How to learn a graph from smooth signals: Supplementary Material

Vassilis Kalofolias
Signal Processing Laboratory 2 (LTS2)
École polytechnique fédérale de Lausanne (EPFL), Switzerland

A Derivations and proofs

A.1 Detailed explanation of eq. (3)

$$\begin{aligned}
\|W \circ Z\|_1 &= \sum_{i=1}^{m}\sum_{j=1}^{m} W_{ij}\|x_i - x_j\|_2^2 \\
&= \sum_{i=1}^{m}\sum_{j=1}^{m} (x_i - x_j)^\top W_{ij}(x_i - x_j) \\
&= 2\sum_{i=1}^{m}\sum_{j=1}^{m} x_i^\top W_{ij} x_i - 2\sum_{i=1}^{m}\sum_{j=1}^{m} x_i^\top W_{ij} x_j \\
&= 2\sum_{i=1}^{m} x_i^\top x_i \sum_{j=1}^{m} W_{ij} - 2\operatorname{tr}\left(X^\top W X\right) \\
&= 2\operatorname{tr}\left(X^\top D X\right) - 2\operatorname{tr}\left(X^\top W X\right) \\
&= 2\operatorname{tr}\left(X^\top L X\right),
\end{aligned}$$

where D is the diagonal matrix with elements $D_{ii} = \sum_i W_{ij}$.

A.2 Proof of proposition 2

Proof. We change variable $\tilde{W} = W/\gamma$ to obtain

$$\begin{aligned}
&F(Z, \alpha, \beta) = \\
&= \gamma \operatorname*{argmin}_{\tilde{W}} \|\gamma\tilde{W} \circ Z\|_{1,1} - \alpha\mathbf{1}^\top \log(\gamma\tilde{W}\mathbf{1}) + \beta\|\gamma\tilde{W}\|_F^2 \\
&= \gamma \operatorname*{argmin}_{\tilde{W}} \gamma\|\tilde{W} \circ Z\|_{1,1} - \alpha\mathbf{1}^\top \log(\tilde{W}\mathbf{1}) + \beta\gamma^2\|\tilde{W}\|_F^2 \\
&= \gamma \operatorname*{argmin}_{\tilde{W}} \|\tilde{W} \circ Z\|_{1,1} - \frac{\alpha}{\gamma}\mathbf{1}^\top \log(\tilde{W}\mathbf{1}) + \beta\gamma\|\tilde{W}\|_F^2 \\
&= \gamma F\left(Z, \frac{\alpha}{\gamma}, \beta\gamma\right),
\end{aligned}$$

where we used the fact that $\log(\gamma\tilde{W}\mathbf{1}) = \log(\tilde{W}\mathbf{1}) + \text{const.}(W)$. The second equality is obtained from the first one for $\gamma = \alpha$. □

A.3 Proof of proposition 3

Proof. For equation (15)

$$\begin{aligned}
&H(Z + \gamma, \alpha, s) = \\
&= \operatorname*{argmin}_{W \in \mathcal{W}_m} \|W \circ Z + \gamma W\|_{1,1} + \alpha\|W\|_F^2 + \alpha\|W\mathbf{1}\|^2 \\
&\qquad \text{s.t.,} \qquad \|W\|_{1,1} = s \\
&= \operatorname*{argmin}_{W \in \mathcal{W}_m} \|W \circ Z\|_{1,1} + \gamma\|W\|_{1,1} + \alpha\|W\|_F^2 + \alpha\|W\mathbf{1}\|^2 \\
&\qquad \text{s.t.,} \qquad \|W\|_{1,1} = s \\
&= \operatorname*{argmin}_{W \in \mathcal{W}_m} \|W \circ Z\|_{1,1} + \gamma s + \alpha\|W\|_F^2 + \alpha\|W\mathbf{1}\|^2 \\
&\qquad \text{s.t.,} \qquad \|W\|_{1,1} = s \\
&= H(Z, \alpha, s),
\end{aligned}$$

because $\|a + b\|_1 = \|a\|_1 + \|b\|_1$ for positive a, b.

For equation (16)we change variable in the optimization and use $\tilde{W} = W/\gamma$ to obtain

$$\begin{aligned}
&H\left(Z, \alpha, s\right) = \\
&\quad = \gamma \operatorname*{argmin}_{\tilde{W} \in \mathcal{W}_m} \|\gamma\tilde{W} \circ Z\|_{1,1} + \alpha\|\gamma\tilde{W}\|_F^2 + \alpha\|\gamma\tilde{W}\mathbf{1}\|^2 \\
&\qquad \text{s.t.,} \qquad \|\gamma\tilde{W}\|_{1,1} = s \\
&\quad = \gamma \operatorname*{argmin}_{\tilde{W} \in \mathcal{W}_m} \gamma\|\tilde{W} \circ Z\|_{1,1} + \gamma^2\alpha\|\tilde{W}\|_F^2 + \gamma^2\alpha\|\tilde{W}\mathbf{1}\|^2 \\
&\qquad \text{s.t.,} \qquad \|\tilde{W}\|_{1,1} = \frac{s}{\gamma} \\
&\quad = \gamma \operatorname*{argmin}_{\tilde{W} \in \mathcal{W}_m} \|\tilde{W} \circ Z\|_{1,1} + \gamma\alpha\|\tilde{W}\|_F^2 + \gamma\alpha\|\tilde{W}\mathbf{1}\|^2 \\
&\qquad \text{s.t.,} \qquad \|\tilde{W}\|_{1,1} = \frac{s}{\gamma} \\
&\quad = \gamma H\left(Z, \alpha\gamma, \frac{s}{\gamma}\right).
\end{aligned}$$

The second equality follows trivially for $\gamma = s$.

□

B Optimization details and algorithm for model of Dong et al. (2015).

To obtain Algorithm 1 (for our model), we need the following:

$$
\begin{aligned}
&K = S \quad (\|S\|_2 = \sqrt{2(m-1)})\\
&\mathrm{prox}_{\lambda f_1}(y) = \max(0, y - \lambda z),\\
&\mathrm{prox}_{\lambda f_2}(y) = \frac{y_i + \sqrt{y_i^2 + 4\alpha\lambda}}{2},\\
&\nabla f_3(w) = 2\beta w,\\
&\zeta = 2\beta \qquad \text{(Lipschitz constant of gradient of } f_3\text{)},
\end{aligned}
$$

where m is the number of nodes of the graph.

To obtain Algorithm 2 (for model by Dong et al. 2015), we need the following:

$$
\begin{aligned}
&K = \mathbf{21} \quad (\|\mathbf{21}\|_2 = 2\sqrt{m(m-1)/2})\\
&\mathrm{prox}_{\lambda f_1}(y) = \max(0, y - \lambda z),\\
&\mathrm{prox}_{\lambda f_2}(y) = s,\\
&\nabla f_3(w) = \alpha(4w + 2S^\top S w),\\
&\zeta = 2\alpha(m+1) \text{ (Lipschitz constant of gradient of } f_3\text{)}.
\end{aligned}
$$

Algorithm 2 Primal dual algorithm for model (14).

```
Input: $z, \alpha, s, w^0 \in \mathcal{W}_v$, $c^0 \in \mathbb{R}_+$, $\gamma$, tolerance $\epsilon$
for $i = 1, \ldots, i_{max}$ do
    $y^i = w^i - \gamma(2\alpha(2w^i + S^\top S w^i) + 2c^i)$
    $\bar{y}^i = c^i + \gamma(2\sum_j w^i_j)$
    $p^i = \max(0, y^i - 2\gamma z)$
    $\bar{p}^i = \bar{y}^i - \gamma s$
    $q^i = p^i - \gamma(2\alpha(2p^i + S^\top S p^i) + 2p^i)$
    $\bar{q}^i = \bar{p}^i + \gamma(2\sum_j p^i_j)$
    $w^i = w^i - y^i + q^i$;
    $c^i = c^i - \bar{y}^i + \bar{q}^i$;
    if $\|w^i - w^{i-1}\|/\|w^{i-1}\| < \epsilon$ and
        $|c^i - c^{i-1}|/|c^{i-1}| < \epsilon$ then
        break
    end if
end for
```

C Artificial smooth data

Table 1 summarizes the different models of smooth signals, that are plotted in figure 1. Samples of these signals on the same non-uniform graph are plotted in figure 2.

D More real data experiments

D.1 Timing comparison on real data

We compare the time needed for different algorightms of graph learning. We use 25 to 200 different images

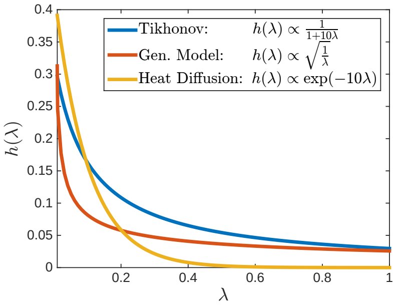

Figure 1: The filters of Table 1 for $\alpha = 10$.

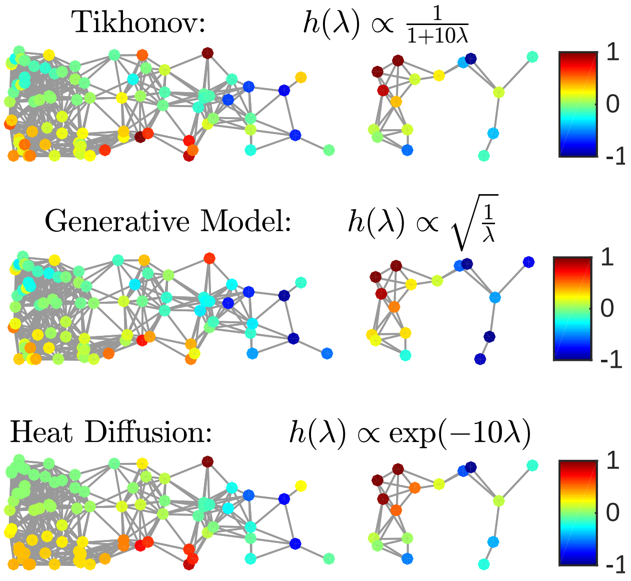
Tikhonov: $h(\lambda) \propto \frac{1}{1+10\lambda}$
1
0
-1
Generative Model: $h(\lambda) \propto \sqrt{\frac{1}{\lambda}}$
1
0
-1
Heat Diffusion: $h(\lambda) \propto \exp(-10\lambda)$
1
0
-1

Figure 2: Different smooth signals on the Non Uniform graph used for our artificial data experiments. All signals are obtained by smoothing the same initial $x_0 \sim \mathcal{N}(0, I)$ with three different filters. This instance of the graph is disconnected with 2 components.

Table 1: Different Types of Smooth Signals.

Concept	Model	Graph filter
Tikhonov	$X = \arg\min_X \frac{1}{2}\|X - X_0\|_F^2 + \frac{1}{\alpha}\operatorname{tr}\left(X^\top L X\right)$	$g(\lambda) = \frac{1}{1+\alpha\lambda}$
Generative model	$X \sim \mathcal{N}\left(0, L^\dagger\right)$	$g(\lambda) = \begin{cases} \frac{1}{\sqrt{\lambda}} & \text{if } \lambda > 0 \\ 0 & \text{if } \lambda = 0 \end{cases}$
Heat diffusion	$X = \exp\left(-\alpha L\right) X_0$	$g(\lambda) = \exp(-\alpha\lambda)$

Table 2: Performance of Different Algorithms on Artificial Data. Each setting has a random graph with 100 nodes and **100** smooth signals from 3 different smoothness models and added 10% noise. Results averaged over 20 random graphs for each setting. F-measure: the higher the better (weights ignored). Edge and degree distances: the lower the better. For relative $\ell - 1$ distances we normalize s.t. $\|w\|_1 = \|w_0\|_1$. For relative $\ell - 2$ distances we normalize s.t. $\|w\|_2 = \|w_0\|_2$. Baseline: for F-measure, the best result by thresholding $\exp(-d^2)$. For edge and degree distances we use $\exp(-d^2/2\sigma^2)$ without thresholding.

	Tikhonov			Generative Model			Heat Diffusion		
	base	**Dong etal**	**Ours**	**base**	**Dong etal**	**Ours**	**base**	**Dong etal**	**Ours**
Rand. Geometric									
F-measure	0.667	0.860	**0.886**	0.671	0.836	**0.858**	0.752	0.837	**0.848**
edge ℓ-1	0.896	0.414	**0.364**	0.851	0.487	**0.468**	0.620	0.526	**0.451**
edge ℓ-2	0.700	0.430	**0.390**	0.692	0.494	**0.477**	0.582	0.535	**0.471**
degree ℓ-1	0.158	0.151	**0.080**	0.268	0.159	**0.128**	0.216	0.225	**0.143**
degree ℓ-2	0.707	0.179	**0.095**	0.679	0.193	**0.145**	0.479	0.264	**0.177**
Non Uniform									
F-measure	0.674	**0.821**	0.817	0.650	**0.779**	0.774	0.763	**0.835**	0.827
edge ℓ-1	0.847	0.547	**0.480**	0.931	0.711	**0.673**	0.612	0.583	**0.491**
edge ℓ-2	0.724	0.545	**0.462**	0.784	0.673	**0.624**	0.565	0.598	**0.464**
degree ℓ-1	0.167	0.190	**0.075**	0.241	0.204	**0.139**	0.235	0.257	**0.132**
degree ℓ-2	0.605	0.228	**0.099**	0.614	0.261	**0.187**	0.433	0.325	**0.164**
Erdős Rényi									
F-measure	0.293	0.595	**0.676**	0.207	0.473	**0.512**	0.358	0.595	**0.619**
edge ℓ-1	1.513	0.837	**0.798**	1.623	1.113	**1.090**	1.401	**0.896**	0.899
edge ℓ-2	1.086	0.712	**0.697**	1.129	0.896	**0.888**	1.045	0.767	**0.759**
degree ℓ-1	0.114	0.129	**0.084**	0.135	0.146	**0.114**	0.185	**0.182**	0.184
degree ℓ-2	0.932	0.202	**0.116**	1.053	0.227	**0.185**	0.875	**0.241**	0.276
Barabási-Albert									
F-measure	0.325	0.564	**0.636**	0.357	0.588	**0.632**	0.349	0.631	**0.711**
edge ℓ-1	1.541	0.939	**0.885**	1.513	0.940	**0.914**	1.473	0.843	**0.774**
edge ℓ-2	1.073	0.802	**0.761**	1.052	0.808	**0.773**	1.049	0.732	**0.672**
degree ℓ-1	0.225	0.309	**0.145**	0.243	0.311	**0.229**	0.281	0.336	**0.181**
degree ℓ-2	0.560	0.378	**0.281**	0.563	0.386	**0.350**	0.570	0.429	**0.319**

of the USPS dataset to learn graphs from different algorithms and report the time in seconds. The results are given in table 3. Log-det (CVX) denotes the CVX solution of the model proposed by Lake and Tenenbaum (2010). Dong etal. (CVX) denotes the CVX solution provided by Dong et al. (2015). We solve the same problem in the form of eq. (14) with Algorithm 2, while our model of eq. (12) is solved by Algorithm 1. CVX is a generic convex optimization tool meant to be used as a black box, and therefore it struggles to solve even moderate sized problems. This effect is stronger when the log of the determinant is used in the objective function. On the other hand, our algorithms are fast proximal based methods tailored specifically for our problems, and are therefore much faster. Note also that the time needed by both our algorithms is linear to the number of iterations that varies according to the parameters and the step size. In these experiments they converged after around 300 and 2000 iterations respectively for algorithms 1 and 2.

D.2 Learning the graph of COIL 20 images

We randomly sample the classes so that the average size increases non-linearly from around 3 to around 60 samples per class. The distribution for one of the instances of this experiment is plotted in fig. 3. We sample from the same distribution 20 times and measure the average performance of the models for different graph densities. For each of the graphs, we run standard spectral clustering (as in the work of Ng, Jordan, Weiss, et al. 2002 but without normalizing the Lapla-

Table 3: Time for solving different graph learning models in seconds. We use a dash for cases that took more than an hour. Log-det is the model by Lake and Tenenbaum (2010), solved with CVX. Our algorithms are run for a tolerance of $\epsilon = 1e - 5$.

Problem size	Log-det (CVX)	Dong etal. (CVX)	Algorithm 1	Algorithm 2
25 nodes	34.29	1.49	0.09	0.10
50 nodes	473.13	4.21	0.19	0.20
100 nodes	-	51.41	0.31	0.47
200 nodes	-	2109.87	0.93	1.75
400 nodes	-	-	1.91	11.41

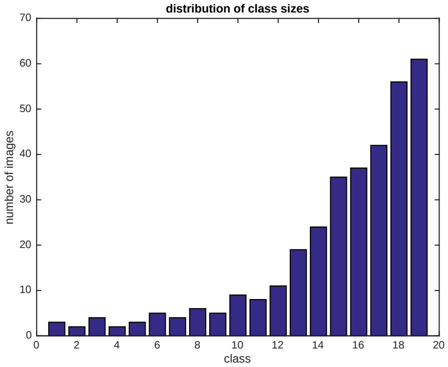

Figure 3: Distribution of class sizes for one of the random instances of the COIL 20 experiments.

cian) with k-means 100 times. For label propagation we choose 100 times a different subset of 50% known labels. We set a baseline by using the same techniques with a k-Nearest neighbors graph (k-NN) with different choices of k.

In Fig. 4 we plot the behavior of different models for different density levels. The horizontal axis is the average number of non-zero edges per node.

The dashed lines of the middle plot denote the number of nodes contained in components without labeled nodes, that can not be classified.

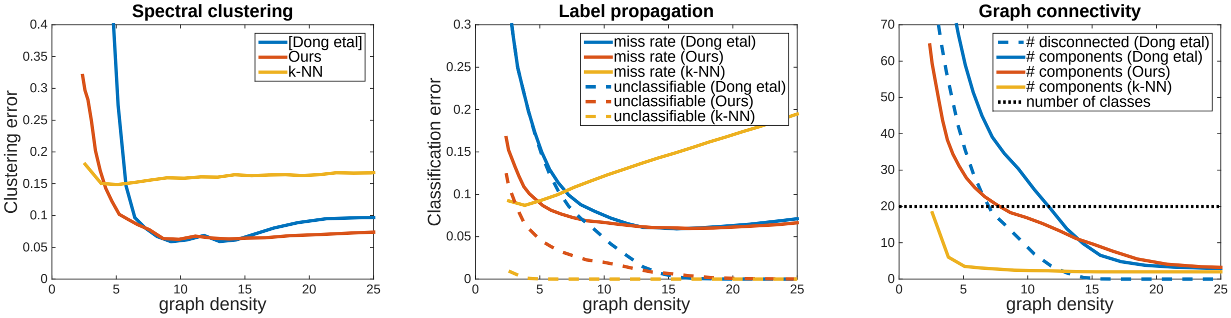

Figure 4: Graph learned from non-uniformly sampled images from COIL 20. Average over 20 different samples from the same non-uniform distribution of images. **Left**: Clustering quality. **Middle**: Label propagation quality. Dashed lines are the number of nodes in components without labeled nodes. **Right**: Number of disconnected components and number of disconnected nodes (Our model and k-NN have no disconnected nodes).