Fast Algorithms for Hypergraph PageRank with Applications to Semi-Supervised Learning

Konstantinos Ameranis [1] **Adela DePavia** [2] **Lorenzo Orecchia** [1] **Erasmo Tani** [1]

Abstract

A fundamental approach to semi-supervised learning is to leverage the structure of the sample space to diffuse label information from annotated examples to unlabeled points. Traditional methods model the input data points as a graph and rely on fast algorithms for solving Laplacian systems of equations, such as those defining PageRank. However, previous work has demonstrated that graph-based models fail to capture higher-order relations, such as group membership, which are better modeled by hypergraphs. Unfortunately, the scalable application of hypergraph models has been hampered by the non-linearity of the hypergraph Laplacian. In this paper, we present highly scalable algorithms for hypergraph primitives, such as hypergraph PageRank vectors and hypergraph Laplacian systems, over general families of hypergraphs. In addition to giving strong theoretical guarantees, we empirically showcase the speed of our algorithms on benchmark instances of semi-supervised learning on categorical data. We exploit their generality to improve semi-supervised manifold clustering via hypergraph models. By providing significant speed-ups on fundamental hypergraph tasks, our algorithms enable the deployment of hypergraph models on a massive scale.

1. Introduction

Many semi-supervised learning problems require learning an approximately continuous function $f : \mathcal{X} \to \mathbb{R}$ over a structured set $\mathcal{X}$, e.g., a network, embedded manifold or metric space, based on observations of potentially noisy function values $f(x) + \epsilon$ on a subset of labelled points $\mathcal{Y} \subset \mathcal{X}$ (van Engelen and Hoos, 2020). Examples of general problems in this family include learning smooth functions (Zhou et al., 2003; Belkin and Niyogi, 2004), learning Lipschitz functions (Kyng et al., 2015; Bungert et al., 2023) and clustering, i.e., learning functions with small total variation (Jung et al., 2019; Trillos and Slepčev, 2016).

If the structure of $\mathcal{X}$ can be encoded well by pairwise relationships, graph-based learning is the standard approach to this problem. Examples include $\mathcal{X}$ as an embedded manifold, which can be approximated by a properly constructed graph over a finite sample of points (Belkin and Niyogi, 2008; Calder and García Trillos, 2022), or a finite metric space, i.e., a length-weighted undirected graph. In this setting, given a weighted graph $G = (V, E, \mathbf{w} \in R_+^E)$, a set $Y \subset V$ of labelled vertices and their values $\mathbf{y} \in \mathbb{R}^Y$[1], the predictor $\hat{\mathbf{x}}$ is typically obtained by minimizing the regularized squared error

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{x} \in \mathbb{R}^V} \frac{1}{2} \cdot \|\mathbf{x} - \mathbf{y}\|_2^2 + \frac{1}{\lambda} \cdot \mathcal{R}(\mathbf{x}), \qquad (1)$$

where $\mathcal{R}$ is a regularizer appropriately penalizing discontinuous predictors over G, and $\lambda > 0$ is an inverse regularization parameter. The most common choice of regularizer $\mathcal{R}$ is the Dirichlet energy $\mathcal{E}_G : \mathbb{R}^V \to \mathbb{R}$ of G, which is the quadratic form of the graph Laplacian operator $\mathbf{L}_G$:

$$\mathcal{E}_G(\mathbf{x}) = \frac{1}{2} \sum_{\{i,j\} \in E} \mathbf{w}_{ij}(\mathbf{x}_i - \mathbf{x}_j)^2 = \frac{1}{2}\mathbf{x}^T \mathbf{L}_G \mathbf{x}.$$

This formulation appeals both because it captures a natural notion of smoothness in terms of Fourier coefficients (Smola and Kondor, 2003) and because the computation of $\hat{\mathbf{x}}$ reduces to the solution of the Laplacian system of linear equations $(\lambda \cdot \mathbf{I} + \mathbf{L}_G)\hat{\mathbf{x}} = \lambda \mathbf{y}$, for which nearly-linear-time algorithms exist (Spielman and Teng, 2004; Gao et al., 2023).

If the Euclidean norm in the error term of Problem 1 is replaced with the norm $\|\cdot\|_{\mathbf{D}}$, where $\mathbf{D}$ is the diagonal matrix of weighted degrees of G, we obtain the Laplacian system $(\lambda \cdot \mathbf{D} + \mathbf{L}_G)\hat{\mathbf{x}} = \lambda \mathbf{y}$. For $\lambda = {}^{2\alpha}/_{1-\alpha}$, the solution of this system is exactly a scaling of the personalized PageRank vector $\mathbf{ppr}_\alpha(\mathbf{y})$ with teleportation parameter $\alpha \in [0, 1]$ and seed $\mathbf{y} \in \mathbb{R}^V$ (Fountoulakis et al., 2019). This vector plays an important role in data mining applications, such as edge

[1]Department of Computer Science, University of Chicago, Chicago, USA [2]Computational and Applied Mathematics, University of Chicago, Chicago, USA. Correspondence to: Lorenzo Orecchia <orecchia@uchicago.edu>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024.

[1]We assume $\mathbf{y}_i = 0$ for $i \notin Y$.

prediction (Backstrom and Leskovec, 2011) and local graph partitioning (Andersen et al., 2006).

In this work, we develop fast algorithms for the hypergraph generalization of Laplacian systems, which allow us to deploy hypergraph regularizers in Problem 1 and to compute hypergraph personalized PageRank vectors (Takai et al., 2020) on large-scale hypergraphs.

From Graphs to Hypergraphs Increasing complexity in real-world data has led to the need to model higher order relationships (Benson et al., 2016; Tsourakakis et al., 2017), such as co-authorship and community membership in social networks (Orecchia et al., 2022), categorical data labels in learning applications (Amburg et al., 2020; Gibson et al., 2000) and gene expression patterns in biological systems (Feng et al., 2021). To address this need, research has shifted to models based on the notion of a hypergraph, a generalization $G = (V, H \subseteq 2^V, \mathbf{w} \in \mathbb{R}_+^H)$ of a graph in which each hyperedge $h \subset H$ is a subset of vertices of arbitrary cardinality (Çatalyürek et al., 2022). Early work in this area showed that such models are strictly more expressive than those based on graphs (Agarwal et al., 2006; Ihler et al., 1993), engendering a strong interest in generalizing the Laplacian regularization paradigm to hypergraph models (Hein et al., 2013). In contrast to graphs, hypergraphs admit many natural choices of Dirichlet-like energy functionals, as there are many ways of measuring the variance of a function $\mathbf{x} \in \mathbb{R}^V$ over a hyperedge $h \subseteq 2^V$.

The most common choice of hypergraph potential $\mathcal{U}_\infty$ measures the hyperedge variance by the squared ℓ_∞-norm:

$$\mathcal{U}_\infty(\mathbf{x}) \stackrel{\text{def}}{=} \frac{1}{2}\sum_{h\in H} \mathbf{w}_h \frac{\max_{i,j\in h}(\mathbf{x}(i)-\mathbf{x}(j))^2}{2} \quad (2)$$
$$= \frac{1}{2}\sum_{h\in H} \mathbf{w}_h \min_{u\in\mathbb{R}} \|\mathbf{x}_h - u\vec{\mathbf{1}}_h\|_\infty^2.,$$

where $\mathbf{x}_h \in \mathbb{R}^h$ is the restriction of the vector $\mathbf{x} \in \mathbb{R}^V$ to the entries $i \in h$. In the first study applying hypergraphs to semi-supervised learning and clustering of categorical data, Hein et al. (2013) showed that this hypergraph potential led to superior performance over graph-based approaches. The same potential has also been intensely studied in the field of approximation algorithms, where it is used to derive Cheeger-like inequalities and algorithmic results for natural hypergraph partitioning problems (Louis, 2015; Chan et al., 2018; Lau et al., 2023b). Based on this connection to spectral theory, Yoshida (2019) and Li and Milenkovic (2018) have put forward a larger class of hypergraph potential functions for which it is possible to derive Cheeger-like inequalities. These general hypergraph potentials are derived from submodular hyperedge cut functions and can be used to model directed (Lau et al., 2023b) and mutual information clustering (Yoshida, 2019) (Section 2).

In this paper, we consider a large class of hypergraph potentials of the form

$$\mathcal{U}(\mathbf{x}) \stackrel{\text{def}}{=} \frac{1}{2}\sum_{h\in E} \mathbf{w}_h \min_{u\in\mathbb{R}} \mathcal{Q}_h(\mathbf{x}_h - u\vec{\mathbf{1}}_h), \quad (3)$$

where each hyperedge potential $\mathcal{Q}_h : \mathbb{R}^h \to \mathbb{R}$ is a convex, 2-positively homogeneous function[2] measuring the local variance of $\mathbf{x}$ with respect to hyperedge h. This very expressive class of potentials includes potentials discussed in previous work and allows the incorporation of a notion of directedness to each hyperedge. We discuss the role of directnedness and give examples later in the section on our contributions.

The Problem: Hypergraph Laplacian Systems Problems of the form of Equation 1 are naturally generalized to hypergraphs by substituting the graph Dirichlet energy $\mathcal{E}_G$ with a hypergraph potential $\mathcal{U}(\mathbf{x})$. If we use the diagonal positive matrix $\mathbf{M} \in \mathbb{R}^{V\times V}$ to measure the mean squared error, e.g., $\mathbf{M} = \mathbf{I}$ or $\mathbf{M} = \mathbf{D}$, the corresponding optimality condition takes the following form, for given inputs $\lambda \geq 0$ and $\mathbf{s} = \lambda\mathbf{y}$:

$$\mathbf{s} \in \lambda\mathbf{M}\mathbf{x} + \mathcal{L}(\mathbf{x}), \quad (4)$$

where we denote by $\mathcal{L}(\mathbf{x})$ the subdifferential $\partial\mathcal{U}(\mathbf{x})$ of the hypergraph potential $\mathcal{U}$ at $\mathbf{x}$. The problem of finding $\mathbf{x}$ satisfying the inclusion is known as a hypergraph Laplacian system (Fujii et al., 2021) and *is the main focus of our paper*. By analogy with the graph case, the set-valued operator $\mathcal{L}$ is known as the hypergraph Laplacian operator (Louis, 2015). Indeed, when each hyperedge $h = \{i, j\}$ has cardinality 2 and $\mathbf{x}$'s variance over it is measured by a squared norm, the operator $\mathcal{L}$ reduces to the linear graph Laplacian $\mathbf{L}_G$.

We follow the conventions used for linear systems (Vishnoi et al., 2013) and consider the the following variational formulation of the hypergraph Laplacian system 4:

$$G(\mathbf{x}) \stackrel{\text{def}}{=} \mathcal{U}(\mathbf{x}) + \frac{\lambda}{2}\|\mathbf{x}\|_{\mathbf{M}}^2 - \langle \mathbf{s}, \mathbf{x}\rangle, \ \text{OPT} \stackrel{\text{def}}{=} \min_{\mathbf{x}} G(\mathbf{x}) \quad (5)$$

As we can only solve Problem 4 approximately, we need to introduce an appropriate error measure. To this end, we say that a candidate solution $\mathbf{x}$ is an ϵ-approximate solution to Problem 4 if $\mathbf{x}$ yields an ϵ-multiplicative approximation to OPT, i.e., $|G(\mathbf{x})-\text{OPT}|/|\text{OPT}| \leq \epsilon$. The reason for choosing this multiplicative notion of error, rather than additive error in Problem 1, stems from the homogeneity of the hypergraph Laplacian systems of Problem 4. As any scaling of $\mathbf{s}$ is just reflected in an equal scaling of the optimal solution $\mathbf{x}$, it makes sense to consider error measures that do not depend on the absolute scale of the input $\mathbf{s}$.

[2] A function $g : \mathbb{R}^n \to \mathbb{R}^k$ is p-positively homogeneous if for all $\alpha > 0$ and all $\mathbf{x} \in \mathbb{R}^n$, we have $g(\alpha\mathbf{x}) = \alpha^p g(\mathbf{x})$.

Previous Work The growing importance and promise of hypergraph-based methods call for fast, practical algorithms for Problem 4, particularly for general hypergraph potential functions. However, development of such algorithms has so far been limited. Early work by Hein et al. (2013) proposes a primal-dual hybrid gradient algorithm (Chambolle and Pock, 2011) for the $\mathcal{U}_\infty$ hypergraph potential, but does not provide a convergence analysis. Li et al. (2020) deploys techniques from decomposable submodular minimization to obtain very precise approximations, i.e., with a runtime dependence of $\log(1/\epsilon)$, for hypergraph potentials derived from submodular functions (see Section 2), such as $\mathcal{U}_\infty$, but requires super-quadratic time in the size of the hypergraph, making it impractical for large datasets. Other works obtain even higher polynomial dependencies by relying on generic convex solvers (Takai et al., 2020).

Our contributions In this paper, we describe and analyze a fast algorithm for computing ϵ-approximate solutions to the hypergraph Laplacian systems of Equation 4. Despite the non-linearity of the hypergraph Laplacian, our algorithm takes a surprisingly similar form and shares favorable properties with the classical gradient descent algorithm used for solving graph Laplacian linear systems, i.e., the Richardson iteration (Spielman, 2018).

Similarly to its graph counterpart, the running time of our algorithm depends on a notion of condition number of the instance hypergraph G. For $\mathbf{M} = \mathbf{D}$, this condition number is the Poincaré constant λ_G of G with respect to $\mathbf{D}$, a direct generalization (up to constants) of the graph spectral gap:

$$\lambda_G \stackrel{\text{def}}{=} \min_{\mathbf{x} \perp \mathbf{D}\vec{\mathbf{1}}} \frac{\mathcal{U}(\mathbf{x})}{\frac{1}{2}\|\mathbf{x}\|_{\mathbf{D}}^2}, \tag{6}$$

We show in Theorem 3.1 in Section 3 that, when $\mathbf{M} = \mathbf{D}$, our algorithm runs in linear time in the size of the hypergraph, with an inverse dependence on λ_G. The only discrepancy with the graph case is the dependence on the error parameter ϵ. Whereas it is $\log(1/\epsilon)$ for gradient descent on graph Laplacian systems, it is $O(1/\epsilon^2)$ for our algorithm due to the non-smoothness of hypergraph potentials.

Like gradient descent, our algorithm can be applied in a preconditioned fashion to improve results for specific classes of hypergraph potentials. As an example, in Theorem 3.2, we use a graph Laplacian preconditioner based on the clique expansion of the hypergraph to obtain a faster algorithm for the $\mathcal{U}_\infty$ hypergraph potential. The resulting running time *does not depend on* λ_G, but requires an additional multiplicative factor of $\max_{h \in H} |h|$, the rank of the hypergraph. For bounded-rank hypergraphs, this is the first nearly-linear-time algorithm for hypergraph Laplacian systems, matching the seminal result of Spielman and Teng (Spielman and Teng, 2004) for graph Laplacian systems.

A significant advantage of our algorithm is its applicability, without any change, to all hypergraph potentials in the general family described in Equation 3. By comparison, the method of Hein et al. (2013), the only viable competitor to our algorithm on large datasets, is closely tailored to the $\mathcal{U}_\infty$ hypergraph potential. In Section 5, we exploit the flexibility of our method to investigate the performance of different hypergraph potentials in semi-supervised manifold clustering, leading to formulations that greatly outperform graph-based methods as the dimension of the manifold increases.

The generalization to convex 2-positively homogenous hypergraph potentials formulated in Equation 3 also allows us to compute hypergraph PageRank vectors for directed models. A particularly simple example is that of the directed potential over graphs

$$\mathcal{U}^{\text{directed}}(\mathbf{x}) = \frac{1}{2} \sum_{(i,j) \in E} w_{ij} (\mathbf{x}_i - \mathbf{x}_j)_+^2,$$

which fits the requirements of our work and is also discussed in Lau et al. (2023a). Even for this simple graph potential, no simple PageRank algorithm was known prior to our work. Following work of Liu et al. (2021b), our algorithm then enables local partitioning of directed graphs under the conductance objective. Other directed hypergraph potentials of interest in the setting of partitioning are described by Yoshida (2018).

To verify the expected performance of our algorithm, we empirically compare it with previous methods on a benchmark dataset for hypergraph-based semi-supervised learning (Section 6). The results show a tenfold speed-up over competitors, with comparable accuracy.

Paper organization All references to sections with alphabetic indexes refer to the Supplementary Material. All proofs of claimed statements are included in the Supplementary Material. For mathematical notation, definitions and background, see Section A.

2. General Hypergraph Potentials

We will denote a hypergraph by a tuple $G = (V, H \subseteq 2^V, \mathbf{w} \in \mathbb{R}^H_{>0})$, where V is a set of vertices, H a set of hyperedges, each with a positive weight w_h. The degree $\deg(i) \stackrel{\text{def}}{=} \sum_{h \ni i} w_h$ of a vertex $i \in V$ is the sum of the weights of hyperedges incident to i. We denote by $\mathbf{D} \in \mathbb{R}^{V \times V}$ the diagonal matrix of degrees of G.

In this section, we establish key properties of the hypergraph potentials in Equation 3. For the rest of this section we assume familiarity with properties of submodular and convex functions: for further background on these topics, see Section A.

Convex, 2-Positively Homogeneous Functions The class of 2-positively homogeneous function is very expressive. These functions generalize squared norms while retaining many of their favorable properties. We now state some simple facts that will be useful in the following sections. Their proof can be found in Appendix B.

Lemma 2.1. *Let $F : \mathbb{R}^n \to R$ be a convex 2-positively homogeneous function. Then:*

1. *(Non-negativity) F is everywhere non-negative,*
2. *(Non-negative Fenchel dual) The Fenchel dual F^* of F is everywhere non-negative,*
3. *(Generalized Euler's Identity (Yang and Wei, 2008)) For all $\mathbf{x} \in \mathbb{R}^n$ and $\mathbf{z} \in \partial F(\mathbf{x})$, it holds that*
$$\langle \mathbf{x}, \mathbf{z} \rangle = 2F(\mathbf{x}),$$
4. *(Homogeneous subdifferential) The subdifferential $\partial F(\cdot)$ is 1-positively homogeneous.*

Hypergraph Potentials from Submodular Cut Functions Yoshida (2018) and Li and Milenkovic (2018) study a large class of hypergraph partitioning problems , which generalize minimum conductance problem over graphs. They are based on submodular hyperedge cut functions $\delta_h : 2^h \to [0, 1]$ that satisfy $\delta_h(\emptyset) = 0$ and $\delta_h(h) = 0$. Their framework encompasses many cut functions that are prominent in applications, including cardinality-based cut functions (Veldt et al., 2020) and the mutual information cut function (Gretton et al., 2003; Xiong et al., 2005).

For any choice of submodular hyperedge cut functions δ_h, it is possible to construct an associated hypergraph potential by considering the weighted sum of their squared *Lovász extensions* $\bar{\delta}_h : \mathbb{R}^h \to \mathbb{R}$. Yoshida (2019) showed that such hypergraph potentials are of crucial importantce for hypergraph partitioning tasks, as they obey a hypergraph analogue of Cheeger inequality. The following lemma, which is proved in Section B.1 of the supplementary material, implies that the class described by Equation 3 includes all potentials arising from hyperedge submodular cut functions.

Lemma 2.2. *For any choice of submodular cut functions $\delta_h : 2^h \to [0, 1], \delta(\emptyset) = \delta(h) = 0$, the hypergraph potential*
$$\mathcal{U}(\mathbf{x}) = \frac{1}{2} \sum_{h \in H} \mathbf{w}_h \cdot (\bar{\delta}_h(\mathbf{x}))^2$$
takes the form of Equation 3. In particular, it is convex and 2-positively homogeneous.

Norms bounds We make the following normalization assumption. Notice that this causes no loss in generality as any additional scaling can be incorporated into the hypergraph weight w_h.

Assumption 2.3. $\mathcal{Q}_h(\mathbf{x}) \leq \|\mathbf{x}\|_2^2$ for every $h \in H, \mathbf{x} \in \mathbb{R}^V$.

With this assumption, we can show the following bounds relating $\mathcal{U}(\mathbf{x})$ with the squared degree norm. The proof appears in Section B of the Supplementary Material.

Lemma 2.4. *For every connected hypergraph G, $\lambda_G > 0$. For any hypergraph and any choice of hyperedge potentials $\{\mathcal{Q}_h\}_{h \in H}$, for all $\mathbf{x} \perp_{\mathbf{D}} \vec{\mathbf{1}}$ the potential energy functional $\mathcal{U}$ satisfies:*
$$\lambda_G \leq \frac{\mathcal{U}(\mathbf{x})}{\frac{1}{2}\|\mathbf{x}\|_{\mathbf{D}}^2} \leq 1. \tag{7}$$

3. Hypergraph Laplacian Systems

Our main algorithm, Algorithm 1, is a first-order method for optimizing problems of the form $\min_{\mathbf{x}} F(\mathbf{x}) - \langle \mathbf{x}, \mathbf{s} \rangle$, when F is convex and 2-positively homogeneous. Its theoretical guarantees are proved in Section 4. In this section, we describe the application of Algorithm 1 to the solution of hypergraph Laplacian systems, as defined in Equation 5,and compare it to the classical application of gradient descent to graph Laplacian systems.

For a Laplacian system $(\lambda\mathbf{D} + \mathbf{L})\mathbf{x} = \mathbf{s}$, gradient descent with respect to the $\|\cdot\|_{\mathbf{D}}$ norm, also known as the Richardson iteration, performs the following update:
$$\mathbf{x}_{t+1} = \mathbf{x}_t - \frac{1}{2} \cdot \mathbf{D}^{-1}(\lambda\mathbf{D}\mathbf{x}_t + \mathbf{L}\mathbf{x}_t - \mathbf{s}),$$
which is a step in the direction of steepest descent of the objective in Equation 5 with step length $1/2$. For hypergraph Laplacian systems when $\mathbf{M} = \mathbf{D}$, we apply Algorithm 1 with $F(\mathbf{x}) = \mathcal{U}(x) + \lambda/2 \cdot \|x\|_{\mathbf{D}}^2$ and $\mathbf{R} = \mathbf{D}$. Because $u_{\mathbf{R}}$ is equal to 1 by Lemma 2.4, the resulting update step is:
$$\hat{\mathbf{x}}_{t+1} = \hat{\mathbf{x}}_t - \frac{\epsilon}{2} \cdot \mathbf{D}^{-1}(\lambda\mathbf{D}\mathbf{x}_t + \mathcal{L}(\mathbf{x}_t) - \mathbf{s}),$$
where $\mathcal{L}(\mathbf{x}_t)$ denotes an arbitrary element of the subdifferential $\partial\mathcal{U}(\mathbf{x}_t)$. This is exactly the same as the gradient descent step above, with a shorter step length of ϵ.

We can now state our first result on solving hypergraph Laplacian systems with $\mathbf{M} = \mathbf{D}$. Its proof, which appears in Section B.2, relies on the analysis of Algorithm 1 in the next section.

Theorem 3.1. *For a connected hypergraph with potential $\mathcal{U}(\mathbf{x})$, Algorithm 1 with prox-generating function $1/2 \cdot \|\cdot\|_{\mathbf{D}}^2$ computes a multiplicative ϵ-approximation to the optimum of Problem 5 with $\mathbf{M} = \mathbf{D}$ in $O\left(\frac{1+\lambda}{(\lambda_G+\lambda)\epsilon^2}\right)$ iterations. Each iteration requires linear time in the size of the hypergraph and a single subgradient computation.*

A similar statement can be obtained for any diagonal $\mathbf{M}$ by applying Theorem 4.1 and considering the Poincaré constant of G with respect to $\mathbf{M}$.

At this time, it is imperative to remark that, to the best of our knowledge, no previously known first-order method can give a guarantee comparable to that of Theorem 3.1, as standard applications of subgradient descent algorithms (Nemirovskij and Yudin, 1983; Nesterov, 2014) fail because the non-differentiable quadratic objective in Equation 5 is both non-smooth and non-Lipschitz. Indeed, our analysis requires a careful modification of subgradient descent to control the quadratic error terms and obtain a multiplicative error guarantee in place of the standard additive guarantee. See Subsection 4.1 for more details.

Compared to the graph case, our running time suffers from a polynomial rather than a logarithmic dependence on ϵ, which is a consequence of the non-smoothness of our potential $\mathcal{U}$. Just as in the case of graphs, the convergence of our algorithm is inversely proportional to λ_G, i.e., slowing down when this analogue of spectral gap is small and the hypergraph is more poorly connected.

In many applications, such as semi-supervised learning and ranking, the value of λ is typically taken to be a constant, so that the above running time can be assumed to be independent of λ_G. However, in some algorithmic settings, such as when using PageRank for partitioning (Takai et al., 2020), one may want to choose $\lambda \approx \lambda_G$, causing the above running time to grow large for small values of λ_G.

Hence, a natural question is whether one can obtain hypergraph analogues of the fast Laplacian solver of Spielman and Teng (2004) running in nearly-linear-time in the size of the graph with no dependence on λ_G or any other graph parameters.

We answer this question positively for the hypergraph potential $\mathcal{U} = \mathcal{U}_\infty$ on bounded-rank hypergraphs, i.e., hypergraphs with hyperedge of bounded constant cardinality. For these hypergraphs, we obtain a λ_G-independent running time by selecting the preconditioner $\mathbf{R}$ for Algorithm 1 to be the Laplacian of the clique expansion of the instance hypergraph. This allows us to derive much sharper bounds for the quantities $u_{\mathbf{R}}$ and $\ell_{\mathbf{R}}$ in the analysis of Algorithm 1 and yields the following theorem, which is proven in Section B.2.

Theorem 3.2. *For any connected hypergraph $G = (V, H, w)$ with potential $\mathcal{U}_\infty(\mathbf{x})$, one can construct a graph with Laplacian matrix $\mathbf{L}$, such that Algorithm 1 with prox-generating function $1/2\|\cdot\|_{\mathbf{L}}^2$ computes a multiplicative ϵ-approximation to the optimum of Problem 5 in $O(\max_{h\in H}|h|/\epsilon^2)$ iterations. Each iteration requires solving a graph Laplacian system in $\mathbf{L}$.*

Note that this method incurs a factor of $\max_{h\in H}|h|$ (the *rank* of G) in its running time, and hence it may not be suited for hypergraphs with large hyperedges.

Algorithm 1 with prox-generating function $1/2 \cdot \|\cdot\|_{\mathbf{R}}^2$ for $\min F(\mathbf{x}) - \langle \mathbf{s}, \mathbf{x}\rangle$ in Theorem 4.1

Input: Oracle for arbitrary $\mathbf{z} \in \partial F(\mathbf{x})$ on input $\mathbf{x}$; $T \in \mathbb{N}$; $\epsilon \in (0,1)$.
$\eta = \epsilon/2u_{\mathbf{R}}$ ▷ Step-size
$\hat{\mathbf{x}}_0 = \eta\mathbf{R}^{-1}\mathbf{s}$ ▷ Initialization
for $t = 0, ..., T-1$ **do**
  $\hat{\mathbf{x}}_{t+1} \leftarrow \hat{\mathbf{x}}_t - \eta\mathbf{R}^{-1}(\mathbf{z}_t - \mathbf{s})$ with $\mathbf{z}_t \in \partial F(\hat{\mathbf{x}}_t)$
end for
$\bar{\mathbf{x}}_T = \frac{1}{T}\sum_{t=0}^{T-1}\hat{\mathbf{x}}_t$
return $\mathbf{x}_T^{\text{out}} = (1 - \epsilon/2)\cdot\bar{\mathbf{x}}_T$

4. First-order methods for convex 2-positively homogenous functions

In this section, we describe and analyze Algorithm 1, a first-order method for the solution of optimization problems of the form

$$\text{OPT}_{F,\mathbf{s}} = \min_{\mathbf{x}\in\mathbb{R}^n} F(\mathbf{x}) - \langle \mathbf{s}, \mathbf{x}\rangle\,, \tag{8}$$

where $\mathbf{s} \in \mathbb{R}^n$ and $F : \mathbb{R}^n \to \mathbb{R}$ is a convex, 2-positively homogeneous function. It follows from Lemma 2.1 and the properties reviewed in Section A that $\text{OPT}_{F,\mathbf{s}} \stackrel{\text{def}}{=} -F^*(\mathbf{s})$ is always non-positive.

As F is potentially non-differentiable, we only assume access to an oracle that, on input $\mathbf{x} \in \mathbb{R}^n$ returns an arbitrary vector in the subdifferential $\partial F(\mathbf{x})$. Algorithm 1 also requires a choice of prox-generating function, i.e. a preconditioner, in the form the square of a Mahalanobis norm $\|\cdot\|_{\mathbf{R}}$. The computational cost to compute a prox step with respect to preconditioner $\|\cdot\|_{\mathbf{R}}$ is the time to solve a system of linear equations in $\mathbf{R}$.

Under these assumptions, we show that Algorithm 1 converges to a multiplicative approximation of $\text{OPT}_{F,\mathbf{s}}$ at a rate that only depends on the Poincaré constants $\ell_{\mathbf{R}}, u_{\mathbf{R}}$, i.e the optimal constants satisfying:

$$\forall\,\mathbf{x} \in \mathbb{R}^n \setminus \{0\} \;:\; \ell_{\mathbf{R}} \le \frac{F(\mathbf{x})}{\frac{1}{2}\|\mathbf{x}\|_{\mathbf{R}}^2} \le u_{\mathbf{R}}. \tag{9}$$

The following guarantee, our main optimization result, is proved in Section B.3 in the Supplementary Material. Notice that the absolute value appears in the error expression, because $\text{OPT}_{F,\mathbf{s}}$ is non-positive.

Theorem 4.1. *For any convex 2-positive homogeneous F satisfying Equation 9 with respect to some $\ell_{\mathbf{R}}, u_{\mathbf{R}} > 0$, error parameter $\epsilon > 0$, and choice of $T = O\left(\frac{u_{\mathbf{R}}}{\ell_{\mathbf{R}}\epsilon^2}\right)$, the point $\mathbf{x}_T^{out}$ produced by Algorithm 1 satisfies*

$$F(\mathbf{x}_T^{\text{out}}) - \left\langle \mathbf{s}, \mathbf{x}_T^{\text{out}}\right\rangle - \text{OPT}_{F,\mathbf{s}} \le \epsilon\cdot|\text{OPT}_{F,\mathbf{s}}|.$$

To emphasize the novelty of this result, we remark again that, to the best of our knowledge, the condition in Equation 9 cannot be exploited by any existing first-order methods for this problem. It is only equivalent to $\ell_{\mathbf{R}}$-smoothness and $u_{\mathbf{R}}$ strong convexity when F itself arises from a squared Mahalanobis norm $\|\cdot\|_{\mathbf{A}}^2$, which is not the case in our hypergraph applications.

4.1. Sketch of Proof of Theorem 4.1

The proof is based on a modified analysis of subgradient descent on the following auxiliary iterates:

$$\mathbf{x}_t \stackrel{\text{def}}{=} \hat{\mathbf{x}}_t - \eta \mathbf{R}^{-1}\mathbf{s}.$$

In particular, notice that $\mathbf{x}_0 = \mathbf{0}$. With this definition, the first step of the proof closely follows the standard analysis of optimistic mirror descent (Rakhlin and Sridharan, 2013), where the optimistic part of the step is performed, from $\mathbf{x}_t$ to $\hat{\mathbf{x}}_t$ with respect to fixed component $-\mathbf{s}$ of the subgradient of the objective function. We summarize this result in the following lemma, which is shown in Appendix B.

Lemma 4.2. *For all $t \in \mathbb{N}$,*

$$\begin{aligned}&(F(\hat{\mathbf{x}}_t) - \langle s, \hat{\mathbf{x}}_t\rangle) - \mathrm{OPT}_{F,\mathbf{s}} \\ &\quad\le \frac{1}{2\eta}\|\mathbf{x}_t - \mathbf{x}^*\|_{\mathbf{R}}^2 - \frac{1}{2\eta}\|\mathbf{x}_{t+1} - \mathbf{x}^*\|_{\mathbf{R}}^2 + \frac{\eta}{2}\|\mathbf{z}_t\|_{\mathbf{R}^{-1}}^2.\end{aligned}$$

At this stage, rather than summing this guarantee over all iterations, as done in the classical analysis of mirror descent, we bound the error term $\eta/2 \cdot \|\mathbf{z}_t\|_{\mathbf{R}^{-1}}^2$ directly in terms of the suboptimality gap of $\hat{\mathbf{x}}_t$. This technical step is a novel feature of our analysis and crucially exploits the 2-positive homogeneity of the objective function. Its proof is also found in Appendix B.

Corollary 4.3. *Under the assumptions of Theorem 4.1:*

$$\frac{1}{2}\|\mathbf{z}_t\|_{\mathbf{R}^{-1}}^2 \le u_{\mathbf{R}} \cdot F(\hat{\mathbf{x}}_t) \le \frac{\epsilon}{2\eta} \cdot F(\hat{\mathbf{x}}_t)$$

After combining the results of Corollary 4.3 and Lemma 4.2 we can finally sum over all $t \in \{0, 1, \ldots, T-1\}$ while telescoping terms to find:

$$\begin{aligned}&\sum_{t=0}^{T-1}(1 - \epsilon/2) \cdot F(\hat{\mathbf{x}}_t) - \langle s, \hat{\mathbf{x}}_t\rangle \\ &\qquad\le T \cdot \mathrm{OPT}_{F,\mathbf{s}} + \frac{1}{2\eta}\|\mathbf{x}_0 - \mathbf{x}^*\|_{\mathbf{R}}^2\end{aligned}$$

The rest of the proof is a short sequence of simple calculations, which are included in Section B.

5. Manifold Learning via Hypergraph Laplacians

Manifold learning problems include identification and recovery of some lower-dimensional parametrization of datapoints embedded in some higher-dimensional space. Many approaches utilize spectral methods on graphs that are constructed from the geometric embedding of datapoints (Belkin and Niyogi, 2001; 2003; Yan et al., 2006; Talmon et al., 2013). These methods are theoretically justified by results showing the asymptotic convergence of the graph Laplacian to the Laplace-Beltrami operator of the underlying manifold (Belkin and Niyogi, 2008; Trillos and Slepčev, 2016). In this section, we initiate a methodological study of the potential advantages of using hypergraph models in the context of manifold learning. In particular, we consider semi-supervised community detection tasks in the presence of manifold structure. We crucially exploit the generality of our algorithm to explore novel hypergraph potentials studied in this work to propose a novel weighted hypergraph potential, discussed in Section 5.1. We present a high-level description of our experiments and findings, and refer readers to Section C for full details, methodology, and further figures.

Consider a point cloud of n data points in d dimensions. We represent these data points with vectors $\{\mathbf{p}_i \in \mathbb{R}^d\}_{i=1}^n$. We assume that each of these points are sampled uniformly from one of two manifolds and subject to additive Gaussian noise. In addition, we assume we have access to labels on a small subset of these datapoints, indicating which manifold each datapoint was drawn from. The goal is to use these seeded labels to estimate which manifold other (unlabeled) points belong to. We give a formal description of this problem setup in Section C.

In these experiments we use our algorithmic implementation of hypergraph Personalized PageRank (Problem 4 with $\mathbf{M} = \mathbf{D}$) to propagate this partial label information to unlabeled nodes. To form a (hyper)graph corresponding to data points $\{\mathbf{p}_i \in \mathbb{R}^d\}_{i=1}^n$, we first initialize one node v_i for each datapoint $\mathbf{p}_i$. We then add (hyper)edges connecting nodes whose data points are close in $\mathbb{R}^d$. In this work, we specifically consider k-nearest neighbor (hyper)edges endowed with a weighted version of the ℓ_∞-hypergraph potential $\mathcal{U}_\infty$. We motivate this weighted potential further in Section 5.1 below. A full description of these constructions is included in Section C.

We estimate labels at each node by computing a personalized PageRank vector, where the seed vector $\mathbf{s}$ is constructed using the set of partial revealed labels. In Figure 1, we compare the performance of the resultant graph- versus hypergraph-PPR estimates on different datasets. In order to assess the performance of the estimates as binary classifiers, we report the area under the curve (AUC) value of the

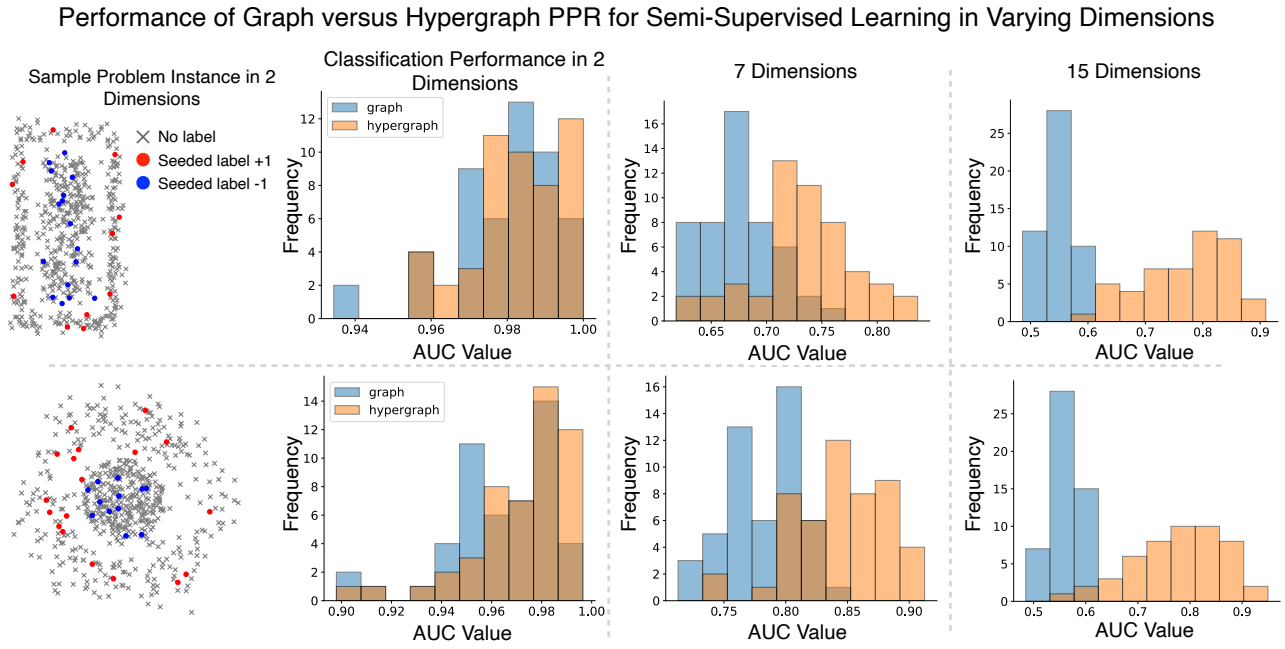


Figure 1: Graph- versus hypergraph-based approaches to semi-supervised learning of manifolds. Top row: results with concentric hyperrectangles. Bottom row: results with concentric hyperspheres. These manifolds are sampled with noise, and partial labels are revealed. A sample instance in two dimensions of the noisily sampled point cloud and partially revealed labels is plotted on the left. A nearest-neighbors graph and hypergraph are formed based on the sampled data. The graph and hypergraph are endowed with weighted potentials, as described in Section 5.1. The personalized PageRank vector seeded at the vector of partial-labels is used to estimate community membership at each sampled point. AUC values are reported over 50 independent trials. Hypergraph PPR clustering exhibits an advantage over the graph method, and this gap widens with increasing dimension.

corresponding receiver operator curve. Values closer to 1 indicate perfect separation between the two communities, while an AUC value of 0.5 corresponds to no-better-than-random performance. We sample data from two manifold learning settings: one in which the two manifolds correspond to concentric hyperspheres, and one in which the two manifolds correspond to concentric hyperrectangles. In particular, these datasets allow us to examine how the estimate performance varies with the ambient dimension of the underlying manifolds.

By comparing AUC values over 50 independent trials, we find that the estimates produced by the hypergraph PPR out-perform estimates produced by graph PPR in high-dimensional settings. In particular the gap between the graph- and hypergraph-estimates is robust and increases with ambient dimension. The separation in increasing dimensions is particularly interesting because well-known results establish that semi-supervised learning with graph Laplacians suffers from the curse of dimensionality (Nadler et al., 2009; Cabannes et al., 2021). While modified schemes for overcoming these issues have been proposed (Calder and Slepčev, 2020; Cabannes et al., 2021), the comparative success of the hypergraph Laplacian for semi-supervised learning in these high dimensional problems raises interesting questions for future work about the convergence of graph versus hypergraph Laplacians in high dimensional settings. Partial theoretical justification in favor of ℓ_∞-like hypergraph potentials can be found in classical metric embedding results that prove that any metric can be isometrically embedded into L_∞, while L_2 metrics are the least expressive L_p metrics (Matousek, 2013).

5.1. Gaussian-kernel Hypergraph Potentials

This work developed algorithms for an expressive class of hypergraph potentials, as captured by Equation 3. In this section, we leverage this generality to propose novel weighting schemes for hypergraph potentials. These weighting schemes incorporate geometric information from the sampled point cloud, in analogy to well-studied graph weighting schemes which enjoy known convergence to continuous manifold operators (Belkin and Niyogi, 2008; Trillos and Slepčev, 2016)

For similar semi-supervised tasks on graphs, a weighted

Laplacian is often constructed, where edge-weights w_{ij} scale as

$$w_{ij}^{(p)} = \exp(-\|\mathbf{p}_i - \mathbf{p}_j\|_2^2/2) \tag{10}$$

Here $\mathbf{p}_i, \mathbf{p}_j \in \mathbb{R}^d$ denote points in the point-cloud associated with nodes i and j. This choice of weights causes the Laplacian potential to more heavily penalize discrepancies between points which are located close together in space. This intuition is strengthened by known theoretical results showing that under appropriate scaling, such choice of edge weights causes the graph Laplacian on sampled point clouds to converge to the manifold Laplace-Beltrami operator (Trillos and Slepčev, 2016; Belkin and Niyogi, 2008).

In our design of hypergraph potentials for manifold learning, we adopt the intuition above by similarly incorporating geometric information from the point cloud. For every point $\mathbf{p}_i \in \mathbb{R}^d$ in the point cloud, we initialize a node v_i in the hypergraph. We then construct *k-nearest neighbor* hyperedges: each hyperedge $h \in H$ is composed of a "central neighbor" v_{i_h} and its k-nearest neighbors, $v_{i_1}, \ldots, v_{i_k} \subset V$. For each hyperedge $h \in H$ we compute a *hyperedge centroid*, a point $\overline{\mathbf{p}}_h \in \mathbb{R}^d$ defined as

$$\overline{\mathbf{p}}_h = \frac{1}{|h|} \sum_{v_i \in h} \mathbf{p}_i.$$

For each hyperedge and every node $v_i \in h$ we then use this centroid to define the following node weights:

$$\forall v_i \in h, w_{i,h}^{(p)} = \exp(-\|\mathbf{p}_i - \overline{\mathbf{p}}_h\|_2^2/2)$$

where $\|\mathbf{p}_i - \overline{\mathbf{p}}_h\|_2$ measures the distance between the embedding of node v_i and the centroid of hyperedge h in embedding space. We then form weighted potential

$$\mathcal{U}^{(P)}(\mathbf{x}) = \frac{1}{2} \sum_{h \in E} \min_{u \in \mathbb{R}} \|\mathbf{W}_h^{(P)}(\mathbf{x}_h - u\mathbf{1})\|_\infty^2 \tag{11}$$

where

$$\mathbf{W}_h^{(P)} = \text{diag}(w_{i_h,h}^{(p)}, w_{i_1,h}^{(p)}, \ldots, w_{i_k,h}^{(p)})$$

where v_{i_h} is the central neighbor of hyperedge h and $v_{i_1}, \ldots, v_{i_k}$ are its k-nearest neighbors in embedding space.

By using this novel weighted potential, rather than simply assigning scalar hyperedge weights w_h as in Equation 2, we penalize hyperedges adaptively. In particular, the potential $\mathcal{U}^{(P)}$ measures not only the maximum discrepancy between values of $\mathbf{x}$ taken at nodes in a hyperedge, but how close the points maximizing this discrepancy are to the centroid of the relevant hyperedge in the point-cloud embedding. In Section C, we present preliminary evidence suggesting $\mathcal{U}^{(P)}$ provides an advantage over the unweighted $\mathcal{U}_\infty$ potential, summarized in Figure 2 of the Supplementary Material.

6. Categorical Clustering Experiments

In this section, we complement the strong theoretical guarantees given so far with an empirical comparison of the performance of our algorithm in Theorem 3.1 against those of previous methods by Hein et al. (2013) and Li et al. (2020). As a benchmark, we adopt the semi-supervised multi-class classification task on a small set of UCI datasets (Kelly et al.) which was considered in previous works on the topic (Zhou et al., 2003; Hein et al., 2013; Li et al., 2020). This setup is particularly suitable to hypergraph-based models because the data contains both numerical and categorical features, the latter of which should be better captured by hypergraphs. The details of each dataset and instructions for recreating them can be found in Appendix D.1.

In order to perform multi-class predictions, given a vector $\mathbf{y}$ of class-labeled points, we create a one-hot label encoding $\mathbf{y}_c$ for each target class c, defined as

$$\mathbf{y}_c(i) = \begin{cases} 1 & \mathbf{y}_i = c \\ -1 & \text{o.w.} \end{cases}$$

We then approximately solve Problem 1 with error measured in the degree norm $\|\cdot\|_{\mathbf{D}}$ to obtain $\mathbf{x}_c$ for each class c. Finally, we select the predicted label to be $\tilde{y} = \arg\max_c \mathbf{x}_c$.

We proceed to compare our implementation with the ones provided by Li et al. (2020) on their performance in the classification task, in their convergence to the minimizer of Problem 1 and in their running time. A short overview of each competing methods is provided in Appendix D.2. Due to lack of space, we present our detailed results in Appendix D.4. The full data used for the comparison and our implementation can be found in the supplementary material.

Results As the number of labeled points varies, all methods except for the Subgradient QDSFM in Li et al. (2020) consistently converge to the minimizer of Problem 1. No method enjoys a clear advantage for the classification task, as even suboptimal points for Problem 1 often yield good classification due to implicit regularization by the algorithm. Crucially, our algorithm achieves the same performance as its competitors while only requiring one tenth of the running time. This appears to be a consequence of the simplicity of each iteration, as our method has the second-best average time per iteration. The best time per iteration is achieved by the Subgradient QDSFM of Li et al. (2020), which, however, requires many more iterations and fails to converge to a minimizer in many instances.

7. Conclusion and Open Problems

We hope that our simple fast algorithms will enable the deployment of hypergraph models to many semi-supervised learning problems over networks and manifolds and the

exploration of richer, more expressive hypergraph potentials. As we look forward, a number of open questions remain.

Nearly-Linear-Time Algorithms In a seminal result, Spielman and Teng (2004) constructed nearly-linear-time algorithms for solving Laplacian systems of linear equations. Whether the same feat is possible for a non-trivial family of hypergraph Laplacian systems, such as those based on the $\mathcal{U}_\infty$ potential, is an intriguing theoretical question. Our result in Theorem 3.2 shows that the hard case consists of hypergraphs with large hyperedges.

Acceleration The algorithms presented in this paper provably converge to the optimal solution at a rate of $O(1/\varepsilon^2)$. However, for many structured convex non-smooth problems, it is possible to obtain a $O(1/\varepsilon)$-convergence via accelerated algorithms (Nesterov, 2005). A natural question is whether one can achieve this improved rate in our multiplicative error setting without worsening the dependency on the hypergraph size. A promising avenue in this sense is given by primal-dual hybrid gradient methods (Esser et al., 2010; Chambolle and Pock, 2011), which were already empirically tested by Hein et al. (2013) in the context of hypergraph Laplacian systems.

Algorithms for Massive Hypergraphs Spectral methods over graphs are a popular algorithm design framework for massive datasets, as they are easy to implement in distributed computational models (Andersen et al., 2006; Das Sarma et al., 2015). We believe it is a worthy challenge to derive similar algorithms for hypergraph Laplacian systems. The similarity of our method with existing graph algorithms and the partial progress of Liu et al. (2021a) for strongly local computation of directed graph models offer some hope in this direction.

Manifold Learning Our empirical results provide preliminary evidence suggesting hypergraphs better approximate manifold structure in high dimensions. As illustrated in Figure 2, the node weighting schemes enabled by our generalized choice of hypergraph potential in Equation (3) are central to achieving these results. These experiments raise the question of how recovery guarantees for hypergraph Laplacian methods scale with dimension on semi-supervised learning tasks, particularly compared to known recovery guarantees for graph Laplacian methods. In the graph setting, it has been shown that specific weighting schemes are necessary to achieve favorable recovery rates in semi-supervised tasks (Calder and Slepčev, 2020; Cabannes et al., 2021). One result of the formalism studied in this paper is that it admits dynamic weighting schemes which can capture geometric information about node embeddings, as discussed in Section 5.1. It would be interesting for future study to examine whether these novel weighting schemes, which generalize the appropriate weightings from the graph setting, can be used to achieve good recovery guarantees.

Closely related to the above discussion is the relationship between discretized hypergraph Laplacians and continuous manifold Laplacian operators. When nodes correspond to points sampled from a manifold, well-known results establish that in the large-sample limit, graph Laplacians with appropriate choice of weighting converge to the Laplace-Beltrami operator (Belkin and Niyogi, 2008; Trillos and Slepčev, 2016). It would be interesting to study whether generalizations of such weighting schemes, such as the weighted potential introduced in Equation (11), can yield convergence results for hypergraph Laplacians. Similar theoretical questions have been raised independently by Saito et al. (2018).

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

Acknowledgements

LO is supported by NSF CAREER 1943510. AD is supported by NSF DGE 2140001.

References

Mushroom Dataset. UCI Machine Learning Repository, 1987. DOI: https://doi.org/10.24432/C5959T.

Sameer Agarwal, Kristin Branson, and Serge Belongie. Higher order learning with graphs. In *Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, pages 17–24, New York, NY, USA, June 2006. Association for Computing Machinery.

Ilya Amburg, Nate Veldt, and Austin R. Benson. Clustering in graphs and hypergraphs with categorical edge labels. In *Proceedings of The Web Conference 2020*, pages 706–717, April 2020.

Reid Andersen, Fan Chung, and Kevin Lang. Local graph partitioning using pagerank vectors. In *2006 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS'06)*, pages 475–486. IEEE, 2006.

Francis Bach. Learning with Submodular Functions: A Convex Optimization Perspective, October 2013.

Lars Backstrom and Jure Leskovec. Supervised random walks: Predicting and recommending links in social networks. In *Proceedings of the Fourth ACM International*

Conference on Web Search and Data Mining, WSDM '11, pages 635–644, New York, NY, USA, February 2011. Association for Computing Machinery.

Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps and spectral techniques for embedding and clustering. *Advances in neural information processing systems*, 14, 2001.

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

Mikhail Belkin and Partha Niyogi. Semi-supervised learning on riemannian manifolds. *Machine learning*, 56: 209–239, 2004.

Mikhail Belkin and Partha Niyogi. Towards a theoretical foundation for Laplacian-based manifold methods. *Journal of Computer and System Sciences*, 74(8):1289–1308, December 2008.

Austin R. Benson, David F. Gleich, and Jure Leskovec. Higher-order organization of complex networks. *Science (New York, N.Y.)*, 353(6295):163–166, July 2016.

Jock Blackard. Covertype Dataset. UCI Machine Learning Repository, 1998. DOI: https://doi.org/10.24432/C50K5N.

Stephen P Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

Leon Bungert, Jeff Calder, and Tim Roith. Uniform Convergence Rates for Lipschitz Learning on Graphs. *IMA Journal of Numerical Analysis*, 43(4):2445–2495, August 2023.

Vivien Cabannes, Loucas Pillaud-Vivien, Francis Bach, and Alessandro Rudi. Overcoming the curse of dimensionality with laplacian regularization in semi-supervised learning. *Advances in Neural Information Processing Systems*, 34: 30439–30451, 2021.

Jeff Calder and Nicolás García Trillos. Improved spectral convergence rates for graph Laplacians on ϵ-graphs and k-NN graphs. *Applied and Computational Harmonic Analysis*, 60:123–175, September 2022.

Jeff Calder and Dejan Slepčev. Properly-weighted graph laplacian for semi-supervised learning. *Applied mathematics & optimization*, 82:1111–1159, 2020.

Antonin Chambolle and Thomas Pock. A First-Order Primal-Dual Algorithm for Convex Problems with Applications to Imaging. *Journal of Mathematical Imaging and Vision*, 40(1):120–145, May 2011.

T-H Hubert Chan, Anand Louis, Zhihao Gavin Tang, and Chenzi Zhang. Spectral properties of hypergraph laplacian and approximation algorithms. *Journal of the ACM (JACM)*, 65(3):1–48, 2018.

Atish Das Sarma, Anisur Rahaman Molla, Gopal Pandurangan, and Eli Upfal. Fast distributed PageRank computation. *Theoretical Computer Science*, 561:113–121, January 2015.

Ernie Esser, Xiaoqun Zhang, and Tony F. Chan. A General Framework for a Class of First Order Primal-Dual Algorithms for Convex Optimization in Imaging Science. *SIAM Journal on Imaging Sciences*, 3(4):1015–1046, January 2010.

Song Feng, Emily Heath, Brett Jefferson, Cliff Joslyn, Henry Kvinge, Hugh D. Mitchell, Brenda Praggastis, Amie J. Eisfeld, Amy C. Sims, Larissa B. Thackray, Shufang Fan, Kevin B. Walters, Peter J. Halfmann, Danielle Westhoff-Smith, Qing Tan, Vineet D. Menachery, Timothy P. Sheahan, Adam S. Cockrell, Jacob F. Kocher, Kelly G. Stratton, Natalie C. Heller, Lisa M. Bramer, Michael S. Diamond, Ralph S. Baric, Katrina M. Waters, Yoshihiro Kawaoka, Jason E. McDermott, and Emilie Purvine. Hypergraph models of biological networks to identify genes critical to pathogenic viral response. *BMC Bioinformatics*, 22(1):287, May 2021.

Richard Forsyth. Zoo Dataset. UCI Machine Learning Repository, 1990. DOI: https://doi.org/10.24432/C5R59V.

Kimon Fountoulakis, Farbod Roosta-Khorasani, Julian Shun, Xiang Cheng, and Michael W. Mahoney. Variational perspective on local graph clustering. *Mathematical Programming*, 174(1):553–573, March 2019.

Kaito Fujii, Tasuku Soma, and Yuichi Yoshida. Polynomial-time algorithms for submodular Laplacian systems. *Theoretical Computer Science*, 892:170–186, 2021.

Yuan Gao, Rasmus Kyng, and Daniel A. Spielman. Robust and Practical Solution of Laplacian Equations by Approximate Elimination, June 2023.

David Gibson, Jon Kleinberg, and Prabhakar Raghavan. Clustering categorical data: An approach based on dynamical systems. *The VLDB Journal*, 8(3):222–236, February 2000.

Arthur Gretton, Ralf Herbrich, and Alexander J Smola. The kernel mutual information. In *2003 IEEE International Conference on Acoustics, Speech, and Signal Processing, 2003. Proceedings.(ICASSP'03).*, volume 4, pages IV–880. IEEE, 2003.

Matthias Hein, Simon Setzer, Leonardo Jost, and Syama Sundar Rangapuram. The Total Variation on Hypergraphs - Learning on Hypergraphs Revisited. In *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc., 2013.

Edmund Ihler, Dorothea Wagner, and Frank Wagner. Modeling hypergraphs by graphs with the same mincut properties. *Information Processing Letters*, 45(4):171–175, March 1993.

Alexander Jung, Alfred O. Hero III, Alexandru Mara, Saeed Jahromi, Ayelet Heimowitz, and Yonina C. Eldar. Semi-supervised Learning in Network-Structured Data via Total Variation Minimization. *IEEE Transactions on Signal Processing*, 67(24):6256–6269, December 2019.

Markelle Kelly, Rachel Longjohn, and Kolby Nottingham. The uci machine learning repository. Date Retrieved: February 1, 2024.

Rasmus Kyng, Anup Rao, Sushant Sachdeva, and Daniel A Spielman. Algorithms for lipschitz learning on graphs. In *Conference on Learning Theory*, pages 1190–1223. PMLR, 2015.

Lap Chi Lau, Kam Chuen Tung, and Robert Wang. Cheeger inequalities for directed graphs and hypergraphs using reweighted eigenvalues. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing*, pages 1834–1847, 2023a.

Lap Chi Lau, Kam Chuen Tung, and Robert Wang. Cheeger Inequalities for Directed Graphs and Hypergraphs using Reweighted Eigenvalues. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing*, STOC 2023, pages 1834–1847, New York, NY, USA, June 2023b. Association for Computing Machinery.

Pan Li and Olgica Milenkovic. Submodular hypergraphs: p-laplacians, cheeger inequalities and spectral clustering. In *International Conference on Machine Learning*, pages 3014–3023. PMLR, 2018.

Pan Li, Niao He, and Olgica Milenkovic. Quadratic Decomposable Submodular Function Minimization: Theory and Practice. *Journal of Machine Learning Research*, 21 (106):1–49, 2020.

Meng Liu, Nate Veldt, Haoyu Song, Pan Li, and David F Gleich. Strongly local hypergraph diffusions for clustering and semi-supervised learning. In *Proceedings of the Web Conference 2021*, pages 2092–2103, 2021a.

Meng Liu, Nate Veldt, Haoyu Song, Pan Li, and David F. Gleich. Strongly Local Hypergraph Diffusions for Clustering and Semi-supervised Learning. In *Proceedings of the Web Conference 2021*, pages 2092–2103, Ljubljana Slovenia, April 2021b. ACM.

Anand Louis. Hypergraph markov operators, eigenvalues and approximation algorithms. In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*, pages 713–722, 2015.

Jiri Matousek. Lecture Notes on Metric Embeddings. 2013.

Tom Mitchell. Twenty Newsgroups Dataset. UCI Machine Learning Repository, 1999. DOI: https://doi.org/10.24432/C5C323.

Boaz Nadler, Nathan Srebro, and Xueyuan Zhou. Semi-supervised learning with the graph laplacian: The limit of infinite unlabelled data. *Advances in neural information processing systems*, 22:1330–1338, 2009.

Arkadij Semenovič Nemirovskij and David Borisovich Yudin. Problem complexity and method efficiency in optimization. 1983.

Yu. Nesterov. Smooth minimization of non-smooth functions. *Mathematical Programming*, 103(1):127–152, May 2005.

Yurii Nesterov. *Introductory Lectures on Convex Optimization: A Basic Course*. Springer Publishing Company, Incorporated, 1 edition, 2014.

Lorenzo Orecchia, Konstantinos Ameranis, Charalampos Tsourakakis, and Kunal Talwar. Practical Almost-Linear-Time Approximation Algorithms for Hybrid and Overlapping Graph Clustering. In *Proceedings of the 39th International Conference on Machine Learning*, pages 17071–17093. PMLR, June 2022.

Alexander Rakhlin and Karthik Sridharan. Online Learning with Predictable Sequences. In *Proceedings of the 26th Annual Conference on Learning Theory*, pages 993–1019. PMLR, 2013.

Shota Saito, Danilo P. Mandic, and Hideyuki Suzuki. Hypergraph p-Laplacian: A Differential Geometry View. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1), April 2018.

Alexander J. Smola and Risi Kondor. Kernels and Regularization on Graphs. In Bernhard Schölkopf and Manfred K. Warmuth, editors, *Learning Theory and Kernel Machines*, Lecture Notes in Computer Science, pages 144–158, Berlin, Heidelberg, 2003. Springer.

Daniel A. Spielman. Iterative solvers for linear equations, 2018.

Daniel A. Spielman and Shang-Hua Teng. Nearly-linear time algorithms for graph partitioning, graph sparsification, and solving linear systems. In *Proceedings of the Thirty-Sixth Annual ACM Symposium on Theory of Computing*, STOC '04, pages 81–90, New York, NY, USA, June 2004. Association for Computing Machinery.

Yuuki Takai, Atsushi Miyauchi, Masahiro Ikeda, and Yuichi Yoshida. Hypergraph clustering based on pagerank. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1970–1978, 2020.

Ronen Talmon, Israel Cohen, Sharon Gannot, and Ronald R Coifman. Diffusion maps for signal processing: A deeper look at manifold-learning techniques based on kernels and graphs. *IEEE signal processing magazine*, 30(4): 75–86, 2013.

Nicolás García Trillos and Dejan Slepčev. Continuum limit of total variation on point clouds. *Archive for Rational Mechanics and Analysis*, 220(1):193–241, April 2016.

Charalampos E. Tsourakakis, Jakub Pachocki, and Michael Mitzenmacher. Scalable Motif-aware Graph Clustering. In *Proceedings of the 26th International Conference on World Wide Web*, WWW '17, pages 1451–1460. International World Wide Web Conferences Steering Committee, 2017.

Jesper E. van Engelen and Holger H. Hoos. A survey on semi-supervised learning. *Machine Learning*, 109(2): 373–440, February 2020.

Nate Veldt, Austin R Benson, and Jon Kleinberg. Minimizing localized ratio cut objectives in hypergraphs. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1708–1718, 2020.

Nisheeth K Vishnoi et al. Lx= b. *Foundations and Trends® in Theoretical Computer Science*, 8(1–2):1–141, 2013.

Huilin Xiong, MNS Swamy, and M Omair Ahmad. Optimizing the kernel in the empirical feature space. *IEEE transactions on neural networks*, 16(2):460–474, 2005.

Shuicheng Yan, Dong Xu, Benyu Zhang, Hong-Jiang Zhang, Qiang Yang, and Stephen Lin. Graph embedding and extensions: A general framework for dimensionality reduction. *IEEE transactions on pattern analysis and machine intelligence*, 29(1):40–51, 2006.

Fuchun Yang and Zhou Wei. Generalized Euler identity for subdifferentials of homogeneous functions and applications. *Journal of Mathematical Analysis and Applications*, 337(1):516–523, January 2008.

Yuichi Yoshida. Cheeger Inequalities for Submodular Transformations, 2018.

Yuichi Yoshida. Cheeger inequalities for submodular transformations. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 2582–2601. SIAM, 2019.

Dengyong Zhou, Olivier Bousquet, Thomas Lal, Jason Weston, and Bernhard Schölkopf. Learning with local and global consistency. *Advances in neural information processing systems*, 16, 2003.

Umit V. Çatalyürek, Karen D. Devine, Marcelo Fonseca Faraj, Lars Gottesbüren, Tobias Heuer, Henning Meyerhenke, Peter Sanders, Sebastian Schlag, Christian Schulz, Daniel Seemaier, and Dorothea Wagner. More Recent Advances in (Hyper)Graph Partitioning. 2022.

Supplementary Material

A. Notation and mathematical preliminaries

For $T \in \mathbb{N}$, we denote by $[T]$ the set $\{1, 2, \ldots, T\}$.

Hypergraph preliminaries A weighted hypergraph $G = (V, H, \mathbf{w})$ is a collection of vertices V and hyperedges $H \subseteq 2^V$ with non-negative hyperedge weights $\mathbf{w} \in \mathbb{R}^H_{\geq 0}$. The degree $\deg(i) \stackrel{\text{def}}{=} \sum_{h \ni i} w_h$ of a vertex $i \in V$ is the sum of the weights of hyperedges containing i. Given a set $S \subseteq V$, its *volume* is the size of the degrees of its vertices: $\mathrm{vol}(S) \stackrel{\text{def}}{=} \sum_{i \in S} \deg(i)$. We denote by $\mathbf{D} \in \mathbb{R}^{V \times V}$ the diagonal matrix of degrees of G. A hypergraph is said to be *k-uniform* if every hyperedge has cardinality k. In this sense, a graph is simply a 2-uniform hypergraph. For any $S \subseteq V$ we denote by $\partial S \subseteq H$ the *boundary* of the set S, i.e. $h \in \partial S$ if $\exists i, j \in h$ such that $i \in S, j \in \overline{S}$.

Linear algebra preliminaries We denote vectors and matrices in boldface e.g. $\mathbf{x} \in \mathbb{R}^V$ and $\mathbf{A} \in \mathbb{R}^{n \times n}$. Given a vector $\mathbf{x} \in \mathbb{R}^V$ the vector $\mathbf{x}_h \in \mathbb{R}^h$ is the restriction of $\mathbf{x}$ on h, i.e. the vector with entries $\mathbf{x}(i)$ for $i \in h$. We use $\vec{\mathbf{1}}$ to denote the all-one vector in $\mathbb{R}^n$. Given two vectors $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$, we denote by $\langle \mathbf{x}, \mathbf{y} \rangle$ the standard inner product between $\mathbf{x}$ and $\mathbf{y}$, and we write $\mathbf{x} \perp \mathbf{y}$ to indicate $\langle \mathbf{x}, \mathbf{y} \rangle = 0$. Similarly, we use $\langle \mathbf{x}, \mathbf{y} \rangle_{\mathbf{A}}$ and $\mathbf{x} \perp_{\mathbf{A}} \mathbf{y}$ for the same expressions with respect to the inner product given by a positive definite operator $\mathbf{A}$. We may also use $\mathbf{1}_h$ to denote the all-one vector in $\mathbb{R}^h$. We recall that a norm $\|\cdot\| : \mathbb{R}^n \to \mathbb{R}$ is any function satisfying the following properties:

Triangle inequality $\|\mathbf{x} + \mathbf{y}\| \leq \|\mathbf{x}\| + \|\mathbf{y}\|$ for every $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$,

Absolute homogeneity $\|\alpha \mathbf{x}\| = |\alpha| \|\mathbf{x}\|$ for every $\mathbf{x} \in \mathbb{R}^n$ and $\alpha \in \mathbb{R}$,

Positive definiteness $\|\mathbf{x}\| \geq 0$ for all $\mathbf{x} \in \mathbb{R}^n$ and $\|\mathbf{x}\| = 0 \iff \mathbf{x} = 0$.

For any $p \in \mathbb{N}$ the ℓ_p-norm is the norm given by:

$$\|\mathbf{x}\|_p \stackrel{\text{def}}{=} \sqrt[p]{\sum_{i \in n} |\mathbf{x}(i)|^p}$$

and the ℓ_∞-norm is given by:

$$\|\mathbf{x}\|_\infty \stackrel{\text{def}}{=} \max_{i \in [n]} |\mathbf{x}(i)|.$$

Given a norm $\|\cdot\|$ on $\mathbb{R}^V$, its dual norm is the norm defined as:

$$\|\mathbf{y}\|_* \stackrel{\text{def}}{=} \max_{\|\mathbf{x}\| \leq 1} \langle \mathbf{x}, \mathbf{y} \rangle .$$

For any positive-definite matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$ the norm induced by $\mathbf{A}$ is $\|\mathbf{x}\|_{\mathbf{A}} \stackrel{\text{def}}{=} \sqrt{\mathbf{x}^\top \mathbf{A} \mathbf{x}}$. Its dual norm is the norm $\|\mathbf{x}\|_{\mathbf{A}^{-1}}$.

Subgradients and convexity The subdifferential of a convex function $f : \mathbb{R}^V \to \mathbb{R}$ at a point $\mathbf{x}$ is the set:

$$\partial f(\mathbf{x}) \stackrel{\text{def}}{=} \{\mathbf{y} \in \mathbb{R}^n \mid f(\mathbf{z}) \geq f(\mathbf{x}) + \langle \mathbf{z} - \mathbf{x}, \mathbf{y} \rangle , \forall \mathbf{z} \in \mathbb{R}^n\}.$$

When f is a convex function, $\partial f(\mathbf{x})$ is non-empty at every point $\mathbf{x}$ in the interior of the domain of f. Moreover, if f is convex and differentiable then $\partial f(\mathbf{x}) = \{\nabla f(\mathbf{x})\}$ for every $\mathbf{x}$ in the interior of the domain of f. An element of the subdifferential of f at $\mathbf{x}$ is called a *subgradient* of f at $\mathbf{x}$.

Fenchel Duality Given a convex function $f : \mathbb{R}^n \to \mathbb{R}$, its convex conjugate is the function $f^* : \mathbb{R}^n \to \mathbb{R}$ given by:

$$f^*(\mathbf{z}) = \sup_{\mathbf{x} \in \mathbb{R}^n} \{\langle \mathbf{x}, \mathbf{z} \rangle - f(\mathbf{x})\} .$$

The convex conjugate enjoys several properties relating to optimizing f. In particular for convex f,

$$\inf_{\mathbf{x}} f(\mathbf{x}) = \sup_{\mathbf{z}} f^*(\mathbf{z}).$$

A commonly used fact is that, for any norm $\|\cdot\|$ the convex conjugate of the function $\frac{1}{2}\|\cdot\|^2$ is the function $\frac{1}{2}\|\cdot\|_*^2$ where $\|\cdot\|_*$ is the dual norm as defined above.

We refer the reader to the book of Boyd and Vandenberghe (2004) for more details.

Submodular functions Given a finite set h a function $\delta : 2^h \to \mathbb{R}$ is submodular if, for all $A, B \subseteq h$ we have:

$$\delta(A) + \delta(B) \geq \delta(A \cap B) + \delta(A \cup B).$$

The Base polyhedron of a submodular function $\delta : 2^h \to \mathbb{R}$ is the set:

$$B(\delta) \stackrel{\text{def}}{=} \{\mathbf{y} \in \mathbb{R}^h \mid \mathbf{y}(h) = \delta(h), \mathbf{y}(S) \leq \delta(S) \text{ for every } S \subseteq h\},$$

where, for any $S \subseteq h$, $\mathbf{y}(S) \stackrel{\text{def}}{=} \sum_{i \in S} \mathbf{y}(i)$. The Lovász extension of δ is the function $\bar{\delta} : \mathbb{R}^h \to \mathbb{R}$ given by:

$$\bar{\delta}(\mathbf{x}) \stackrel{\text{def}}{=} \max_{\mathbf{y} \in B(\delta)} \langle \mathbf{y}, \mathbf{x} \rangle .$$

Submodular functions are a central object of study in combinatorial optimization and have found numerous applications throughout Computer Science. We refer the reader to the monograph by Bach (Bach, 2013) for more information on submodular functions and their applications.

B. Deferred proofs

In this section, we provide the proofs to all the technical results in the paper.

B.1. Deferred proofs from Section 2

Proof of Lemma 2.1. We begin the proof of (1) by proving that $F(\mathbf{0}) = 0$. By the continuity of convex functions and the 2-positive homogeneity of f, we have:

$$F(\mathbf{0}) = \lim_{\alpha \downarrow 0} F(\alpha \mathbf{1}) = \lim_{\alpha \downarrow 0} \alpha^2 \cdot F(\mathbf{1}) = 0.$$

Furthermore, the function F is symmetric about zero. That is, for every $\mathbf{x} \in \mathbb{R}^n$ we have:

$$F(\mathbf{x}) = (-1)^2 F(\mathbf{x}) = F(-\mathbf{x}).$$

By convexity of F and symmetry under negation, we can show $F(\mathbf{0}) \leq F(\mathbf{x})$:

$$F(\mathbf{0}) = F\left(\frac{1}{2}\mathbf{x} + \frac{1}{2}(-\mathbf{x})\right) \leq \frac{1}{2}\left(F(\mathbf{x}) + F(-\mathbf{x})\right) = F(\mathbf{x}).$$

This, in combination with the fact that $F(\mathbf{0}) = 0$, implies that F is everywhere non-negative.

The proof of (2) follows from the non-negativity of F and the fact that $F(\mathbf{0}) = 0$:

$$F^*(\mathbf{z}) = \sup_{\mathbf{x} \in \mathbb{R}^V} \{\langle \mathbf{x}, \mathbf{z} \rangle - F(\mathbf{x})\} \geq \langle \mathbf{0}, \mathbf{z} \rangle - F(\mathbf{0}) = 0.$$

We next prove (3). By the definition of subdifferential and the convexity of F, we have for $\mathbf{y} \in \mathbb{R}^n$:

$$F(\mathbf{y}) \geq F(\mathbf{x}) + \langle \mathbf{z}, \mathbf{y} - \mathbf{x} \rangle.$$

Letting $\mathbf{y} = \alpha \mathbf{x}$ for an arbitrary $\alpha > 0$ and applying 2-positive homogeneity, we deduce:

$$(\alpha^2 - 1)F(\mathbf{x}) \geq (\alpha - 1)\langle \mathbf{z}, \mathbf{x} \rangle.$$

For $\alpha > 1$, we can divide through by $(\alpha - 1) > 0$ and consider the limit:

$$\langle \mathbf{z}, \mathbf{x} \rangle \leq \lim_{\alpha \downarrow 1} \frac{\alpha^2 - 1}{\alpha - 1} \cdot F(\mathbf{x}) = 2F(\mathbf{x}),$$

where the last equality follows from L'Hôpital's rule. Similarly, for $\alpha < 1$:

$$\langle \mathbf{z}, \mathbf{x} \rangle \geq \lim_{\alpha \uparrow 1} \frac{\alpha^2 - 1}{\alpha - 1} \cdot F(\mathbf{x}) = 2F(\mathbf{x}),$$

thus establishing the result.

To prove (4), we use the definition of the subdifferential. For all $\mathbf{x} \in \mathbb{R}^n$, $\mathbf{z} \in \partial F(\mathbf{x})$ if and only if $\forall \mathbf{y} \in \mathbb{R}^n$,

$$F(\mathbf{y}) \geq F(\mathbf{x}) + \langle \mathbf{z}, \mathbf{y} - \mathbf{x} \rangle.$$

Consider any $\alpha > 0$. Multiplying both sides of the above equation by α^2 and leveraging the 2-positive homogeneity of F yields

$$\alpha^2 F(\mathbf{y}) \geq \alpha^2 F(\mathbf{x}) + \alpha^2 \langle \mathbf{z}, \mathbf{y} - \mathbf{x} \rangle$$
$$F(\alpha \mathbf{y}) \geq F(\alpha \mathbf{x}) + \langle \alpha \mathbf{z}, \alpha \mathbf{y} - \alpha \mathbf{x} \rangle.$$

Because the above holds for every $\mathbf{y} \in \mathbb{R}^n$, it implies by definition that $\alpha \mathbf{z} \in \partial F(\alpha \mathbf{x})$.

□

Proof of Lemma 2.2. By definition, for every $h \in H$, $\delta_h \geq 0$. Moreover, the Lovász extension of a set function δ is convex whenever δ is submodular. Hence $\mathcal{U}(\mathbf{x})$ is a conic combination of squares of non-negative convex functions, which implies it is convex.

Moreover, by definition of the Lovász extension, for any submodular cut function we have that for any $\alpha \in \mathbb{R}_{>0}$

$$\bar{\delta}(\alpha \mathbf{x}) = \max_{\mathbf{y} \in B(\delta)} \langle \mathbf{y}, \alpha \mathbf{x} \rangle = \alpha \max_{\mathbf{y} \in B(\delta)} \langle \mathbf{y}, \mathbf{x} \rangle = \alpha \bar{\delta}(\mathbf{x}),$$

and thus δ is 1-positive homogeneous. In particular, the square of any 1-positive homogeneous function is 2-positive homogeneous. I.e. for every $\alpha \geq 0$:

$$\bar{\delta}_h(\alpha \mathbf{x})^2 = \left(\alpha \bar{\delta}_h(\mathbf{x})\right)^2 = \alpha^2 \bar{\delta}_h(\mathbf{x})^2.$$

Since $\delta_h(h) = \delta_h(\varnothing) = 0$, the function $\bar{\delta}_h(\cdot)$ is invariant under translation by the $\vec{\mathbf{1}}$ vector, and in particular:

$$\min_{u \in \mathbb{R}} \bar{\delta}_h(\mathbf{x} - u\vec{\mathbf{1}}) = \bar{\delta}(\mathbf{x}).$$

□

Proof of Lemma 2.4. The lower bound in Equation (7)follows by definition of λ_G. The upper bound relies on Assumption 2.3:

$$\begin{aligned} 2\mathcal{U}(\mathbf{x}) &= \sum_{h \in E} w_h \min_{u \in \mathbb{R}} \mathcal{Q}_h(\mathbf{x}_h - u\vec{\mathbf{1}}_h) \leq \sum_{h \in E} w_h \mathcal{Q}_h(\mathbf{x}_h) \leq \sum_{h \in E} w_h \|\mathbf{x}_h\|_2^2 \\ &= \sum_{i \in V} \sum_{h : h \ni i} w_h \mathbf{x}(i)^2 = \sum_{i \in V} \deg(i) \mathbf{x}(i)^2 = \|\mathbf{x}\|_{\mathbf{D}}^2. \end{aligned}$$

In order to establish that $\lambda_G > 0$, we proceed by contradiction. Assume $\lambda_G = 0$. Then, there is a non-zero vector $\mathbf{x} \in \mathbb{R}^V \perp_{\mathbf{D}} \mathbf{1}$ with $\mathcal{U}(\mathbf{x}) = 0$. Consider the cut $S \subset V$ consisting of $i \in V$ having $\mathbf{x}_i > 0$. Because $\langle \mathbf{x}, \mathbf{1} \rangle_{\mathbf{D}} = 0$, the cut S is non-empty. We claim that S constitutes a connected component of G and completes the proof of the lemma. By way of contradiction, suppose a hyperedge h is cut by S, i.e. $h \cap S, h \cap \bar{S} \neq \emptyset$. Then, $\mathbf{x}$ is not constant over h, so that $\mathbf{x}_h - u\mathbf{1}_h \neq 0$ for all u. Because $\mathcal{Q}_h$ is non-negative by Lemma 2.1, this implies that hyperedge h makes a positive contribution to $\mathcal{U}(\mathbf{x})$. □

B.2. Deferred Proofs from Section 3

Proof of Theorem 3.1. The only obstacle to applying Theorem 4.1 directly is the fact the lowerbound with $\lambda_G > 0$ in Equation 9 only applies when $\mathbf{x}$ is restricted to the space $\mathbb{R}^V \perp_{\mathbf{D}} \mathbf{1}$. To bypass this obstacle, we show how the solution to the resolvent optimization problem in Equation 5 over $\mathbb{R}^V$ can be reduced to that over $\mathbb{R}^V \perp_{\mathbf{D}} \mathbf{1}$. To this end, we decompose a solution $\mathbf{x}$ into its component along $\mathbf{1}$ and its component orthogonal to $\mathbf{1}$, with respect to the $\mathbf{D}$-inner product.

$$\mathbf{x} = \mathbf{x}_\perp + \pi(\mathbf{x}) = \mathbf{x}_\perp + \frac{\langle \mathbf{x}, \mathbf{1} \rangle_{\mathbf{D}}}{\|\mathbf{1}\|_{\mathbf{D}}^2} \mathbf{1}.$$

By its definition, $\mathcal{U}(\mathbf{x})$ is invariant under shifts by multiples of $\mathbf{1}$, so that we can decompose the optimization problem 5 into two separate optimization problems:

$$\begin{aligned} \text{OPT} \stackrel{\text{def}}{=} \min_{\mathbf{x} \in \mathbb{R}^V} \mathcal{U}(\mathbf{x}) + \frac{\lambda}{2} \|\mathbf{x}\|_{\mathbf{D}}^2 - \langle \mathbf{s}, \mathbf{x} \rangle = \\ \min_{\substack{\mathbf{x}_\perp \in \mathbb{R}^V \perp_{\mathbf{D}} \mathbf{1} \\ \alpha \in \mathbb{R}}} \mathcal{U}(\mathbf{x}_\perp + \alpha \mathbf{1}) + \frac{\lambda}{2} \cdot \|\mathbf{x}_\perp + \alpha \mathbf{1}\|_{\mathbf{D}}^2 - \langle \mathbf{s}, \mathbf{x}_\perp + \alpha \mathbf{1} \rangle = \\ \min_{\mathbf{x}_\perp \in \mathbb{R}^V \perp_{\mathbf{D}} \mathbf{1}} \mathcal{U}(\mathbf{x}_\perp) + \frac{\lambda}{2} \cdot \|\mathbf{x}_\perp\|_{\mathbf{D}}^2 - \langle \mathbf{s}, \mathbf{x}_\perp \rangle + \min_{\alpha \in \mathbb{R}} \frac{\lambda}{2} \cdot \alpha^2 \cdot \|\mathbf{1}\|_{\mathbf{D}}^2 - \alpha \cdot \langle \mathbf{s}, \mathbf{1} \rangle \end{aligned} \tag{12}$$

If $\lambda = 0$ and $\langle \mathbf{s}, \mathbf{1} \rangle \neq 0$, then the optimization problem is unbounded, which can be easily detected by our algorithm. If $\lambda = 0$ and $\langle \mathbf{s}, \mathbf{1} \rangle = 0$, we have successfully reduced the problem to the form of Theorem 4.1. Hence, we may assume that $\lambda > 0$.

The second minimization is optimized by choosing $\alpha = {}^{\langle \mathbf{s}, \mathbf{1} \rangle}/_{\lambda \cdot \|\mathbf{1}\|_{\mathbf{D}}^2}$, which yields:

$$\text{OPT} = \min_{\mathbf{x}_\perp \in \mathbb{R}^V \perp_{\mathbf{D}} \mathbf{1}} \mathcal{U}(\mathbf{x}_\perp) + \frac{\lambda}{2} \cdot \|\mathbf{x}_\perp\|_{\mathbf{D}}^2 - \langle \mathbf{s}, \mathbf{x}_\perp \rangle - \frac{\langle \mathbf{s}, \mathbf{1} \rangle^2}{\lambda \cdot \|\mathbf{1}\|_{\mathbf{D}}^2} = \text{OPT}_\perp - \frac{\langle \mathbf{s}, \mathbf{1} \rangle^2}{\lambda \cdot \|\mathbf{1}\|_{\mathbf{D}}^2},$$

where $\text{OPT}_\perp$ denotes the optimum of the minimization problem restricted to $\mathbb{R}^V \perp_{\mathbf{D}} \mathbf{1}$. By our assumptions on $\mathcal{U}$, we have the following norm bounds:

$$\forall \mathbf{x} \in \mathbb{R}^V \perp_{\mathbf{D}} \mathbf{1}, \ \frac{(\lambda_G + \lambda)}{2} \cdot \|\mathbf{x}\|_{\mathbf{D}}^2 \leq \mathcal{U}(\mathbf{x}) + \frac{\lambda}{2} \|\mathbf{x}\|_{\mathbf{D}}^2 \leq \frac{(1 + \lambda)}{2} \cdot \|\mathbf{x}\|_{\mathbf{D}}^2.$$

Hence, by Theorem 4.1, after $O\left({}^{1+\lambda}/_{(\lambda_G + \lambda)\epsilon^2}\right)$ iterations, Algorithm 1 outputs $\mathbf{x}_\perp$ such that

$$\mathcal{U}(\mathbf{x}_\perp) + \frac{\lambda}{2} \cdot \|\mathbf{x}_\perp\|_{\mathbf{D}}^2 - \langle \mathbf{s}, \mathbf{x}_\perp \rangle \leq \text{OPT}_\perp + \epsilon \cdot |\text{OPT}_\perp|$$

Finally, the algorithm for Problem 5 will return:

$$\mathbf{x}_{\text{out}} = \mathbf{x}_\perp + \frac{\langle \mathbf{s}, \mathbf{1} \rangle}{\lambda \cdot \|\mathbf{1}\|_{\mathbf{D}}^2} \mathbf{1}.$$

By Equation 12, we have:

$$\begin{aligned} \mathcal{U}(\mathbf{x}_{\text{out}}) + \frac{\lambda}{2} \cdot \|\mathbf{x}_{\text{out}}\|_{\mathbf{D}}^2 - \langle \mathbf{s}, \mathbf{x}_{\text{out}} \rangle = \mathcal{U}(\mathbf{x}_\perp) + \frac{\lambda}{2} \cdot \|\mathbf{x}_\perp\|_{\mathbf{D}}^2 - \langle \mathbf{s}, \mathbf{x}_\perp \rangle - \frac{\langle \mathbf{s}, \mathbf{1} \rangle^2}{\lambda \cdot \|\mathbf{1}\|_{\mathbf{D}}^2} \leq \\ \text{OPT}_\perp + \epsilon \cdot |\text{OPT}_\perp| - \frac{\langle \mathbf{s}, \mathbf{1} \rangle^2}{\lambda \cdot \|\mathbf{1}\|_{\mathbf{D}}^2} = \text{OPT} + \epsilon \cdot |\text{OPT}_\perp| \leq \text{OPT} + \epsilon \cdot |\text{OPT}| \end{aligned}$$

This proves the approximation result and the bound on the number of iterations. Each iteration requires updating a vector by $\eta \mathbf{D}^{-1} \mathbf{s}$, which requires linear time in the size of the hypergraph, and updating in the heat diffusion direction $\eta \mathbf{D}^{-1} \mathcal{L}(\mathbf{x}_t)$. □

Proof of Theorem 3.2. We now prove Theorem 3.2. The statement of the theorem is a simple consequence of the following lemma, together with Theorem 4.1.

Lemma B.1. *Let $G = (V, H, w)$ be a weighted hypergraph and let $\mathcal{U}_\infty$ be the standard hypergraph potential on G. Then there exists a graph $G' = (V, E, w')$ with Laplacian matrix $\mathbf{L}$ such that:*

$$\frac{1}{\max_{h\in H}|h|-1} \leq \frac{\mathcal{U}(\mathbf{x})}{\frac{1}{2}\|\mathbf{x}\|_{\mathbf{L}}^2} \leq 1$$

for all $\mathbf{x} \in \mathbb{R}^V$.

We now prove Lemma B.1.

Proof of Lemma B.1. Consider the graph G' on the same vertex set V as G, with edge set given by:

$$E = \{ij \mid \exists h \in H, \{i,j\} \subseteq h\},$$

and with edge weights given by:

$$w'_{ij} = \sum_{h:\{i,j\}\subseteq h} 2 \cdot \frac{w_h}{|h|}.$$

We then have:

$$\begin{aligned}\frac{1}{2}\|\mathbf{x}\|_{\mathbf{L}}^2 &= \frac{1}{2}\sum_{ij\in E} w'_{ij}(\mathbf{x}(i)-\mathbf{x}(j))^2 = \sum_{ij\in E}\sum_{h:\{i,j\}\subseteq h}\frac{w_h}{|h|}(\mathbf{x}(i)-\mathbf{x}(j))^2 = \sum_{h\in H}\sum_{\{i,j\}\subseteq h}\frac{w_h}{|h|}(\mathbf{x}(i)-\mathbf{x}(j))^2 \\ &= \sum_{h\in H}\frac{w_h}{|h|}\sum_{\{i,j\}\subseteq h}(\mathbf{x}(i)-\mathbf{x}(j))^2 = \sum_{h\in H} w_h \sum_{i\in h}(\mathbf{x}(i)-\overline{\mathbf{x}}_h)^2,\end{aligned}$$

where $\overline{\mathbf{x}}_h$ is the average value of the entries of $\mathbf{x}_h$. In particular, this gives:

$$\frac{1}{2}\|\mathbf{x}\|_{\mathbf{L}}^2 = \sum_{h\in H} w_h \sum_{i\in h}(\mathbf{x}(i)-\overline{\mathbf{x}}_h)^2 \geq \sum_{h\in H}\frac{w_h}{2}\cdot\max_{i,j\in h}(\mathbf{x}(i)-\mathbf{x}(j))^2 = \mathcal{U}_\infty(\mathbf{x}), \tag{13}$$

where the inequality follows from the fact that the term $\sum_{i\in h}(\mathbf{x}(i)-\overline{\mathbf{x}}_h)^2$ contains both $\max_{i\in h}(\mathbf{x}(i)-\overline{\mathbf{x}}_h)^2$ and $\min_{i\in h}(\mathbf{x}(i)-\overline{\mathbf{x}}_h)^2$, together with the inequality $2(a^2+b^2) \geq (a+b)^2$. On the other hand, we also get:

$$\begin{aligned}\frac{1}{2}\|\mathbf{x}\|_{\mathbf{L}}^2 &= \sum_{h\in E}\frac{w_h}{|h|}\sum_{\{i,j\}\subseteq h}(\mathbf{x}(i)-\mathbf{x}(j))^2 \leq \sum_{h\in E}\frac{w_h}{|h|}\binom{|h|}{2}\max_{i,j\in h}(\mathbf{x}(i)-\mathbf{x}(j))^2 \\ &= \sum_{h\in H}\frac{|h|-1}{2}w_h\max_{i,j\in h}(\mathbf{x}(i)-\mathbf{x}(j))^2 \leq \frac{\max_{h\in H}|h|-1}{1}\cdot\mathcal{U}_\infty(\mathbf{x}).\end{aligned} \tag{14}$$

The result then follows from (13) and (14). □

This also concludes the proof of Theorem 3.2 □

B.3. Deferred Proofs from Section 4

Proof of Lemma 4.2. Recall the definition of auxiliary iterates $\mathbf{x}_t$:

$$\mathbf{x}_t \stackrel{\text{def}}{=} \hat{\mathbf{x}}_t - \eta R^{-1}\mathbf{s}.$$

Thus by the definition of Algorithm 1 for any $t \in \mathbb{N}$ we have:

$$\mathbf{x}_{t+1} = \hat{\mathbf{x}}_t - \eta R^{-1}\mathbf{s} = \hat{\mathbf{x}}_t - \eta R^{-1}(\mathbf{z}_t - \mathbf{s}) - \eta R^{-1}\mathbf{s} = \hat{\mathbf{x}}_t - \eta R^{-1}\mathbf{z}_t$$

For any $t \in \mathbb{N}$, we have:

$$\frac{1}{2}\|\mathbf{x}_{t+1}-\mathbf{x}^*\|_{\mathbf{R}}^2 = \frac{1}{2}\|\hat{\mathbf{x}}_t - \eta\mathbf{R}^{-1}\mathbf{z}_t - \mathbf{x}^*\|_{\mathbf{R}}^2$$

$$
\begin{aligned}
&= \frac{1}{2}\|\hat{\mathbf{x}}_t - \mathbf{x}^*\|_{\mathbf{R}}^2 - \eta\left\langle \hat{\mathbf{x}}_t - \mathbf{x}^*, \mathbf{z}_t\right\rangle + \frac{\eta^2}{2}\|\mathbf{R}^{-1}\mathbf{z}_t\|_{\mathbf{R}}^2 \\
&= \frac{1}{2}\|\hat{\mathbf{x}}_t - \mathbf{x}^*\|_{\mathbf{R}}^2 - \eta\left\langle \hat{\mathbf{x}}_t - \mathbf{x}^*, \mathbf{z}_t\right\rangle + \frac{\eta^2}{2}\|\mathbf{z}_t\|_{\mathbf{R}^{-1}}^2 \\
&= \frac{1}{2}\|\mathbf{x}_t + \eta\mathbf{R}^{-1}\mathbf{s} - \mathbf{x}^*\|_{\mathbf{R}}^2 - \eta\left\langle \hat{\mathbf{x}}_t - \mathbf{x}^*, \mathbf{z}_t\right\rangle + \frac{\eta^2}{2}\|\mathbf{z}_t\|_{\mathbf{R}^{-1}}^2 \\
&= \frac{1}{2}\|\mathbf{x}_t - \mathbf{x}^*\|_{\mathbf{R}}^2 - \eta\left\langle \hat{\mathbf{x}}_t - \mathbf{x}^*, \mathbf{z}_t - \mathbf{s}\right\rangle + \frac{\eta^2}{2}\left(\|\mathbf{z}_t\|_{\mathbf{R}^{-1}}^2 - \|\mathbf{s}\|_{\mathbf{R}^{-1}}^2\right) \\
&\leq \frac{1}{2}\|\mathbf{x}_t - \mathbf{x}^*\|_{\mathbf{R}}^2 + \eta\left[\mathrm{OPT}_{F,\mathbf{s}} - (F(\hat{\mathbf{x}}_t) - \langle s, \hat{\mathbf{x}}_t\rangle)\right] + \frac{\eta^2}{2}\|\mathbf{z}_t\|_{\mathbf{R}^{-1}}^2
\end{aligned}
$$

where the last inequality follows from the convexity of F and the positivity of $\|s\|_{\mathbf{R}^{-1}}^2$. The lemma follows from re-arranging terms. □

Proof of Corollary 4.3. Taking the Fenchel dual of $\|\cdot\|_{\mathbf{R}^{-1}}$,

$$
\begin{aligned}
\frac{1}{2}\|\mathbf{z}_t\|_{\mathbf{R}^{-1}}^2 &= \max_{\mathbf{y}} \langle \mathbf{z}_t, \mathbf{y}\rangle - \frac{\|\mathbf{y}\|_{\mathbf{R}}^2}{2} \leq \max_{\mathbf{y}} \langle \mathbf{z}_t, \mathbf{y}\rangle - \frac{F(\mathbf{y})}{u_{\mathbf{R}}} \\
&\leq \max_{\mathbf{x}} u_{\mathbf{R}} \cdot \langle \mathbf{z}_t, \mathbf{x}\rangle - \frac{F(u_{\mathbf{R}} \cdot \mathbf{x})}{u_{\mathbf{R}}} \\
&= u_{\mathbf{R}} \cdot \max_{\mathbf{x}} \langle \mathbf{z}_t, \mathbf{x}\rangle - F(\mathbf{x}).
\end{aligned}
$$

The last expression is maximized by $\hat{\mathbf{x}}_t$ such that $\mathbf{z}_t \in \partial F(\hat{\mathbf{x}}_t)$. Hence, by Lemma 2.1, we have:

$$
\frac{1}{2}\|\mathbf{z}_t\|_{\mathbf{R}^{-1}}^2 \leq u_{\mathbf{R}} \cdot (2F(\hat{\mathbf{x}}_t) - F(\hat{\mathbf{x}}_t)) = u_{\mathbf{R}} \cdot F(\hat{\mathbf{x}}_t).
$$

The statement of the corollary follows from the definition of η. □

Proof of Theorem 4.1. After combining the results of Corollary 4.3 and Lemma 4.2, we proceed to sum over all iterations t while telescoping terms. This yields:

$$
\begin{aligned}
&\sum_{t=0}^{T-1}(1 - \epsilon/2) \cdot F(\hat{\mathbf{x}}_t) - \langle s, \hat{\mathbf{x}}_t\rangle \\
&\qquad\qquad \leq T \cdot \mathrm{OPT}_{F,\mathbf{s}} + \frac{1}{2\eta}\|\mathbf{x}_0 - \mathbf{x}^*\|_{\mathbf{R}}^2
\end{aligned}
$$

By convexity and homogeneity, the left-hand side can be lower bounded as:

$$
\begin{aligned}
&\sum_{t=0}^{T-1}(1 - \epsilon/2)F(\hat{\mathbf{x}}_t) - \langle \mathbf{s}, \hat{\mathbf{x}}_t\rangle \geq \\
&\frac{T}{1 - \epsilon/2} \cdot \left(F(\mathbf{x}_T^{\text{out}}) - \left\langle \mathbf{s}, \mathbf{x}_T^{\text{out}}\right\rangle\right)
\end{aligned}
$$

Conversely, the last term on the right-hand side can be easily bounded by the lower inequality in Equation 9 and the fact that $\mathbf{x}_0 = \mathbf{0}$:

$$
\frac{1}{2}\|\mathbf{x}_0 - \mathbf{x}^*\|_{\mathbf{R}}^2 = \frac{1}{2}\|\mathbf{x}^*\|_{\mathbf{R}}^2 \leq \frac{1}{\ell_{\mathbf{R}}}F(\mathbf{x}^*) = \frac{|\mathrm{OPT}_{F,\mathbf{s}}|}{\ell_{\mathbf{R}}}
$$

Combining our bounds for the left- and right-hand sides, multiplying by $(1 - \epsilon/2)T^{-1}$, and recalling that $\mathrm{OPT}_{F,\mathbf{s}} \leq 0$, we derive the bound:

$$
\left(F(\mathbf{x}_T^{\text{out}}) - \left\langle \mathbf{s}, \mathbf{x}_T^{\text{out}}\right\rangle\right) \leq (1 - \epsilon/2)\left(\mathrm{OPT}_{F,\mathbf{s}} + \frac{|\mathrm{OPT}_{F,\mathbf{s}}|}{\eta\ell_{\mathbf{R}}T}\right)
$$

$$\leq \mathrm{OPT}_{F,\mathbf{s}} + \left(\frac{\epsilon}{2} + \frac{u_{\mathbf{R}}}{\ell_{\mathbf{R}}} \cdot \frac{2}{\epsilon T}\right) \cdot |\mathrm{OPT}_{F,\mathbf{s}}|$$

Finally, setting $T \geq \frac{u_{\mathbf{R}}}{\ell_{\mathbf{R}}} \cdot \frac{4}{\epsilon^2}$ yields the required multiplicative bound completing the proof of Theorem 4.1. □

C. Manifold Learning via Hypergraph Laplacians

Code for conducting all experiments in Appendix C and Appendix D is publically available on GitHub.[3]

Problem setup. Let $\mathcal{M}_1, \mathcal{M}_2$ denote manifolds embedded in $\mathbb{R}^d$. Let $\mu_1(\cdot)$, $\mu_2(\cdot)$ denote the uniform distributions over $\mathcal{M}_1$ and $\mathcal{M}_2$ respectively. We consider sampling n data points in $\mathbb{R}^d$ in the following manner: let $\boldsymbol{\theta} \in \{-1, +1\}^n$. The ith data point is then sampled as

$$\mathbf{p}_i \sim \begin{cases} \mu_1 * \rho_\sigma & \text{if } \boldsymbol{\theta}_i = -1 \\ \mu_2 * \rho_\sigma & \text{if } \boldsymbol{\theta}_i = +1 \end{cases}$$

where $\rho_\sigma(\cdot)$ denotes the density of the standard multivariate normal distribution in $\mathbb{R}^d$ with covariance $\sigma^2 \mathbb{I}_d$ for parameter σ.

We assume access to a small set of labels. We denote by $S \subset [n]$ the indices of the labeled points. These partial labels are then defined by the entries of vector $\mathbf{s} \in \mathbb{R}^n$:

$$\mathbf{s}_i = \begin{cases} \boldsymbol{\theta}_i & \text{if } i \in S \\ 0 & \text{otherwise} \end{cases}$$

The goal of the manifold learning task is to propagate information from the labeled set S to all data points.

(Hyper)graph construction. To form a (hyper)graph corresponding to data points $\{\mathbf{p}_i \in \mathbb{R}^d\}_{i=1}^n$, we first initialize one node v_i for each data point $\mathbf{p}_i$. We then add (hyper)edges connecting nodes whose data points are close in $\mathbb{R}^d$. In this work, we specifically consider k-nearest neighbor (hyper)graphs. For each vertex v_i, we identify its k nearest neighbors $v_{j_1}, \ldots, v_{j_k}$ based on Euclidean distances between corresponding data points $\mathbf{p}_i, \mathbf{p}_j$. To construct the k-nearest neighbor graph, we then initialize an edge between v_i and each of its k-nearest neighbors: $(v_i, v_{j_1}), \ldots, (i, v_{j_k})$. To construct the k-nearest neighbor hypergraph, we initialize a single hyperedge containing v_i and all of its k-nearest neighbors: $\{v_i, v_{j_1}, \ldots, v_{j_k}\}$.

We define the weighted hypergraph potential $\mathcal{U}^{(P)}$ using the sampled point cloud data matrix $P \in \mathbb{R}^{n \times d}$ as described in Section 5.1. For the graph results displayed in Figure 1, we use analogous distance-to-centroid Gaussian weights. In particular, we note that for graph edges using distance-to-centroid simply amounts to adjusting the bandwidth parameter in the traditional Gaussian kernel, defined in Equation 10: in the graph the centroid for each edge (i, j) is always point

$$\overline{\mathbf{p}}_{(i,j)} = \frac{1}{2}(\mathbf{p}_i + \mathbf{p}_j)$$

so the corresponding node weights are

$$w^{(p)}_{i,(i,j)} = \exp\left(-\|\mathbf{p}_i - \overline{\mathbf{p}}_{(i,j)}\|_2^2\right) = \exp\left(-\|\mathbf{p}_i - \mathbf{p}_j\|_2^2/4\right) = w^{(p)}_{j,(i,j)}.$$

Label estimation. To estimate $\boldsymbol{\theta}$, we use the personalized PageRank vector to propagate information from the partial labels in vector $\mathbf{s}$ to all nodes in the (hyper)graph. We use teleportation constant $\alpha = 0.5$. For the hypergraph, we use Algorithm 1 with 50 iterations to compute the PPR vector, denoted by $\mathbf{p}_\alpha(\mathbf{s})$, which is a scaled solution to Problem 4 with $\mathbf{M} = \mathbf{D}$. In the graph setting, we compute the PPR by solving the linear system

$$\left(\mathbb{I}_n + \frac{1-\alpha}{2\alpha}\mathbf{L}\mathbf{D}^{-1}\right)\mathbf{p}_\alpha(\mathbf{s}) = \mathbf{s},$$

where $\mathbf{L}$ denotes the weighted graph Laplacian, as described above.

[3]Full link address: `https://github.com/Orecchia-Research-Group/hypergraph_diffusions`

	2 DIMENSIONS	7 DIMENSIONS	HIGHER DIMENSIONS
$\mathbf{s}\boldsymbol{\ell}^{(1)}$	[1 3]	[1 3 1 1 1 1 1]	[1 3 1 1 . . . 1 1 1]
$\mathbf{s}\boldsymbol{\ell}^{(2)}$	[2 4]	[2 4 2 2 2 2 2]	[2 4 2 2 . . . 2 2 2]

Table 1: Dimensions of hyperrectangles used in manifold learning experiments.

Manifold sampling. We study two manifold learning problems: one with concentric hyperspheres, and one with concentric hyperrectangles. In the concentric hypersphere problem, the two manifolds are defined as:

$$\mathcal{M}_1 \stackrel{\text{def}}{=} \{\mathbf{p} \in \mathbb{R}^d : \|\mathbf{p}\|_2 \leq 1\}, \qquad \mathcal{M}_2 \stackrel{\text{def}}{=} \{\mathbf{p} \in \mathbb{R}^d : \|\mathbf{p}\|_2 = 2\}.$$

The standard deviation of the additive noise in this setting is $\sigma = 0.4$.

A hyperrectangle in $\mathbb{R}^d$ centered at the origin is defined by a vector of side lengths $\mathbf{s}\boldsymbol{\ell} \in \mathbb{R}^d$ as

$$\mathcal{R}(\mathbf{s}\boldsymbol{\ell}) \stackrel{\text{def}}{=} \{\mathbf{p} \in \mathbb{R}^d : |\mathbf{p}_i| \leq \mathbf{s}\boldsymbol{\ell}_i/2 \; \forall i \in [d]\}.$$

For example, a 1×2 rectangle in $\mathbb{R}^2$ centered at the origin is defined by side lengths $\mathbf{s}\boldsymbol{\ell} = [1 \;\; 2]$.

In the concentric hyperrectangles problem, the first manifold $\mathcal{M}_1$ is the hyperrectangle $\mathcal{R}(\mathbf{s}\boldsymbol{\ell}^{(1)})$. The second manifold, $\mathcal{M}_2$, is defined as the *surface* of hyperrectangle $\mathcal{R}(\mathbf{s}\boldsymbol{\ell}^{(2)})$. For example, in two dimensions we use $\mathbf{s}\boldsymbol{\ell}^{(1)} = [1, 3]$ and $\mathbf{s}\boldsymbol{\ell}^{(2)} = [2, 4]$, so $\mathcal{M}_1$ is the interior of the 1×3 rectangle centered at the origin, whereas $\mathcal{M}_2$ is the surface of the 2×4 rectangle centered at the origin. Values of $\mathbf{s}\boldsymbol{\ell}^{(1)}, \mathbf{s}\boldsymbol{\ell}^{(2)}$ used in varying dimensions are detailed in Table 1. The standard deviation of the additive noise in the hyperrectangles problem is $\sigma = 0.1$.

Weighted versus Unweighted Potentials. In Section 5.1, we introduce a novel weighted hypergraph potential. In Figure 1, we report the performance of label estimates produced by calculating the personalized PageRank vector corresponding to these weighted potentials. In Figure 2 we further present preliminary evidence suggesting that the weighted potential $\mathcal{U}^{(P)}$, defined in Equation 11, leads to better estimates compared to its unweighted counterpart $\mathcal{U}^{\infty}$, presented in Equation 2. The estimates produced by the weighted potential, displayed in Figure 2, exhibit AUC values closer to 1 than those produced by the unweighted potential. In particular, this divide becomes increasingly apparent in higher dimensions.

Confusion matrices of PPR for Semi-Supervised Learning for different (hyper)graph constructions and label estimation strategies. As seen in Figure 3, in high dimensions, using a hypergraph **always** yields superior results to using a graph. In most cases, including distribution information leads to higher accuracy results. Note that the confusion matrix when using balanced predictions can be described by a single number, since every row and column sum up to one. In the case of high dimensional hyperspheres, using the sign function misclassifies most of the points as belonging to the inner sphere, which is expected as distances along the surface of the outer sphere become ever greater as dimensions increase, while the distance to the center stays the same. Including distribution information can fix this issue. Surprisingly, the same behavior is not observed in weighted hyperrectangles where we achieve very good results even without label distribution information. All experiments were repeated fifty times to reduce any variance.

D. Categorical Clustering Experiments

D.1. Dataset details and construction

In the datasets we use there are three kinds of features, binary, categorical and numerical. For each binary feature we define a hyperedge over all the nodes that possess that feature. For categorical features, a hyperedge is created over the nodes belonging to each category. Finally, for numerical features we quantize them into 10 bins of equal size. Because only ten (mutually exclusive) bins are used, the average rank is of the same order as the number of nodes. Statistics for all datasets are shown in Table 2. We briefly explain each dataset's instances and features.

Zoo (Forsyth, 1990) Each instance is a colloquially named animal (e.g. antelope) that refers to multiple species, specifically a collection of subfamilies. The label for each instance is its taxonomical class. There are 16 binary features indicating the presence of certain morphological characteristics.

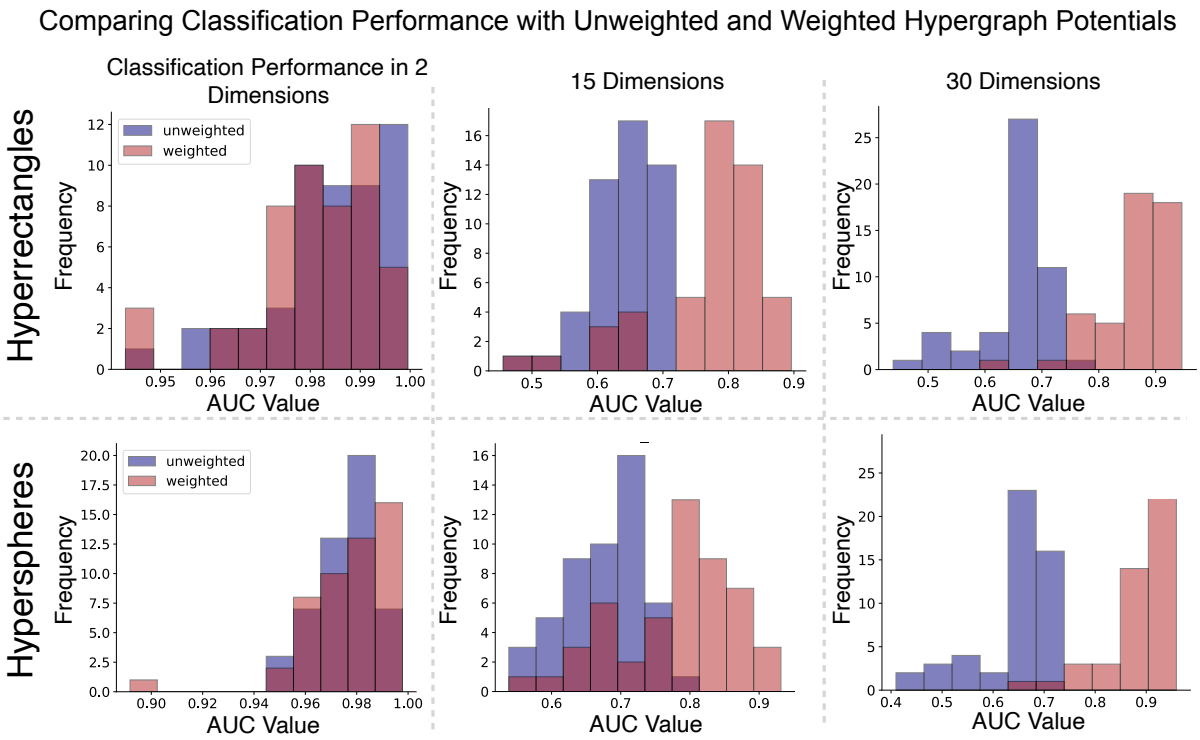


Figure 2: Comparing performance of the novel weighted hypergraph potential $\mathcal{U}^{(P)}$, defined in Equation 11, to the canonical (unweighted) hypergraph potential $\mathcal{U}^{\infty}$, defined in Equation 2. On each row, we display in blue the histogram of AUC values achieved by the PPR vector estimates for the semi-supervised learning problem using the unweighted potential, and we display in red the histogram of AUC values achieved using the weighted potential. Results are aggregated over 50 independent trials. Top row: results for hyperrectangle manifolds. Bottom row: results for hypersphere manifolds. In these semi-supervised manifold learning problems, the weighted hypergraph potential displays an advantage over unweihgted potentials, with greater discrepancy emerging in higher dimensions.

Mushroom (mis, 1987) Each instance is a species of mushroom and the target is whether that species is poisonous or not. Features are all categorical or binary morphological characteristics.

Covertype (Blackard, 1998) Each instance is a 30x30 meter cell of forest, classified by the US Forest Service (USFS) as one of seven types (Spruce/Fir, Lodgepole Pine, Ponderosa Pine, Cottonwood/Willow, Aspen, Douglas-fir, Krummholz). There are 10 numerical features, 4 binary columns classifying the surrounding wilderness area and 40 binary columns describing the soil type. In truth, this is 12 features, 10 numerical and 2 categorical, but they appear as 54 instead. Instead of working on the full 581,012 instances of all classes, instead we create two separate datasets, one from classes 4 and 5 (Cottonwood/Willow vs Aspen) and one from classes 6 and 7 (Douglas-fir vs Krummholz). We are referring to these datasets as covertype45 and covertype67.

Newsgroups (Mitchell, 1999) Every instance represents a usenet article. Each feature represents the presence of one of 100 common words. Because only the most common words are used, only 10,267 of 16,242 data points are different. (Hein et al., 2013) notes that the minimum error achieved by any deterministic classifier is 9.6%.

D.2. Methods

We compare a C++ implementation of our algorithm from Theorem 3.1 with the algorithms evaluated by Li et al. (2020), which also include the earlier PDHG method from Hein et al. (2013). These are also mainly implemented in C++ with a MATLAB wrapper.Here, we briefly review each of their four algorithms.

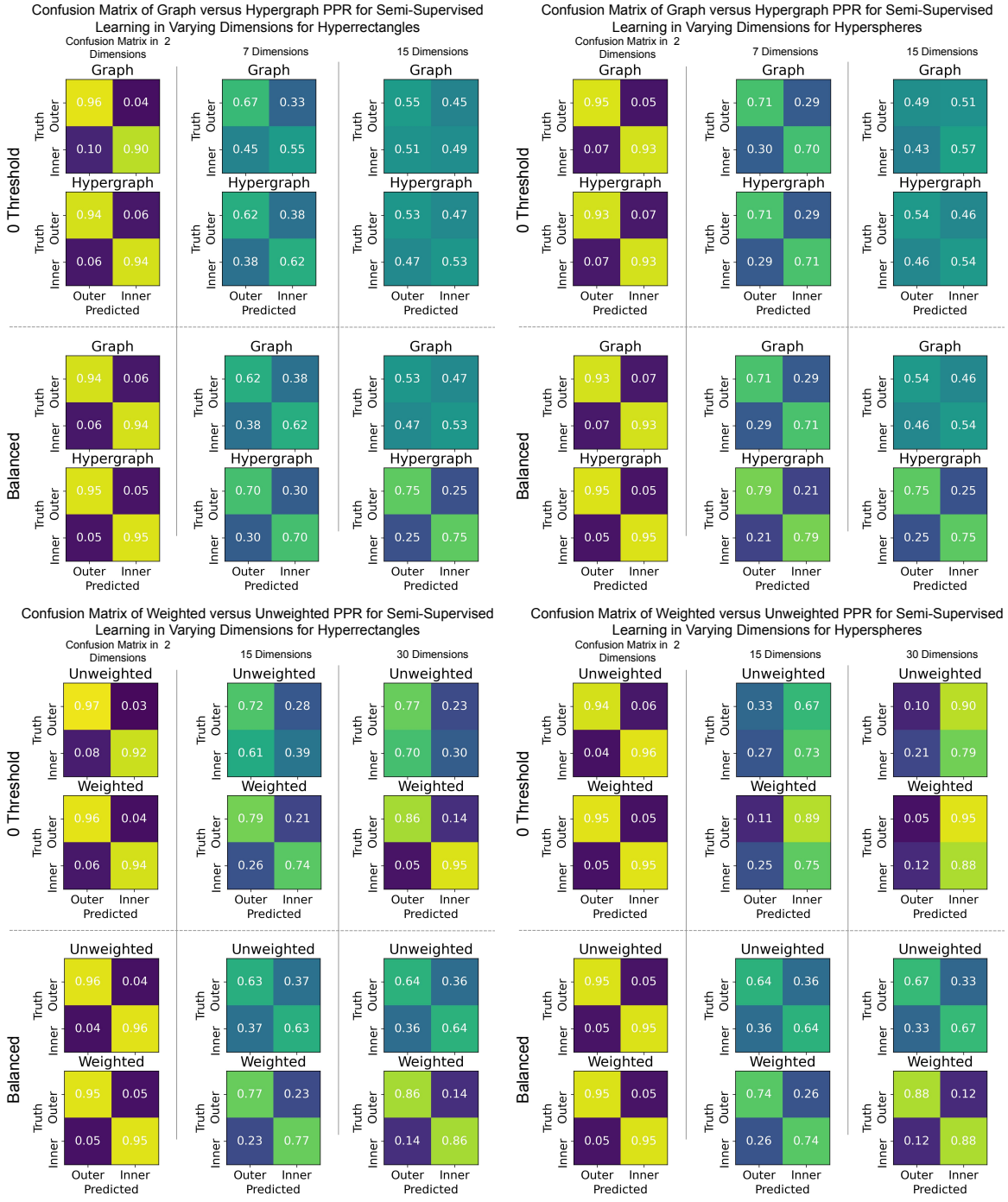


Figure 3: For determining predicted labels, we tried two different approaches: Using the sign function and assigning half the points to each category. The second requires knowledge of the size of each cluster, which is stronger, but not an unreasonable assumption. In the **top rows** is the comparison between **graphs** and **hypergraphs**. In the **bottom rows** we compare the effect that **Gaussian weights** have on the predictions. The **first columns** have results on the different dimension **hyperrectangles**, while the **last columns** show the same results on hyperspheres.

Name	**|F|**	**|V|**	**|H|**	$\sum_{\mathbf{v}\in\mathbf{V}} \frac{\deg_{\mathbf{v}}}{\|\mathbf{V}\|}$	$\sum_{\mathbf{h}\in\mathbf{H}} \frac{\|\mathbf{h}\|}{\|\mathbf{H}\|}$
Zoo (Forsyth, 1990)	16	101	36	16	44.89
Mushroom (Kelly et al.)	22	8124	112	21	1523.25
Covertype45 (Blackard, 1998)	54	12240	144	13.44	1142.51
Covertype67 (Blackard, 1998)	54	37877	144	13.02	3425.01
Newsgroups (Mitchell, 1999)	100	16242	100	4.03	654.51

Table 2: UCI Hypergraph datasets: for each hypergraph, we give name, number of features, vertices, hyperedges and average degree and rank of hyperedge.

PDHG QDSFM *Primal-Dual Hybrid Gradient* descent for *Quadratic Decomposable Submodular Function Minimization*. This is the primal-dual method proposed by (Hein et al., 2013). This algorithm maintains both primal and dual variables. At each iteration, it performs a proximal step on the dual variables of each hyperedge and a global gradient step on the primal variables.

QRCDM *Quadratic Random Coordinate Descent Method*. At every iteration, the algorithm randomly picks one hyperedge to perform coordinate descent. The authors show a linear convergence ($\log \frac{1}{\epsilon}$) for a number of iterations that grows linearly with the number of hyperedges and also depends on the width of the degree.

QRCDM AP *Quadratic Random Coordinate Descent Method* with *Alternative Projection*. Despite its name, this method does not perform random coordinate descent, but instead operates over all hyperedges. It alternatively performs projections over the product of cones and hyperplanes to attempt a best-approximation of the dual problem.

Subgradient QDSFM This method is closely related to our, but it does not enjoy our theoretical proof of convergence. In the practical evaluation, it seems to be the only one not converging to the optimizer.

Graph	PDHG QDSFM	QRCDM	QRCDM AP	Subgradient QDSFM	T=300	T=3000
zoo	0.14s	0.16s	0.14s	0.28s	0.03s	0.28s
mushroom	9.03s	19.44s	12.47s	9.55s	0.77s	7.60s
covertype45	7.98s	8.64s	8.57s	8.84s	0.75s	7.43s
covertype67	27.59s	29.27s	26.96s	25.65s	2.18s	21.75s
newsgroups	6.43s	7.93s	7.31s	11.90s	0.61s	5.98s

Table 3: Timing comparison of our method for 300 (T=300) and 3000 (T=3000) iterations with the methods provided by (Li et al., 2020). PDHG QDSFM and QRCDM AP run for 300 iterations. QRCDM runs for 300 * $|H|$ iterations, which can be thought as 300 epochs. Subgradient QDSFM runs for 15000 iterations. Our implementation performs iterations ten times faster.

Graph	PDHG QDSFM	QRCDM	QRCDM AP	Subgradient QDSFM	T=300	T=3000
zoo	0.475s	0.533s	0.467s	0.019s	0.098s	0.094s
mushroom	30.086s	64.787s	41.565s	0.636s	2.563s	2.533s
covertype45	26.586s	28.791s	28.567s	0.589s	2.493s	2.476s
covertype67	91.983s	97.554s	89.868s	1.710s	7.274s	7.249s
newsgroups	21.434s	26.422s	24.351s	0.793s	2.030s	1.993s

Table 4: Comparisons of average time *per 1000 iterations*. As QRCDM is a randomized coordinate descent algorithm which processes one hyperedge per iteration, we consider the average time for "epoch", i.e., $|H|$ iterations.

D.3. Execution

For a fixed set of labeled nodes with labels $\mathbf{y}$, we encode membership in target class $c \in C$ by the one-hot vector $\mathbf{y}_c(i) = \begin{cases} 1 & Y_i = c \\ -1 & \text{o.w.} \end{cases}$. The algorithms are then asked to approximately solve the following personalized PageRank vector

problem, corresponding to $\lambda = 1$ and $\alpha = 1/3$, for each class c:

$$\min P_{\mathbf{y},c}(\mathbf{x}) \stackrel{\text{def}}{=} \min_{\mathbf{x}} \frac{1}{2} \cdot \|\mathbf{x} - \mathbf{y}_c\|_{\mathbf{D}}^2 + \mathcal{U}_\infty(\mathbf{x}).$$

We chose this single value of λ as it proved representative of the relative behavior of each method. It also ensured that equal weight was given to both components of the objective, so that no algorithm focusing on a single component would be unfairly advantaged. Let $\mathbf{x}_{\mathbf{y},c}$ be the minimizers of the PageRank vector problem above. The final multiclass prediction was is by choosing, for each point $i \in V$, the class c maximizing $\mathbf{x}_{\mathbf{y},c}(i)$.

D.4. Evaluation

Table 4 displays average per iteration times for all methods. We have compiled graphical plots of the classification error and average function value $1/|C| \cdot \sum_c P_{\mathbf{y},c}(\mathbf{x}_{\mathbf{y},c})$ for the solution provided by each method in Figure 4. In order to reduce variance, each reported result is the average of ten runs.

Following previous works, we started by running all methods for an approximately equal amount of time. This resulted in 300 iterations each for PDHG QDSFM, QRCDM and QRCDM AP, 15000 iterations for Subgradient QDSFM and 3000 iterations for our method. With this choice of iteration counts, all methods perform very similarly on the classification error. In some cases, PDHG appeared to have very slight edge, possibly due to a better choice of initialization point or the regularizing effect of the primal-dual iteration. However, subgradient QDSFM appears to diverge further from the optimum point as the number of revealed points increases.

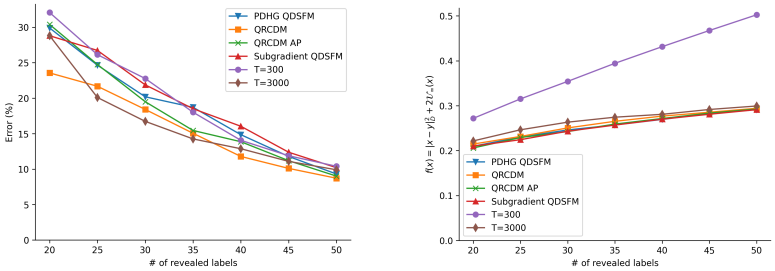
Note that with only 300 iterations, our method does not converge always to the minimizer, but often outperforms other methods in the prediction task, and achieves comparable function values, while only running for one tenth of the time.This is probably due to the very small per iteration time of our algorithm.

The method closest to our time per iteration is Subgradient QDSFM which does comparable amount of work per iteration, but has disappointing results in convergence even after a very large number of iterations.

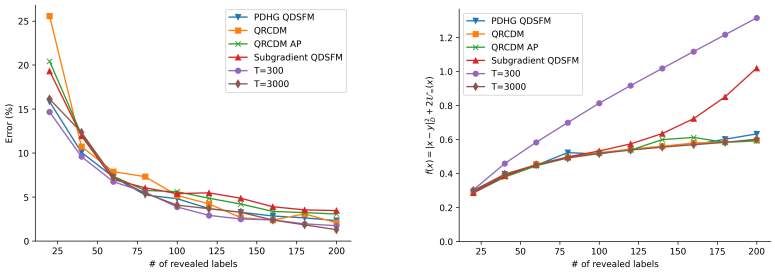
Another interesting trend is that in some cases, when our algorithm has not converged yet to the minimizer, the solution provided outperforms other methods in the classification task. This trend is more pronounced in covertype45 and is probably due to a suboptimal choice of λ and implicit regularization effects by the algorithm.

The abysmal classification error of all methods in the newsgroups dataset is explained by the poor discriminatory nature of the features used in its creation. Specifically, only the presence or absence of one hundred words is recorded, making groups of instances indistinguishable in every way, even though they belong to different clusters.

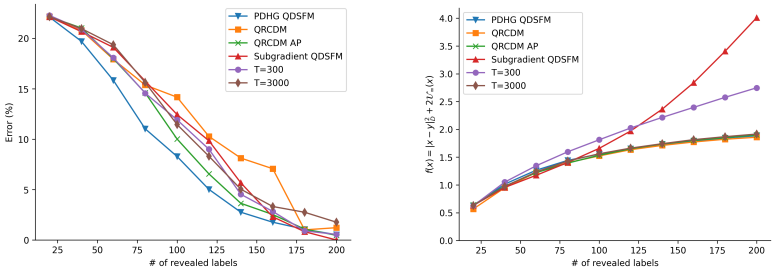
All experiments were run on a server with a 24core Intel Xeon Silver 4116 CPU @ 2.10GHz processor and 128gb RAM. All code is single threaded and can be found in our supplementary material together with the datasets. Additionally, the Python script used to generate the following image is also included.



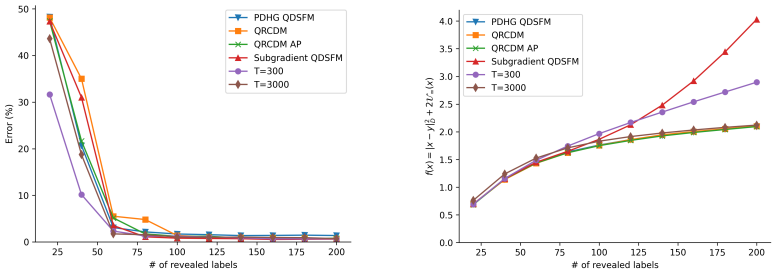
(a) zoo



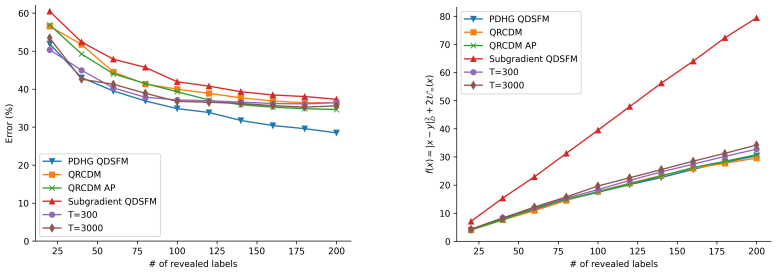
(b) mushroom



(c) covertype45



(d) covertype67



(e) newsgroups

Figure 4: Classification error and function value results for each method. Each method was run 10 times and an average was taken over them to reduce variance.