1 Graph Kernels by Spectral Transforms

Xiaojin Zhu
Jaz Kandola
John Lafferty
Zoubin Ghahramani

Many graph-based semi-supervised learning methods can be viewed as imposing smoothness conditions on the target function with respect to a graph representing the data points to be labeled. The smoothness properties of the functions are encoded in terms of Mercer kernels over the graph. The central quantity in such regularization is the spectral decomposition of the graph Laplacian, a matrix derived from the graph's edge weights. The eigenvectors with small eigenvalues are smooth, and ideally represent large cluster structures within the data. The eigenvectors having large eigenvalues are rugged, and considered noise.

Different weightings of the eigenvectors of the graph Laplacian lead to different measures of smoothness. Such weightings can be viewed as *spectral transforms*, that is, as transformations of the standard eigenspectrum that lead to different regularizers over the graph. Familiar kernels, such as the diffusion kernel resulting by solving a discrete heat equation on the graph, can be seen as simple parametric spectral transforms.

The question naturally arises whether one can obtain effective spectral transforms automatically. In this paper we develop an approach to searching over a nonparametric family of spectral transforms by using convex optimization to maximize kernel alignment to the labeled data. Order constraints are imposed to encode a preference for smoothness with respect to the graph structure. This results in a flexible family of kernels that is more data-driven than the standard parametric spectral transforms. Our approach relies on a quadratically constrained quadratic program (QCQP), and is computationally practical for large datasets.

1.1 The Graph Laplacian

We are given a labeled dataset of input-output pairs $(X_l, Y_l) = \{(x_1, y_1), \ldots, (x_l, y_l)\}$ and an unlabeled dataset $X_u = \{x_{l+1}, \ldots, x_n\}$. We form a graph $\mathsf{g} = (V, E)$ where the vertices V are $x_1, \ldots, x_n$, and the edges E are represented by an $n \times n$ matrix W. Entry W_{ij} is the edge weight between nodes i, j, with $W_{ij} = 0$ if i, j are not connected. The entries of W have to be non-negative and symmetric, but it is not necessary for W itself to be positive semi-definite. Let D be the diagonal degree matrix with $D_{ii} = \sum_j W_{ij}$ being the total weight on edges connected to node i. The *combinatorial graph Laplacian* is defined as $L = D - W$, which is also called the unnormalized Laplacian. The *normalized graph Laplacian* is $\mathfrak{L} = D^{-1/2} L D^{-1/2} = \mathbf{I} - D^{-1/2} W D^{-1/2}$.

graph Laplacian

In graph-based semi-supervised learning the Laplacian L (or $\mathfrak{L}$) is a central object. Let us denote the eigenvalues of L by $\lambda_1 \leq \ldots \leq \lambda_n$, and the complete orthonormal set of eigenvectors by $\phi_1 \ldots \phi_n$. Therefore the spectral decomposition of the Laplacian is given as $L = \sum_{i=1}^{n} \lambda_i \phi_i \phi_i^\top$. We refer readers to Chung [1997] for a discussion of the mathematical aspects of this decomposition, but briefly summarize two relevant properties:

spectral decomposition

Theorem 1.1 *The Laplacian L is positive semi-definite, i.e., $\lambda_i \geq 0$.*

Indeed, it is not hard to show that for any function $f : [n] \to \mathbb{R}$,

$$f^\top L f = \frac{1}{2} \sum_{i,j} W_{ij} \left(f(i) - f(j)\right)^2 \geq 0 \tag{1.1}$$

where the inequality holds because W has non-negative entries.

Equation (1.1) measures the *smoothness* of f on the graph[1]. Roughly speaking, f is smooth if $f(i) \approx f(j)$ for those pairs with large W_{ij}. This is sometimes informally expressed by saying that f varies slowly over the graph, or that f follows the data manifold. In particular, the smoothness of an eigenvector is

smoothness of f

$$\phi_i^\top L \phi_i = \lambda_i \tag{1.2}$$

Thus, eigenvectors with smaller eigenvalues are smoother. Since $\{\phi_i\}$ forms a basis on $\mathbb{R}^n$, we can always write any function f as

$$f = \sum_{i=1}^{n} \alpha_i \phi_i \,, \alpha_i \in \mathbb{R} \tag{1.3}$$

and equation (1.1) which measures the smoothness of f can be re-expressed as

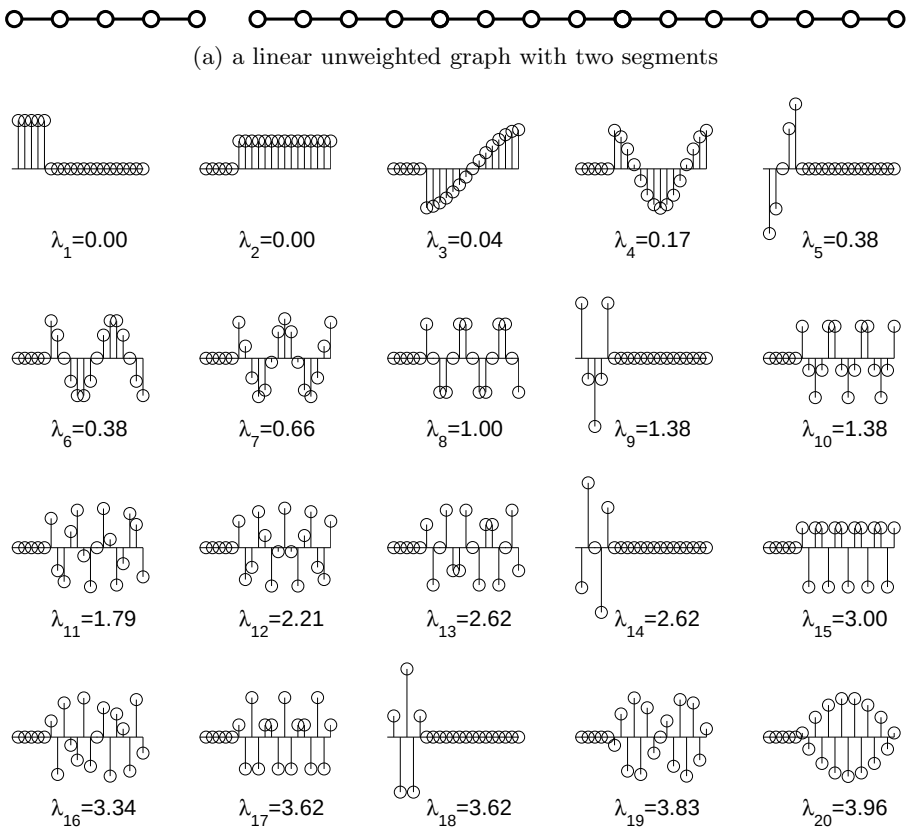
$$f^\top L f = \sum_{i=1}^{n} \alpha_i^2 \lambda_i \tag{1.4}$$

1. Note that a smaller value means smoother f.

For semi-supervised learning a smooth function f is part of what we seek, because this is the prior knowledge encoded by the graph—but we also require that the function f fits the labels Y_l on the inputs X_l.

Theorem 1.2 *The graph* g *has k connected components if and only if $\lambda_i = 0$ for $i = 1, 2, \ldots, k$.*

The corresponding eigenvectors $\phi_1, \ldots, \phi_k$ of L are constant on the nodes within the corresponding connected component, and zero elsewhere. Note λ_1 is always 0 for any graph (Chung [1997]). We will make use of this property later.



(a) a linear unweighted graph with two segments

(b) the eigenvectors and eigenvalues of the Laplacian L

Figure 1.1 A simple graph and its Laplacian spectral decomposition. Note the eigenvectors become rougher with larger eigenvalues.

As an example, Figure 1.1(a) shows an unweighted graph ($W_{ij} = 1$ if there is an edge) consisting of two linear segments. The spectral decomposition of its Laplacian L is shown in (b). Note that the eigenvectors do indeed look smoother for small λ_i, and that the graph has two connected components.

1.2 Kernels by Spectral Transforms

Kernel methods are increasingly being used for classification because of their conceptual simplicity, theoretical properties, and good performance on many tasks. It is attractive to create kernels specifically for semi-supervised learning. We restrict ourselves to transduction, i.e., the unlabeled data X_u is also the test data. As a result we only need to consider kernel matrices $K \in \mathbb{R}^{n \times n}$ on nodes $1, \ldots, n$ in the graph.

In particular, we want K to respect the smoothness preferences encoded in a graph. That is, as a regularizer the kernel should penalize functions that are not smooth over the graph. To establish a link to the graph, we consider K having the form

$$K = \sum_{i=1}^{n} \mu_i \phi_i \phi_i^\top \tag{1.5}$$

where ϕ are the eigenvectors of the graph Laplacian L, and $\mu_i \geq 0$ are the eigenvalues of K. Since K is the non-negative sum of outer products, it is positive semi-definite, i.e., a kernel matrix.

The matrix K defines a Reproducing Kernel Hilbert Space (RKHS) with norm

$$\|f\|_K^2 = \langle f, f \rangle_K = \sum_{i=1}^{n} \frac{\alpha_i^2}{\mu_i} \tag{1.6}$$

for a function $f = \sum_{i=1}^{n} \alpha_i \phi_i$. Note if some $\mu_i = 0$ the corresponding dimension is not present in the RKHS, and we might define $\frac{1}{0} = 0$ here.

In many learning algorithms, regularization is expressed as an increasing function of $\|f\|_K$. From a semi-supervised learning point of view, we want f to be penalized if it is not smooth with respect to the graph. Comparing the smoothness of f in equation (1.4) with equation (1.6), we find this can be achieved by making μ_i small if the Laplacian eigenvalue λ_i is large, and vice versa.

Indeed, Chapelle et al. [2002] and Smola and Kondor [2003] both suggest a general principle for creating a semi-supervised kernel K from the graph Laplacian. Define a *spectral transformation* function $r : \mathbb{R}_+ \to \mathbb{R}_+$ that is non-negative and decreasing. Set the kernel spectrum by $\mu_i = r(\lambda_i)$ to obtain the kernel

spectral transformation

$$K = \sum_{i=1}^{n} r(\lambda_i) \phi_i \phi_i^\top \tag{1.7}$$

Note that r essentially reverses the order of the eigenvalues, so that smooth ϕ_i's have larger eigenvalues in K. Since r is decreasing, a greater penalty is incurred if a function is not smooth.

The transform r is often chosen from a parametric family, resulting in some familiar kernels. For example Chapelle et al. [2002] and Smola and Kondor [2003] list the following transformations on $\mathcal{L}$

- regularized Laplacian: $r(\lambda) = \frac{1}{\lambda+\epsilon}$
- diffusion kernel: $r(\lambda) = \exp\left(-\frac{\sigma^2}{2}\lambda\right)$
- one step random walk: $r(\lambda) = (\alpha - \lambda)$ with $\alpha \geq 2$
- p-step random walk: $r(\lambda) = (\alpha - \lambda)^p$ with $\alpha \geq 2$
- inverse cosine: $r(\lambda) = \cos(\lambda\pi/4)$
- step function: $r(\lambda) = 1$ if $\lambda \leq \lambda_{\text{cut}}$

Each has its own special interpretation. The regularized Laplacian is also known as the Gaussian field kernel (Zhu et al. [2003]). Of course there are many other natural choices for r. Although the general principle of equation (1.7) is appealing, it does not address the question of which parametric family to use. Moreover, the hyperparameters (e.g., σ or ϵ above) in a particular parametric family may not suit the task at hand, resulting in overly constrained kernels.

Is there an optimal spectral transformation? The following sections address this question. The short answer is yes, in a certain sense. We select a spectral transformation that optimizes kernel alignment to the labeled data, while imposing an ordering constraint but otherwise not assuming any parametric form. Kernel alignment is a surrogate for classification accuracy, and, importantly, leads to a convex optimization problem.

1.3 Kernel Alignment

The empirical kernel alignment (Cristianini et al. [2001], Lanckriet et al. [2004]) assesses the fitness of a kernel to training labels. The alignment has a number of convenient properties: it can be efficiently computed before any training of the kernel machine takes place, and based only on training data information. The empirical alignment can also be shown to be sharply concentrated around its expected value, allowing it to be estimated from finite samples. A connection between high alignment and good generalization performance has been established in Cristianini et al. [2001].

Frobenius product

As we will compare matrices, we introduce here the Frobenius product $\langle ., . \rangle_F$ between two square matrices M and N of the same size:

$$\langle M, N \rangle_F = \sum_{ij} m_{ij} n_{ij} = \text{Tr}(MN)$$

The *empirical kernel alignment* compares the $l \times l$ kernel matrix K_{tr} on the labeled training set $x_1, \ldots, x_l$, and a target matrix T derived from the labels $y_1, \ldots, y_l$. One such target matrix is $T_{ij} = 1$ if $y_i = y_j$, and -1 otherwise. Note for binary $\{+1, -1\}$ training labels $Y_l = (y_1 \ldots y_l)^\top$ this is simply the rank one matrix $T = Y_l Y_l^\top$. The empirical kernel alignment is defined as follows.

Definition 1.3 (Empirical Kernel Alignment) *Let K_{tr} be the kernel matrix*

restricted to the training points, and T the target matrix on training data. We define the empirical kernel alignment as:

empirical kernel alignment

$$\hat{A}(K_{tr}, T) = \frac{\langle K_{tr}, T\rangle_F}{\sqrt{\langle K_{tr}, K_{tr}\rangle_F \langle T, T\rangle_F}} \tag{1.8}$$

The empirical alignment is essentially the cosine between the matrices K_{tr} and T. The range of the alignment is $[0, 1]$. The larger its value the closer is the kernel to the target. This quantity is maximized when $K_{tr} \propto T$.

1.4 Optimizing Alignment using QCQP for Semi-Supervised Learning

Having introduced the alignment quantity, now let us consider the problem of semi-supervised kernel construction using a principled non-parametric approach. In short, we will learn the spectral transformation $\{\mu_i \equiv r(\lambda_i)\}$ (1.7) by optimizing the resulting kernel alignment, with certain restrictions. Notice we no longer assume a parametric function $r()$, instead we work with the transformed eigenvalues μ_i's directly.

When the kernel matrix is defined as $K = \sum_{i=1}^{n} \mu_i \phi_i \phi_i^\top$ and the target T given, the kernel alignment between the labeled submatrix K_{tr} and T is a convex function in μ_i's. Nonetheless in general we have to make sure K is a valid kernel matrix, i.e. it is positive semi-definite. This is a Semi-Definite Program (SDP), which has high computational complexity (Boyd and Vandenberge [2004]). We thus restrict $\mu_i \geq 0, \forall i$. This guarantees K to be positive semi-definite, and reduces the optimization problem into a *quadratically constrained quadratic program* (QCQP), which is computationally more efficient. In a QCQP both the objective function and the constraints are quadratic as illustrated below,

quadratically constrained quadratic programs

$$\text{minimize} \qquad \frac{1}{2} x^\top P_0 x + q_0^\top x + r_0 \tag{1.9}$$

$$\text{subject to} \qquad \frac{1}{2} x^\top P_i x + q_i^\top x + r_i \leq 0 \quad i = 1 \cdots m \tag{1.10}$$

$$Ax = b \tag{1.11}$$

where $P_i \in \mathcal{S}_+^n,\ i = 0, \ldots, m$, where $\mathcal{S}_+^n$ defines the set of square symmetric positive semi-definite matrices. In a QCQP, we minimize a convex quadratic function over a feasible region that is the intersection of ellipsoids. The number of iterations required to reach the solution is comparable to the number required for linear programs, making the approach feasible for large datasets.

Previous work using kernel alignment did not take into account that the "building blocks" $K_i = \phi_i \phi_i^\top$ were derived from the graph Laplacian with the goal of semi-supervised learning. As such, the μ_i's can take arbitrary non-negative values and there is no preference to penalize components that do not vary smoothly over the graph. This shall be rectified by requiring smoother eigenvectors to receive larger coefficients, as shown in the next section.

1.5 Semi-Supervised Kernels with Order Constraints

order constraints

We would like to maintain a decreasing order on the spectral transformation μ_i to reflect the prior knowledge encoded in the graph, that smooth functions are preferred. This motivates the set of *order constraints*

$$\mu_i \geq \mu_{i+1}, \quad i = 1 \cdots n-1 \tag{1.12}$$

And we can specify the desired semi-supervised kernel as follows.

order constrained kernel

Definition 1.4 *(order constrained kernel) An* order constrained *semi-supervised kernel K is the solution to the following convex optimization problem:*

$$\begin{aligned} max_K \quad & \hat{A}(K_{tr}, T) && (1.13)\\ subject\ to \quad & K = \textstyle\sum_{i=1}^{n} \mu_i K_i && (1.14)\\ & \mu_i \geq 0 && (1.15)\\ & \mathrm{Tr}(K) = 1 && (1.16)\\ & \mu_i \geq \mu_{i+1}, \quad i = 1 \cdots n-1 && (1.17) \end{aligned}$$

where T is the training target matrix, $K_i = \phi_i \phi_i^\top$ and ϕ_i's are the eigenvectors of the graph Laplacian.

convex optimization

This formulation is an extension to the original kernel alignment of Lanckriet et al. [2004], with the addition of order constraints, and with special components K_i's from the graph Laplacian. Since $\mu_i \geq 0$ and K_i's are outer products, K will automatically be positive semi-definite and hence a valid kernel matrix. The trace constraint is needed to fix the scale invariance of kernel alignment. It is important to notice the order constraints are convex and as such, Definition 1.4 is a convex optimization problem.

The problem is equivalent to

$$\begin{aligned} \max_K \quad & \langle K_{tr}, T \rangle_F && (1.18)\\ \text{subject to} \quad & \langle K_{tr}, K_{tr} \rangle_F \leq 1 && (1.19)\\ & K = \textstyle\sum_{i=1}^{n} \mu_i K_i && (1.20)\\ & \mu_i \geq 0 && (1.21)\\ & \mu_i \geq \mu_{i+1}, \quad i = 1 \cdots n-1, && (1.22) \end{aligned}$$

where the trace constraint is replaced by (1.19) (up to a constant factor). Let $\mathrm{vec}(A)$ be the column vectorization of a matrix A. Defining

$$M = \left[\mathrm{vec}(K_{1,tr}) \cdots \mathrm{vec}(K_{m,tr})\right] \tag{1.23}$$

it is not hard to show that the problem can then be expressed as

$$\max_{\mu} \quad \text{vec}(T)^\top M\mu \tag{1.24}$$
$$\text{subject to} \quad ||M\mu|| \leq 1 \tag{1.25}$$
$$\mu_i \geq 0 \tag{1.26}$$
$$\mu_i \geq \mu_{i+1}, \quad i = 1 \cdots n-1 \tag{1.27}$$

The objective function is linear in μ, and there is a simple cone constraint, making it a quadratically constrained quadratic program (QCQP).

We can further improve the kernel. Consider a graph that has a single connected component, i.e., any node can reach any other node via one or more edges. Such graphs are common in practice. By the basic property of the Laplacian we know $\lambda_1 = 0$, and the corresponding eigenvector ϕ_i is a constant. Therefore $K_1 = \phi_i \phi_i^\top$ is a constant matrix. Such a constant matrix acts as a bias term in the graph kernel, as in equation (1.7). We should not constrain μ_1 as in Definition 1.4, but allow the bias of the kernel to vary freely. This motivates the following definition:

bias term

improved order constrained kernel

Definition 1.5 *(improved order constrained kernel) An* improved order constrained *semi-supervised kernel K is the solution to the same problem in Definition 1.4, but the order constraints* (1.17) *apply only to non-constant eigenvectors:*

$$\mu_i \geq \mu_{i+1}, \quad i = 1 \cdots n-1, \text{ and } \phi_i \text{ not constant} \tag{1.28}$$

It should be pointed out that the improved order constrained kernel is identical to the order constrained kernel when the graph has disjoint components. This is because the first k eigenvectors are piece-wise constant over the components, but not constant over all, when the graph has $k > 1$ connected components. We in fact would like to emphasize these eigenvectors because they might correspond to natural clusters in data. Thus we will still enforce the order constraints on them. The definition in (1.28) is meant to target μ_1 in *connected* graphs only. As discussed above, in this situation μ_1 is the bias term of the kernel. The only 'improvement' in the improved order constrained kernel is that we do not constrain such bias term. As the experiments show later, this improves the quality of the kernels markedly.

In practice we do not need all n eigenvectors of the graph Laplacian, or equivalently all n K_i's. The first $m < n$ eigenvectors with the smallest eigenvalues work well empirically. Also note we could have used the fact that K_i's are from orthogonal eigenvectors ϕ_i to further simplify the expression. However we leave it as is, making it easier to incorporate other kernel components if necessary.

It is illustrative to compare and contrast the order constrained semi-supervised kernels to other related kernels. We call the original kernel alignment solution in Lanckriet et al. [2004] a *maximal-alignment* kernel. It is the solution to Definition 1.4 without the order constraints (1.17). Because it does not have the additional constraints, it maximizes kernel alignment among all spectral transformation. The hyperparameters σ and ϵ of the diffusion kernel and Gaussian field kernel (described in Section 1.2) can be learned by maximizing the alignment score also, although

maximal-alignment kernel

different information usage

the optimization problem is not necessarily convex. These kernels use different information from the original Laplacian eigenvalues λ_i. The maximal-alignment kernels ignore λ_i altogether. The order constrained semi-supervised kernels only use the *order* of λ_i and ignore their actual values. The diffusion and Gaussian field kernels use the actual values. In terms of the degree of freedom in choosing the spectral transformation μ_i's, the maximal-alignment kernels are completely free. The diffusion and Gaussian field kernels are restrictive since they have an implicit parametric form and only one free parameter. The order constrained semi-supervised kernels incorporates desirable features from both approaches.

1.6 Experimental Results

We evaluate kernels on seven datasets. The datasets and the corresponding graphs are summarized in Table 1.1. **baseball-hockey**, **pc-mac** and **religion-atheism** are binary document categorization tasks taken from the 20-newsgroups dataset. The distance measure is the cosine similarity between tf.idf vectors. **one-two**, **odd-even** and **ten digits** are handwritten digits recognition tasks originally from the Cedar Buffalo binary digits database. **one-two** is digits "1" vs. "2"; **odd-even** is the artificial task of classifying odd "1, 3, 5, 7, 9" vs. even "0, 2, 4, 6, 8" digits, such that each class has several well defined internal clusters; **ten digits** is 10-way classification. **isolet** is isolated spoken English alphabet recognition from the UCI repository. For these datasets we use Euclidean distance on raw features. We use 10-nearest-neighbor (10NN) unweighted graphs on all datasets except isolet which is 100NN. For all datasets, we use the smallest $m = 200$ eigenvalue and eigenvector pairs from the graph Laplacian. These values are set arbitrarily without optimizing and do not create an unfair advantage to the order constrained kernels. For each dataset we test on five different labeled set sizes. For a given labeled set size, we perform 30 random trials in which a labeled set is randomly sampled from the whole dataset. All classes must be present in the labeled set. The rest is used as unlabeled (test) set in that trial.

dataset	instances	classes	graph
baseball-hockey	1993	2	cosine similarity 10NN unweighted
pc-mac	1943	2	cosine similarity 10NN unweighted
religion-atheism	1427	2	cosine similarity 10NN unweighted
one-two	2200	2	Euclidean 10NN unweighted
odd-even	4000	2	Euclidean 10NN unweighted
ten digits	4000	10	Euclidean 10NN unweighted
isolet	7797	26	Euclidean 100NN unweighted

Table 1.1 Summary of datasets

We compare a total of 8 different types of kernels. Five are semi-supervised kernels: **improved order constrained** kernels, **order constrained** kernels, **Gaussian field** kernels (Section 1.2), **diffusion** kernels (Section 1.2), and **maximal-alignment** kernels (Section 1.5). Three are standard supervised kernels, which do not use unlabeled data in kernel construction: **linear** kernels, **quadratic** kernels, and **radial basis function (RBF)** kernels.

We compute the spectral transformation for improved order constrained kernels, order constrained kernels and maximal-alignment kernels by solving the QCQP using the standard solver SeDuMi/YALMIP, see Sturm [1999] and Löfberg [2004]. The hyperparameters in the Gaussian field kernels and diffusion kernels are learned with the *fminbnd()* function in Matlab to maximize kernel alignment. The bandwidth of the RBF kernels are learned using 5-fold cross validation on labeled set accuracy. Here and below we use cross validation – it is done independent of and after kernel alignment methods, to optimize a quantity not related to the proposed kernels.

We apply the 8 kernels to the same support vector machine (SVM) in order to compute the accuracy on unlabeled data. For each task and kernel combination, we choose the bound on SVM slack variables C with 5-fold cross validation on labeled set accuracy. For multiclass classification we perform one-against-all and pick the class with the largest margin.

Table 1.2 through Table 1.8 list the results. There are two rows for each cell: the upper row is the average *test(unlabeled) set accuracy* with one standard deviation; the lower row is the average *training(labeled) set kernel alignment*, and in parenthesis the average *run time in seconds* for QCQP on a 2.4GHz Linux computer. Each number is averaged over 30 random trials. To assess the statistical significance of the results, we perform paired t-test on test accuracy. We highlight the best accuracy in each row, and those that cannot be distinguished from the best with paired t-test at significance level 0.05.

We find that:

- The five semi-supervised kernels tend to outperform the three standard supervised kernels. It shows that with properly constructed graphs, unlabeled data can help classification.
- The order constrained kernel is often quite good, but the improved order constrained kernel is even better. All the graphs on these datasets happen to be connected. Recall this is when the improved order constrained kernel differs from the order constrained kernel, by not constraining the bias term. Obviously a flexible bias term is important for classification accuracy.
- Figure 1.2 shows the spectral transformation μ_i of the five semi-supervised kernels for different tasks. These are the average of the 30 trials with the largest labeled set size in each task. The x-axis is in increasing order of λ_i (the original eigenvalues of the Laplacian). The mean (thick lines) and ± 1 standard deviation (dotted lines) of only the top 50 μ_i's are plotted for clarity. The μ_i values are scaled vertically for easy comparison among kernels. As expected the maximal-alignment kernels'

	semi-supervised kernels					standard kernels		
Training set size	Improved Order	Order	Gaussian Field	Diffusion	Max-align	RBF $\sigma = 200$	Linear	Quadratic
10	**95.7** ± 8.9	**93.9** ±12.0	63.1 ±15.8	65.8 ±22.8	**93.2** ± 6.8	53.6 ± 5.5	68.1 ± 7.6	68.1 ± 7.6
	0.90 (2)	0.69 (1)	0.35	0.44	0.95 (1)	0.11	0.29	0.23
30	**98.0** ± 0.2	**97.3** ± 2.1	91.8 ± 9.3	59.1 ±17.9	96.6 ± 2.2	69.3 ±11.2	78.5 ± 8.5	77.8 ±10.6
	0.91 (9)	0.67 (9)	0.25	0.39	0.93 (6)	0.03	0.17	0.11
50	**97.9** ± 0.5	**97.8** ± 0.6	96.7 ± 0.6	93.7 ± 6.8	97.0 ± 1.1	77.7 ± 8.3	84.1 ± 7.8	75.6 ±14.2
	0.89 (29)	0.63 (29)	0.22	0.36	0.90 (27)	0.02	0.15	0.09
70	**97.9** ± 0.3	**97.9** ± 0.3	96.8 ± 0.6	**97.5** ± 1.4	97.2 ± 0.8	83.9 ± 7.2	87.5 ± 6.5	76.1 ±14.9
	0.90 (68)	0.64 (64)	0.22	0.37	0.90 (46)	0.01	0.13	0.07
90	**98.0** ± 0.5	**98.0** ± 0.2	97.0 ± 0.4	97.8 ± 0.2	97.6 ± 0.3	88.5 ± 5.1	89.3 ± 4.4	73.3 ±16.8
	0.89 (103)	0.63 (101)	0.21	0.36	0.89 (90)	0.01	0.12	0.06

Table 1.2 Baseball vs. Hockey

	semi-supervised kernels					standard kernels		
Training set size	Improved Order	Order	Gaussian Field	Diffusion	Max-align	RBF $\sigma = 100$	Linear	Quadratic
10	**87.0** ± 5.0	**84.9** ± 7.2	56.4 ± 6.2	57.8 ±11.5	71.1 ± 9.7	51.6 ± 3.4	63.0 ± 5.1	62.3 ± 4.2
	0.71 (1)	0.57 (1)	0.32	0.35	0.90 (1)	0.11	0.30	0.25
30	**90.3** ± 1.3	**89.6** ± 2.3	76.4 ± 6.1	79.6 ±11.2	85.4 ± 3.9	62.6 ± 9.6	71.8 ± 5.5	71.2 ± 5.3
	0.68 (8)	0.49 (8)	0.19	0.23	0.74 (6)	0.03	0.18	0.13
50	**91.3** ± 0.9	90.5 ± 1.7	81.1 ± 4.6	87.5 ± 2.8	88.4 ± 2.1	67.8 ± 9.0	77.6 ± 4.8	75.7 ± 5.4
	0.64 (31)	0.46 (31)	0.16	0.20	0.68 (25)	0.02	0.14	0.10
70	**91.5** ± 0.6	90.8 ± 1.3	84.6 ± 2.1	90.5 ± 1.2	89.6 ± 1.6	74.7 ± 7.4	80.2 ± 4.6	74.3 ± 8.7
	0.63 (70)	0.46 (56)	0.14	0.19	0.66 (59)	0.01	0.12	0.08
90	**91.5** ± 0.6	**91.3** ± 1.3	86.3 ± 2.3	**91.3** ± 1.1	90.3 ± 1.0	79.0 ± 6.4	82.5 ± 4.2	79.1 ± 7.3
	0.63 (108)	0.45 (98)	0.13	0.18	0.65 (84)	0.01	0.11	0.08

Table 1.3 PC vs. MAC

	semi-supervised kernels					standard kernels		
Training set size	Improved Order	Order	Gaussian Field	Diffusion	Max-align	RBF $\sigma = 130$	Linear	Quadratic
10	**72.8** ±11.2	70.9 ±10.9	55.2 ± 5.8	60.9 ±10.7	60.7 ± 7.5	55.8 ± 5.8	60.1 ± 7.0	61.2 ± 4.8
	0.50 (1)	0.42 (1)	0.31	0.31	0.85 (1)	0.13	0.30	0.26
30	**84.2** ± 2.4	83.0 ± 2.9	71.2 ± 6.3	80.3 ± 5.1	74.4 ± 5.4	63.4 ± 6.5	63.7 ± 8.3	70.1 ± 6.3
	0.38 (8)	0.31 (6)	0.20	0.22	0.60 (7)	0.05	0.18	0.15
50	**84.5** ± 2.3	83.5 ± 2.5	80.4 ± 4.1	83.5 ± 2.7	77.4 ± 6.1	69.3 ± 6.5	69.4 ± 7.0	70.7 ± 8.5
	0.31 (28)	0.26 (23)	0.17	0.20	0.48 (27)	0.04	0.15	0.11
70	**85.7** ± 1.4	85.3 ± 1.6	83.0 ± 2.9	**85.4** ± 1.8	82.3 ± 3.0	73.1 ± 5.8	75.7 ± 6.0	71.0 ±10.0
	0.29 (55)	0.25 (42)	0.16	0.19	0.43 (51)	0.03	0.13	0.10
90	**86.6** ± 1.3	**86.4** ± 1.5	84.5 ± 2.1	**86.2** ± 1.6	82.8 ± 2.6	77.7 ± 5.1	74.6 ± 7.6	70.0 ±11.5
	0.27 (86)	0.24 (92)	0.15	0.18	0.40 (85)	0.02	0.12	0.09

Table 1.4 Religion vs. Atheism

	semi-supervised kernels					standard kernels		
Training set size	Improved Order	Order	Gaussian Field	Diffusion	Max-align	RBF $\sigma = 1000$	Linear	Quadratic
10	**96.2** ± 2.7	90.6 ±14.0	58.2 ±17.6	59.4 ±18.9	85.4 ±11.5	78.7 ±14.3	85.1 ± 5.7	85.7 ± 4.8
	0.87 (2)	0.66 (1)	0.43	0.53	0.95 (1)	0.38	0.26	0.30
20	**96.4** ± 2.8	**93.9** ± 8.7	87.0 ±16.0	83.2 ±19.8	94.5 ± 1.6	90.4 ± 4.6	86.0 ± 9.4	90.9 ± 3.7
	0.87 (3)	0.64 (4)	0.38	0.50	0.90 (3)	0.33	0.22	0.25
30	**98.2** ± 2.1	97.2 ± 2.5	**98.1** ± 2.2	**98.1** ± 2.7	96.4 ± 2.1	93.6 ± 3.1	89.6 ± 5.9	92.9 ± 2.8
	0.84 (8)	0.61 (7)	0.35	0.47	0.86 (6)	0.30	0.17	0.24
40	98.3 ± 1.9	96.5 ± 2.4	98.9 ± 1.8	**99.1** ± 1.4	96.3 ± 2.3	94.0 ± 2.7	91.6 ± 6.3	94.9 ± 2.0
	0.84 (13)	0.61 (15)	0.36	0.48	0.86 (11)	0.29	0.18	0.21
50	98.4 ± 1.9	95.6 ± 9.0	99.4 ± 0.5	**99.6** ± 0.3	96.6 ± 2.3	96.1 ± 2.4	93.0 ± 3.6	95.8 ± 2.3
	0.83 (31)	0.60 (37)	0.35	0.46	0.84 (25)	0.28	0.17	0.20

Table 1.5 One vs. Two

	semi-supervised kernels					standard kernels		
Training set size	Improved Order	Order	Gaussian Field	Diffusion	Max-align	RBF $\sigma = 1500$	Linear	Quadratic
10	**69.6** ± 6.5	**68.8** ± 6.1	65.5 ± 8.9	**68.4** ± 8.5	55.7 ± 4.4	65.0 ± 7.0	63.1 ± 6.9	65.4 ± 6.5
	0.45 (1)	0.41 (1)	0.32	0.34	0.86 (1)	0.23	0.25	0.27
30	**82.4** ± 4.1	**82.0** ± 4.0	79.6 ± 4.1	**83.0** ± 4.2	67.2 ± 5.0	77.7 ± 3.5	72.4 ± 6.1	76.5 ± 5.1
	0.32 (6)	0.28 (6)	0.21	0.23	0.56 (6)	0.10	0.11	0.16
50	87.6 ± 3.5	87.5 ± 3.4	85.9 ± 3.8	**89.1** ± 2.7	76.0 ± 5.3	81.8 ± 2.7	74.4 ± 9.2	81.3 ± 3.1
	0.29 (24)	0.26 (25)	0.19	0.21	0.45 (26)	0.07	0.09	0.12
70	89.2 ± 2.6	89.0 ± 2.7	89.0 ± 1.9	**90.3** ± 2.8	80.9 ± 4.4	84.4 ± 2.0	73.6 ±10.0	83.8 ± 2.8
	0.27 (65)	0.24 (50)	0.17	0.20	0.39 (51)	0.06	0.07	0.12
90	**91.5** ± 1.5	**91.4** ± 1.6	90.5 ± 1.4	**91.9** ± 1.7	85.4 ± 3.1	86.1 ± 1.8	66.1 ±14.8	85.5 ± 1.6
	0.26 (94)	0.23 (97)	0.16	0.19	0.36 (88)	0.05	0.07	0.11

Table 1.6 Odd vs. Even

	semi-supervised kernels					standard kernels		
Training set size	Improved Order	Order	Gaussian Field	Diffusion	Max-align	RBF $\sigma = 2000$	Linear	Quadratic
50	**76.6** ± 4.3	71.5 ± 5.0	41.4 ± 6.8	49.8 ± 6.3	70.3 ± 5.2	57.0 ± 4.0	50.2 ± 9.0	66.3 ± 3.7
	0.47 (26)	0.21 (26)	0.15	0.16	0.51 (25)	-0.62	-0.50	-0.25
100	**84.8** ± 2.6	83.4 ± 2.6	63.7 ± 3.5	72.5 ± 3.3	80.7 ± 2.6	69.4 ± 1.9	56.0 ± 7.8	77.2 ± 2.3
	0.47 (124)	0.17 (98)	0.12	0.13	0.49 (100)	-0.64	-0.52	-0.29
150	**86.5** ± 1.7	**86.4** ± 1.3	75.1 ± 3.0	80.4 ± 2.1	84.5 ± 1.9	75.2 ± 1.4	56.2 ± 7.2	81.4 ± 2.2
	0.48 (310)	0.18 (255)	0.11	0.13	0.50 (244)	-0.66	-0.53	-0.31
200	**88.1** ± 1.3	**88.0** ± 1.3	80.4 ± 2.5	84.4 ± 1.6	86.0 ± 1.5	78.3 ± 1.3	60.8 ± 7.3	84.3 ± 1.7
	0.47 (708)	0.16 (477)	0.10	0.11	0.49 (523)	-0.65	-0.54	-0.33
250	**89.1** ± 1.1	**89.3** ± 1.0	84.6 ± 1.4	87.2 ± 1.3	87.2 ± 1.3	80.4 ± 1.4	61.3 ± 7.6	85.7 ± 1.3
	0.47 (942)	0.16 (873)	0.10	0.11	0.49 (706)	-0.65	-0.54	-0.33

Table 1.7 Ten Digits (10 classes)

	semi-supervised kernels					standard kernels		
Training set size	Improved Order	Order	Gaussian Field	Diffusion	Max-align	RBF $\sigma = 30$	Linear	Quadratic
50	**56.0** ± 3.5	42.0 ± 5.2	41.2 ± 2.9	29.0 ± 2.7	50.1 ± 3.7	28.7 ± 2.0	30.0 ± 2.7	23.7 ± 2.4
	0.27 (26)	0.13 (25)	0.03	0.11	0.31 (24)	-0.89	-0.80	-0.65
100	**64.6** ± 2.1	59.0 ± 3.6	58.5 ± 2.9	47.4 ± 2.7	63.2 ± 1.9	46.3 ± 2.4	46.6 ± 2.7	42.0 ± 2.9
	0.26 (105)	0.10 (127)	-0.02	0.08	0.29 (102)	-0.90	-0.82	-0.69
150	**67.6** ± 2.6	65.2 ± 3.0	65.4 ± 2.6	57.2 ± 2.7	**67.9** ± 2.5	57.6 ± 1.5	57.3 ± 1.8	53.8 ± 2.2
	0.26 (249)	0.09 (280)	-0.05	0.07	0.27 (221)	-0.90	-0.83	-0.70
200	71.0 ± 1.8	70.9 ± 2.3	70.6 ± 1.9	64.8 ± 2.1	**72.3** ± 1.7	63.9 ± 1.6	64.2 ± 2.0	60.5 ± 1.6
	0.26 (441)	0.08 (570)	-0.07	0.06	0.27 (423)	-0.91	-0.83	-0.72
250	71.8 ± 2.3	73.6 ± 1.5	73.7 ± 1.2	69.8 ± 1.5	**74.2** ± 1.5	68.8 ± 1.5	69.5 ± 1.7	66.2 ± 1.4
	0.26 (709)	0.08 (836)	-0.07	0.06	0.27 (665)	-0.91	-0.84	-0.72

Table 1.8 ISOLET (26 classes)

spectral transformation is zigzagged, diffusion and Gaussian field's are very smooth, while (improved) order constrained kernels are in between.

- The order constrained kernels (green) have large μ_1 because of the order constraint on the constant eigenvector. Again this seems to be disadvantageous — the spectral transformation tries to balance it out by increasing the value of other μ_i's, so that the bias term K_1's relative influence is smaller. On the other hand the improved order constrained kernels (black) allow μ_1 to be small. As a result the rest μ_i's decay fast, which is desirable.

In summary, the improved order constrained kernel is consistently the best among all kernels.

1.7 Conclusion

We have proposed and evaluated a novel approach for semi-supervised kernel construction using convex optimization. The method incorporates order constraints, and the resulting convex optimization problem can be solved efficiently using a QCQP. In this work the base kernels were derived from the graph Laplacian, and no parametric form for the spectral transformation was imposed, making the approach more general than previous approaches. Experiments show that the method is both computationally feasible and results in improvements to classification performance when used with support vector machines.

There are several future directions:

- In both order constrained kernels and improved order constrained kernels, we are learning a large number of parameters $\mu_1, \ldots, \mu_n$ based on l labeled examples. Usually $l \ll n$, which suggests the danger of overfitting. However we have to consider two mitigating factors: the first is that we in practice only learn the top $m < n$ parameters and set the rest at zero; the second is that the μ's are order-constrained, which reduces the effective complexity. One interesting question for future research is an estimate of the effective number of parameters in these methods.
- The QCQP problem may be transformed into a standard Quadratic Program (QP), which may result in further improvements in computational efficiency.
- The alignment is one example of a cost function that can be optimized. With a fixed kernel, margin based upper bounds on misclassification probability can be derived. As such, other cost functions that directly optimize quantities such as the margin can also be used. This approach has been considered in the work of Chapelle and Vapnik [2000] where the so called span bound was introduced and optimized using gradient descent; and in Lanckriet et al. [2004], Bousquet and Herrmann [2002] where optimization of tighter Rademacher complexity bounds has been proposed.

Acknowledgment

We thank Olivier Chapelle and the anonymous reviewers for their comments and suggestions.

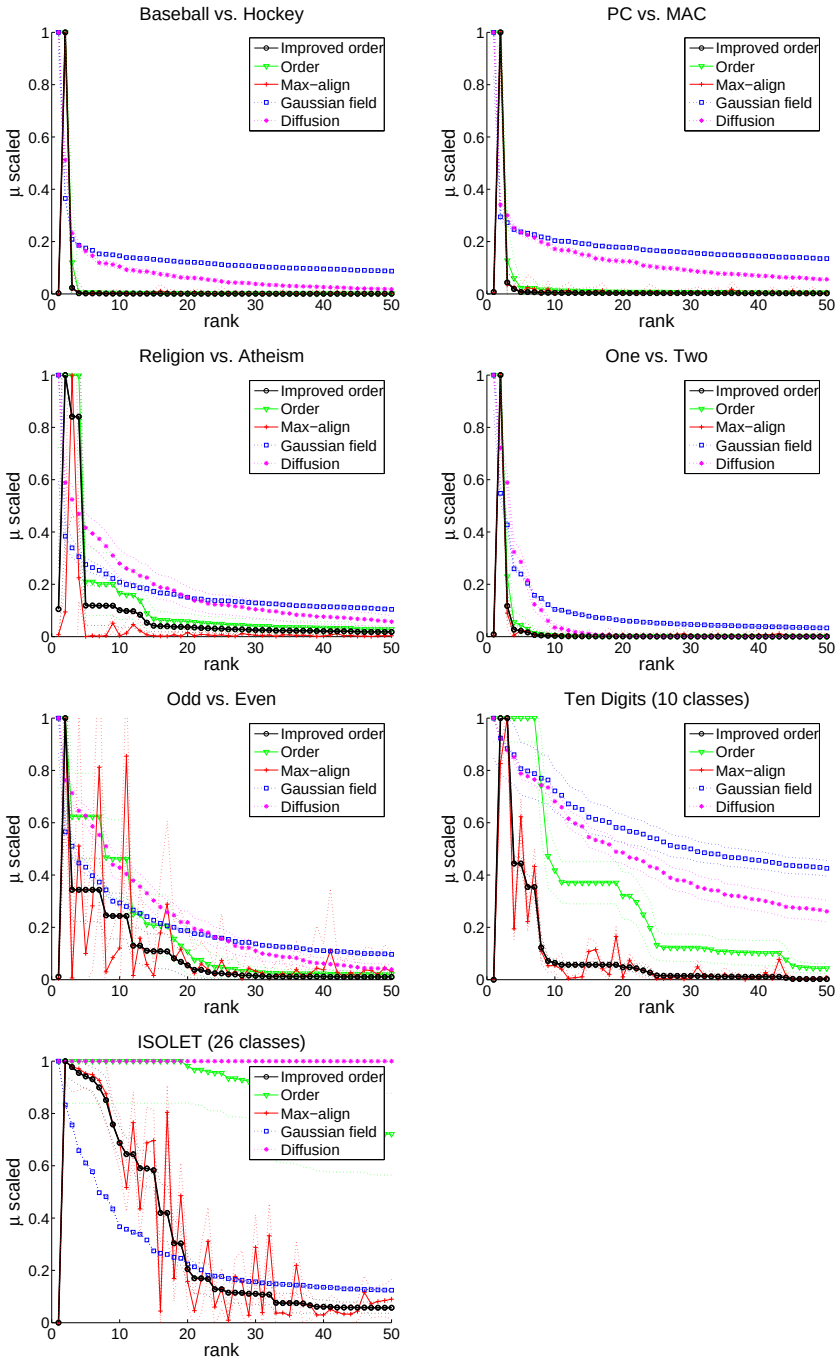


Figure 1.2 Comparison of spectral transformation for the 5 semi-supervised kernels.

References

O. Bousquet and D. Herrmann. On the complexity of learning the kernel matrix. In *Advances in NIPS 14*, 2002.

S. Boyd and L. Vandenberge. *Convex Optimization*. Cambridge University Press, Cambridge UK, 2004.

O. Chapelle and V. Vapnik. Model selection for support vector machines. In *Advances in NIPS 12*, 2000.

Olivier Chapelle, Jason Weston, and Bernhard Schölkopf. Cluster kernels for semi-supervised learning. In *Advances in Neural Information Processing Systems, 15*, volume 15, 2002.

F. R. K. Chung. *Spectral graph theory, Regional Conference Series in Mathematics, No. 92*. American Mathematical Society, 1997.

Nello Cristianini, John Shawe-Taylor, Andre Elisseeff, and Jaz Kandola. On kernel-target alignment. In *Advances in NIPS*, 2001.

G.R.G. Lanckriet, N. Cristianini, P. Bartlett, L. El Ghaoui, and M.I. Jordan. Learning the kernel matrix with semidefinite programming. *Journal of Machine Learning Research*, 5:27–72, 2004.

J Löfberg. *YALMIP 3*, 2004. http://control.ee.ethz.ch/ joloef/yalmip.msql.

A. Smola and R. Kondor. Kernels and regularization on graphs. In *Conference on Learning Theory, COLT/KW*, 2003.

J. F. Sturm. Using SeDuMi 1.02, a matlab toolbox for optimization over symmetric cones. *Optimization Methods and Software*, 11-12(8):625–653, 1999. Special issue on Interior Point Methods (CD supplement with software).

Xiaojin Zhu, John Lafferty, and Zoubin Ghahramani. Semi-supervised learning: From Gaussian fields to Gaussian processes. Technical Report CMU-CS-03-175, Carnegie Mellon University, 2003.