# Visualizing Graphs with Structure Preserving Embedding

**Blake Shaw**
Department of Computer Science
Columbia University
New York, NY 10027
blake@cs.columbia.edu

**Tony Jebara**
Department of Computer Science
Columbia University
New York, NY 10027
jebara@cs.columbia.edu

## Abstract

Structure Preserving Embedding (SPE) is a method for embedding graphs in low-dimensional Euclidean space such that the embedding preserves the graph's global topological properties. Specifically, topology is preserved if a connectivity algorithm can recover the original graph from only the coordinates of its nodes after embedding. Given an input graph and an algorithm for linking embedded nodes, SPE learns a low-rank kernel matrix by means of a semidefinite program with linear constraints that captures the connectivity structure of the input graph. The SPE cost function ensures that the learned kernel is low-rank and thus the resulting embedding uses low-dimensional coordinates for each node that reproduce the original graph when processed by a connectivity algorithm (such as $k$-nearest neighbors, or $b$-matching). SPE provides significant improvements in terms of visualization and lossless compression of graphs, outperforming popular methods such as spectral embedding and spring embedding. Furthermore, we find that many classical graphs and networks can be properly embedded using only a few dimensions.

## 1 Introduction

SPE accepts as input an unweighted graph consisting of $N$ nodes and $|E|$ edges represented as a symmetric adjacency matrix $A \in \{0,1\}^{N^2}$ specifying which pairs of nodes are connected, and finds a low-dimensional *structure preserving* embedding of the input graph in Euclidean space. The embedding can be represented as a positive semi-definite kernel matrix $K \in \mathbb{R}^{N^2}$ that specifies all pairwise affinities between nodes, and by singular value decomposition (SVD) or principal component analysis (PCA) of the matrix $K$, specifies a unique set of coordinates for each node $\vec{y_i} \in \mathbb{R}^d$ for $i = 1, \ldots, N$. Given a connectivity algorithm $\mathcal{G}$ (such as $k$-nearest neighbors, $b$-matching, or maximum weight spanning tree) that accepts as input an embedding and returns an adjacency matrix, an embedding is *structure preserving* if when the embedding is processed by the connectivity algorithm $\mathcal{G}$, the result is exactly the input graph: $\mathcal{G}(K) = A$. This article proposes SPE, an efficient optimization based on semidefinite programming for finding an embedding $K$ such that $K$ is both low-rank and structure preserving.

Traditional graph embedding algorithms such as Spectral Embedding and Spring Embedding do not explicitly preserve structure according to our definition and thus in practice produce poor visualizations of many simple classical graphs. In Figure 1 we see the classical Mobius ladder graph and the resulting visualizations from the two methods. The spectral embedding looks degenerate, not resembling a Mobius band in any regard. Also, the eigenspectrum indicates that the embedding is 6-dimensional when we expect to be able to embed this graph using fewer dimensions. Two spring embeddings are shown. The left spring embedding is a good diagram of what the graph should look

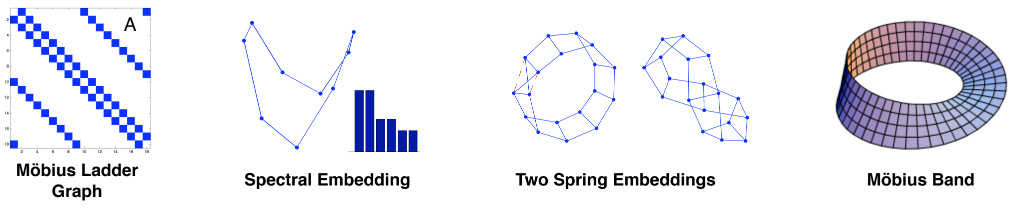

Figure 1: Embedding the classical Mobius Ladder Graph. Given the adjacency matrix (left), the visualizations produced by spectral embedding and spring embedding (middle) do not accurately capture the graph topology.

like; however, we see that the famous twist of the Mobius strip is not accurately captured. Given the coordinates in Euclidean space produced by this method, any simple neighbor-finding algorithm $\mathcal{G}$ would connect nodes along the red dotted lines, not the blue ones specified by the connectivity matrix, and thus the inherent connectivity of the embedding disagrees with the actual connectivity of the graph. The spring embedding on the right shows a common poor random initialization, resulting in a local minima that is no longer visually recognizable or accurate. We are motivated to find a simple tool for properly visualizing graphs such as the Mobius ladder.

## 2 Preserving Graph Structure

For a variety of different connectivity algorithms $\mathcal{G}$ we can define a set of linear constraints on an embedding $K$ to enforce that $\mathcal{G}(K) = A$. We consider $k$-nearest-neighbors, epsilon balls, maximum weight matchings ($b$-matchings), and maximum weight spanning trees.

**Definition 1.** *We define the* distance *between any pair of points $(i, j)$, with respect to a given positive semidefinite kernel matrix $K$ as $D_{ij} = K_{ii} + K_{jj} - 2K_{ij}$.*

When $\mathcal{G}$ is $k$-nearest neighbors algorithm, each node is connected to the $k$ other nodes with the smallest distance. This is enforced via $D_{ij} > (1 - A_{ij}) \max_m(A_{im}D_{im})$. When $\mathcal{G}$ is the epsilon-ball algorithm, each node is connected to all other nodes with distance less then $\epsilon$, we enforce that $D_{ij}(A_{ij} - \frac{1}{2}) \leq \epsilon(A_{ij} - \frac{1}{2})$.

**Definition 2.** *Given a kernel matrix $K$, we can define the* weight *between two points $(i, j)$ as the negated pairwise distance between them: $W_{ij} = -D_{ij} = -K_{ii} - K_{jj} + 2K_{ij}$.*

Note that the $W$ matrix is simply a linear function of $K$. Given $W$, a maximum weight subgraph or $b$-matching algorithm $\mathcal{G}$ finds the connectivity matrix $\mathcal{G}(K)$ that has maximum weight while enforcing a set of degree constraints $b_i$ for $i = 1...N$: $\mathcal{G}(K) = \arg\max_A \sum_{ij} W_{ij}A_{ij}$ s.t. $\sum_j A_{ij} = b_i, A_{ij} = A_{ji}, A_{ii} = 0, A_{ij} \in \{0, 1\}$. Similarly, a maximum weight spanning tree algorithm $\mathcal{G}$ returns the maximum weight subgraph $\mathcal{G}(K)$ such that $\mathcal{G}(K) \in \mathcal{T}$, where $\mathcal{T}$ is the set of all tree graphs: $\mathcal{G}(K) = \arg\max_A \sum_{ij} W_{ij}A_{ij}$ s.t $A \in \mathcal{T}$. Unfortunately, for both these algorithms, the constraints on $K$ to make it structure preserving cannot be enumerated with a small set of linear inequalities, as is the case with $k$-nearest neighbors and other greedy algorithms; in fact, there can be an exponential number of constraints of the form: $\sum_{ij} W_{ij}A_{ij} \geq \sum_{ij} W_{ij}\hat{A}_{ij}$. However, in Section 4 we show a cutting plane approach such that the exponential enumeration is avoided and the most violated inequalities are introduced sequentially. It has been shown that cutting-plane optimizations such as this converge in a finite number of steps [5] and perform well in practice.

## 3 Reducing Dimensionality

We propose an objective function that favors low-dimensional embeddings. Spectral embedding is known to reduce the dimensionality of the embedding since it uses the most dominant eigenvectors of $A = V\Lambda V^T$ corresponding to the largest $k$ eigenvalues as the embedding coordinates first. We are interested in recovering an embedding, or equivalently, a positive semidefinite kernel matrix $K \succeq 0$ that is low-rank. Consider the following objective function $\max_{K \succeq 0} \operatorname{tr}(KA)$ and limit the trace norm of $K$ to avoid the objective function from growing unboundedly. We claim that this objective function attempts to recover a low-rank version of spectral embedding.

**Theorem 1.** *The objective function* $\max_{K\succeq 0} \operatorname{tr}(KA)$ *subject to* $\operatorname{tr}(K) \leq 1$ *recovers a matrix* $K$ *that up to a scaling factor is the rank-r matrix approximation of A, where r is at most the multiplicity of the largest eigenvalue of A.*

*Proof.* Rewrite the matrices in terms of the eigendecomposition of the positive semidefinite matrix $K = U\Lambda U^T$ and the symmetric matrix $A = V\tilde{\Lambda}V^T$, and insert into the objective function:

$$\max_{K\succeq 0} \operatorname{tr}(KA) \quad = \quad \max_{\Lambda\in\mathcal{L}, U\in\mathcal{O}} \operatorname{tr}(U\Lambda U^T V\tilde{\Lambda}V^T)$$

where $\mathcal{L}$ is the set of positive semidefinite diagonal matrices and $\mathcal{O}$ is the set of orthonormal matrices also known as the Stiefel manifold. Since $\operatorname{tr}(MN) = \operatorname{tr}(NM)$, we can rewrite the above as

$$\begin{aligned}\max_{K\succeq 0} \operatorname{tr}(KA) &= \max_{\Lambda\in\mathcal{L}, U\in\mathcal{O}} \operatorname{tr}((V^T U)\Lambda(V^T U)^T\tilde{\Lambda}) \\ &= \max_{\Lambda\in\mathcal{L}, R\in\mathcal{O}} \operatorname{tr}(R\Lambda R^T\tilde{\Lambda})\end{aligned}$$

where we have defined the orthonormal matrix $R = V^T U$. Consider the optimization over $R$ for a given $\Lambda$. This problem is known as a homogenous quadratically constrained quadratic program (QQP) [1] and has the solution $R^*$ as the product of the diagonalizers of $\Lambda$ and $\tilde{\Lambda}$. Since both $\Lambda$ and $\tilde{\Lambda}$ are already diagonal matrices, $R^*$ must be a permutation matrix. Similarly, if the diagonal entries of $\Lambda$ and $\tilde{\Lambda}$ are arranged in decreasing order, then $R^* = I$ and the optimal eigenvectors of $K$ must satisfy $U = V$. At this setting, we obtain the following maximum

$$\max_{R\in\mathcal{O}} \operatorname{tr}(R\Lambda R^T\tilde{\Lambda}) \quad = \quad \lambda^T\tilde{\lambda}.$$

Here $\lambda$ is a vector containing the diagonal entries of $\Lambda$ in decreasing order $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_n$ and $\tilde{\lambda}$ is a vector containing the diagonal entries of $\tilde{\Lambda}$ in decreasing order, i.e. $\tilde{\lambda}_1 \geq \tilde{\lambda}_2 \geq \ldots \geq \tilde{\lambda}_n$. Therefore, the full optimization problem can be rewritten in terms of an optimization over non-negative eigenvalues

$$\max_{K\succeq 0, \operatorname{tr}(K)\leq 1} \operatorname{tr}(KA) \quad = \quad \max_{\lambda\geq 0, \lambda^T\mathbf{1}\leq 1} \lambda^T\tilde{\lambda}$$

where we also have to satisfy $\lambda^T\mathbf{1} \leq 1$ since $\operatorname{tr}(K) \leq 1$. If $A$ has a unique largest eigenvalue (with multiplicity 1), to maximize the objective function $\lambda^T\tilde{\lambda}$ we simply set $\lambda_1 = 1$ and the remaining $\lambda_i = 0$ for $i = 2, \ldots, n$ which produces the maximum $\tilde{\lambda}_1$, the top eigenvalue of $A$

$$\max_{K\succeq 0, \operatorname{tr}(K)\leq 1} \operatorname{tr}(KA) \quad = \quad \tilde{\lambda}_1.$$

Thus, if $A$ has a unique largest eigenvalue, the maximization problem produces $K$ as a rank-1 approximation (up to scaling) of $A$. The rank-1 solution must be $K = vv^T$ where $v$ is the leading eigenvector of $A$. If there are ties in $A$ for its top eigenvalues, $K$ is a conic combination of the outer products of the top $r$ eigenvectors of $A$, in other words, the rank $r$ approximation where $r$ is at most the multiplicity of the top eigenvalue of $A$. □

Given this objective function and its preference for low-rank embeddings, we next combine it with constraints that ensure the connectivity algorithm will reconstruct the desired graph.

## 4 Algorithm

Combining the constraints from Section 2, the convex objective function from Section 3, and some additional common constraints $\mathcal{K} = \{K \succeq 0, \mathrm{tr(K)} \leq 1, \sum_{\mathrm{ij}} \mathrm{K_{ij}} = 0, \xi \geq 0\}$ to limit the trace norm of $K$, center $K$, and slacken all constraints, we have Structure Preserving Embedding. If $\mathcal{G}$ is a greedy algorithm such as $k$-nearest neighbors, the steps of SPE are shown in Table 1.

If $\mathcal{G}$ is a maximum weight subgraph method, there exist an exponential number of constraints of the form: $\mathrm{tr(WA)} - \mathrm{tr(W\tilde{A})} \geq \triangle(\tilde{\mathrm{A}}, \mathrm{A}) - \xi \forall \tilde{\mathrm{A}} \in \mathcal{G}$ where $\triangle(\tilde{A}, A) = \frac{1}{N^2}\sum_{ij} |\tilde{A}_{ij} - A_{ij}|$, and $\tilde{A} \in \mathcal{G}$

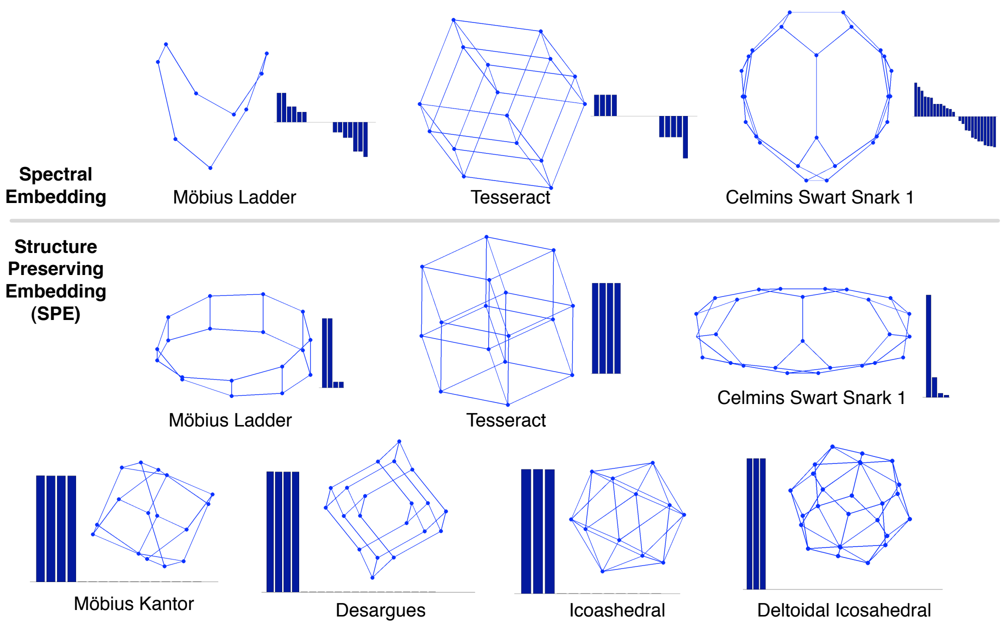

Figure 2: Embeddings of a variety of classical graphs with spectral embedding (above), and SPE (below). Eigenspectrums are shown to the right. Note that SPE is able to find a small number of dimensions that highlight many of the symmetries of these graphs.

states that $\tilde{A}$ is in the family of graphs formed by a connectivity algorithm $\mathcal{G}$, such as $b$-matchings or trees. To avoid enumerating the exponential number of constraints, we start by running the optimization without any structure preserving constraints and then add a constraint at each iteration corresponding to the biggest violator. Given a learned kernel $\hat{K}$ from the last iteration, we find the biggest violator by computing the connectivity $\tilde{A}$ that maximizes $\mathrm{tr}(\hat{\mathrm{W}}\tilde{\mathrm{A}})$ s.t. $\tilde{A} \in \mathcal{G}$ using a maximum weight subgraph method. We then add the constraint to our optimization as follows $(\mathrm{tr}(\mathrm{WA}) - \mathrm{tr}(\mathrm{W}\tilde{\mathrm{A}})) \geq \triangle(\tilde{\mathrm{A}}\mathrm{A}) - \xi$. The first iteration yields a rank 1 solution, which typically violates many constraints, but after several iterations, the algorithm converges when $|\mathrm{tr}(\mathrm{WA_i}) - \mathrm{tr}(\mathrm{WA})| \leq \epsilon$, where $\epsilon$ is an input parameter. Table 2 summarizes the cutting-plane formulation of SPE.

SPE is implemented in MATLAB as a Semidefinite Program using Yalmip and CSDP and has complexity similar to other dimensionality reduction SDPs such as Semidefinite Embedding. The complexity is $O(N^3 + \mathcal{C}^3)$ [7] where $\mathcal{C}$ denotes the number of constraints and (for the $k$-nearest neighbor connectivity algorithm) we typically have $\mathcal{C} \propto |E|$. However, in practice many constraints are inactive and working set methods (now common practice for SDPs) do much better. Running SPE on graphs of a 1000 edges takes only a few minutes on a standard workstation. For the cutting plane formulation, we add constraints iteratively, and it can be shown that the algorithm will converge in polynomial time [5] for quadratic programs with linear constraints. Since we have a semidefinite program, these guarantees do not carry over immediately, although in practice the cutting plane al-

| | |
|---|---|
| Input | $A \in \{0,1\}^{N^2}$, connectivity algorithm $\mathcal{G}$, and parameters $C$. |
| Step 1 | Solve SDP $\tilde{K} = \arg\max_{K \in \mathcal{K}} \mathrm{tr}(\mathrm{KA}) - \mathrm{C}\xi$ s.t. $D_{ij} > (1 - A_{ij}) \max_m (A_{im} D_{im}) - \xi$ where $D_{ij} = K_{ii} + K_{jj} - 2K_{ij}$ |
| Step 2 | Apply KPCA to $\tilde{K}$ and use the top eigenvectors as embedding coordinates |

Table 1: Structure Preserving Embedding algorithm for $k$-nearest neighbor constraints.

| Input | $A \in \{0,1\}^{N^2}$, connectivity algorithm $\mathcal{G}$, and parameters $C, \epsilon$. |
|---|---|
| Step 1 | Solve SDP $\tilde{K} = \arg\max_{K \in \mathcal{K}} \mathrm{tr}(\mathrm{KA}) - \mathrm{C}\xi$. and compute $\tilde{W}_{ij} = -K_{ii} - K_{jj} + 2K_{ij}$ |
| Step 2 | Use $\mathcal{G}$, $\tilde{K}$ to find biggest violator $\tilde{A} = \arg\max_{A} \mathrm{tr}(\tilde{\mathrm{W}}\mathrm{A})$. |
| Step 3 | If $\lvert \mathrm{tr}(\tilde{\mathrm{W}}\tilde{\mathrm{A}}) - \mathrm{tr}(\tilde{\mathrm{W}}\mathrm{A}) \rvert > \epsilon$, add constraint $\mathrm{tr}(\mathrm{WA}) - \mathrm{tr}(\mathrm{W}\tilde{\mathrm{A}}) \geq \triangle(\tilde{\mathrm{A}}, \mathrm{A}) - \xi$ and go to Step 1 |
| Step 4 | Apply KPCA to $\tilde{K}$ and use the top eigenvectors as embedding coordinates |

Table 2: Structure Preserving Embedding algorithm with cutting-plane constraints.

gorithm works well and has also been successfully deployed in settings beyond structured prediction and quadratic programming [6].

## 5 Experiments

We present visualization results on a variety of synthetic and real-world datasets with SPE, highlighting the improvements over purely spectral methods. Figure 2 shows a variety of classical graphs visualized by Spectral Embedding and SPE. Note that Spectral Embedding typically finds many eigenvectors with dominant eigenvalues, and thus needs many more coordinates for accurate visualization, as compared to SPE which finds compact and accurate embeddings.

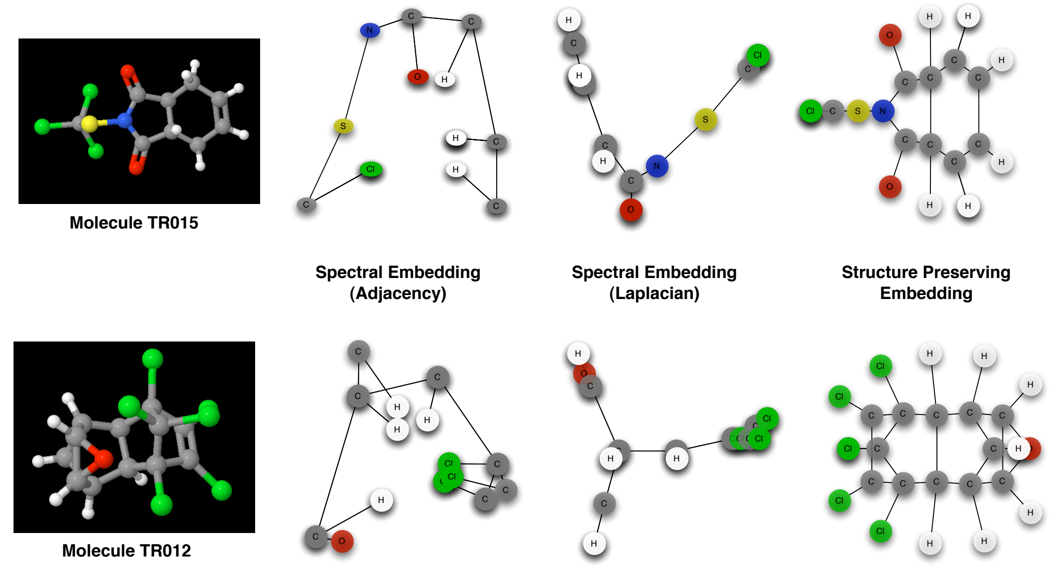
Molecule TR015
Spectral Embedding (Adjacency)
Spectral Embedding (Laplacian)
Structure Preserving Embedding
Molecule TR012

Figure 3: Two comparisons of molecule embeddings (top row and bottom). The SPE embedding (right) more closely resembles the true physical embedding of the molecule (left), despite being given only connectivity information.

Figure 3 shows an embedding of two organic compounds. The true physical embedding in 3D space is shown on the left. Given only connectivity information SPE is able to produce coordinates for each atom that better resemble the true physical coordinates. Figure 4 shows a visualization of a social network by SPE, as well as two variants of Spectral Embedding. The eigensepctrum shown next to each embedding reveals that both spectral embeddings are very high-dimensional, where SPE requires far fewer dimensions to accurately capture the structure of the data. Also, note that the embedding that uses the graph Laplacian is dominated by the degree distribution of the network. Nodes with high degree require neighbors to be very far away. The SPE embedding was created using $b$-matching constraints, since we expect friendship to be better described as a mutual process than a greedy one, although the $k$-nearest neighbor constraints yield a similar solution. SPE represents the network quite compactly, only requiring 6-dimensions to describe its topology.

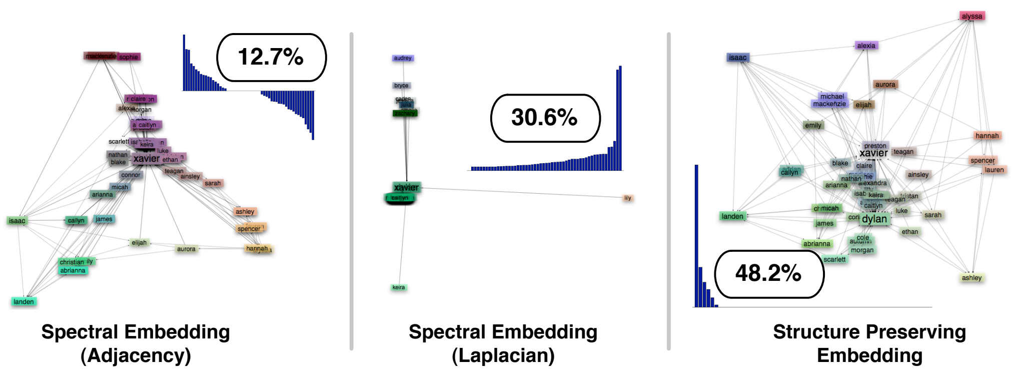

Figure 4: Visualizing a Facebook social network. The eigenspectrum is presented next to each embedding, as well as the percentage of eigenenergy contained in the top 2 dimensions. We see that in 2D, SPE is able to capture a higher percentage of the modes of variation in the data compared to purely spectral methods.

## 6 Conclusion

SPE offers significant improvements over current graph embedding algorithms in terms of both the quality of the resulting visualizations as well as the amount of information compression. SPE allows us to accurately visualize many interesting network structures ranging from classical graphs to social network data using relatively few dimensions.

## References

[1] K. Anstreicher and H. Wolkowicz. On Lagrangian relaxation of quadratic matrix constraints. *SIAM Journal on Matrix Analysis and Applications*, 22(1):41–55, 2001.

[2] S. Arora, S. Rao, and U.V. Vazirani. Expander flows, geometric embeddings and graph partitioning. In *Symposium on Theory of Computing*, 2004.

[3] G. Di Battista, P. Eades, R. Tamassia, and I.G. Tollis. *Graph Drawing: algorithms for the visualization of graphs*. Prentice Hall, 1999.

[4] F. R. K. Chung. *Spectral Graph Theory (CBMS Regional Conference Series in Mathematics, No. 92) (Cbms Regional Conference Series in Mathematics)*. American Mathematical Society, February 1997.

[5] T. Finley and T. Joachims. Training structural SVMs when exact inference is intractable. In *International Conference on Machine Learning (ICML)*, 2008.

[6] D. Sontag and T. Jaakkola. New outer bounds on the marginal polytope. In J.C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 1393–1400. MIT Press, Cambridge, MA, 2008.

[7] K. Q. Weinberger, B. D. Packer, and L. K. Saul. Nonlinear dimensionality reduction by semidefinite programming and kernel matrix factorization. In *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics*, Barbados, January 2005.