# Differential Geometric Regularization for Supervised Learning of Classifiers

**Qinxun Bai** QINXUN@CS.BU.EDU
Department of Computer Science, Boston University, Boston, MA 02215 USA

**Steven Rosenberg** SR@MATH.BU.EDU
Department of Mathematics and Statistics, Boston University, Boston, MA 02215 USA

**Zheng Wu** WUZHENG@BU.EDU
The Mathworks Inc. Natick, MA 01760 USA

**Stan Sclaroff** SCLAROFF@BU.EDU
Department of Computer Science, Boston University, Boston, MA 02215 USA

## Abstract

We study the problem of supervised learning for both binary and multiclass classification from a unified geometric perspective. In particular, we propose a geometric regularization technique to find the submanifold corresponding to an estimator of the class probability $P(y|\boldsymbol{x})$. The regularization term measures the volume of this submanifold, based on the intuition that overfitting produces rapid local oscillations and hence large volume of the estimator. This technique can be applied to regularize any classification function that satisfies two requirements: firstly, an estimator of the class probability can be obtained; secondly, first and second derivatives of the class probability estimator can be calculated. In experiments, we apply our regularization technique to standard loss functions for classification, our RBF-based implementation compares favorably to widely used regularization methods for both binary and multiclass classification.

## 1. Introduction

In supervised learning for classification, the idea of regularization seeks a balance between a perfect description of the training data and the potential for generalization to unseen data. Most regularization techniques are defined in the form of penalizing some functional norms. For instance, one of the most successful classification methods, the support vector machine (SVM) (Vapnik, 1998; Schölkopf & Smola, 2002) and its variants (Bartlett et al., 2006; Steinwart, 2005), use a RKHS norm as a regularizer. While functional norm based regularization is widely-used in machine learning, we feel that there is important local geometric information overlooked by this approach.

In many real world classification problems, if the feature space is meaningful, then all samples that are locally within a small enough neighborhood of a training sample should have class probability $P(y|\boldsymbol{x})$ similar to the training sample. For instance, a small enough perturbation of RGB values at some pixels of a human face image should not change dramatically the likelihood of correct identification of this image during face recognition. However, such "small local oscillations" of the class probability are not explicitly incorporated by penalizing commonly used functional norms. For instance, as reported by Goodfellow et al. (2014), linear models and their combinations can be easily fooled by barely perceptible perturbations of a correctly predicted image, even though a $L^2$ regularizer is adopted.

Geometric regularization techniques have also been studied in machine learning. Belkin et al. (2006) employed geometric regularization in the form of the $L2$ norm of the gradient magnitude supported on a manifold. This approach exploits the geometry of the marginal distribution $P(\boldsymbol{x})$ for semi-supervised learning, rather than the geometry of the class probability $P(y|\boldsymbol{x})$. Other related geometric regularization methods are motivated by the success of level set methods in image segmentation (Cai & Sowmya, 2007; Varshney & Willsky, 2010) and Euler's Elastica in image processing (Lin et al., 2012; 2015). In particular, the Level Learning Set (Cai & Sowmya, 2007) combines a counting function of training samples and a geometric penalty on the surface area of the decision boundary. The Geometric Level Set (Varshney & Willsky, 2010) generalizes this

*Proceedings of the $33^{rd}$ International Conference on Machine Learning*, New York, NY, USA, 2016. JMLR: W&CP volume 48. Copyright 2016 by the author(s).

idea to standard empirical risk minimization schemes with margin-based loss. Along this line, the Euler's Elastica Model (Lin et al., 2012; 2015) proposes a regularization technique that penalizes both the gradient oscillations and the curvature of the decision boundary. However, all three methods focus on the geometry of the decision boundary supported in the domain of the feature space, and the "small local oscillation" of the class probability is not explicitly addressed.

In this work, we argue that "small local oscillations" of the class probability can be characterized by the geometry of a specific submanifold in the product space of the feature domain and the probabilistic output space. Let $\boldsymbol{f} : \mathcal{X} \rightarrow \Delta^{L-1}$ be a class probability estimator, where $\mathcal{X}$ is the feature space and $\Delta^{L-1}$ is the probabilistic simplex for $L$ classes. From a geometric perspective, if we regard $\{(\boldsymbol{x}, \boldsymbol{f}(\boldsymbol{x}))|\boldsymbol{x} \in \mathcal{X}\}$, the *functional graph* (in the geometric sense) of $\boldsymbol{f}$, as a submanifold in $\mathcal{X} \times \Delta^{L-1}$, then "small local oscillations" can be measured by the local volume or the more sensitive local curvature of this submanifold.

In our approach, the learning process can be viewed as a submanifold fitting problem that is solved by a geometric flow method. In particular, our approach finds a submanifold by iteratively fitting the training samples in a curvature or volume decreasing manner without any *a priori* assumptions on the geometry of the submanifold in $\mathcal{X} \times \Delta^{L-1}$. We use gradient flow methods to find an optimal direction, i.e. at each step we find the vector field pointing in the optimal direction to move $\boldsymbol{f}$. As we will see in the next section, this regularization approach naturally handles binary and multiclass classification in a unified way, while previous decision boundary based techniques (and most functional regularization approaches) are originally designed for binary classification, and rely on "one versus one", "one versus all" or more efficiently a binary coding strategy (Varshney & Willsky, 2010) to generalize to multiclass case.

In experiments, a radial basis function (RBF) based implementation of our formulation compares favorably to widely used binary and multiclass classification methods on datasets from the UCI repository and real-world datasets including the Flickr Material Database (FMD) and the MNIST Database of handwritten digits.

In summary, our contributions are:

- A geometric perspective on overfitting and a regularization approach that exploits the geometry of a robust class probability estimator for classification,
- A unified gradient flow based algorithm for both binary and multiclass classification that can be applied to standard loss functions, and
- A RBF-based implementation that achieves promising experimental results.

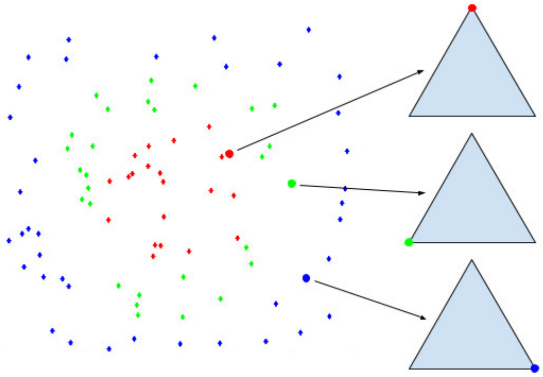


*Figure 1.* Example of three-class learning, i.e., $L = 3$, where the input space $\mathcal{X}$ is $2d$. Training samples of the three classes are marked with red, green and blue dots respectively. The class label for each training sample corresponds to a vertex of the simplex $\Delta^{L-1}$. As a result, each mapped training point $(\boldsymbol{x}_i, \boldsymbol{z}_i)$ lies on one face (corresponding to its label $y_i$) of the space $\mathcal{X} \times \Delta^2$.

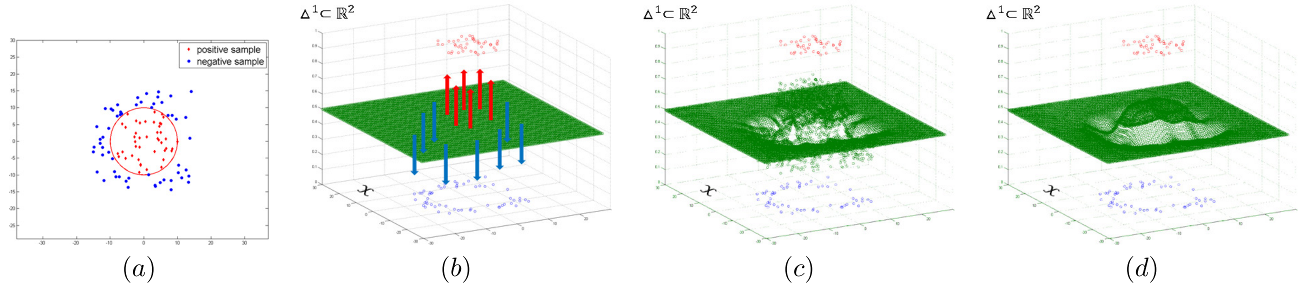
## 2. Method Overview

In our work, we propose a regularization scheme that exploits the geometry of a robust class probability estimator and suggest a gradient flow based approach to solve for it. In the following, we will describe our approach. Related mathematical notation is summarized in Table 2.

Following the probabilistic setting of classification, given a sample (feature) space $\mathcal{X} \subset \mathbb{R}^N$, a label space $\mathcal{Y} = \{1, \ldots, L\}$, and a finite training set of labeled samples $\mathcal{T}_m = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^m$, where each training sample is generated i.i.d. from distribution $P$ over $\mathcal{X} \times \mathcal{Y}$, our goal is to find a $h_{\mathcal{T}_m} : \mathcal{X} \rightarrow \mathcal{Y}$ such that for any new sample $\boldsymbol{x} \in \mathcal{X}$, $h_{\mathcal{T}_m}$ predicts its label $\hat{y} = h_{\mathcal{T}_m}(\boldsymbol{x})$. The optimal generalization risk (Bayes risk) is achieved by the classifier $h^*(\boldsymbol{x}) = \operatorname{argmax}\{\eta^\ell(\boldsymbol{x}), \ell \in \mathcal{Y}\}$, where $\boldsymbol{\eta} = (\eta^1, \ldots, \eta^L)$ with $\eta^\ell : \mathcal{X} \rightarrow [0, 1]$ being the $\ell^{\text{th}}$ class probability, i.e. $\eta^\ell(\boldsymbol{x}) = P(y = \ell|\boldsymbol{x})$.

Our regularization approach exploits the geometry of the class probability estimator, and can be regarded as a "hybrid" plug-in/ERM scheme (Audibert & Tsybakov, 2007). A regularized loss minimization problem is setup to find an estimator $\boldsymbol{f} : \mathcal{X} \rightarrow \Delta^{L-1}$, where $\Delta^{L-1}$ is the standard $(L-1)$-simplex in $\mathbb{R}^L$, and $\boldsymbol{f} = (f^1, \ldots, f^L)$ is an estimator of $\boldsymbol{\eta}$ with $f^\ell : \mathcal{X} \rightarrow [0, 1]$. The estimator $\boldsymbol{f}$ is then "plugged-in" to get the classifier $h_{\boldsymbol{f}}(\boldsymbol{x}) = \operatorname{argmax}\{f^\ell(\boldsymbol{x}), \ell \in \mathcal{Y}\}$.

Figure 1 shows an example of the setup of our approach, for a synthetic three-class classification problem. The submanifold corresponding to estimator $\boldsymbol{f}$ is the graph (in the geometric sense) of $\boldsymbol{f}$: $\operatorname{gr}(\boldsymbol{f}) = \{(\boldsymbol{x}, f^1(\boldsymbol{x}), \ldots, f^L(\boldsymbol{x})) :$



$(a)$ $(b)$ $(c)$ $(d)$

*Figure 2.* Example of binary learning via gradient flow. As shown in $(a)$, the feature space $\mathcal{X}$ is $2d$, training points are sampled uniformly within the region $[-15, 15] \times [-15, 15]$, and labeled by the function $y = \operatorname{sign}(10 - \|\boldsymbol{x}\|_2)$ (the red circle). In the initialization step, shown in $(b)$, positive and negative training points map to the two faces of the space $\mathcal{X} \times \Delta^1$ respectively. Our gradient flow method starts from a neutral function $\boldsymbol{f}_0 \equiv \frac{1}{2}$ and moves towards the negative direction (red and blue arrows) of the penalty gradient $\nabla \mathcal{P}_{\boldsymbol{f}_0}$. Figure $(c)$ shows the submanifold $(\mathrm{gr}(\boldsymbol{f}_1))$ one step after $(b)$. The submanifold then continues to evolve towards $-\nabla \mathcal{P}_{\boldsymbol{f}_t}$ step by step and the final output after convergence of the algorithm is shown in $(d)$.

$\boldsymbol{x} \in \mathcal{X}\} \subset \mathcal{X} \times \Delta^{L-1}$. We denote a point in the space $\mathcal{X} \times \Delta^{L-1}$ as $(\boldsymbol{x}, \boldsymbol{z}) = (x^1, \dots, x^N, z^1, \dots, z^L)$, where $\boldsymbol{x} \in \mathcal{X}$ and $\boldsymbol{z} \in \Delta^{L-1}$. Then in this product space, a training pair $(\boldsymbol{x}_i, y_i = \ell)$ naturally maps to the point $(\boldsymbol{x}_i, \boldsymbol{z}_i) = (\boldsymbol{x}_i, 0, \dots, 1, \dots, 0)$, with the one-hot vector $\boldsymbol{z}_i$ (with the 1 in its $\ell$-th slot) at the vertex of $\Delta^{L-1}$ corresponding to $P(y = y_i|\boldsymbol{x}) = 1$.

We point out two properties of this geometric setup. Firstly, it inherently handles multiclass classification, with binary classification as a special case. Secondly, while the dimension of the ambient space, i.e. $\mathbb{R}^{N+L}$, depends on both the feature dimension $N$ and number of classes $L$, the intrinsic dimension of the submanifold $\mathrm{gr}(\boldsymbol{f})$ only depends on $N$.

### 2.1. Variational formulation

We want $\mathrm{gr}(\boldsymbol{f})$ to approach the mapped training points while remaining as flat as possible, so we impose a penalty on $\boldsymbol{f}$ consisting of an empirical loss term $\mathcal{P}_{\mathcal{T}_m}$ and a geometric regularization term $\mathcal{P}_G$. For $\mathcal{P}_{\mathcal{T}_m}$, we can choose either the widely-used cross-entropy loss function for multiclass classification or the simpler Euclidean distance function between the simplex coordinates of the graph point and the mapped training point. For $\mathcal{P}_G$, we would ideally consider an $L^2$ measure of the Riemann curvature of $\mathrm{gr}(\boldsymbol{f})$, as the vanishing of this term gives optimal (i.e., locally distortion free) diffeomorphisms from $\mathrm{gr}(\boldsymbol{f})$ to $\mathbb{R}^N$. However, the Riemann curvature tensor takes the form of a combination of derivatives up to third order, and the corresponding gradient vector field is even more complicated and inefficient to compute in practice. As a result, we measure the graph's volume, $\mathcal{P}_G(\boldsymbol{f}) = \int_{\mathrm{gr}(\boldsymbol{f})} \mathrm{dvol}$, where dvol is the induced volume from the Lebesgue measure on the ambient space $\mathbb{R}^{N+L}$.

More precisely, we find the function that minimizes the following penalty $\mathcal{P}$:

$$\mathcal{P} = \mathcal{P}_{\mathcal{T}_m} + \lambda \mathcal{P}_G : \mathcal{M} = \mathrm{Maps}(\mathcal{X}, \Delta^{L-1}) \to \mathbb{R} \qquad (1)$$

on the set $\mathcal{M}$ of smooth functions from $\mathcal{X}$ to $\Delta^{L-1}$, where $\lambda$ is the tradeoff parameter between empirical loss and regularization. It is important to note that any relative scaling of the domain $\mathcal{X}$ will not affect the estimate of the class probability $\boldsymbol{\eta}$, as scaling will distort $\mathrm{gr}(\boldsymbol{f})$ but will not change the critical function estimating $\boldsymbol{\eta}$.

### 2.2. Gradient flow and geometric foundation

The standard technique for solving variational formulas is the Euler-Lagrange PDE. However, due to our geometric term $\mathcal{P}_G$, finding the minimal solutions of the Euler-Lagrange equations for $\mathcal{P}$ is difficult, instead, we solve for $\operatorname{argmin} \mathcal{P}$ using gradient flow in functional space $\mathcal{M}$.

A simple but intuitive simulated example of binary learning using gradient flow for our approach is given in Figure 2. For the explanation purposes only, we replace $\mathcal{M}$ with a finite dimensional Riemannian manifold $M$. Without loss of generality, we also assume that $\mathcal{P}$ is smooth, then it has a differential $d\mathcal{P}_{\boldsymbol{f}} : T_{\boldsymbol{f}}M \to \mathbb{R}$ for each $\boldsymbol{f} \in M$, where $T_{\boldsymbol{f}}M$ is the tangent space to $M$ at $\boldsymbol{f}$. Since $d\mathcal{P}_{\boldsymbol{f}}$ is a linear functional on $T_{\boldsymbol{f}}M$, there is a unique tangent vector, denoted $\nabla \mathcal{P}_{\boldsymbol{f}}$, such that $d\mathcal{P}_{\boldsymbol{f}}(\boldsymbol{v}) = \langle \boldsymbol{v}, \nabla \mathcal{P}_{\boldsymbol{f}} \rangle$ for all $\boldsymbol{v} \in T_{\boldsymbol{f}}M$. $\nabla \mathcal{P}_{\boldsymbol{f}}$ points in the direction of maximal increase of $\mathcal{P}$ at $\boldsymbol{f}$. Thus, the solution of the negative gradient flow $d\boldsymbol{f}_t/dt = -\nabla \mathcal{P}_{\boldsymbol{f}_t}$ is a flow line of steepest descent starting at an initial $\boldsymbol{f}_0$. For a dense open set of initial points, flow lines approach a local minimum of $\mathcal{P}$ at $t \to \infty$. We always choose the initial function $\boldsymbol{f}_0$ to be the "neutral" choice $\boldsymbol{f}_0(\boldsymbol{x}) \equiv (\frac{1}{L}, \dots, \frac{1}{L})$ which reasonably assigns equal conditional probability to all classes.

Similar gradient flow procedures are widely used in variational problems, such as level set methods (Osher & Sethian, 1988; Sethian, 1999), Mumford-Shah functional (Mumford & Shah, 1989), etc. In the classification literature, Varshney & Willsky (2010) were the first to use gradient flow methods for solving level set based energy functions, then followed by Lin et al. (2012; 2015) to solve Euler's Elastica models. In our case, we are exploiting the geometry in the space $\mathcal{X} \times \Delta^{L-1}$, rather than standard vector spaces.

Since our gradient flow method is actually applied on the infinite dimensional manifold $\mathcal{M}$, we have to understand both the topology and the Riemannian geometry of $\mathcal{M}$. For the topology, we put the Fréchet topology on $\mathcal{M}' = \mathrm{Maps}(\mathcal{X}, \mathbb{R}^L)$, the set of smooth maps from $\mathcal{X}$ to $\mathbb{R}^L$, and take the induced topology on $\mathcal{M}$. Intuitively speaking, two functions in $\mathcal{M}$ are close if the functions and all their partial derivatives are pointwise close. Since $\mathcal{M}$ is a closed Fréchet submanifold with corners inside the vector space $\mathcal{M}'$, we can identify each $T_{\boldsymbol{f}}\mathcal{M}$ with a closed cone inside $\mathcal{M}'$. For the Riemannian metric on $\mathcal{M}$, we restrict the $L^2$ metric on $\mathcal{M}'$ to $T_{\boldsymbol{f}}\mathcal{M}$: $\langle \phi_1, \phi_2 \rangle := \int_{\mathcal{X}} \phi_1(\boldsymbol{x})\phi_2(\boldsymbol{x})\mathrm{dvol}_{\boldsymbol{x}}$, with $\phi_i \in \mathcal{M}'$ and $\mathrm{dvol}_{\boldsymbol{x}}$ being the volume form of the induced Riemannian metric on the graph of $\boldsymbol{f}$. (Strictly speaking, the volume form is pulled back to $\mathcal{X}$ by $\boldsymbol{f}$, usually denoted by $\boldsymbol{f}^*\mathrm{dvol}$.)

The differential $d\mathcal{P}_{\boldsymbol{f}}$ is linear as above, and by a direct calculation, there is a unique tangent vector $\nabla\mathcal{P}_{\boldsymbol{f}} \in T_{\boldsymbol{f}}\mathcal{M}$ such that $d\mathcal{P}_{\boldsymbol{f}}(\phi) = \langle \nabla\mathcal{P}_{\boldsymbol{f}}, \phi \rangle$ for all $\phi \in T_{\boldsymbol{f}}\mathcal{M}$. Thus, we can construct the gradient flow equation. However, unlike the case of finite dimensions, the existence of flow lines is not automatic. Assuming the existence of flow lines, a generic initial point flows to a local minimum of $\mathcal{P}$. In any case, our RBF-based implementation in §3 mimicking gradient flow is well defined.

Note that we think of $\mathcal{X}$ as large enough so that the training data actually is sampled well inside $\mathcal{X}$. This allows us to treat $\mathcal{X}$ as a closed manifold in our gradient calculations, so that boundary effects can be ignored. A similar *natural boundary condition* is also adopted by previous work (Varshney & Willsky, 2010; Lin et al., 2012; 2015).

### 2.3. More on related work

There exist some other works that are related to some aspects of our work. Most notably, Sobolev regularization, involves functional norms of a certain number of derivatives of the prediction function. For instance, the manifold regularization (Belkin et al., 2006) mentioned in §1 uses a Sobolev regularization term,

$$\int_{x\in\mathcal{M}} \|\nabla_{\mathcal{M}} f\|^2 dP(x), \tag{2}$$

where $f$ is a smooth function on manifold $\mathcal{M}$. A discrete version of (2) corresponds to the graph Laplacian regularization (Zhou & Schölkopf, 2005). Lin et al. (2015) discussed in detail the difference between a Sobolev norm and a curvature-based norm for the purpose of exploiting the geometry of the decision boundary.

For our purpose, while imposing, say, a high Sobolev norm[1], will also lead to a flattening of the hypersurface $\mathrm{gr}(\boldsymbol{f})$, these norms are not specifically tailored to measuring the flatness of $\mathrm{gr}(\boldsymbol{f})$. In other words, a high Sobolev norm bound will imply the volume bound we desire, but not *vice versa*. As a result, imposing high Sobolev norm constraints (regardless of computational difficulties) overshrinks the hypothesis space from a learning theory point of view. In contrast, our regularization term (given in (11)) involves only the combination of first derivatives of $\boldsymbol{f}$ that specifically address the geometry behind the "small local oscillation" prior observed in practice.

Our training procedure for finding the optimal graph of a function is, in a general sense, also related to the manifold learning problem (Tenenbaum et al., 2000; Roweis & Saul, 2000; Belkin & Niyogi, 2003; Donoho & Grimes, 2003; Zhang & Zha, 2005; Lin & Zha, 2008). The most closely related work is (Donoho & Grimes, 2003), which seeks a flat submanifold of Euclidean space that contains a dataset. Again, there are key differences. Since the goal of (Donoho & Grimes, 2003) is dimensionality reduction, their manifold has high codimension, while our functional graph has codimension $L-1$, which may be as low as 1. More importantly, we do not assume that the graph of our target function is a flat (or volume minimizing) submanifold, and we instead flow towards a function whose graph is as flat (or volume minimizing) as possible. In this regard, our work is related to a large body of literature on Morse theory in finite and infinite dimensions, and on mean curvature flow (Chen et al., 1999; Mantegazza, 2011).

## 3. Example Formulation: RBFs

We now illustrate our approach using an RBF representation of our estimator $\boldsymbol{f}$. RBFs are also used by previous geometric classification methods (Varshney & Willsky, 2010; Lin et al., 2012; 2015).

Given values of $\boldsymbol{f}$ are probabilistic vectors, it is common to represent $\boldsymbol{f}$ as a "softmax" output of RBFs, i.e.

$$f^j = \frac{e^{h^j}}{\sum_{l=1}^{L} e^{h^l}}, \qquad \text{where } h^j = \sum_{i=1}^{m} a_i^j \varphi_i(\boldsymbol{x}), \\ \text{for } j = 1, \ldots, L, \tag{3}$$

where $\varphi_i(\boldsymbol{x}) = e^{-\frac{1}{c}\|\boldsymbol{x}-\boldsymbol{x}_i\|^2}$ is the RBF function centered at training sample $\boldsymbol{x}_i$, with kernel width parameter $c$.

Estimating $\boldsymbol{f}$ becomes an optimization problem for the $m \times L$ coefficient matrix $A = (a_i^\ell)$. The following equation determines $A$:

$$[\boldsymbol{h}(\boldsymbol{x}_1), \ldots, \boldsymbol{h}(\boldsymbol{x}_m)]^T = GA, \text{ where } G_{ij} = \varphi_j(\boldsymbol{x}_i). \tag{4}$$

[1] "High Sobolev norm" is the conventional term for Sobolev norm with high order of derivatives.

To plug this RBF representation into our gradient flow scheme, the gradient vector field $\nabla\mathcal{P}_{\boldsymbol{f}}$ is evaluated at each sample point $\boldsymbol{x}_i$, and $A$ is updated by

$$A \leftarrow A - \tau G^{-1}\left[\nabla\mathcal{P}_{\boldsymbol{h}}(\boldsymbol{x}_1), \ldots, \nabla\mathcal{P}_{\boldsymbol{h}}(\boldsymbol{x}_m)\right]^T, \qquad (5)$$

where $\tau$ is the step-size parameter, and

$$\nabla\mathcal{P}_{\boldsymbol{h}}(\boldsymbol{x}_i) = \left[\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{h}}\right]^T_{\boldsymbol{x}_i} \nabla\mathcal{P}_{\boldsymbol{f}}(\boldsymbol{x}_i). \qquad (6)$$

Here $\nabla\mathcal{P}_{\boldsymbol{h}}(\boldsymbol{x}_i)$ denotes the gradient vector field w.r.t. $\boldsymbol{h}$ evaluated at $\boldsymbol{x}_i$, and the $L \times L$ Jacobian matrix $\left[\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{h}}\right]_{\boldsymbol{x}_i}$ can be obtained in closed form from (3). In the following subsections, we give exact forms of the empirical penalty $\mathcal{P}_{\mathcal{T}_m}$ and the geometric penalty $\mathcal{P}_G$, and discuss the computation of $\nabla\mathcal{P}_{\boldsymbol{h}}$ for both penalty terms.

### 3.1. The empirical penalty $\mathcal{P}_{\mathcal{T}_m}$

We consider two widely-used loss functions for the empirical penalty term $\mathcal{P}_{\mathcal{T}_m}$.

**Quadratic loss.** Since $\mathcal{P}_{\mathcal{T}_m}$ measures the deviation of $\mathrm{gr}(\boldsymbol{f})$ from the mapped training points, it is natural to choose the quadratic function of the Euclidean distance in the simplex $\Delta^{L-1}$,

$$\mathcal{P}_{\mathcal{T}_m}(\boldsymbol{f}) = \sum_{i=1}^{m} \|\boldsymbol{f}(\boldsymbol{x}_i) - \boldsymbol{z}_i\|^2, \qquad (7)$$

where $\boldsymbol{z}_i$ is the one-hot vector corresponding to the ground truth label of $\boldsymbol{x}_i$. The gradient vector w.r.t. $\boldsymbol{f}$ evaluated at $\boldsymbol{x}_i$ is

$$\nabla\mathcal{P}_{\mathcal{T}_m,\boldsymbol{f}}(\boldsymbol{x}_i) = 2(\boldsymbol{f}(\boldsymbol{x}_i) - \boldsymbol{z}_i).$$

The gradient vector w.r.t. $\boldsymbol{h}$ evaluated at $\boldsymbol{x}_i$ is

$$\nabla\mathcal{P}_{\mathcal{T}_m,\boldsymbol{h}}(\boldsymbol{x}_i) = 2\left[\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{h}}\right]^T_{\boldsymbol{x}_i} (\boldsymbol{f}(\boldsymbol{x}_i) - \boldsymbol{z}_i), \qquad (8)$$

evaluation of $\left[\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{h}}\right]^T_{\boldsymbol{x}_i}$ is the same as in (6).

**Cross-entropy loss.** The cross-entropy loss function is also widely-used for probabilistic output in classification,

$$\mathcal{P}_{\mathcal{T}_m}(\boldsymbol{f}) = -\sum_{i=1}^{m}\sum_{\ell=1}^{L} z_i^{\ell} \log f^{\ell}(\boldsymbol{x}_i), \qquad (9)$$

whose gradient vector field w.r.t. $\boldsymbol{h}$ evaluated at $\boldsymbol{x}_i$ is

$$\nabla\mathcal{P}_{\mathcal{T}_m,\boldsymbol{h}}(\boldsymbol{x}_i) = \boldsymbol{f}(\boldsymbol{x}_i) - \boldsymbol{z}_i. \qquad (10)$$

### 3.2. The geometric penalty $\mathcal{P}_G$

As discussed in §2, we wish to penalize graphs for excessive curvature and we use the following function, which measures the volume of the $\mathrm{gr}(\boldsymbol{f})$:

$$\mathcal{P}_G(\boldsymbol{f}) = \int_{\mathrm{gr}(\boldsymbol{f})} \mathrm{dvol} = \int_{\mathrm{gr}(\boldsymbol{f})} \sqrt{\det(g)}\,dx^1 \ldots dx^N, \qquad (11)$$

where $g = (g_{ij})$ with $g_{ij} = \delta_{ij} + f_i^a f_j^a$, is the Riemmanian metric on $\mathrm{gr}(\boldsymbol{f})$ induced from the standard dot product on $\mathbb{R}^{N+L}$. We use the summation convention on repeated indices. Note that this regularization term is clearly very different from the standard Sobolev norm of any order.

It is standard that $\nabla\mathcal{P}_G = -\mathrm{Tr}\ \mathrm{II} \in \mathbb{R}^{N+L}$ on the space of all embeddings of $\mathcal{X}$ in $\mathbb{R}^{N+L}$, where $\mathrm{Tr}\ \mathrm{II}$ is the trace of second fundamental form of $\mathrm{gr}(\boldsymbol{f})$. If we restrict to the submanifold of graphs of $\boldsymbol{f} \in \mathcal{M}'$, it is easy to calculate that the gradient of geometric penalty (11) is

$$\nabla\mathcal{P}_{G,\boldsymbol{f}} = V_{G,\boldsymbol{f}} = -\mathrm{Tr}\ \mathrm{II}^L, \qquad (12)$$

where $\mathrm{Tr}\ \mathrm{II}^L$ denotes the last $L$ components of $\mathrm{Tr}\ \mathrm{II}$. Then the geometric gradient w.r.t. $\boldsymbol{h}$ is

$$\nabla\mathcal{P}_{G,\boldsymbol{h}} = V_{G,\boldsymbol{h}} = -\left[\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{h}}\right]^T \mathrm{Tr}\ \mathrm{II}^L. \qquad (13)$$

Evaluation of $\left[\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{h}}\right]$ and $\mathrm{Tr}\ \mathrm{II}^L$ at $\boldsymbol{x}_i$ leads to $\nabla\mathcal{P}_{G,\boldsymbol{h}}(\boldsymbol{x}_i)$.

The formulation given above is general in that it encompasses both the binary and the multiclass cases. For both cases, evaluation of $\left[\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{h}}\right]$ at the training points is the same as that in (6), and evaluation of $\mathrm{Tr}\ \mathrm{II}^L$ at any point $\boldsymbol{x}$ can be performed explicitly by the following theorem.

**Theorem 1.** *For $\boldsymbol{f} : \mathbb{R}^N \to \Delta^{L-1}$, $\mathrm{Tr}\ \mathrm{II}^L$ for $\mathrm{gr}(\boldsymbol{f})$ is given by*

$$\begin{aligned}\mathrm{Tr}\ \mathrm{II}^L \quad = \quad & (g^{-1})^{ij}\Big(f_{ji}^1 - (g^{-1})^{rs} f_{rs}^a f_i^a f_j^1, \ldots, \\ & f_{ji}^L - (g^{-1})^{rs} f_{rs}^a f_i^a f_j^L\Big), \qquad (14)\end{aligned}$$

*where $f_i^a, f_{ij}^a$ denote partial derivatives of $f^a$.*

The proof is in the supplemental materials. Note that for our RBF representation (3), the partial derivatives $f_i^a, f_{ij}^a$ can be easily obtained in closed form.

**Simplex constraint.** The class probability estimator $\boldsymbol{f} : X \to \Delta^{L-1}$ always takes values in $\Delta^{L-1} \subset \mathbb{R}^L$, as guaranteed by the "softmax" representation of $\boldsymbol{f}$ (3). However, the geometric gradient vector field $\nabla P_{G,f} \in T_{\boldsymbol{f}}\mathcal{M}'$, the set of vector fields along $\boldsymbol{f}$ with values in $\mathbb{R}^L$, may not lie in $T_{\boldsymbol{f}}\mathcal{M}$, and in particular may not take values in $T\Delta^{L-1}$. There are two ways to enforce this constraint for the geometric gradient vector field. First, since our initial function $\boldsymbol{f}_0$ takes values at the center of $\Delta^{L-1}$, we can orthogonally project the geometric gradient vector $V_{G,\boldsymbol{f}}$ to $V'_{G,\boldsymbol{f}}$in the tangent space $Z = \{(y^1, \dots, y^L) \in \mathbb{R}^L : \sum_{\ell=1}^{L} y^\ell = 0\}$ of the simplex. More simply, we can also select $L-1$ of the $L$ components of $\boldsymbol{f}(\boldsymbol{x})$, call the new function $\boldsymbol{f}' : \mathcal{X} \to R^{L-1}$, and compute the $(L-1)$-dimensional gradient vector $V_{G,\boldsymbol{f}'}$ following (12) and (14). The omitted component of the desired $L$-gradient vector is determined by $-\sum_{\ell=1}^{L-1} V^{\ell}_{G,\boldsymbol{f}'}$, by the definition of tangent space $Z$. Our implementation follows this second approach, where we choose the $(L-1)$ components of $\boldsymbol{f}$ by omitting the component corresponding to the class with least number of training samples.

### 3.3. Algorithm summary

Algorithm 1 gives a summary of the classifier learning procedure. Input to the algorithm is the training set $\mathcal{T}_m$, RBF kernel width $c$, trade-off parameter $\lambda$, and step-size parameter $\tau$. For initialization, our algorithm first initializes the function values of $\boldsymbol{h}$ and $\boldsymbol{f}$ for every training point, and then constructs matrix $G$ and solves for $A$ by (4). In the subsequent steps, at each iteration, our algorithm first evaluates the gradient vector field $\nabla\mathcal{P}_{\boldsymbol{h}}$ at every training point, then updates coefficient matrix $A$ by (5). For the overall penalty function $\mathcal{P} = \mathcal{P}_{\mathcal{T}_m} + \lambda\mathcal{P}_G$, we compute the total gradient vector field $\nabla\mathcal{P}_{\boldsymbol{h}}$ evaluated at $\boldsymbol{x}_i$ as follows.

For quadratic loss, it is:

$$\nabla\mathcal{P}_{\boldsymbol{h}}(\boldsymbol{x}_i) = \left[\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{h}}\right]^T_{\boldsymbol{x}_i} \left(2(\boldsymbol{f}(\boldsymbol{x}_i) - \boldsymbol{z}_i) - \lambda \mathrm{Tr}\ \mathrm{II}^L_{\boldsymbol{x}_i}\right). \quad (15)$$

For cross-entropy loss, it is:

$$\nabla\mathcal{P}_{\boldsymbol{h}}(\boldsymbol{x}_i) = \boldsymbol{f}(\boldsymbol{x}_i) - \boldsymbol{z}_i - \lambda \left[\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{h}}\right]^T_{\boldsymbol{x}_i} \mathrm{Tr}\ \mathrm{II}^L_{\boldsymbol{x}_i}. \quad (16)$$

Our algorithm iterates until it converges or reaches the maximum iteration number.

The same algorithm applies to both the quadratic loss and the cross-entropy loss. To evaluate the total gradient vectors $\nabla\mathcal{P}_{\boldsymbol{h}}(\boldsymbol{x}_i)$ in each iteration, for the quadratic loss, we use (8) and (13) to compute the total gradient vector (15); for the cross-entropy loss, we use (10) and (13) instead to compute (16). The remaining steps of the procedure are exactly the same for both loss functions.

The final predictor learned by our algorithm is given by

$$F(\boldsymbol{x}) = \mathrm{argmax}\{f^\ell(\boldsymbol{x}), \ell \in \{1, 2, \cdots, L\}\}. \quad (17)$$

**Algorithm 1** Geometric regularized classification

```
Input: training data $\mathcal{T}_m = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^m$, RBF kernel width $c$, trade-off parameter $\lambda$, step-size $\tau$
Initialize: $\boldsymbol{h}(\boldsymbol{x}_i) = (1, \dots, 1), \boldsymbol{f}(\boldsymbol{x}_i) = (\frac{1}{L}, \dots, \frac{1}{L})$, $\forall i \in \{1, \cdots, m\}$, construct matrix $G$ and solve $A$ by (4)
for $t = 1$ to $T$ do
  – Evaluate the total gradient vector $\nabla\mathcal{P}_{\boldsymbol{h}}(\boldsymbol{x}_i)$ at every training point according to (15) or (16).
  – Update the $A$ by (5).
end for
Output: class probability estimator $\boldsymbol{f}$ given by (3).
```

## 4. Experiments

To evaluate the effectiveness of the proposed regularization approach, we compare our RBF-based implementation with two groups of related classification methods. The first group of methods are standard RBF-based methods that use different regularizers than ours. The second group of methods are previous geometric regularization methods.

In particular, the first group includes the Radial Basis Function Network (RBN), SVM with RBF kernel (SVM) and the Import Vector Machine (IVM) (Zhu & Hastie, 2005) (a greedy search variant of the standard RBF kernel logistic regression classifier). Note that both SVM and IVM use RKHS regularizers and the IVM also uses the similar cross-entropy loss as Ours-CE.

The second group includes the Level Learning Set classifier (Cai & Sowmya, 2007) (LLS), the Geometric Level Set classifier (Varshney & Willsky, 2010) (GLS) and the Euler's Elastica classifier (Lin et al., 2012; 2015) (EE). Note that both GLS and EE use RBF representations and EE also uses the same quadratic distance loss as Ours-Q.

We test both the quadratic loss version (Ours-Q) and the cross-entropy loss version (Ours-CE) of our implementation.

### 4.1. UCI datasets

We tested our classification method on four binary classification datasets and four multiclass classification datasets. Given that Varshney & Willsky (2010) has covered several methods on our comparing list and their implementation is publicly available, we choose to use the same datasets as (Varshney & Willsky, 2010) and carefully follow the exact experimental setup. Tenfold cross-validation error is reported. For each of the ten folds, the kernel-width constant $c$ and tradeoff parameter $\lambda$ are found using fivefold cross-validation on the training folds. All dimensions of input sample points are normalized to a fixed range $[0, 1]$ throughout the experiments. We select $c$ from the set of val-

*Table 1.* Tenfold cross-validation error rate (percent) on four binary and four multiclass classification datasets from the UCI machine learning repository. $(L, N)$ denote the number of classes and input feature dimensions respectively. We compare both the quadratic loss version (Ours-Q) and the cross-entropy loss version (Ours-CE) of our method with 6 RBF-based classification methods and (or) geometric regularization methods: SVM with RBF kernel (SVM), Radial basis function network (RBN), Level learning set classifier (Cai & Sowmya, 2007) (LLS), Geometric level set classifier (Varshney & Willsky, 2010) (GLS), Import Vector Machine (Zhu & Hastie, 2005) (IVM), Euler's Elastica classifier (Lin et al., 2012; 2015) (EE). The mean error rate averaged over all eight datasets is shown in the bottom row. Top performance for each dataset is shown in bold.

| Dataset($L, N$) | RBN | SVM | IVM | LLS | GLS | EE | Ours-Q | Ours-CE |
|---|---|---|---|---|---|---|---|---|
| Pima(2,8) | 24.60 | 24.12 | 24.11 | 29.94 | 25.94 | **23.33** | 23.98 | 24.51 |
| WDBC(2,30) | 5.79 | 2.81 | 3.16 | 6.50 | 4.40 | **2.63** | **2.63** | **2.63** |
| Liver(2,6) | 35.65 | 28.66 | 29.25 | 37.39 | 37.61 | 26.33 | **25.74** | 26.31 |
| Ionos.(2,34) | 7.38 | **3.99** | 21.73 | 13.11 | 13.67 | 6.55 | 6.83 | 6.26 |
| Wine(3,13) | 1.70 | 1.11 | 1.67 | 5.03 | 3.92 | 0.56 | **0.00** | **0.00** |
| Iris(3,4) | 4.67 | **2.67** | 4.00 | 3.33 | 6.00 | 4.00 | 3.33 | 3.33 |
| Glass(6,9) | 34.50 | 31.77 | **29.44** | 38.77 | 36.95 | 32.28 | 29.87 | **29.44** |
| Segm.(7,19) | 13.07 | 3.81 | 3.64 | 14.40 | 4.03 | 8.80 | **2.47** | 2.73 |
| all-avg | 15.92 | 12.37 | 14.63 | 18.56 | 16.57 | 13.06 | **11.86** | 11.90 |

ues $\{1/2^5, 1/2^4, 1/2^3, 1/2^2, 1/2, 1, 2, 4, 8\}$ and $\lambda$ from the set of values $\{1/1.5^4, 1/1.5^3, 1/1.5^2, 1/1.5, 1, 1.5\}$ that minimizes the fivefold cross-validation error. The step-size $\tau = 0.1$ and iteration number $T = 5$ are fixed over all datasets. We used the same settings for both loss functions.

Table 1 reports the results of this experiment. The top performer for each dataset is marked in bold, and the averaged performance of each method over all testing datasets is summarized in the bottom row. The numbers for RBN, LLS and GLS are copied from Table 1 of (Varshney & Willsky, 2010). Results for SVM and IVM are obtained by running publicly available implementations for SVM (Chang & Lin, 2011) and IVM (Roscher et al., 2012). Results for EE are obtained by running an implementation provided by the authors of (Lin et al., 2012). When running these implementations, we followed the same experimental setup as described above and exhaustively searched for the optimal range for the kernel bandwidth and the trade-off parameter via cross-validation.

As shown in the last row of Table 1, two versions of our approach are overall the top two performers among all reported methods. In particular, Ours-Q attains top performance on four out of the eight benchmarks, Ours-CE attains top performance on three out of the eight benchmarks. The performance of the two versions of our method are very close, which shows the robustness of our geometric regularization approach cross different loss functions for classification. Note that three pairs of comparisons, IVM vs Ours-CE, GLS vs Ours-Q/Ours-CE, and EE vs Ours-Q are of particular interest. We are going to discuss them in detail respectively.

The IVM method of kernel logistic regression uses the same RBF-based implementation and very similar cross-entropy loss as our cross-entropy version Ours-CE, and both methods handle the multiclass case inherently. The main difference lies in regularization, i.e., the standard RKHS norm regularizer vs our geometric regularizer. Ours-CE outperforms IVM on six of the eight benchmars in Table 1, and achieves equal performance on one of the remaining two, and is only slightly behind on "PIMA". The overall superior performance of Ours-CE demonstrates the advantage of the proposed geometric regularization over the standard RKHS norm regularization.

The GLS method uses the same RBF-based implementation as ours and also exploits volume geometry for regularization. As described in §1, however, there are key differences between the two regularization techniques. GLS measures the volume of the decision boundary supported in $\mathcal{X}$, while our approach measures the volume of a submanifold supported in $\mathcal{X} \times \Delta^{L-1}$ that corresponds to the class probability estimator. Our regularization technique handles the binary and multiclass cases in a unified framework, while the decision boundary based techniques, such as GLS (and EE), were inherently designed for the binary case and rely on a binary coding strategy to train $\log_2 L$ decision boundaries to generalize to the multiclass case. In our experiments, both Ours-Q and Ours-CE outperform GLS on all the benchmarks we have tested. This demonstrates the effectiveness of exploiting the geometry of the class probability in addressing the "small local oscillation" for classification.

The EE method of Euler's Elastica model uses the same RBF-based implementation and the same quadratic loss as our quadratic loss version Ours-Q. The main difference, again, lies in regularization, i.e., a combination of 1-Sobolev norm and curvature penalty on the decision boundary vs our volume penalty on the submanifold corresponding to the class probability estimator. Since EE adopts a

*Table 2.* Notations

| |
|---|
| $h_{\boldsymbol{f}}(\boldsymbol{x}) = \underset{\ell \in \mathcal{Y}}{\operatorname{argmax}}\, f^{\ell}(\boldsymbol{x}),$ : plug-in classifier of $\boldsymbol{f} : \mathcal{X} \to \Delta^{L-1}$ |
| $\Delta^{L-1}$ : the standard$(L-1)$-simplex in $\mathbb{R}^L$; |
| $\boldsymbol{\eta}(\boldsymbol{x}) = (\eta^1(\boldsymbol{x}), \ldots, \eta^L(\boldsymbol{x}))$ : class probability: $\eta^{\ell}(\boldsymbol{x}) = P(y = \ell \mid \boldsymbol{x})$ |
| $\mathcal{M} : \{\boldsymbol{f} : \mathcal{X} \to \Delta^{L-1} : \boldsymbol{f} \in C^{\infty}\}$ |
| $\mathcal{M}' : \{\boldsymbol{f} : \mathcal{X} \to \mathbb{R}^L : \boldsymbol{f} \in C^{\infty}\}$ |
| $T_f\mathcal{M}$ : the tangent space to $\mathcal{M}$ at some $f \in \mathcal{M}$ |
| The graph of $f \in \mathcal{M}$ (or $\mathcal{M}'$) : $\mathrm{gr}(\boldsymbol{f}) = \{(\boldsymbol{x}, f(\boldsymbol{x})) : \boldsymbol{x} \in \mathcal{X}\}$ |
| $g_{ij} = \frac{\partial f}{\partial x^i}\frac{\partial f}{\partial x^j}$ : The Riemannian metric on $\mathrm{gr}(\boldsymbol{f})$induced from the standard dot product on $\mathbb{R}^{N+L}$ |
| $(g^{ij}) = g^{-1},$ with $g = (g_{ij})_{i,j=1,\ldots,N}$ |
| $\mathrm{dvol} = \sqrt{\det(g)}dx^1 \ldots dx^N$, the volume element on $\mathrm{gr}(\boldsymbol{f})$ |
| $\{e_i\}_{i=1}^N$ : a smoothly varying orthonormal basis of the tangent spaces $T_{(\boldsymbol{x},\boldsymbol{f}(\boldsymbol{x}))}\mathrm{gr}(\boldsymbol{f})$of the graph of $\boldsymbol{f}$ |
| $\mathrm{Tr\ II}$ : the trace of the second fundamental form of $\mathrm{gr}(\boldsymbol{f})$, $\mathrm{Tr\ II} \in \mathbb{R}^{N+L}$<br>$\mathrm{Tr\ II} = \left(\sum_{i=1}^N D_{e_i} e_i\right)^{\perp}$ : with $\perp$ the orthogonal projection to the subspace perpendicular to the<br>tangent space of $\mathrm{gr}(\boldsymbol{f})$ and $D_y w$ the directional derivative of $w$ in $y$ direction |
| $\mathrm{Tr\ II}^L$ : the projection of $\mathrm{Tr\ II}$ onto the last $L$ coordinates of $\mathbb{R}^{N+L}$ |
| $\nabla\mathcal{P}$ : the gradient vector field of a function $\mathcal{P} : \mathcal{M} \to \mathbb{R}$ on a possibly infinite dimensional manifold $\mathcal{M}$ |

combination of sophisticated geometric measures on the decision boundary, which fit specifically the binary case, it achieves top performance on binary datasets. However, as explained in §1, the geometry of the class probability for general classification, which is captured by our approach, cannot be captured by decision boundary based techniques. That is the reason why Ours-Q, a general scheme for both the binary and multiclass case, outperforms EE on all four multiclass datasets, while it still achieves top performance on binary datasets. This again demonstrates our geometric perspective and regularization approach that exploits the geometry of the class probability.

### 4.2. Real-world datasets

To test the scalability of our method to high dimensional and large-scale problems, we also conduct experiments on two real-world datasets, i.e., the Flickr Material Database (FMD) (Sharan et al., 2009) for image classification and the MNIST (LeCun et al., 1998) Database of handwritten digits.

**FMD (4096 dimensional).** The FMD dataset contains 10 categories of images with 100 images per category. We extract image features using the SIFT descriptor augmented by its feature coordinates, implemented by the VLFeat library (Vedaldi & Fulkerson, 2008). With this descriptor, Bag-of-visual-words uses 4096 vector-quantized visual words, histogram square rooting, followed by L2 normalization. We compare our method with an SVM classifier with RBF kernels, using exactly the same 4096 dimensional feature. Our method achieves a correct classification rate of $48.8\%$ while the SVM baseline achieves $46.4\%$. Note that while recent works (Qi et al., 2015; Cimpoi et al., 2015) report better performance on this dataset, the effort focuses on better feature design, not on the classifier itself. The features used in those works, such as local texture descriptors and CNN features, are more sophisticated.

**MNIST (60,000 samples).** The MNIST dataset contains 10 classes ($0 \sim 9$) of handwritten digits with $60,000$ samples for training and $10,000$ samples for testing. Each sample is a $28 \times 28$ grey scale image. We use $1000$ RBFs to represent our function $\boldsymbol{f}$, with RBF centers obtained by applying K-means clustering on the training set. Note that our learning and regularization approach still handles all the $60,000$ training samples as described by Algorithm 1. Our method achieves an error rate of $2.74\%$. While there are many results reported on this dataset, we feel that the most comparable method with our representation is the Radial Basis Function Network with $1000$ RBF units (LeCun et al., 1998), which achieves an error rate of $3.6\%$. This experiment shows the potential that our geometric regularization approach scales to larger datasets.

## 5. Conclusion

We have introduced a new geometric perspective on regularization for classification that exploits the geometry of a robust class probability estimator. Under this perspective, we propose a general regularization approach that applies to both binary and multiclass cases in a unified way. In experiments with an example formulation based on RBFs, our implementation achieves favorable results comparing with widely used RBF-based classification methods and previous geometric regularization methods. We plan to extend this work to machine learning problems beyond classification, and to state-of-the-art deep learning models.

## Acknowledgments

We would like to thank Shivani Agarwal for her very helpful feedback on an earlier version of this paper. This work was supported in part through US National Science Foundation grants 1029430 and 0910908.

## References

Audibert, Jean-Yves and Tsybakov, Alexandre. Fast learning rates for plug-in classifiers. *Annals of Statistics*, 35 (2):608–633, 2007.

Bartlett, Peter L, Jordan, Michael I, and McAuliffe, Jon D. Convexity, classification, and risk bounds. *Journal of the American Statistical Association*, 101(473):138–156, 2006.

Belkin, Mikhail and Niyogi, Partha. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, 15(6):1373–1396, 2003.

Belkin, Mikhail, Niyogi, Partha, and Sindhwani, Vikas. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of Machine Learning Research*, 7:2399–2434, 2006.

Cai, Xiongcai and Sowmya, Arcot. Level learning set: A novel classifier based on active contour models. In *Proc. European Conf. on Machine Learning (ECML)*, pp. 79–90. 2007.

Chang, Chih-Chung and Lin, Chih-Jen. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

Chen, Yun-Gang, Giga, Yoshikazu, and Goto, Shun'ichi. Uniqueness and existence of viscosity solutions of generalized mean curvature flow equations. In *Fundamental Contributions to the Continuum Theory of Evolving Phase Interfaces in Solids*, pp. 375–412. Springer, Berlin, 1999.

Donoho, David and Grimes, Carrie. Hessian eigenmaps: Locally linear embedding techniques for high-dimensional data. *Proceedings of the National Academy of Sciences*, 100(10):5591–5596, 2003.

Goodfellow, Ian J, Shlens, Jonathon, and Szegedy, Christian. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

LeCun, Yann, Bottou, Léon, Bengio, Yoshua, and Haffner, Patrick. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Lin, Tong and Zha, Hongbin. Riemannian manifold learning. *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, 30(5):796–809, 2008.

Lin, Tong, Xue, Hanlin, Wang, Ling, and Zha, Hongbin. Total variation and Euler's elastica for supervised learning. *Proc. International Conf. on Machine Learning (ICML)*, 2012.

Lin, Tong, Xue, Hanlin, Wang, Ling, Huang, Bo, and Zha, Hongbin. Supervised learning via euler's elastica models. *Journal of Machine Learning Research*, 16:3637–3686, 2015.

Mantegazza, Carlo. *Lecture Notes on Mean Curvature Flow*, volume 290 of *Progress in Mathematics*. Birkhäuser/Springer Basel AG, Basel, 2011.

Mumford, David and Shah, Jayant. Optimal approximations by piecewise smooth functions and associated variational problems. *Communications on Pure and Applied Mathematics*, 42(5):577–685, 1989.

Osher, Stanley and Sethian, James. Fronts propagating with curvature-dependent speed: Algorithms based on Hamilton-Jacobi formulations. *Journal of Computational Physics*, 79(1):12–49, 1988.

Roscher, Ribana, Förstner, Wolfgang, and Waske, Björn. I 2 vm: incremental import vector machines. *Image and Vision Computing*, 30(4):263–278, 2012.

Roweis, Sam and Saul, Lawrence. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290 (5500):2323–2326, 2000.

Schölkopf, Bernhard and Smola, Alexander. *Learning with kernels: Support vector machines, regularization, optimization, and beyond*. MIT press, 2002.

Sethian, James Albert. *Level set methods and fast marching methods: evolving interfaces in computational geometry, fluid mechanics, computer vision, and materials science*, volume 3. Cambridge University Press, 1999.

Sharan, Lavanya, Rosenholtz, Ruth, and Adelson, Edward. Material perception: What can you see in a brief glance? *Journal of Vision*, 9(8):784–784, 2009.

Steinwart, Ingo. Consistency of support vector machines and other regularized kernel classifiers. *IEEE Trans. Information Theory*, 51(1):128–142, 2005.

Tenenbaum, Joshua, De Silva, Vin, and Langford, John. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.

Vapnik, Vladimir Naumovich. *Statistical learning theory*, volume 1. Wiley New York, 1998.

Varshney, Kush and Willsky, Alan. Classification using geometric level sets. *Journal of Machine Learning Research*, 11:491–516, 2010.

Vedaldi, A. and Fulkerson, B. VLFeat: An open and portable library of computer vision algorithms. http://www.vlfeat.org/, 2008.

Zhang, Zhenyue and Zha, Hongyuan. Principal manifolds and nonlinear dimensionality reduction via tangent space alignment. *SIAM Journal on Scientific Computing*, 26 (1):313–338, 2005.

Zhou, Dengyong and Schölkopf, Bernhard. Regularization on discrete spaces. In *Pattern Recognition*, pp. 361–368. Springer, 2005.

Zhu, Ji and Hastie, Trevor. Kernel logistic regression and the import vector machine. *Journal of Computational and Graphical Statistics*, 2005.