Local High-order Regularization on Data Manifolds

Kwang In Kim
Lancaster University

James Tompkin
Harvard SEAS

Hanspeter Pfister
Harvard SEAS

Christian Theobalt
MPI for Informatics

Abstract

The common graph Laplacian regularizer is well-established in semi-supervised learning and spectral dimensionality reduction. However, as a first-order regularizer, it can lead to degenerate functions in high-dimensional manifolds. The iterated graph Laplacian enables high-order regularization, but it has a high computational complexity and so cannot be applied to large problems. We introduce a new regularizer which is globally high order and so does not suffer from the degeneracy of the graph Laplacian regularizer, but is also sparse for efficient computation in semi-supervised learning applications. We reduce computational complexity by building a local first-order approximation of the manifold as a surrogate geometry, and construct our high-order regularizer based on local derivative evaluations therein. Experiments on human body shape and pose analysis demonstrate the effectiveness and efficiency of our method.

1. Introduction

The graph Laplacian regularizer is established as one of the most popular regularizers for semi-supervised learning [5], spectral clustering [20, 13], and dimensionality reduction [3]. The underlying assumption for using the graph Laplacian regularizer is that data lie on a low-dimensional sub-manifold, and the object (e.g., a function) of interest should be regularized as defined on the manifold rather than as defined on the entire ambient space. By measuring local pairwise deviations of the function values in the ambient space, the graph Laplacian regularizer approximates the first-order variations on the manifold, thereby enabling us to regularize the function based on its first-order energy without having to know the manifold analytically.

Despite its solid theoretical background [4, 9] and success in many applications, the graph Laplacian regularizer has an important shortcoming that makes its usage less favorable on data lying in high-dimensional manifolds: as we will discuss, as a first-order regularizer, the null space of the graph Laplacian regularizer contains discontinuous functions on manifolds with dimensionality larger than 2 [15, 24].

Recently, Zhou and Belkin [24] proposed an iterated graph Laplacian approach that avoids this *degeneracy* and enables regularization on high-dimensional manifolds. The price for the non-degeneracy and the resulting simplicity of the algorithm is high computational complexity: the iterated graph Laplacian regularizer is constructed by taking powers of the graph Laplacian matrix, which makes the original matrix denser and, accordingly, for large-scale problems (e.g., $O(100{,}000)$) it cannot be directly applied efficiently.

We propose an empirical regularizer which avoids degeneracy and leads to a sparse matrix. Our algorithm is based on the local linear approximation of the manifold: At each point, the corresponding neighborhood is projected onto its tangent space, where the high-order derivatives of the function are defined in this surrogate geometry. Instead of explicitly calculating high-order derivatives and measuring the corresponding complexity of the function, we measure its reproducing kernel Hilbert space (RKHS) norm. Similar to the graph Laplacian, its sparsity is explicitly controlled based on the local neighborhood structure. We present experimental results on human body shape and pose datasets, which show that our method is superior to graph Laplacian and iterated graph Laplacian techniques in terms of accuracy and computational complexity.

As this paper is equation and symbol rich, we summarize all symbols and notation conventions on the first page of the supplemental material.

2. Problem statement

While our proposed regularizer can be used in clustering and dimensionality reduction, as with the graph Laplacian and iterated graph Laplacian regularizers, we focus on *semi-supervised learning* which enables us to compare numerically the performance of each algorithm.

For a set of data points $\mathcal{X} = \{X_1,\ldots,X_u\} \subset \mathbb{R}^n$ plus the corresponding labels $\{Y_1,\ldots,Y_l\} \subset \mathbb{R}$ for the first l points in $\mathcal{X}$ where $l \ll u$, the goal of semi-supervised learning is to infer the labels of the remaining $u-l$ data points in $\mathcal{X}$. Our approach is based on regularized empirical risk minimization:

$$\underset{f:\mathbb{R}^n\to\mathbb{R}}{\operatorname{argmin}} \sum_{i=1}^{l} (Y_i - f(X_i))^2 + \lambda \mathcal{R}(f), \tag{1}$$

where $\mathcal{R}(\cdot)$ is the regularization functional. Here, we use the standard squared loss function for simplicity, though our framework is applicable to any convex loss function. This problem can be solved either by reconstructing the underlying function f or by identifying its evaluation $f|_{\mathcal{X}}$ on $\mathcal{X}$. In this paper, we focus on the second case, which is often called *transductive learning*.

Most semi-supervised learning algorithms can be characterized by how the unlabeled data points of $\mathcal{X}$ are used to construct a

corresponding regularizer $\mathcal{R}(f|_{\mathcal{X}})$. One of the best established regularizers is the graph Laplacian L [13]:

$$\mathcal{R}_L(\mathbf{f}) := \mathbf{f}^\top L\mathbf{f} = \sum_{i,j=1}^{u} [W]_{ij}(f_i - f_j)^2, \tag{2}$$

where $f_i = f(X_i)$, $\mathbf{f} := f|_{\mathcal{X}} = [f_1, \dots, f_u]^\top$, and W is a non-negative input similarity matrix which is typically defined based on a Gaussian:

$$[W]_{ij} = \exp\left(-\frac{\|X_i - X_j\|^2}{b}\right). \tag{3}$$

One way of justifying the use of the graph Laplacian comes from its limit case behavior as $u \to \infty$ and $b \to 0$: When the data $\mathcal{X}$ is generated from an underlying manifold M with dimension $m \leq n$, i.e., the corresponding probability distribution P has support in M, the graph Laplacian converges to the Laplace-Beltrami operator Δ on M [4, 9]. The Laplace-Beltrami operator can be used to measure the first-order variations of a continuously differentiable function f on M:

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

where g is the *Riemannian metric*, and dV is the corresponding *natural volume element* [12] of M. The second equality is the result of Stokes' theorem. Accordingly, a graph Laplacian-based regularizer $\mathcal{R}_L$ can be regarded as an empirical estimate of the first-order variation of f on M based on $\mathcal{X}$.

However, the convergence of the graph Laplacian L to the Laplace-Beltrami operator Δ reveals an important shortcoming for it to be used as the standard regularizer for high-dimensional data: For high-dimensional manifolds ($m > 1$), the null space of Δ includes discontinuous functions on M. This is suggested by the *Sobolev embedding theorem* that states that, in general, any (semi-)norm induced by differential operators with order $d \leq m/2$ will have discontinuous functions in its null space [18]. In particular, the norm $\|\cdot\|_\Delta$ in Eq. 4 which measures the first-order variation has a null space consisting only of continuous functions (in particular, constant functions) when $m = 1$ only. For $m > 1$, the null space of Δ contains some discontinuous functions as a subset of L^2 space which are equivalent almost everywhere to constant functions, except for the set of *measure zero* [7]. In other words, there are "spiky" functions f, e.g., Dirac delta functions, with norm $\|f\|_\Delta^2 = 0$ (Fig. 1).

This is especially important in semi-supervised learning because we actively minimize the regularized risk of attaining a zero value by such a function (Eq. 1). While this has been well-known in statistics, its effect on semi-supervised learning has only recently been analyzed by Nadler *et al.* [15]. They showed that, in the limit case (i.e., $u \to \infty$), where $\mathcal{R}_L$ is used, indeed the null space of the empirical risk functional (Eq. 1) includes a function f which is zero everywhere except for the labeled data points $\{X_1, \dots, X_l\}$, where f agrees with the given labels, and no generalization is obtained.

In practice, due to the finite number of data points u, the learned function f (more precisely, its evaluation $\mathbf{f}$ on $\mathcal{X}$) is not a Dirac delta function exactly, but is a very steep, sheer-sided spike which peaks at the labeled data points (Fig. 1). For discrete problems, e.g., classification, where only relative values of f are relevant, it is possible to normalize the output values based on the local distribution of f to soften such peaks, as exemplified in [22]. However, this technique is not applicable for learning continuous functions.

Zhou and Belkin [24] presented the first approach that explicitly prevents this degenerate case in semi-supervised learning. They proposed using powers of graph Laplacian (or *iterated graph Laplacian*) as a regularizer:

$$\mathcal{R}_{L^p}(\mathbf{f}) := \mathbf{f}^\top L^p \mathbf{f}, \tag{5}$$

with $p > \frac{m}{2}$. In the limit case as $u \to \infty$, L^p converges to Δ^p, which corresponds to the penalizer of (selected) $\lceil \frac{p}{2} \rceil$-th order variations in the context similar to Eq. 4 [24]:

$$\|f\|_{\Delta^p}^2 = \int_M f(X)[\Delta^p f|_X]dV(X), \tag{6}$$

which is infinite when f is discontinuous. The ability to regularize over higher-order derivatives avoids the degenerate case of learning discontinuous functions.

One of the major limitations of iterated graph Laplacian is that, due to the density of the resulting matrix L^p, it cannot be directly applied to large-scale problems. For a non-iterated graph Laplacian, finding the minimizer of Eq. 1 with $\mathcal{R}_L$ requires building and solving a linear system of size $u \times u$. Even for large-scale problems (e.g., $u \approx 10^5$), this is affordable since the corresponding weight matrix W can be well-approximated by a sparse matrix constructed from a k-nearest neighbor (NN) graph. However, in general, iterating L (taking powers L^p) makes a sparse matrix denser. This is especially true when p is large, which is required for high-dimensional data, as suggested by the Sobolev embedding theorem. For instance, with $u = 50{,}000$, solving Eq. 1 with iterated graph Laplacian is $15\times$ slower (Sec. 6) than the Laplacian case.

3. Local high-order regularization

Our goal is to build a new regularizer that shares the desirable properties of both penalizing discontinuous functions with L^p and being sparse in L for fast computation. To achieve this goal, we build a global regularization matrix G based on local regularizers evaluated at each point in $\mathcal{X}$.

First, we take a class of high-order manifold operators as regularizers by adopting the regularization framework of Yuille and Grzywacz [23]. These regularizers correspond to generalizations of Eq. 4:[1]

$$\|f\|_D^2 := \int_M \sum_{k=1}^{\infty} c_k |D^k f|_X|^2 dV(X), \tag{7}$$

[1] As a special case, when $c_p = 1$ and $\{c_k\}_{k \neq p} = 0$, $\|\cdot\|_D^2$ becomes $\|\cdot\|_{\Delta^p}^2$ (Eq. 6). In general, different choices of differential operators are possible, e.g., Hessian, rather than the powers of Δ and ∇. This choice was motivated by the demonstrated empirical success of the resulting regularizer in many applications [23], and the computational efficiency as facilitated by the use of the corresponding Gaussian RKHS as discussed in Sec. 4.

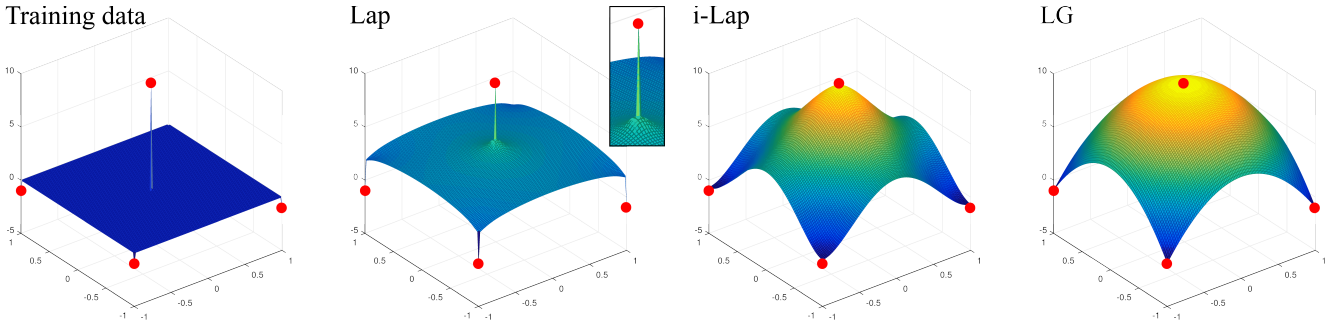

Figure 1. Example on 2D data. Section 5 contains details of this toy example; the surface in the training data plot is to help with visualization only, and no regularization has taken place. The *Lap* result largely fails to regularize, apart from points very near to the original training data. These spikes can be seen in the zoom inlay. The result of *i-Lap* looks *hyperbolic* because its null space includes polynomials. In this example, both *i-Lap* and *LG* are acceptable since they lead to smooth functions. Inspired by [24].

$$D^k f = \begin{cases} \Delta^k f, & \text{for even } k \\ \nabla(\Delta^k f), & \text{for odd } k \end{cases} \tag{8}$$

$$|D^k f|^2 = \begin{cases} (D^k f)^2, & \text{for even } k \\ g(D^k f, D^k f), & \text{for odd } k \end{cases} \tag{9}$$

where k is the order of the derivative operator, and coefficients $c_k \geq 0$.

For a known manifold with known metric and *Christoffel symbols* [12], the derivative operators in Eq. 8 are easy to calculate. However, in most practical applications, the manifold is not directly observed but is only indirectly observed as a point cloud of sampled data points $\mathcal{X} \subset \mathbb{R}^n$, where M is a (m-dimensional) sub-manifold of $\mathbb{R}^n$. Accordingly, direct calculation of Eq. 8 is infeasible.

A local first-order approximation D_0. We bypass this problem by using a local first-order approximation $T_X(M)$ of manifold M at each point X (M_X) in $\mathbb{R}^n$ as a proxy geometry for M near X. Since $T_X(M)$ is identified with $\mathbb{R}^m$, evaluating the derivative operators in Eq. 8 on X boils down to the calculation of the derivative operators in Euclidean geometry. In particular, evaluating the Laplace-Beltrami operator becomes the calculation of the Laplacian operator:

$$D_0^2 f|_X = \Delta_0 f|_X = \sum_{r=1}^{m} \partial_r^2 f|_X. \tag{10}$$

Subscript 0 denotes operators defined on the proxy geometry, where $\Delta_0[\cdot]|_X$ is the Laplacian defined at $T_X(M)$. ∂_r is shorthand for $\frac{\partial}{\partial x^r}$. Practically, the dimension of m is unknown and so is a hyper-parameter.

With a manifold approximation, the next step is to construct approximations of Eq. 8 and Eq. 10 given $\mathcal{X}$ and $f|_{\mathcal{X}}$. Suppose that for each data point X_i, the corresponding k-NN $N_k(X_i) \subset \mathcal{X}$ are identified. First, we estimate the first-order approximation $T_{X_i}(M)$ by performing principal component analysis on $N_k(X_i)$ [6]: The representations $\{\mathbf{x}_j\}_{j=1}^k$ of $N_k(X_i)$ on $T_{X_i}(M)$ are given as the first m-principal components of $N_k(X_i)$. Then, at X_i, the approximation of the Laplacian in Eq. 10 is obtained by fitting a smooth interpolation φ^i in (x) to $\{f(X_j)\}_{j=1}^k$ and then extracting the trace of the resulting Hessian $H\varphi^i$ of φ^i, which we denote as $S^{(2)}(X_i)$. The surrogate function φ^i can be a (constrained) second-order polynomial h^i (for Δ) or a Gaussian kernel interpolation q^i (for Δ^k, $k>0$):

$$h^i(\mathbf{x}) = f(X_i) + \sum_{r=1}^{m} [a^i]_r x^r + \sum_{r=1,s=r}^{m} [b^i]_{r,s} x^r x^s, \tag{11}$$

$$q^i(\mathbf{x}) = f(X_i) + \sum_{j=1}^{k} [\alpha^i]_j K(\mathbf{x}_j, \mathbf{x}), \tag{12}$$

where $\mathbf{x} = [x^1, \ldots, x^m]^\top$, and

$$K(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{\|\mathbf{x}-\mathbf{x}'\|^2}{\sigma^2}\right). \tag{13}$$

The coefficients $\{a^i, b^i\}$ and $\{\alpha^i\}$ of h^i and q^i, respectively, are calculated as the standard least squares fit:

$$[a^i, b^i] = \underset{w \in \mathbb{R}^{m+m(m+1)/2}}{\operatorname{argmin}} \sum_{j=1}^{k} \Big(f(X_j) - h^i(\mathbf{x}_j) \Big)^2, \tag{14}$$

$$\alpha^i = \underset{a \in \mathbb{R}^k}{\operatorname{argmin}} \sum_{j=1}^{k} \Big(f(X_j) - q^i(\mathbf{x}_j) \Big)^2, \tag{15}$$

where w is a vector of linear and quadratic coefficients in the second-order polynomials.

By combining these estimates of the local Laplacians and re-arranging the variables, one can construct a matrix B as a new regularizer on a point cloud $\mathcal{X}$:

$$\|f\|_{\Delta_0}^2 \approx \mathcal{R}_B(\mathbf{f}) = \mathbf{f}^\top B \mathbf{f} = \sum_{i=1}^{u} f(X_i) S^{(2)}(X_i). \tag{16}$$

To evaluate the squared Laplacian operator $\Delta_0^2|_{X_i}$, we calculate the corresponding fourth-order derivatives of φ. In the case when $\varphi = q$, the derivatives of φ of any order are easily calculated by noting that the derivative of a Gaussian function can be evaluated based on the original Gaussian and the combinations of Hermite polynomials [10]. The corresponding empirical regularizer $\mathcal{R}_E$ based on a finite number of points $\mathcal{X}$ can be

constructed similarly to Eq. 16:

$$\mathcal{R}_E(\mathbf{f}) = \sum_{k=1}^{\infty} c_k \mathbf{f}^\top E^{(k)} \mathbf{f} := \sum_{i=1}^{u} \mathcal{S}_{X_i}(f), \tag{17}$$

where k indexes the order of derivatives, $\mathcal{S}_{X_i}(f) = \sum_{k=1}^{\infty} c_k |S^{(k)}(X_i)|^2$, $S^{(k)}(X_i)$ corresponds to an empirical approximation of $D^k f|_{X_i}$, and $E^{(k)}(X_i)$ is the corresponding regularization matrix.

Summary Our regularizer $\mathcal{R}_E$ is constructed by combining a set of local high-order regularizers, each of which is obtained based on a local first-order approximation of M. This avoids explicit calculation of high-order derivatives on M. Our regularizer $\mathcal{R}_E(\mathbf{f})$ is explicitly given as a sparse matrix E, i.e., $\mathcal{R}_E(\mathbf{f}) = \mathbf{f}^\top E \mathbf{f}$, where E is obtained by aligning the local matrices $\{E^{(k)}\}$. Since this is a combination of local high-order regularizers, it is a global high-order regularizer, and therefore it avoids the degeneracy of the graph Laplacian regularizer. As a combination of local high-order regularizers, $\mathcal{R}_E$ is a global high-order regularizer, and therefore it avoids the degeneracy of the graph Laplacian regularizer.

Explicitly calculating $\{E^{(k)}\}$ is both numerically unstable and computationally demanding. Therefore, we propose a stable approximation of $\mathcal{R}_E$ in Sec. 4. Before we explain this more-practical implementation, for interested readers, we discuss the relationship between the operators D and D_0.

3.1. Relation between D and D_0.

The regularizer $\mathcal{R}_E$ depends on the local first-order approximation $T_X(M)$ at each X. If the M is smoothly embedded in the ambient space $\mathbb{R}^n$, especially in the sense that the corresponding *second fundamental form* [12] is bounded, then the approximation error is third-order: Let $d_X := d_X(\cdot,\cdot)$ be the geodesic distance between two points on M in the neighborhood $\mathcal{N}(X)$ of X,[2] then the distance $\tilde{d}_X$ between these points in the proxy geometry $T_X(M)$ is related as [4, 9]

$$d_X = \tilde{d}_X + \mathcal{O}(d_X^3). \tag{18}$$

The use of local first-order approximations to a manifold is justified by its success in many applications (e.g., [19, 6]). We support this approximation further by noting that the corresponding orthonormal coordinates in $T_X(M)$ can be regarded as approximations of *Riemannian normal coordinates* [11]. In a Riemannian normal coordinate chart centered at a point X, the manifold appears Euclidean up to second-order. Specifically, at X, the corresponding Riemannian metric g becomes Euclidean: the first order derivatives vanish, and evaluating the Laplace-Beltrami operator boils down to the calculation of the Laplacian in Euclidean space:

$$\Delta f|_X = \sum_{r,s=1}^{m} \frac{\partial_r (g^{rs} \sqrt{\det g} \partial_s f)}{\sqrt{\det g}} = \Delta_0 f|_X, \tag{19}$$

where $\partial_r = \frac{\partial}{\partial x^r}$, $\delta_s^r = \sum_t g^{rt} g_{ts}$, δ_s^r: $\delta_s^r = 1$ if $r = s$ and 0, otherwise, $g_{rs} = g(\partial_r, \partial_s)$, and $\det g$ is the determinant of the matrix evaluation $\{g_{rs}\}$. Using this setup, similarly to the graph Laplacian L case, one can show the convergence of the matrix B (Eq. 16) to Δ in the limit case as $u \to \infty$, the diameter ϵ of N_k is controlled carefully:

Definition 1 (Audibert and Tsybakov [1]) *For given constants $c_0, \epsilon_0 > 0$, a Lebesgue measurable set $A \subset \mathbb{R}^m$ is called* (c_0, ϵ_0)-regular *if*

$$\lambda[A \cap \mathcal{B}(\mathbf{x}, \epsilon)] \geq c_0 \lambda[\mathcal{B}(\mathbf{x}, \epsilon)], \forall \epsilon \in [0, \epsilon_0], \forall \mathbf{x} \in A,$$

where $\lambda[S]$ is the Lebesgue measure of $S \subset \mathbb{R}^m$ [7]. We fix constants $c_0, \epsilon_0 > 0$ and $0 < \mu_{\min} < \mu_{\max} < \infty$ and a compact $\mathcal{C} \subset \mathbb{R}^d$. We say that the strong density assumption *is satisfied if the distribution P is supported on a compact (c_0, ϵ_0)-regular set $A \subseteq \mathcal{C}$ and has a density μ w.r.t. λ bounded above and below by between μ_{min} and μ_{max}*

$$\mu_{\min} \leq \mu(\mathbf{x}) \leq \mu_{\max}, \forall \mathbf{x} \in A \text{ and } \mu(\mathbf{x}) = 0 \text{ otherwise}.$$

Proposition 1 *If Hessian Hf on M is Lipschitz continuous with the Lipschitz constant γ, and the natural volume element dV is bounded in the sense that the underlying probability distribution P satisfies strong density assumption, then there are constants $C_1, C_2, \mu_0 > 0$ such that with probability larger than $1 - (m^2 + 3m)\exp(-C_2 u \epsilon^m)$:*

$$|tr[Hh(\mathbf{x})] - \Delta f(X)|^2 \leq \frac{k}{u\epsilon^m} \frac{C_1 \epsilon^2 \gamma^2}{\mu_0}, \tag{20}$$

where $tr[A]$ calculates the trace of A, $k = |\mathcal{X} \cap \mathcal{B}(X, \epsilon)|$, and $\mathcal{B}(X, \epsilon)$ is the ϵ-neighborhood of X in coordinates, i.e. $\mathcal{B}(X, \epsilon) := \{X' : \|\mathbf{x} - \mathbf{x}'\|_{T_X(M)} \leq \epsilon\}$, with $\mathbf{x}'$ being the coordinate representation of X'.

The proof of this convergence be found in the supplemental material. For simplicity of proof, we use the ϵ-neighborhood $\mathcal{B}(X, \epsilon)$ instead of k-NNs $N_k(X)$. It can be easily modified for the k-NN case (see supplemental material). Accordingly, in Eq. 20, ϵ is the only parameter to be controlled to obtain the convergence. The role of ϵ is similar to the width of the Laplacian *weight function* (Eq.3) in [4]: Roughly, decreasing ϵ guarantees that the local surrogate function h is flexible enough to well-approximate f. However, it should not shrink too fast to ensure that there are sufficient data points k in $\mathcal{B}(X, \epsilon)$ to prevent h from *overfitting* to f. This leads to the condition that ϵ^m-shrink should be slower than u-increase, so that $u\epsilon^m \to \infty$. The number of neighborhoods k in Eq. 20, given as $|\mathcal{B}(X, \epsilon) \cap \mathcal{X}|$, is automatically controlled by sampling $\mathcal{X}$ from P. This leads to $\mathcal{O}(\frac{k}{u\epsilon^m}) = 1$ (see supplemental material) guaranteeing quadratic (ϵ^2) convergence. All other constants C_1, C_2, μ_0, and γ are independent of u.

The strong density assumption is moderate. In particular, it holds for any compact manifold with a continuous distribution.

In general, the derivatives of the metric g with orders higher than 2 are non-vanishing even in normal coordinates. In this case, for instance, $\Delta_0^2 f|_X$ deviates from $\Delta^2 f|_X$ in third-order:

$$\Delta^2 f|_X = \Delta_0^2 f|_X + \mathcal{D}^3(f|_X), \tag{21}$$

[2] The *injectivity radius* $\text{inj}(X)$ of $X \in M$ is always positive [12]. Here, we assume that $\mathcal{N}(X) \subset \text{inj}(X)$.

where $\mathcal{D}^3(f|_X)$ contains selected derivatives of f at X up to third-order.[3]

However, since they agree at the highest (fourth) order, Δ_0^2 shares two important properties with Δ^2 which are precisely what leads to a *proper* regularizer for $m<4$. When $m<4$, and the metric g and the embedding $\hat{\imath}: M \to \mathbb{R}^n$ are smooth:

1. $c_2\Delta_0 + c_4\Delta_0^2$ with $c_2, c_4 > 0$, has the null space consisting of truly constant functions (i.e., excluding the degenerate functions which deviate from constant functions on sets of measure zero), and
2. The evaluation of the corresponding norm defined similarly to Eq. 4 is infinite for any discontinuous functions.

This property extends to general high-order cases: The approximation error of $\Delta_0^k|_X$ to $\Delta^k|_X$ is of order $k-1$ and, for a manifold with dimension $m \geq 4$, the regularizers $\|\cdot\|_{D_0}^2$ that replaces D^k with D_0^k in $\|\cdot\|_D^2$ (Eq. 7) with $c^1, \ldots, c^{\lfloor m/2+1 \rfloor} > 0$ share the same null space with $\|\cdot\|_D^2$. Furthermore, their evaluations on any discontinuous functions produce infinite value.

4. Local Gaussian regularization

The regularization cost functional $\mathcal{R}_E$ (Eq. 17) has both the desired properties of being a high-order regularizer and of leading to a sparse system. However, evaluating it requires explicitly calculating the powers of the Laplacian evaluation $\Delta_0^k f|_{X_i}$ at each point $X_i \in \mathcal{X}$ and for each non-zero coefficient c_k. This is not only tedious but also numerically unstable since, in practice, the corresponding high-order derivatives are estimated by fitting a function φ^i to only a small number (k) of data points $N_k(X_i)$: fitting a high-order polynomial (as an extension of h^i in Eq. 12) is very unstable in general. While this can be resolved with smooth Gaussian interpolation i.e. $\varphi^i = q^i$, due to the existence of high-order polynomials contained in the derivatives of q^i (Eq. 12), the resulting derivative estimates can still be unstable, i.e., perturbed significantly with respect to slight variations of f.

We focus on a special case of the regularization functional $\mathcal{R}_E$, with a specific choice of derivative operator contribution $\{c_k\}$, which enables us to bypass the explicit evaluation of individual derivatives D^k while retaining the desired properties of being a sparse, robust, and high-order regularizer.

First, the stability problem in evaluating derivatives can be addressed by taking integral averages of derivative evaluations ($D^k f$; Eq. 8) and the corresponding magnitude $|D^k f|$ within a neighborhood $\mathcal{U}(X_i)$ of X_i, rather than their point evaluations at X_i. For instance, for derivative operators of even powers, instead of $|D_0^{2k} f|_{X_i}|$ (Eq. 7), we use:

$$|\widetilde{D}_0^{2k} f|_{X_i}| = \frac{1}{\mathrm{vol}(\mathcal{U}(X_i))} \int_{\mathcal{U}(X_i)} [\Delta_0^k \varphi^i|_{\mathbf{x}}]^2 d\mathbf{x}, \tag{22}$$

where $\mathrm{vol}(A)$ measures the volume of $A \subset T_{X_i}(M)$, which is a fixed constant given M.

This still requires explicit calculation of derivatives. However, for the special case of Eq. 7 where the coefficients $\{c_k\}$ are given as:

$$c_k = \frac{\sigma^{2k}}{k! 2^k}, \tag{23}$$

with σ^2 as defined in (13) we can efficiently calculate an approximation: First, the *local energy* of $\varphi^i = q^i$ over T_{X_i} defined as

$$\|q^i\|_D^2 := \sum_{k=1}^{\infty} c_k \int_{T_{X_i}(M)} |D^k q^i|_{\mathbf{x}}|^2 d\mathbf{x} = \|q^i\|_K^2, \tag{24}$$

can be analytically evaluated as the corresponding Gaussian reproducing kernel Hilbert space (RKHS) norm $\|\cdot\|_K$: The second equality is one of the central results in regularization theory [23], established by obtaining q^i as the solution of a minimization that combines the energy in Eq. 7 with an empirical loss in Eq. 15. This is always possible as q^i has k degrees of freedom, and leads to an Euler-Lagrange equation that renders k as Green's function of our operator D.

Second, we note that, for large u, the local energy (Eq. 24) well approximates the sum of local stabilized derivations (Eq. 22). For a Gaussian function $K(\mathbf{x}_j, \cdot)$, its value and derivatives decrease rapidly as the corresponding points of evaluation deviate from center X_j (depending on its *width* σ^2). Accordingly, its support is *effectively* limited within a neighborhood $\mathcal{U}'(X_j)$. Since $D^k q^i$ is a kernel expansion of $N_k(X_i)$, its support is limited to a larger neighborhood $\mathcal{N}(X_i)$ of X_i that encompasses $\{\mathcal{U}'(X_j), \forall X_j \in N_k(X_i)\}$. Then, we set $\mathcal{U}(X_i)$ by $\mathcal{N}(X_i)$ and obtain the local energy $\|q^i\|_D^2$ as a replacement of the integrand in (7).

In general, for given $\mathcal{U}(X_i)$, this approximation becomes more accurate as σ^2 and $N_k(X_i)$ decrease to zero, which is the case as $u \to \infty$ (see accompanying supplemental material). However, for practical applications, we do not tune σ^2 or $N_k(X_i)$ to minimize error or to achieve a desired level of accuracy since explicitly calculating the corresponding error is tedious (see Appendix). More importantly, having too small σ^2 or $N_k(X_i)$ for finite u will lead to a bad interpolation function: a Gaussian kernel interpolation with small σ^2 may lead to a highly non-linear function q^i that overfits to $\{f(X_j)\}_{j=1}^k$. While we propose setting σ^2 and $N_k(X_i)$ as decreasing functions with respect to u so that the approximation becomes exact as $u \to \infty$, for practical applications with fixed u (including our experiments), we implicitly determine the diameter of $N_k(X_i)$ based on the selected k-NN, and regard k and σ^2 as hyper-parameters. As described in Sec. 6, σ^2 is actually adaptively determined based on $N_k(X_i)$ and accordingly only $N_k(X_i)$ is tuned.

Now, we build a new regularizer $\mathcal{R}_G$ as a combination of local regularizers on $\varphi^i - f(X_i)$ for $i = 1, \ldots, u$, similarly to

[3] This can be easily verified by expanding the derivatives in normal coordinates at X:

$$\Delta^2 f = \sum_{i,j,r,s=1}^{m} \Big(\partial_i \partial_j [g^{rs} \partial_r \partial_s f] + \partial_i \partial_j [\partial_r [\partial g^{rs}] \partial_s f] + \frac{1}{2} \partial_i \partial_j \Big[g^{rs} \sum_{t,u=1}^{m} g^{tu} \partial_r [\partial g_{tu}] \partial_s f \Big] \Big).$$

Eq. 17 in Section 3:

$$\mathcal{R}_G(\mathbf{f}) = \sum_{i=1,\ldots,u} {\mathbf{f}^i}^\top \mathbf{G}^i \mathbf{f}^i \tag{25}$$

with:

$$\begin{aligned} {\mathbf{f}^i}^\top \mathbf{G}^i \mathbf{f}^i &= \|f(X_i) - \varphi^i(\cdot)\|_K^2 && (26)\\ &= {\mathbf{f}^i}^\top (I - \mathbf{1}\mathbf{1}^i)^\top (\mathbf{K}^i)^+ (I - \mathbf{1}\mathbf{1}^i)\mathbf{f}^i, && (27)\end{aligned}$$

where $[\mathbf{K}]_{lm} = K(\mathbf{x}_l, \mathbf{x}_m)$, $\mathbf{f}^i = [f(X_1), \ldots, f(X_k)]^\top$, $\mathbf{K}^+$ is the Moore-Penrose pseudoinverse of $\mathbf{K}$, and $\mathbf{1}\mathbf{1}^i$ is an indicator matrix whose element is zero except for the $l(i)$-th column that consists of ones with $l(i)$ being the index of X_i in $N_k(X_i)$.

5. Augmenting null spaces

Our local Gaussian regularizer completely eliminates the possibility of generating degenerate functions and so provides a valid regularization on high-dimensional manifolds. Further, it is designed as a combination of *local* regularizers (Eq. 25) and so is tailored to incorporate a priori knowledge of the local behavior of functions. In particular, it is easy to tune the regularizer such that it does not penalize functions with desirable properties (i.e., to augment the null space of the regularizer so that it contains those functions). One good choice for $\mathbf{f}$ are geodesic functions: both Donoho and Grimes [6] and Kim et al. [11] have demonstrated that *geodesic functions*, which are linear along geodesics, i.e., nothing more than linear functions in Euclidean space, are preferred over other functions since they correspond to the most *natural* parametrization of the underlying data.

The geodesic functions are completely characterized by their local behavior. In particular, in the Riemannian normal coordinates, they are locally linear functions. Accordingly, we can easily add geodesic functions to the null space of the global regularizer $\mathcal{R}_G(\mathbf{f})$ by including linear functions in the null space of the local regularizers (Eq. 27): We fit a linear function to $\mathbf{f}^i$ and *subtract* the resulting function from $\mathbf{f}^i$ before we fit the non-linear function (Eq. 12). This can be easily incorporated into new local regularization matrices:

$$\begin{aligned} (\mathbf{G}')^i &= \|f(X_i) - \varphi_L^i(\cdot) - \varphi^i(\cdot)\|_K^2 && (28)\\ &= (\mathbf{L}^i)^\top (\mathbf{K}^i)^+ \mathbf{L}^i, && (29)\end{aligned}$$

where $\varphi_L^i(\cdot)$ is the linear regressor fitting $\mathbf{f}^i$ in normal coordinates (i.e., $\varphi_L^i(\mathbf{x}) = (\Phi_L^i)^+ (I - \mathbf{1}\mathbf{1}^i)\mathbf{f}^i\mathbf{x}$), $\Phi_L^i \in \mathbb{R}^{k \times m}$ is the design matrix whose rows correspond to the normal coordinate values of $N_k(X_i)$, and

$$\mathbf{L}^i = I - \mathbf{1}\mathbf{1}^i - \Phi_L^i (\Phi^i)_L^+ (I - \mathbf{1}\mathbf{1}^i). \tag{30}$$

The new regularization functional $\mathcal{R}_{G'}$, in which $\{(\mathbf{G}')^i\}$ replaces $\{\mathbf{G}^i\}$, has a richer null space: a one-dimensional space of constant functions plus an m-dimensional space of geodesic functions. This null space should not be confused with the *too large* null space of the original graph Laplacian regularizer. The null space of our updated local Gaussian regularizer does not include any degenerate functions.

While this setup does not cause any noticeable increase of computational complexity, in our preliminary MoCap experiments (see Sec. 6), this reduced error rates by around 3%. Accordingly, throughout the entire experiments, we use this new local Gaussian regularizer.

Algorithm 1: The construction of the regularization functional $\mathcal{R}_{G'}$ from a point cloud $\mathcal{X}$.

Input: $\mathcal{X} = \{X_1, \ldots, X_u\}$, manifold dimension n, k.
Output: G'.

```
Initialization: Find k nearest neighbors, e.g., build KD-tree;
for i = 1,...,u do
    Construct the local approximation
    M at X_i using n-dimensional PCA of N_k(X_i);
    Calculate the local regularization
    matrix G^i for N_k(X_i) in the PCA representation:
    (G')^i = (L^i)^T (K^i)^{+1} L^i (Eqs. 29 and 30);
end
Re-arrange {(G')^i} according to the indices
of {f^i} in f to construct matrix G' s.t. f^T G' f = R_{G'}(f);
```

$\mathcal{R}_{G'}$ construction pseudocode is in Algorithm 6. Supplemental MATLAB code is available on the author's webpage. This real code references the pseudocode to aid explanation.

6. Experiments

To demonstrate our algorithm performance, we consider examples of estimating continuous values in human body shape and pose analysis: the MoCap database [2] of optical motion capture data and the CAESAR human body database [17]. For comparison, we performed experiments with existing graph Laplacian (*Lap*) [13, 3] and iterated graph Laplacian (*i-Lap*) [24] regularizers.

Toy example. We uniformly sample $10,000$ data points in $[-1,1] \times [-1,1]$. Five points (four corners and center) were assigned labels in $\{-1, 10\}$ (red dots in Fig. 1). While the original graph Laplacian (*Lap*) produces a "spiky" function, the iterated graph Laplacian (*i-Lap*) and our regularizer (*LG*: local Gaussian) produced smooth functions, which demonstrate the importance of high-order regularization.

MoCap database. This contains $50,000$ entries describing human body poses captured with an optical marker-based system [2]. For each *pose* entry, inverse kinematics is applied to recover skeletal joint angles represented as axis-angle $(\hat{e}, \theta)$. A body model comprising a surface mesh consisting of $6,449$ vertices is deformed via surface skinning by embedding this skeleton of 62 joints, leading to 42 degrees of freedom parameterized by the joint angles. The locations of end effectors (left/right hand, left/right foot, and head) were separately recorded from the surface mesh model. These constitute a 15 (5×3)-dimensional coarse, mid-level representation (Figure 3). The task is to estimate the 42-dimensional joint angles from the mid-level representation. This is useful for retrieval and indexing of motion data, e.g., for motion capture with motion priors of similar poses [2], fast

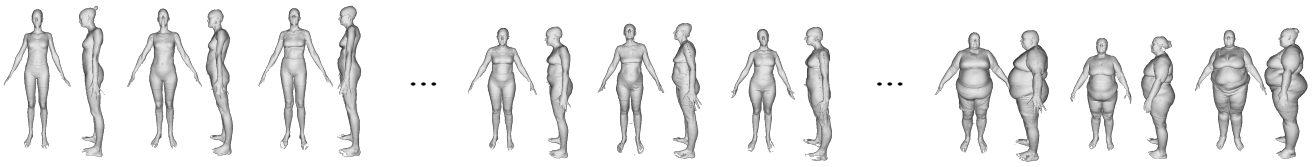

Figure 2. The CAESAR database contains 4,258 3D scans of human beings, along with ground-truth body measurements taken with calipers. Here, we see variation in female shape across the database.

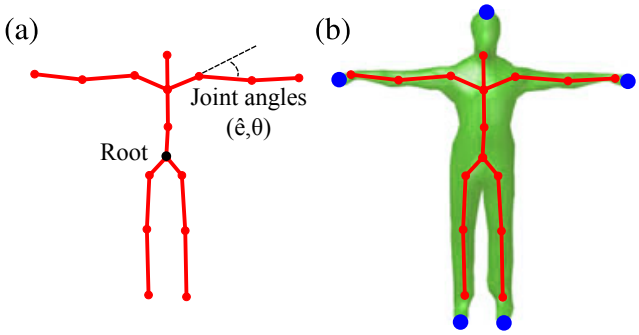

Figure 3. (a) Skeletal kinematic chain. (b) End effectors (blue) recovered from a geometric model fit to the skeleton. Each joint angle is in angle-axis form, with axis $\hat{e}$ and angle θ.

Table 1. Mean L2-reconstruction error on the MoCap dataset.

Algorithm	*Lap*	*i-Lap*	*LG*
Joint angles error	1.62	1.24	**1.16**
Joint locations error	1.22	0.72	**0.50**

MoCap data indexing in authoring tools [14], or synthesis of motions from sparse sensor data with pose priors [21].

We randomly chose 100 labels, with the remaining data points used as unlabeled examples. The experiment was repeated 10 times with different sets of labeled examples and the results were averaged (Table 1). We also show the corresponding results measured in the 186 (62×3)-dimensional joint location space that is restored by applying forward kinematics. Both in terms of joint angle and position error, we outperform the competing methods.

CAESAR database. This contains 4,258 3D scans of human beings, along with ground-truth measurements of their bodies obtained with calipers (Fig. 2). Detailed description and example usages of this dataset can be found in [17, 8]. With a technique based on the work of Pishchulin et al. [16], we fit a statistical body model to each of the scans, which is able to represent body variations such as height, hip and belly girth, limb length, and so on. Each body scan is represented as a vector in 20-dimensional feature space spanned by a linear shape basis.

Table 2 shows absolute error in semi-supervised learning performance when comparing the three regularizers, over different numbers of labeled items. Each experiment was repeated 10 times and averaged. In most cases, our approach significantly improves performance. The worse performance of *LG* over *i-Lap* for some cases is caused by over-fitting in cross-validation.

Parameters. There are four hyper-parameters in our algorithm: the number (k) of nearest neighbors, the dimensionality (m) of the manifold, the regularization parameter (λ), and the local scale parameter (σ; see Eq. 23). In preliminary experiments, the performance of our algorithm varied significantly with respect to the first three parameters, while it was rather robust to σ variations. We decide σ adaptively for each point X_i, at 0.1 times the mean distance between X_i and the elements of $N_k(X_i)$ while the remaining three hyper-parameters were optimized by 5-fold cross-validation (CV) where, in each run, a subset of labeled points were left out while all unlabeled data points are kept. There are three and four hyper-parameters for *Lap* and *i-Lap*, respectively: λ, k, and the parameter b for building the graph Laplacian (Eq. 3) for *Lap* and the iteration parameter p for *i-Lap* (Eq. 5). These parameters were tuned in the same way as for *LG*. Across Table 2, k varied from 20 to 40, m from 10 to 17, λ from $10e^{-8}$ to $10e^{-5}$, b from 5 to 300, and p from 1 to 4.

Computation complexity and time. For each algorithm, this depends on the number of data points u, the number of nearest neighbors k, and the number of non-zeros entries of the resulting regularization matrix that lies in-between $O(uk)$ and $O(uk^2)$, depending on the well-behavedness of neighborhoods (where $O(uk^2)$ corresponds to random neighbors). The most time-consuming component of each algorithm is solving the corresponding system.

For the MoCap dataset, with $u = 50,000$, $k = 20$, and $p=4$ for *i-Lap*, it took 30, 50, and 40 seconds for *Lap*, *i-Lap*, and *LG* to build the regularization matrices, respectively. The corresponding sparsity, defined as the number of nonzero entries divided by the number of all entries in the regularization matrix, is 0.0005, 0.0400, and 0.0017 for *Lap*, *i-Lap*, and *LG*, respectively. This resulted in the run-times for solving the systems of roughly 50, 720, and 120 seconds, respectively, on an Intel Xeon 3GHz CPU in MATLAB. For the CAESAR dataset, with $u=4,258$, run-times were only a few seconds. The improvement in computation time for large sets, coupled with the accuracy improvements demonstrated, makes our new regularizer a good alternative to *Lap* and *i-Lap*.

7. Discussion

We focused on constructing analytic solutions of Eq. 1. In general, an iterative solver can be used instead (i.e., gradient descent). In this case, the iterated Laplacian *i-Lap* need not be computed explicitly as its action on a vector can be computed by iterating matrix-vector multiplications. We briefly explored this possiblity:

Table 2. Mean absolute error for estimating 6 ground truth parameters from the CAESAR dataset. Bold face marks the best results. The *Deviation from mean* replaces the evaluation of each $f(X_i)$ with the mean of each output variable (calculated from the entire data set). This presents an idea of the difficulty of the estimation problem for each parameter.

# Labels	Algorithm	*Age*	*Arm length*	*Shoulder breadth*	*Weight*	*Sit height*	*Foot length*
	Deviation from mean	10.89	35.98	36.13	13.94	39.50	15.57
20	*Lap*	**10.89**	30.23	32.69	12.80	32.58	13.80
	i-Lap	12.46	19.54	25.34	6.30	20.54	10.30
	LG	12.55	**17.92**	**20.64**	**3.17**	**19.31**	**9.87**
50	*Lap*	10.79	24.28	28.88	10.99	26.05	11.14
	i-Lap	**10.61**	17.43	21.14	6.62	18.39	**8.20**
	LG	11.03	**16.30**	**16.15**	**2.25**	**16.49**	8.34
100	*Lap*	10.64	20.62	26.00	9.60	21.72	9.46
	i-Lap	10.21	16.97	19.33	5.08	17.65	**7.99**
	LG	**9.85**	**15.07**	**15.39**	**1.98**	**15.59**	8.05
200	*Lap*	10.45	18.23	23.07	8.09	18.99	8.38
	i-Lap	9.99	16.49	17.56	4.11	17.25	7.81
	LG	**9.40**	**13.96**	**14.93**	**1.77**	**12.42**	**7.76**
500	*Lap*	10.00	16.44	19.39	6.02	17.31	7.75
	i-Lap	9.52	15.62	15.84	2.93	16.65	7.59
	LG	**8.93**	**13.42**	**14.53**	**1.60**	**11.94**	**7.54**

During gradient evaluation, the number of matrix-vector multiplcations increases from 1 to p: For MoCap (u=50,000, p=4), *i-Lap* iterative optimization was around five times slower than analytic optimization, and three times slower than our iterative LG optimization. For *i-Lap* with $p>4$, analytic optimization is not feasible and the iterative *i-Lap* could be used; however, our LG requires no iteration. This suggests that LG can still be faster than *i-Lap*. For larger-scale problems, both methods need iteration.

Local first-order approximation approaches, like ours, are supported by their success in manifold learning and regularization [19, 6]. However, local first-order approximations result in the corresponding derivatives being exact up to second order, but at third order and higher, the derivatives may deviate from the underlying covariant derivatives. Nevertheless, since the highest-order terms agree, calculating the Euclidean derivatives therein enables us to completely eliminate the possibility of generating degenerate functions.

Furthermore, the number of hyper-parameters to be tuned (the other parameter σ^k is adaptively decided) is the same as for classical graph Laplacian and is one smaller than for iterated graph Laplacian. Combined with the observed empirical performance of our algorithm, and the computationally efficient regularization, this supports its usage.

Our local Gaussian interpolation varies σ^k with the local neighborhood size $N_k(X)$ (instead of making it constant per dataset), which desires rigorous limit case behavior analysis. Further future work should address the theoretical analysis of our regularizer (e.g., error bound), and the possible benefit to spectral clustering and dimensionality reduction.

8. Conclusion

We have presented the local Gaussian regularizer: a new high-order regularization framework on data manifolds. Our algorithm does not suffer from the degeneracy of graph Laplacian-based regularizers. Further, it leads to a sparse regularization matrix, thereby facilitating application to large-scale datasets. Experiments on human body shape and pose analysis demonstrate the improved accuracy and faster execution time of our new algorithm.

Acknowledgements

This work has been benefited from discussions with Matthias Hein, and from the dataset processing and model fitting work of Leonid Pishchulin and Thomas Helten. Kwang In Kim thanks EPSRC EP/M00533X/1 and EP/M006255/1, James Tompkin and Hanspeter Pfister thank NSF CGV-1110955, and James Tompkin and Christian Theobalt thank the Intel Visual Computing Institute. Part of this work was completed while Kwang In Kim and James Tompkin were at Max Planck Institute for Informatics.

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. 4

[2] A. Baak, M. Müller, G. Bharaj, H.-P. Seidel, and C. Theobalt. A data-driven approach for real-time full body pose reconstruction from a depth camera. In *Proc. ICCV*, pages 1092–1099, 2011. 6

[3] M. Belkin and P. Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, 15(6):1373–1396, 2003. 1, 6

[4] 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, 2, 4

[5] O. Chapelle, B. Schölkopf, and A. Zien. *Semi-Supervised Learning*. MIT Press, Cambridge, MA, 2006. 1

[6] 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. 3, 4, 6, 8

[7] R. M. Dudley. *Real Analysis and Probability*. Cambridge University Press, Cambridge, 2nd edition, 2002. 2, 4

[8] S. Hauberg, O. Freifeld, and M. J. Black. A geometric take on metric learning. In *NIPS*, pages 2033–2041, 2012. 7

[9] 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, 2, 4

[10] M. Kara. An analytical expression for arbitrary derivatives of Gaussian functions $\exp(ax^2)$. *Internal Journal of Physical Sciences*, 4(4):247–249, 2009. 3

[11] 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. 4, 6

[12] J. M. Lee. *Riemannian Manifolds- An Introduction to Curvature*. Springer, New York, 1997. 2, 3, 4

[13] U. Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395–416, 2007. 1, 2, 6

[14] M. Müller, T. Röder, and M. Clausen. Efficient content-based retrieval of motion capture data. *ACM Trans. Graphics (Proc. SIGGRAPH)*, 24(3):677–685, 2005. 7

[15] B. Nadler, N. Srebro, and X. Zhou. Statistical analysis of semi-supervised learning: the limit of infinite unlabelled data. In *NIPS*, pages 1330–1338, 2009. 1, 2

[16] L. Pishchulin, S. Wuhrer, T. Helten, C. Theobalt, and B. Schiele. Building statistical shape spaces for 3d human modeling. In *arXiv:1503.05860*, 2015. 7

[17] K. Robinette, H. Daanen, and E. Paquet. The CAESAR project: a 3-D surface anthropometry survey. In *Proc. 3-D Digital Imaging and Modeling*, pages 380–386, 1999. 6, 7

[18] S. Rosenberg. *The Laplacian on a Riemannian Manifold: An Introduction to Analysis on Manifolds*. Cambridge University Press, Cambridge, 1997. 2

[19] S. Roweis and L. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, pages 2323–2326, 2000. 4, 8

[20] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE TPAMI*, 22(8):888–905, 2000. 1

[21] J. Tautges, A. Zinke, B. Krüger, J. Baumann, A. Weber, T. Helten, M. Müller, H.-P. Seidel, and B. Eberhardt. Motion reconstruction using sparse accelerometer data. *ACM Trans. Graphics*, 30(3):18:1–18:12, May 2011. 7

[22] S.-F. C. X.-M. Wu, Z. Li. Analyzing the harmonic structure in graph-based learning. In *NIPS*, pages 3129–3137, 2013. 2

[23] A. L. Yuille and N. M. Grzywacz. The motion coherence theory. In *Proc. ICCV*, pages 344–353, 1988. 2, 5

[24] X. Zhou and M. Belkin. Semi-supervised learning by higher order regularization. *JMLR W&CP (Proc. AISTATS)*, pages 892–900, 2011. 1, 2, 3, 6