hessian corresponding to the fitted local polynomial may deviate from the true Hessian.

In the supplementary material, we prove that, for a submanifold M with a bounded second fundamental form and for a reasonably general assumption on the underlying probability distribution P on $M \subset \mathbb{R}^n$ (see [1] for details), the estimated second fundamental form and the shape operator converge point-wise to the true second fundamental form and the shape operator as the number of data points tends to infinity, while the diameter of the neighborhood $N_k(X_\alpha)$ tends to zero. We also demonstrate this with a toy example.

Re-weighted graph Laplacian. The constructed approximation of D_p (using Equs. 9, 10, and 15) can be straightforwardly applied for anisotropic diffusion and regularization by replacing D in Equation 3 with D_p at each p. The corresponding Laplace-Beltrami operator (Δ_D) can either be explicitly constructed (*e.g.*, for regularization) or indirectly evaluated (*e.g.*, for diffusion). In either case, we must be able to construct a vector D_pZ for an input vector Z.

However, in many practical applications, no gradient vector is explicitly constructed. For instance, the graph Laplacian L of $\mathcal{X} \subset \mathbb{R}^n$ as an approximation of the Laplace-Beltrami operator on M is constructed based only on pairwise similarity evaluations in $\mathbb{R}^n$:

$$L = G - W, \tag{18}$$

where $[W]_{\alpha\beta} = w(\|X_\alpha - X_\beta\|)$, $w : \mathbb{R}^n \to \mathbb{R}$ is a decreasing function, and G is a diagonal matrix containing the column sums of W.

In this case, the scalar-valued operator d_p (11) can be used instead: In graph Laplacian-based regularization, one minimizes $\mathbf{f}^\top L \mathbf{f}$ where $\mathbf{f} = [f(X_1), \ldots, f(X_l)]^\top$. This corresponds to penalizing the pair-wise deviations of the

evaluations of f. The amount of penalization $[W]_{\alpha\beta}$ for a pair $(f(X_\alpha), f(X_\beta))$ is proportional to the *length* of a *vector* $X_\beta - X_\alpha \in \mathbb{R}^n = T_{X_\alpha}(\mathbb{R}^n)$. As in the construction of normal coordinates in M, when $X_\beta \in N_k(X_\alpha)$, $X_\beta - X_\alpha$[5] can be regarded as an approximation of a vector lying in $T_{X_\alpha}(M)$ after a suitable projection onto $T_{X_\alpha}(M)$ (which will be denoted as Z_β). Now we can apply the scalar-valued operator d_{X_α} to Z_β. This may result in scaling Z_β but does not change its direction. However, instead of explicitly constructing $d_{X_\alpha}(Z_\beta)$, we scale $[W]_{\alpha\beta}$ depending on $\|d_{X_\alpha}(Z_\beta)\|$ since $[W]_{\alpha\beta}$ completely determines the graph Laplacian regularization process.

Finally, the re-weighted Laplacian L_r is constructed based on a *re-weighted* function w' defined as:

$$w'(\|X_\alpha - X_\beta\|) = w(\|X_\alpha - X_\beta\|) \cdot \phi(d_{X_\alpha}(X_\beta - X_\alpha)), \tag{19}$$

where $\phi(X) = X^2$ and for w we use a standard Gaussian function after the projection with a scale parameter σ^2 ($w(\|X_\alpha - X_\beta\|) = \exp(-\|Z_\beta\|^2/\sigma^2)$). As before (see the end of the previous section), one could plug in any nonlinear function to $\phi(X)$ and control the diffusivity accordingly.

In summary, in the original graph Laplacian-based regularization, the deviation $(f(X_\alpha) - f(X_\beta))^2$ is penalized only based on the length of the vector $X_\beta - X_\alpha$, while our algorithm additionally takes into account the extrinsic curvature of M along the direction of $X_\beta - X_\alpha$. The resulting new graph Laplacian will henceforth be referred to as a *re-weighted graph Laplacian*. The new graph Laplacian can be subsequently normalized as desired.

5. Experiments

Our estimation of the second fundamental form can be used either directly on an analytically presented manifold (*e.g.*, using Eq. 3) or for constructing a re-weighted graph Laplacian that can be applied to any point cloud. In this section, we focus on the second case and compare the performance of our re-weighted graph Laplacian (r-Lap; L_r) with classical graph Laplacian (Lap; L). We consider two application scenarios in which the graph Laplacian has been particularly successful: semi-supervised learning and spectral clustering. For all experiments, following conventions, the graph Laplacians are normalized.

Throughout the experiments, the main computational bottleneck shared by r-Lap and Lap was the computation of the k-NN graph structure. Given that, the run-time spent constructing r-Lap is, on average, around twice as long as that of Lap. With the k-NN structure, building r-Lap for the MNIST dataset of size 70,000 took around 3 minutes on a 3GHz machine (see 'Spectral clustering' paragraph later).[6]

[5]More precisely, it is the corresponding push forward with respect to i^{-1}: $i_*^{-1}(X_\beta - X_\alpha)$.

[6]The code is available on the authors' website.

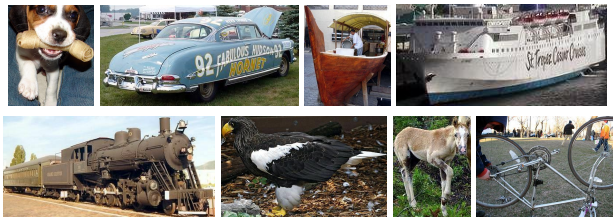

Figure 4. Examples of C-PASCAL dataset. Each image is focused on an object of interest and is obtained by cropping the annotated bounding box from PASCAL VOC challenge 2008 training image [8]. This enables direct comparison of classification performances of Lap and r-Lap without any external object detection.

Semi-supervised learning. We adopted four standard datasets (USPS, COIL2, BCI, and Text) for semi-supervised learning [4] and the cropped PASCAL (C-PASCAL) dataset used by Ebert et al. [8]. While similar to Lap in that r-Lap can be used for general multi-class classification problems, instead we focus on binary classification problems. This facilitates direct comparison of the regularization performances of r-Lap and Lap and disregards the effect of any multi-class combination methods. The C-PASCAL data set is constructed from the PASCAL VOC challenge 2008 training set [9] by cropping the sub-windows using bounding box annotations [8] (Fig. 4). It contains 6,175 images of objects from 20 classes. From these classes, we randomly choose 5 pairs of classes to construct 5 binary classification problems. We used the histogram of oriented gradient (HOG) features as provided by Ebert *et al*. [8]. We refer to Chapelle *et al*. [4] for the details of the remaining datasets.

For all experiments except for C-PASCAL, we randomly chose 100 labels for each class, with the remaining data points used as unlabeled examples. For C-PASCAL, we used 50 labels such that a sufficient number of unlabeled points were available for each class. For sets of labeled data points $\{(X_i, Y_i)\}_{i=1}^{l'}$ and unlabeled data points $\{X_i\}_{i=l'+1}^{l}$, the final assignments of the labels were obtained by solving:

$$\underset{f \in \mathbb{R}^n}{\arg\min} \ \frac{1}{l'} \sum_{i=1}^{l'} (Y_i - f(X_i))^2 + \lambda \mathbf{f}^\top B \mathbf{f}, \tag{20}$$

where and $B = L$ or $B = L_r$. For each problem, the experiment was repeated 10 times with different sets of labeled examples and the results were averaged. There are three parameters to be tuned for Lap in this setting which include the parameter (σ^2) for the weight function w, the number (k) of nearest neighbors, and the regularization parameter (λ). For r-Lap, the dimensionality of the manifold (m) is an additional hyper-parameter. These hyper-parameters were optimized by 10-fold cross-validation (CV) where in each run, a subset of labeled points were left-out while all unlabeled data points are kept. For all experiments, we tune the hyper-parameters of Lap first. Then, the parameters of

Algorithm	USPS	COIL2	BCI	Text	C-PASCAL				
					1	2	3	4	5
Lap	6.72	0.47	37.19	22.3	9.80	14.12	11.59	11.37	**6.27**
r-Lap	**5.78**	**0.41**	**35.67**	**20.8**	**8.90**	**11.79**	**11.52**	**10.39**	6.57
Improvement (%)	14.00	12.77	4.09	6.73	9.18	16.50	0.60	8.62	-4.79
Lap (GT)	5.92	**0**	32.60	20.9	6.98	10.17	10.45	10.90	5.94
r-Lap (GT)	**4.94**	**0**	**25.94**	**19.9**	**6.58**	**9.97**	**9.96**	**8.99**	**5.52**
Improvement (%)	15.55	0	20.43	4.79	5.73	1.97	4.69	17.52	7.07

Table 1. Classification performance (error rate) of graph Laplacian (Lap) and re-weighted graph Laplacian (r-Lap). The results obtained with ground-truth parameters are indicated with (GT). The best results are marked with bold face. The performance improvement of r-Lap from Lap is calculated as the reduction of error rate in %.

r-Lap were chosen by restricting the search space of σ^2, k, and λ only at the vicinity of the optimal values for the case of Lap. This resulted in the total number of parameter evaluations for r-Lap being only slightly larger than twice that of Lap. We also report the performance of both algorithms when the ground-truth (GT) hyper-parameters are provided. This keeps the test error minimal during hyper-parameter search. This can also be used to evaluate the utility of each algorithm for interactive settings: The user tries different parameter combinations and chooses the best. If the error rate surface with respect to hyper-parameter is *smooth*, then the user could decide the next search point based on the information gathered thus far. Our preliminary experiments showed that, except for the parameter m (see next paragraph), the error rate surface with respect to hyper-parameter *is* smooth. Accordingly, the active sampling strategy can indeed be exercised (Table 1).

For all but one dataset, the error rate of r-Lap was lower than that of Lap when the parameters were automatically chosen based on CV. This clearly demonstrates the superiority of r-Lap over Lap in semi-supervised learning and supports our claim that exploiting external curvature is useful. When the ground truth hyper-parameters were adopted, the performance difference between r-Lap and Lap is even more pronounced (except for the case of COIL2 in which both algorithms resulted in zero error). This reveals both the strengths and limitations of our algorithm. Potentially, r-Lap can lead to significant improvements over Lap when the parameters are tuned properly (*e.g.*, through user interaction). However, the added parameter over Lap can lead to overfitting when the parameters are optimized with cross-valuation with a limited number of labeled points (as observed in worse performance for r-Lap on C-PASCAL 5). Automatic tuning of hyper-parameters is still an open problem in semi-supervised learning and clustering in which no or only limited number of labeled examples are provided.

Spectral clustering. We used two standard datasets for spectral clustering, USPS and MNIST, which consist of 9,298 and 70,000 digit images respectively. Following the experimental convention adopted by Bühler et al. [3], the hyper-parameter k was fixed at 10 while σ^2 was adaptively determined for each point X_i such that σ_i becomes half of the mean distance from X_i to its k-NNs.

Quantitative evaluation is performed by measuring the error rate: the number of disagreements with ground truth labels for each pattern with the label of the corresponding cluster, normalized by the number of total data points. The label of a cluster is assigned as the mode of the ground truth labels of the patterns that belong to that cluster. For r-Lap, we performed experiments with different values of m. Table 2 shows the results. When $m = 10$, r-Lap boils down to Lap. For some values of m, r-Lap resulted in even higher error rates than Lap while when $m = 10$, the performance of r-Lap and Lap are identical as expected. However when m is properly chosen, r-Lap can produce significant improvements over Lap. Overall, the results imply that r-Lap can provide a reasonable trade-off between the performance and the effort for choosing an additional parameter.

6. Discussion and Conclusion

In graph Laplacian applications, data is often given as a point cloud in a vector space (*e.g.*, Euclidean). However, in some applications, each data point is never explicitly represented but its pair-wise similarity or dissimilarity is provided instead. Our algorithm cannot be directly applied in this case since the estimation of the second fundamental form II exploits the explicit representation of a point in the cloud. Fortunately, this does not require that the representation of each point should be global, *i.e.*, a locally consistent representation of each point within a small neighborhood is sufficient. Here, we could apply any local distance-based embedding technique such as multi-dimensional scaling. Once coordinates are assigned to a point X_α and its neighbors $N_k(X_\alpha)$, II can be straightforwardly calculated.

We regard manifold dimensionality as a hyper-parameter which is tuned either based on cross-validation or explicitly

Algorithm	Lap	r-Lap		
		m	Error rate	Improvement (%)
USPS	0.22	2	0.23	-4.54
		3	0.28	-27.27
		4	**0.15**	31.82
		5	0.21	4.54
		6	0.24	-9.09
		7	0.22	0.00
		8	0.22	0.00
		9	0.22	0.00
		10	0.22	0.00
MNIST	0.31	2	**0.19**	38.71
		3	0.21	32.26
		4	0.32	-3.23
		5	0.25	19.35
		6	0.32	-3.23
		7	0.31	0.00
		8	0.31	0.00
		9	0.31	0.00
		10	0.31	0.00

Table 2. Clustering performance of graph Laplacian (Lap) and reweighted graph Laplacian (r-Lap) with varying dimensionality m.

by the user. Automatic estimation of the dimensionality of a data manifold is an area of active research [16, 12]. In the future, we will investigate combining our algorithm with automatic dimensionality estimation algorithms to make equal the number of hyper-parameters to Lap and r-Lap.

Conclusion. In this paper, we conjectured that curvature information could be exploited to improve regularization on manifolds. We experimentally verified this by developing a curvature-aware anisotropic regularization algorithm on a manifold, and applying it to semi-supervised learning and clustering. As the main building block of our algorithm, we presented a procedure that estimates the second fundamental form and proved its consistency (see supplementary material). This procedure can be applied to general Riemannian submanifolds and accordingly, it could be used in any application that exploits curvatures of manifolds.

Acknowledgment

Matthias Hein greatly aided these ideas, especially the second fundamental form estimate convergence proof.

References

[1] J.-Y. Audibert and A. B. Tsybakov. Fast learning rates for plug-in classifiers. *The Annals of Statistics*, 35(2):608–633, 2007. 5

[2] M. Belkin and P. Niyogi. Towards a theoretical foundation for Laplacian-based manifold methods. *Journal of Computer and System Sciences*, 74(8):1289–1308, 2005. 1

[3] T. Bühler and M. Hein. Spectral clustering based on the graph p-Laplacian. In *Proc. ICML*, pages 81–88, 2009. 7

[4] O. Chapelle, B. Schölkopf, and A. Zien. *Semi-Supervised Learning*. MIT Press, Cambridge, MA, 2010. 6

[5] U. Clarenz, U. Diewald, and M. Rumpf. Anisotropic geometric diffusion in surface processing. In *Proc. Visualization*, pages 397–405, 2000. 2

[6] M. Desbrun, M. Meyer, P. Schröder, and A. H. Barr. Implicit fairing of irregular meshes using diffusion and curvature flow. In *Proc. SIGGRAPH*, pages 317–324, 1999. 2

[7] D. L. Donoho and C. Grimes. Hessian eigenmaps: locally linear embedding techniques for high-dimensional data. *Proc. of the National Academy of Sciences*, 100(10):5591–5596, 2003. 1, 2

[8] S. Ebert, D. Larlus, and B. Schiele. Extracting structures in image collections for object recognition. In *Proc. ECCV*, pages 720–733, 2010. 6

[9] M. Everingham, L. V. Gool, C. K. I. Williams, J. Winn, and A. Zisserman. *The PASCAL VOC Challenge 2008 Results*. 2008. 6

[10] M. Hein, J.-Y. Audibert, and U. von Luxburg. From graphs to manifolds - weak and strong pointwise consistency of graph Laplacians. In *Proc. COLT*, pages 470–485, 2005. 1, 5

[11] M. Hein and M. Maier. Manifold denoising. In *NIPS*, pages 561–568, 2007. 2

[12] B. Kégl. Intrinsic dimension estimation using packing numbers. In *NIPS*, pages 681–688, 2002. 8

[13] K. I. Kim, F. Steinke, and M. Hein. Semi-supervised regression using Hessian energy with an application to semi-supervised dimensionality reduction. In *NIPS*, pages 979–987, 2010. 1, 5

[14] S. Kobayashi and K. Nomizu. *Foundations of Differential Geometry*, volume 1. John Wiley & Sons Inc. (reprint of the 1963 original), New York, 1996. 3

[15] J. M. Lee. *Riemannian Manifolds- An Introduction to Curvature*. Springer, New York, 1997. 2, 3

[16] A. m. Farahmand, C. Szepesvári, and J.-Y. Audibert. Manifold-adaptive dimension estimation. In *Proc. ICML*, pages 265–272, 2007. 8

[17] P. Perona and J. Malik. Scale-space and edge detection using anisotropic diffusion. *IEEE TPAMI*, 12(7):629–639, 1990. 2

[18] D. Tschumperlé and R. Deriche. Vector-valued image regularization with PDEs: a common framework for different applications. *IEEE TPAMI*, 27(4):506–517, 2005. 2

[19] J. Weickert. *Anisotropic Diffusion in Image Processing*. ECMI Series, Teubner-Verlag, Stuttgart, 1998. 2, 4

[20] T. J. Willmore. *Riemannian Geometry*. Oxford University Press, Oxford, 1997. 4

[21] H. Zhao and G. Xu. Triangular surface mesh fairing via Gaussian curvature flow. *Journal of Computational and Applied Mathematics*, 195(1):300–311, 2006. 2