Consistency of Semi-supervised Learning, Stochastic Tug-of-War Games, and the p-Laplacian

Jeff Calder and Nadejda Drenska

Abstract In this paper, we give a broad overview of the intersection of partial differential equations (PDEs) and graph-based semi-supervised learning. The overview is focused on a large body of recent work on PDE continuum limits of graph-based learning, which have been used to prove well-posedness of semi-supervised learning algorithms in the large data limit. We highlight some interesting research directions revolving around *consistency* of graph-based semi-supervised learning and present some new results on the consistency of p-Laplacian semi-supervised learning using the stochastic tug-of-war game interpretation of the p-Laplacian. We also present the results of some numerical experiments that illustrate our results and suggest directions for future work.

1 Introduction

Machine learning refers to algorithms that learn how to perform tasks, like image classification or text generation, from examples or experience, and are not explicitly programmed with step-by-step instructions in the way a human may be instructed to perform a similar task. The recent surge in machine learning and artificial intelligence is being driven by deep learning, which uses deep artificial neural networks and has found applications in nearly all areas of science, engineering, and everyday life [53]. Modern deep learning excels when provided with massive amounts of training data and computational resources. However, there are many applications, specifically with real-world problems, where labeled training data is hard to come by and number in the hundreds or thousands, instead of millions. For

J. Calder (✉)
School of Mathematics, University of Minnesota, Minneapolis, MN, USA
e-mail: jcalder@umn.edu

N. Drenska
Department of Mathematics, Louisiana State University, Baton Rouge, LA, USA
e-mail: ndrenska@lsu.edu

J. A. Carrillo, E. Tadmor (eds.), *Active Particles, Volume 4*, Modeling and Simulation in Science, Engineering and Technology,
https://doi.org/10.1007/978-3-031-73423-6_1

example, in medical image analysis, a human expert, i.e., a highly trained doctor, must annotate images in order to provide data for machine learning algorithms to train in. Obtaining labeled data is thus costly, and there is a tremendous interest in developing machine learning algorithms that perform well with as few labeled examples as possible.

There are many frameworks for learning from limited data. One effective method is *semi-supervised learning*, which makes use of both labeled and unlabeled data in the learning task. In contrast, the most common type of machine learning, called *fully supervised* learning, makes use of only labeled training data. The labeled training data for a fully supervised classification task includes data/label pairs $(x_1, y_1), \dots, (x_n, y_n)$, where $x_i \in \mathbb{R}^d$ and $y_i \in \mathbb{R}^k$. The goal of fully supervised learning can generally be stated as finding, or "learning," a function $f : \mathbb{R}^d \to \mathbb{R}^k$ so that $f(x_i) \approx y_i$ for all i. This can be viewed as an *ill-posed* problem, especially when the number of labeled training points n is small, since there are many possible functions f that fit the training data. Furthermore, the ultimate goal is not just to fit the training data but to learn a function that *generalizes* well to new data that has not been seen before. Semi-supervised learning uses unlabeled data to improve the performance of classification algorithms in the context of small training sets. In many applications, unlabeled data is abundant and easy to obtain. In medical image analysis, for example, unlabeled data would correspond to medical images of a similar type and modality that have not been labeled or annotated by an expert.

An effective technique for exploiting unlabeled data in semi-supervised learning is to utilize a graph structure, which may be intrinsic to the data, or constructed based on similarities between data points. Graphs encode interdependencies between constituent data points that have proven useful for analyzing and representing high-dimensional data. There has been a surge of interest recently in graph-based semi-supervised learning techniques for problems where very few labeled examples are available, which is a setting that is challenging for existing techniques based on Laplacian regularization and harmonic extension. Various methods have been proposed, including p-Laplacian regularization, higher-order Laplacian methods, Poisson learning, and many others. Many of these algorithms are inspired by insights from the theory of partial differential equations (PDEs) or the calculus of variations, by examining the PDE-continuum limits of discrete graph-based learning algorithms, and their well-posedness properties, or lack thereof.

While there has been a substantial amount of work on PDE-continuum limits of graph-based learning, there has been relatively little work on the question of *consistency* of graph-based learning, using these well-developed PDE tools. The basic question of *consistency* is whether the machine learning algorithm is making the correct predictions, under a simplified model for the data. The current PDE-continuum limit results simply describe how the algorithms behave in the large data limit, but have not yet, with few exceptions, been used to prove that they work properly—that is, that they are consistent. It is arguably the case that consistency is more important than well-posedness in the continuum limit, yet the question has rarely been studied, which we suspect is due to the difficulty in defining what

consistency means, and the difficulty in obtaining meaningful results outside of toy settings.

In this paper, we provide a broad overview of semi-supervised learning and its connections to PDEs, and we present some new consistency results for p-Laplacian-based semi-supervised learning. Our new results make use of the tug of war with noise interpretation of the p-Laplacian, for which we also provide a brief literature survey. In particular, one of our results uses the tug-of-war game on a stochastic block model graph, which does not have the geometric structure that is usually required for PDE-based analysis. Our consistency results for the p-Laplacian are preliminary results meant to spark new work, and they certainly leave many questions unanswered. Our overall goal in this paper is to highlight a number of open research problems that will benefit from fruitful collaboration between PDE analysts and more theoretically minded machine learning researchers. It would be interesting in future work to improve these results and extend them to other graph-based semi-supervised learning algorithms and other graph structures.

1.1 Outline

This paper is organized as follows. In Sect. 2, we provide a brief overview of the p-Laplacian and the stochastic tug-of-war game interpretation. In Sect. 3, we give a thorough survey of graph-based semi-supervised learning, and its connections to PDEs and tug-of-war games, with a particular emphasis on the p-Laplacian. Then in Sect. 4, we present some preliminary results on consistency properties of the p-Laplacian using the tug-of-war game interpretation. This includes results both on geometric graphs and stochastic block models, as well as some numerical results to illustrate the main theorems. Finally, we conclude and discuss directions for future work in Sect. 5.

2 Tug-of-War Games and the p-Laplacian

In this section, we provide a brief overview of the p-Laplacian and the connection to stochastic tug-of-war games.

2.1 The p-Laplacian

The p-Laplacian arises as the Euler-Lagrange equation, or necessary conditions, for the nonlinear potential problem in the calculus of variations

$$\min_{u \in W^{1,p}(\Omega)} \int_{\Omega} |\nabla u|^p \, dx, \tag{2.1}$$

where $p \geq 1$ and $\Omega \subset \mathbb{R}^d$, subject to some boundary conditions, such as a Dirichlet condition $u = g$ on $\partial\Omega$. The Euler-Lagrange equation [46] for (2.1) is

$$\Delta_p u := \text{div}\left(|\nabla u|^{p-2}\nabla u\right) = 0 \quad \text{in} \quad \Omega, \tag{2.2}$$

and we call $\Delta_p u$ the p-Laplacian of u. We must take $p \geq 1$ to ensure (2.1) is convex and admits a minimizer. The case of $p = 2$ corresponds to the usual Laplacian $\Delta_p = \Delta$. When $1 \leq p < 2$ the diffusion is singular when $\nabla u = 0$, while for $p > 2$, the diffusion becomes degenerate when $\nabla u = 0$; both cases lead to drastically different properties and regularity theory compared to the uniformly elliptic case of $p = 2$ [45]. Functions that satisfy $\Delta_p u = 0$ are called *p harmonic*.

If we expand the divergence in (2.2), we find that any p-harmonic function u satisfies (provided $\nabla u \neq 0$ when $p < 2$)

$$\Delta_p u = |\nabla u|^{p-2}(\Delta u + (p-2)\Delta_\infty u) = 0 \quad \text{in} \quad \Omega, \tag{2.3}$$

where $\Delta_\infty u$ is the ∞-Laplacian, defined by

$$\Delta_\infty u = \frac{1}{|\nabla u|^2}\nabla u \cdot \nabla^2 u \nabla u = \frac{1}{|\nabla u|^2}\sum_{i,j=1}^{d} u_{x_i x_j} u_{x_i} u_{x_j}, \tag{2.4}$$

and $\nabla^2 u$ is the Hessian of u. The ∞-Laplacian is so named, because it is the $p \to \infty$ limit of the p-Laplacian in the sense that

$$\Delta_\infty u = \lim_{p\to\infty} \frac{1}{p-2}|\nabla u|^{2-p}\Delta_p u,$$

provided again that $\nabla u \neq 0$, which follows directly from (2.3). It is possible to interpret the ∞-Laplacian as the Euler-Lagrange equation for a variational problem like (2.1) with $p = \infty$; we refer the reader to [4] for more details. For a more detailed overview of the p-Laplacian, we also refer to [82].

2.2 Tug-of-War Games

When $p = 2$, there is a well-established classical connection between random walks, or Brownian motions, and harmonic functions. Indeed, any harmonic function u satisfies the mean value property [46]

$$u(x_0) = \fint_{B(x_0,\varepsilon)} u(y)\, dy, \tag{2.5}$$

where $\varepsilon > 0$ is any value for which $B(x_0, \varepsilon) \subset \Omega$, and the notation $\fint$ means

$$\fint_V u\,dx = \frac{1}{|V|}\int_V u\,dx.$$

We can interpret the right-hand side of (2.5) as the expectation of $u(X)$, where X is a random variable uniformly distributed on the ball $B(x_0, \varepsilon)$. Thus, if we define a random walk $X_1, X_2, \ldots$, on Ω, which is a sequence of random variables for which $X_0 = x_0 \in \Omega$ and, conditioned on X_k, X_{k+1} is uniformly distributed on $B(X_k, \varepsilon)$, we have

$$\mathbb{E}\left[u(X_{k+1}) \mid X_k\right] = \fint_{B(X_k,\varepsilon)} u(y)\,dy = u(X_k),$$

provided $B(X_k, \varepsilon) \subset \Omega$, provided we, for the moment, ignore the boundary $\partial\Omega$. Thus, since u is harmonic, we have that $Z_k = u(X_k)$ is a *martingale* [120]. This connection to probability theory allows simple alternative proofs of various estimates in harmonic function theory, such as Harnack's inequality and gradient estimates [79, 80], and has found applications in proving gradient estimates on graphs as well [26].

Over the past 15 years, there has been significant interest in extending these martingale techniques to the p-Laplacian. To do this, however, the definition (2.2) of the p-Laplacian is not very useful. Instead, provided that $\nabla u \neq 0$, we can drop the $|\nabla u|^{p-2}$ term in (2.3) to obtain the equation

$$\Delta u + (p-2)\Delta_\infty u = 0, \tag{2.6}$$

and restrict our attention to $p \geq 2$. Given a smooth function u, we can average the Taylor expansion for u about x_0 to obtain

$$\fint_{B(x_0,\varepsilon)} u(y)\,dy = u(x_0) + \frac{\varepsilon^2}{2(d+2)}\Delta u(x_0) + O(\varepsilon^3),$$

and so

$$\varepsilon^2\Delta u(x_0) = 2(d+2)\fint_{B(x_0,\varepsilon)} u(y)\,dy - 2(d+2)u(x_0) + O(\varepsilon^3). \tag{2.7}$$

Noting that the ∞-Laplacian is the second derivative of u in the direction of the gradient $v = \frac{\nabla u(x_0)}{|\nabla u(x_0)|}$, we have

$$\begin{aligned}\varepsilon^2\Delta_\infty u(x_0) &= \left[u\left(x_0 + \varepsilon v\right) - u(x_0)\right] - \left[u(x_0) - u\left(x_0 - \varepsilon v\right)\right] + O(\varepsilon^3) \\ &= \max_{B(x_0,\varepsilon)} u + \min_{B(x_0,\varepsilon)} u - 2u(x_0) + O(\varepsilon^3).\end{aligned} \tag{2.8}$$

Inserting (2.7) and (2.8) into (2.6), we see that if u is a smooth p-harmonic function, i.e., satisfying (2.6), then

$$u(x_0) = \alpha \fint_{B(x_0,\varepsilon)} u(y)\, dy + \frac{1-\alpha}{2}\left(\max_{B(x_0,\varepsilon)} u + \min_{B(x_0,\varepsilon)} u\right) + O(\varepsilon^3). \tag{2.9}$$

as $\varepsilon \to 0$ where $\alpha = \frac{d+2}{d+p} \in [0,1]$ since $p \geq 2$. Thus, while p-harmonic functions do not satisfy a mean value property for any size ball, (2.9) gives an asymptotic version of a mean value property, with min and max terms arising from the ∞-Laplacian. In fact, the asymptotic mean value property (2.9) *characterizes* p-harmonic functions [89]. It is also an interesting equation to study in its own right; when the $O(\varepsilon^3)$ is dropped, the functions are called *p-harmonious* and studied in detail in [91].

The mean value property (2.9) suggests a way to adapt the random walk construction earlier to p-harmonic functions. We simply define a stochastic process $X_0, X_1, X_2, \ldots$ so that, given X_k, X_{k+1} is defined in the following way: With probability α, we take a random walk step, so X_{k+1} is uniformly distributed on $B(X_k, \varepsilon)$; with probability $\frac{1-\alpha}{2}$, we set $X_{k+1} = \operatorname{argmax}_{B(X_k,\varepsilon)} u$, and likewise with probability $\frac{1-\alpha}{2}$, we set $X_{k+1} = \operatorname{argmin}_{B(X_k,\varepsilon)} u$. When the argmax or argmin is not unique, we make a rule to break ties, which can be deterministic or random. By the definition of this stochastic process, for any continuous function u, we have

$$\mathbb{E}[u(X_{k+1}) \mid X_k] = \alpha \fint_{B(X_k,\varepsilon)} u(y)\, dy + \frac{1-\alpha}{2}\left(\max_{B(X_k,\varepsilon)} u + \min_{B(X_k,\varepsilon)} u\right).$$

In particular, if u is smooth and p-harmonic, then the discussion above shows that

$$\mathbb{E}[u(X_{k+1}) \mid X_k] = u(X_k) + O(\varepsilon^3).$$

Thus, we have recovered the martingale property, at least asymptotically as $\varepsilon \to 0$.

The stochastic process introduced above is often described as a two player *tug-of-war* game with noise. The game involves a token X_k that is moved by two players and by random noise. Player I is trying to move the token to locations that maximize u, while the goal of player II is to move the token to places that minimize u. The game is played by flipping two coins. The first comes up heads with probability α and tails with probability $1-\alpha$. If the first coin comes up heads, the token X_k is moved to a uniformly random point X_{k+1} in the ball $B(X_k, \varepsilon)$. If the first coin comes up tails, then the game switches to a *tug-of-war* game, where a second unbiased coin is flipped to decide which player gets to move the token to decide X_{k+1}. Whichever player wins the second coin flip is allowed to move the token wherever they like in the ball $B(X_k, \varepsilon)$; the idea being that player I will move the token to maximize u, while player II will minimize. The exact goal of each player depends on the boundary condition; they may want to maximize/minimize the value of u when the game stops by hitting the boundary $\partial\Omega$, in which case only the values of u on or

near the boundary need to be specified in the game and the players are assumed to play optimal strategies to maximize or minimize the payoff—the value of u—at the end of the game. The random walk step, and the randomness in choosing between players, is interpreted as *noise*, hence the term tug of war with noise.

There is a close connection between tug-of-war games and nonlocal elliptic equations. Let us define the nonlocal 2 and ∞ Laplacians by

$$\Delta_2^\varepsilon u(x) = \fint_{B(x_0,\varepsilon)} u(y)\,dy - u(x), \quad \text{and} \quad \Delta_\infty^\varepsilon u(x) = \frac{1}{2}\left(\max_{B(x,\varepsilon)} u + \min_{B(x,\varepsilon)} u\right) - u(x).$$

Then if we drop the error term in (2.9) and rearrange, we arrive at the equation

$$\Delta_p^\varepsilon u := \alpha \Delta_2^\varepsilon u + (1-\alpha)\Delta_\infty^\varepsilon u = 0. \tag{2.10}$$

The operator Δ_p^ε on the left above is a nonlocal approximation to the p-Laplacian that arises from the tug-of-war game perspective and is closely related to the graph p-Laplacian discussed in Sect. 3. We can also consider a corresponding nonlocal boundary value problem

$$\begin{cases} \Delta_p^\varepsilon u = 0, & \text{in } \Omega_\varepsilon \\ \quad u = g, & \text{on } \partial_\varepsilon \Omega, \end{cases} \tag{2.11}$$

where g is given, $\partial_\varepsilon \Omega = \partial\Omega + B(0,\varepsilon)$ and $\Omega_\varepsilon = \Omega \setminus \partial_\varepsilon \Omega$. If we choose the stopping time τ to be the first time that the tug-of-war game hits the boundary strip $\partial_\varepsilon \Omega$, then the martingale property and the optional stopping theorem yield

$$u(x) = \mathbb{E}[u(X_\tau) \mid X_1 = x].$$

This gives a representation formula for solutions of the nonlocal p-Laplace equation (2.10) that is useful for studying properties of the solution through martingale techniques. Properties that are independent of the nonlocal scale $\varepsilon > 0$ are inherited by p-harmonic functions by sending $\varepsilon \to 0$.

Tug-of-war games for the p-Laplacian were originally introduced in [105, 106] with a version that holds for $1 \le p \le \infty$ using ideas from earlier work on deterministic games for the 1-Laplacian [69]. The version for $p \ge 2$ described above was introduced and studied in [90]. This work motivated a study of the game-theoretic p-Laplacian on graphs [92] as well as finite difference approaches for numerically approximating solutions [3, 102, 103]. The tug-of-war interpretation of the p-Laplacian has led to simple alternative proofs of regularity for p-harmonic functions, including the Harnack inequality and gradient estimates [5, 83]. In addition, many variants of tug-of-war games have been introduced, including games with bias [107], mixed Neumann/Dirichlet boundary conditions [32], obstacle problems [78], nonlocal tug-of-war for the fractional p-Laplacian [76], time-dependent equations [56], and variants on the core structure of the game [74]. We

also refer the reader to the survey article [77] and two recent books on tug-of-war games [75, 104] for more details.

3 Semi-supervised Learning and PDEs

In this section, we overview graph-based semi-supervised learning and the recent connections to PDEs in the continuum limit, with a specific focus on the p-Laplacian and tug-of-war games.

3.1 Graph-Based Semi-supervised Learning

Semi-supervised learning uses both labeled and unlabeled data. As a toy example, in Fig. 1a, we show the famous two-moons data set with two labeled data points, the blue circle and green square. The black points are the *unlabeled data*, which are not used by fully supervised learning, and will be discussed momentarily. With only two labeled data points, it is difficult to learn a general function f that will correctly classify new data points. In Fig. 1b, we show the result of training a linear kernel support vector machine (SVM) with these two data points, which finds the linear decision boundary with maximal margin—in this case, the line equidistant[1] from the two training points. Given no other information, this may be a reasonable thing to do. However, suppose now that we also have access to the black points, which are *unlabeled* data points. That is, we have access to the coordinates $x_i \in \mathbb{R}^2$ but not the label y_i, which in this case is $y_i \in \{0, 1\}$ for binary classification, and these unlabeled data points are the *only* data points we will want to apply our classifier to in the future. In this case, the linear SVM decision boundary is a poor choice, since it cuts through a dense region of the unlabeled data—the lower moon—where we may not expect a true decision boundary to lie. Instead, we should seek to place the decision boundary in sparse regions between clusters. In Fig. 1c, we show the result of using a semi-supervised learning algorithm on the same data set, which correctly separates the two moons.[2]

There are many ways to incorporate unlabeled data into a machine learning algorithm. One common and successful approach is to utilize *graph-based* learning, where each node in the graph corresponds to a data point, and the edges in the graph correspond to either intrinsic relationships between data points or record some

[1] The line does not appear orthogonal to the vector between the training points, because the axes are scaled differently.

[2] To be precise, we applied graph-based Poisson learning, described below, to obtain label predictions for all unlabeled points, and then we trained a radial basis kernel SVM using the label predictions.

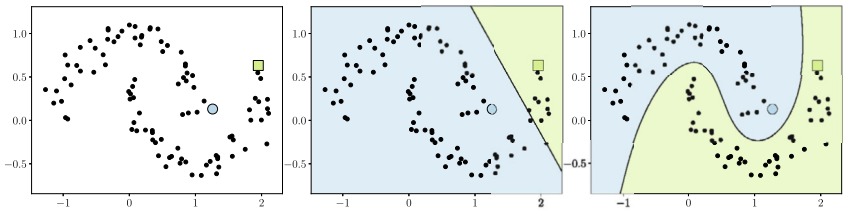

Fig. 1 A toy example comparing fully and semi-supervised learning. In (**a**), we show a data set with two labeled examples—the blue circle and green square—along with 98 unlabeled data points, the black dots. In (**b**), we show the decision regions from training a fully supervised classification algorithm, while in (**c**), we show the decision regions for a semi-supervised learning algorithm, which uses the unlabeled data to inform the decision boundary

notion of similarity between data points. Many types of data have intrinsic graph structures, like molecules in drug discovery problems, citation data sets, or networks like the internet. In other problems, like image classification, the graph structure is not intrinsic but can be constructed as a *similarity graph*, in which similar data points are connected by an edge with a weight that encodes the degree of similarity. Let $\mathcal{X} = \{x_1, \ldots, x_n\}$ be a set of data points, where $x_i \in \mathbb{R}^d$. Here, we assume we have a graph with vertex set $\mathcal{X}$ described by a symmetric weight matrix $W = W^T \in \mathbb{R}^{n \times n}$ in which each entry w_{xy}, for $x, y \in \mathcal{X}$, is nonnegative and encodes a notion of similarity between x and y. A zero weight $w_{xy} = 0$ indicates there is no edge between x and y, while a positive weight $w_{xy} > 0$ indicates the presence of an edge, and the larger the value the stronger the edge. We let $N_x \subset \mathcal{X}$ denote the graph neighbors of the vertex $x \in \mathcal{X}$, which is defined by

$$N_x = \{y \in \mathcal{X} : w_{xy} > 0\}. \tag{3.1}$$

A common way to construct a graph over a data set is the geometric graph construction

$$w_{xy} = \eta\left(\frac{|x - y|}{\varepsilon}\right), \tag{3.2}$$

where $\varepsilon > 0$ is the graph *bandwidth* and $\eta : [0, \infty) \to [0, \infty)$ is a nonnegative function that is typically decreasing with compact support. A common choice is Gaussian weights where $\eta(t) = e^{-t^2}$, with a possible truncation to zero at some distance $t = \tau$. One issue with the geometric graph construction is that one has to choose a very large value for the graph bandwidth $\varepsilon > 0$ to ensure the graph is connected, even in the sparsest regions of the data set, which leads to a very non-sparse weight matrix W that is difficult to work with computationally. To address this, it is common to adjust the bandwidth ε locally to reflect the density (or sparsity) of the data set, which results in various types of k-nearest neighbor graphs. One common construction is

$$w_{xy} = \eta\left(\frac{|x-y|}{\sqrt{\varepsilon_x \varepsilon_y}}\right),$$

where ε_i is the distance from x to its kth nearest neighbor, though other choices are possible. Throughout this section, we'll generally assume a graph with geometric weights of the form (3.2).

3.2 Laplacian Regularization

Suppose now that we have a graph structure over our data set, given by a weight matrix $W = (w_{xy}) \in \mathbb{R}^{n\times n}$ along with a subset of graph nodes $\Gamma \subset \mathcal{X}$ that are labeled, with corresponding labels $y = g(x) \in \mathbb{R}$ for $g : \Gamma \to \mathbb{R}$ a given labeling function, which we are taking to be scalar only for simplicity of discussion. The seminal approach in graph-based semi-supervised learning is *Laplacian regularization*, initially proposed in [135], which propagates the labels from Γ to the rest of the graph by solving the optimization problem

$$\min_{u:\mathcal{X}\to\mathbb{R}} \sum_{x,y\in\mathcal{X}} w_{xy}(u(x)-u(y))^2, \tag{3.3}$$

subject to the constraint that $u(x) = g(x)$ for $x \in \Gamma$. The real-valued solution $u(x)$ is then thresholded to the nearest label to make a class prediction. The energy in (3.3) is called the *graph Dirichlet energy*, and the minimizer $u : \mathcal{X} \to \mathbb{R}$ is exactly the harmonic extension of the labeled data, which satisfies the boundary value problem

$$\begin{cases} \mathcal{L}u(x) = 0, & \text{if } x \in \mathcal{X}\setminus\Gamma \\ u(x) = g(x), & \text{if } x \in \Gamma, \end{cases} \tag{3.4}$$

where $\mathcal{L}$ is the graph Laplacian defined by

$$\mathcal{L}u(x) = \sum_{y\in\mathcal{X}} w_{xy}(u(x)-u(y)). \tag{3.5}$$

In other words, we are seeking the *smoothest* function—in this case harmonic—that correctly classifies the labeled data points. In semi-supervised learning, this is often called the *semi-supervised smoothness assumption* [30], which stipulates that the labeling is smooth in high density regions of the data set. Since the initial development in [134, 135], graph-based learning using Laplacian regularization, and variations thereof, has grown into a wide class of useful techniques in machine learning [2, 8, 10–12, 15, 24, 25, 29, 55, 57, 58, 73, 117, 118, 121, 123, 128, 129, 131–133].

Graph harmonic functions satisfy a mean value property, similar to harmonic functions on $\mathbb{R}^d$. Indeed, if we rearrange the condition $\mathcal{L}u(x) = 0$, we obtain

$$u(x) = \frac{1}{d_x} \sum_{y \in \mathcal{X}} w_{xy} u(y), \tag{3.6}$$

where $d_x = \sum_{y \in \mathcal{X}} w_{xy}$ is the degree of vertex x. This is a local version of the continuous mean value property for harmonic functions. It says that the value $u(x)$ of a graph harmonic function at a node x is equal to the weighted average of the values $u(y)$ at neighboring nodes y, weighted by the graph edge weights w_{xy}. In fact, one way to solve equation (3.4) is by iterating the mean value property

$$u_{k+1}(x) = \frac{1}{d_x} \sum_{y \in \mathcal{X}} w_{xy} u_k(y), \tag{3.7}$$

for $x \in \mathcal{X} \setminus \Gamma$, fixing $u_k(x) = g(x)$ for $x \in \Gamma$. This is exactly the classical Jacobi iteration for solving a linear system, and while it is not the fastest technique—the preconditioned conjugate gradient method is much faster—it has a nice interpretation as *propagating* labels using the neighborhood structure of the graph and is thus called *label propagation* in the literature; see [134]. Equation (3.7) can also be viewed as a diffusion equation on the graph.

Laplacian regularized learning has an important connection to random walks on graphs. Let $X_1, X_2, \ldots$ be a random walk on the vertices $\mathcal{X}$ of the graph, with transition probabilities $P = D^{-1}W$, where $D = \operatorname{diag}(d_{x_1}, \ldots, d_{x_n})$. That is, the probability p_{xy} of the random walker moving from vertex x to vertex y is given by $p_{xy} = d_x^{-1} w_{xy}$. Given a function $u : \mathcal{X} \to \mathbb{R}$ on the vertices of the graph, we can compute

$$\mathbb{E}[u(X_{k+1}) - u(X_k) \mid X_k = x] = \sum_{y \in \mathcal{X}} p_{xy}(u(y) - u(x)) = -\frac{1}{d_x} \mathcal{L}u(x).$$

The object on the right-hand side is called the *random walk* graph Laplacian, and we denote it by $\mathcal{L}_{rw}$; that is,

$$\mathcal{L}_{rw} u(x) = \frac{1}{d_x} \mathcal{L}u(x).$$

The computation above thus yields

$$\mathbb{E}[u(X_{k+1}) \mid X_k] = u(X_k) - \mathcal{L}_{rw} u(X_k), \tag{3.8}$$

and thus, the random walk graph Laplacian is the generator for the random walk. In particular, if u is graph harmonic, so that $\mathcal{L}u(x) = 0$, then $u(X_k)$ is a *martingale*. Defining the stopping time τ as the first time the random walk hits the labeled data

set Γ, the optional stopping theorem yields that the solution u of Laplace learning (3.4) satisfies

$$u(x) = \mathbb{E}[u(X_\tau) \mid X_1 = x] = \sum_{y \in \Gamma} \mathbb{P}(X_\tau = y \mid X_1 = x) g(y). \tag{3.9}$$

Thus, the solution of Laplacian regularized graph-based learning (3.4) can be interpreted as a weighted average of the given labels $g(y)$, weighted by the probability of the random walk hitting $y \in \Gamma$ first, before hitting any other labeled data point. This is a reasonable thing to do for semi-supervised learning at an intuitive level as well. Indeed, if each cluster in the data set is well separated from the others, then provided the label rate is not too low, the random walk is likely to stay in the cluster it started in long enough to hit a labeled data point in that cluster, giving the correct label and propagating labels well within clusters.

At moderately low label rates, Laplacian regularization performs very well for graph-based semi-supervised learning. However, the performance can become quite poor at extremely low label rates, which was first noted in [101] and later in [39]. When there are very few labels, the solution of the Laplace learning problem (3.4) prefers to be nearly constant over the whole graph, with sharp spikes at the labeled data points, as depicted in Fig. 2a. This is the configuration that gives the least graph Dirichlet energy, since the spikes are relatively inexpensive when the number of labeled data points is small. From the random walk perspective, when there are very few labels, the random walk takes very long to hit a labeled data point, so the stopping time τ is very large, and the distribution of the random walker approaches the limiting invariant distribution on the graph, which is proportional to the degree. Thus, by (3.9) we have

$$u(x) \approx \frac{\sum_{y \in \Gamma} d_y g(y)}{\sum_{y \in \Gamma} d_y} \tag{3.10}$$

at very low label rates. When the labels are binary $g(y) \in \{-1, 1\}$, the sign of the right-hand side of (3.10) is constant over the graph and simply chooses the class whose labeled points have the largest cumulative degree, which is one way to explain the very poor performance of Laplace learning at low label rates.

Even in settings where moderate amounts of labeled data will eventually be acquired, it is often necessary to consider, at least initially, problems with very few labels where Laplace learning is not useful. One of those areas is *active learning*, which refers to machine learning algorithms that address the problem of choosing the *best* data points to label in order to obtain superior model performance with as few labels as possible [110]. Active learning methods incorporate a human-in-the-loop that can be queried to label new data points as needed, and the goal is to achieve the highest accuracy with as few labels as possible by intelligently choosing the next point to label, often in a sequential manner. Active learning procedures normally start out with extremely small labeled sets that slowly grow

during the acquisition of new labeled data points, which requires graph-based semi-supervised learning algorithms that perform well both with small and moderate amounts of labeled data. A number of graph-based active learning methods have been proposed [65, 85, 97, 100, 108, 136], including recent methods inspired by the PDE-continuum limits discussed below [98], and recent applications have been found in image classification problems [17, 31, 44, 99].

3.3 PDE-Inspired Insights and Algorithms

Another interpretation of the poor performance of Laplace learning at low label rates is through PDE continuum limits. To explain this, we need to place some additional assumptions on the graph construction. We assume the vertices $\mathcal{X} = \{x_1, \dots, x_n\}$ of the graph are *i.i.d.* random variables distributed on a domain $\Omega \subset \mathbb{R}^d$ with a continuous and positive density $\rho : \Omega \to (0, \infty)$. We assume the random geometric graph construction, which uses weights (3.2) for a given bandwidth $\varepsilon > 0$. This is called a *random geometric graph*, and in this setting, the expectation of the graph Dirichlet energy in (3.3) is

$$\begin{aligned} n^2 \int_\Omega \int_\Omega \eta\left(\frac{|x-y|}{\varepsilon}\right)(u(x)-u(y))^2 \rho(x)\rho(y)\,dxdy \\ \approx C_\eta n^2 \varepsilon^{d+2} \int_\Omega \rho^2 |\nabla u|^2\,dx, \end{aligned} \tag{3.11}$$

provided $u : \Omega \to \mathbb{R}$ is a smooth function and ρ is Lipschitz, where C_η is a positive constant depending only on η. The asymptotics in (3.11) can be verified formally with Taylor expansions of u and ρ and can also be established rigorously in the language of Gamma convergence, as in [49]. Thus, the continuum version of Laplace learning (3.4) is the boundary value problem

$$\begin{cases} \operatorname{div}(\rho^2 \nabla u) = 0, & \text{in } \Omega \setminus \Gamma \\ u = g, & \text{on } \Gamma, \end{cases} \tag{3.12}$$

where $\Gamma \subset \Omega$ is purposely vaguely specified and should encode some notion of the continuum limit of the labeled data set, and $g : \Gamma \to \mathbb{R}$ the continuum limit of the values of the labels. When Γ does not contain the entire boundary $\partial\Omega$, we also must impose homogeneous Neumann conditions on $\partial\Omega \setminus \Gamma$.

The presence of the squared density ρ^2 as a diffusion coefficient in the PDE continuum limit illustrates how Laplace learning is able to diffuse labels quickly in high-density regions where ρ is large and place decision boundaries, which correspond to sharp transitions in the label function u, in sparse regions where ρ is small. However, the presence of the labeled set Γ as a boundary condition is

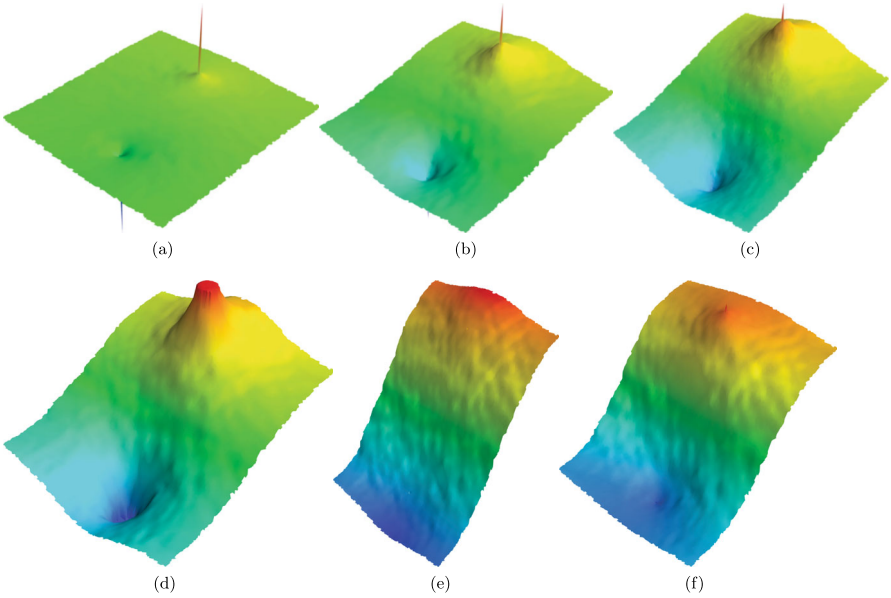

Fig. 2 Comparison of different semi-supervised learning methods on a toy example with two labeled data points with labels $+1$ and -1. The graph consists of $n = 20{,}000$ uniformly distributed random variables on the unit box $[0, 1]^2$ with geometric Gaussian kernel weights with $\varepsilon = 0.05$. Here WNLL stands for Weighted NonLocal Laplacian [111], and PWLL stands for Poisson Weighted Laplace Learning [27, 98]. (**a**) Laplace [135]. (**b**) p-Laplace, $p = 2.05$ [47]. (**c**) p-Laplace, $p = 2.25$ [47]. (**d**) WNLL [111]. (**e**) High-order Laplace, $m = 2$ [128]. (**f**) PWLL [27, 98]

problematic, since (3.12) is well-posed only when Γ is large enough, and regular enough. This is due to the fact that the Sobolev function space $H^1(\Omega)$, like the $L^p(\Omega)$ function spaces, consists of functions that are defined only up to sets of measure zero. So the values of $u(x)$ at particular points $x \in \Omega$ are not well-defined. This means, for example, that Γ cannot be a collection of isolated points, as in Fig. 2a; otherwise, (3.12) is ill-posed. In order for (3.12) to be well-posed, the boundary set Γ must be *large enough* and sufficiently regular so that we can interpret what it means to *evaluate* $u(x)$ for $x \in \Gamma$. Such results are called *trace theorems* in the PDE literature, and the restriction $u|_\Omega$ is the trace of u on Γ; we refer to [46] for more details.

The last decade has seen a surge of interest in developing graph-based learning algorithms that are well-posed with arbitrarily small labeled data sets by considering PDE continuum limits that are insensitive to the labeled data set. One natural idea proposed in [40] is to consider the p-Dirichlet energy on the graph, given by

$$\sum_{x,y\in\mathcal{X}} w_{xy}|u(x) - u(y)|^p. \tag{3.13}$$

In fact, the p-Laplacian was suggested in many other works as well [1, 16, 40–43, 54, 66, 130] though not in the low label rate context. By a similar argument as above, the continuum limit energy should be the weighted p-Dirichlet energy

$$\int_\Omega \rho^2 |\nabla u|^p \, dx.$$

When $p > d$, we have the Sobolev embedding $W^{1,p}(\Omega) \subset C^{0,\alpha}(\Omega)$ [46], which ensures that solutions are Hölder continuous, spikes cannot form, and the continuum boundary value problem

$$\begin{cases} \operatorname{div}(\rho^2 |\nabla u|^{p-2} \nabla u) = 0, & \text{in } \Omega \setminus \Gamma \\ u = g, & \text{on } \Gamma, \end{cases} \tag{3.14}$$

is well-posed without any conditions on Γ. This well-posedness was established rigorously in the continuum limit in [112] when $p > d$, and the number of labeled points is finite and fixed as the number of unlabeled data points tends to infinity. In this case, the authors of [112] identified an additional restriction on the length scale $\varepsilon > 0$ that arises through the nonlocal nature of the graph Laplacian. Namely, in addition to $p > d$, it is necessary that $n\varepsilon^p \ll 1$ to ensure that spikes do not form. Since $n\varepsilon^d \gg \log n$ is necessary to ensure connectivity of the graph,[3] this condition can only be satisfied when $p > d$. The authors of [112] also proposed a modification of p-Laplace learning for which the condition $n\varepsilon^p \ll 1$ is not required, by essentially extending the labels to all nearest neighbors on the graph.

It is worthwhile to pause and note that the condition $n\varepsilon^p \ll 1$ comes from a simple balancing of energy between "spiky" functions and smooth functions on the graph. If we write (3.11) for the p-Laplacian, we see that the graph p-Dirichlet energy of a smooth function scales like $n^2\varepsilon^{p+d}$. On the other hand, the energy of a constant function with spikes at the labeled nodes is

$$\sum_{x\in\Gamma} \sum_{y\in\mathcal{X}} \eta\left(\frac{|x-y|}{\varepsilon}\right) |g(y) - c|^p \approx C|\Gamma| S^p n\varepsilon^d,$$

where C is a constant depending on η, $c = u(x)$ is the constant value of u, S is the size of the spikes, and $|\Gamma|$ is the number of labeled points, all provided the label rate is sufficiently low and the labeled points are sufficiently far apart. In order to ensure that a smooth interpolating function has less energy than a spiky one, in the setting where $|\Gamma|$ is constant with n and ε, we require

$$n^2\varepsilon^{p+d} \ll n\varepsilon^d \iff n\varepsilon^p \ll 1.$$

[3] The quantity $n\varepsilon^d$ is the average number of neighbors of any nodes on the graph, up to a constant.

If we allow the number of labels $|\Gamma|$ to depend on n and ε, then the condition is

$$n^2\varepsilon^{p+d} \ll |\Gamma| n\varepsilon^d \iff \beta := \frac{1}{n}|\Gamma| \gg \varepsilon^p.$$

The quantity β above is the *label rate*—the ratio of labeled data points to all data points—and simply energy balancing arguments show that the label rate β should satisfy $\beta \gg \varepsilon^p$ to ensure convergence to a well-posed continuum problem. The case of finite labels as $n \to \infty$ corresponds to $\beta = n^{-1}$, which recovers the condition $n\varepsilon^p \ll 1$. In [28], it was shown that $\beta \sim \varepsilon^2$ for $p = 2$ is the threshold for a spiky versus smooth continuum limit in a setting that allowed the labeled data set to grow as $n \to \infty$. The proofs in [28] used the random walk interpretation of the 2-Laplacian and martingale arguments. A negative result was also established in [28] for $p > 2$ showing that spikes develop when $\beta \ll \varepsilon^p$. However, a positive result for $p > 2$—that $\beta \gg \varepsilon^p$ is sufficient to prevent spikes from forming—is currently lacking, outside of the finite label regime covered in [112].

As in Sect. 2, we can take the limit as $p \to \infty$ of the graph p-Dirichlet energy. It is more illustrative to do this with the necessary conditions for minimizing the graph p-Dirichlet energy (3.13), which is the graph p-Laplace equation

$$\sum_{y \in \mathcal{X}} w_{xy}|u(x) - u(y)|^{p-2}(u(x) - u(y)) = 0 \tag{3.15}$$

for each $x \in \mathcal{X} \setminus \Gamma$. To take the limit as $p \to \infty$, we separate the terms in the summation above by sign and take the p^{th} root to obtain

$$\left(\sum_{y\,:\,g_{xy}>0}^{n} w_{xy}(u(x) - u(y))^{p-1}\right)^{1/p} = \left(\sum_{y\,:\,g_{yx}>0}^{n} w_{xy}(u(y) - u(x))^{p-1}\right)^{1/p}.$$

where $g_{xy} = w_{xy}(u(x) - u(y))$. The terms with $g_{xy} = 0$ do not contribute and can be neglected. We assume neither sum above is empty; otherwise, all terms in (3.15) are zero, which is trivial. Sending $p \to \infty$ yields the equation

$$\max_{y\,:\,g_{xy}>0}(u(x) - u(y)) = \max_{y\,:\,g_{yx}>0}(u(y) - u(x)).$$

The maximums above are unchanged by replacing the conditions $g_{xy} > 0$ and $g_{yx} > 0$ with $w_{xy} > 0$, as the graph is symmetric so $w_{xy} = w_{yx}$. Making this replacement, dividing by two on both sides, and rearranging yield

$$\mathcal{L}_\infty u(x) := u(x) - \frac{1}{2}\left(\max_{N_x} u + \min_{N_x} u\right) = 0, \tag{3.16}$$

where $\mathcal{L}_\infty$ is called the *graph* ∞*-Laplacian*, and we recall that N_x are the graph neighbors of x defined in (3.1). Notice the maximum and minimum above are over graph neighbors, which satisfy $w_{xy} > 0$, but that the weights in the graph do not enter directly into the operator otherwise. If we were to replace w_{xy} by w_{xy}^p in (3.15), then the weights would appear in the graph ∞-Laplacian, which is more informative of the graph structure and was the approach taken in [20]. We use the unweighted ∞-Laplacian here, since it connects more directly to the tug-of-war interpretation of the graph p-Laplacian, discussed below in this section. It turns out, similar to the continuum ∞-Laplace equation, that graph ∞-harmonic functions also solve L^∞-type variational problems on graphs (see, e.g., [71, 84]).

The graph ∞-Laplacian was first used for semi-supervised learning in [71, 84], where it is called *Lipschitz learning*. In [21], it was shown rigorously using PDE-continuum limits that solutions of the graph ∞-Laplace equation converged to ∞-harmonic functions in the continuum, even for finite isolated labeled data points. This established well-posedness of Lipschitz learning with arbitrarily few labeled examples. Subsequent work [18, 19] established convergence rates of graph ∞-harmonic functions to the continuum. As in Sect. 2, specifically (2.6), it is natural to define a p-Laplacian operator on the graph that is a combination of the 2-Laplacian and ∞-Laplacian. Here we use the definition

$$\mathcal{L}_p = \alpha \mathcal{L}_{rw} + (1-\alpha)\mathcal{L}_\infty, \tag{3.17}$$

where $\alpha = 1/(p-1)$ and $\alpha = 1$ when $p = \infty$. This type of graph p-Laplacian is often called the *game-theoretic* p-Laplacian due to its connection to tug-of-war games, as described in Sect. 2. The p-Laplace operator appearing earlier in (3.15), as the necessary conditions for the p-Dirichlet energy, is usually called the *variational* p-Laplacian and is a different operator on the graph, even though they agree in the continuum. The game-theoretic p-Laplacian on graphs originally appeared in [92] and was proposed for semi-supervised learning in [20], although the latter work considered a weighted version of the ∞-Laplacian. In [20], it was shown that the game-theoretic p-Laplacian is well-posed with arbitrarily few labeled examples when $p > d$,[4] by showing that the continuum limit was the well-posed continuum p-Laplace equation with $p > d$. An interesting and important detail is that the condition $n\varepsilon^p \ll 1$ is not required by the game-theoretic p-Laplacian due to the strong regularization provided by the ∞-Laplacian.

As in Sect. 2, we can develop a tug-of-war game interpretation for the game-theoretic graph p-Laplacian. Any function u on the graph satisfying $\mathcal{L}_p u = 0$ also satisfies, by rearranging (3.17), the equation

$$u(x) = \frac{\alpha}{d_x} \sum_{y \in \mathcal{X}} w_{xy} u(y) + \frac{1-\alpha}{2} \left(\max_{N_x} u + \min_{N_x} u \right), \tag{3.18}$$

[4] The definition of α differs by a constant in [20], so that p has the correct continuum interpretation.

which is the discrete graph version of the continuous identity (2.9), without the $O(\varepsilon)$ error term. As we did in Sect. 2, we can define a discrete stochastic process on the graph that produces the martingale property. We define $X_1, X_2, \dots$ so that, given X_k, the choice of X_{k+1} is with probability α a random walk step from X_k, with probability $\frac{1-\alpha}{2}$ the vertex y maximizing $u(y)$ over neighbors y of X_k and with probability $\frac{1-\alpha}{2}$ the corresponding minimizing vertex. For the same reasons as in Sect. 2, any graph p-harmonic function, satisfying $\mathcal{L}_p u(x) = 0$ for all i, is a martingale when applied to the process X_k, in the sense that

$$\mathbb{E}[u(X_{k+1}) \mid X_k] = u(X_k).$$

This allows us to apply martingale techniques to study properties of the solution to the game-theoretic p-Laplacian on the graph. The variational graph p-Laplacian (3.15) does not have any such stochastic tug-of-war interpretation. To our knowledge, no existing works have used the tug-of-war interpretation of the graph p-Laplacian to study properties of semi-supervised p-Laplacian learning.

Figure 2b and c show the results of the game-theoretic p-Laplacian applied to the toy two labeled point problem, illustrating how the p-Laplacian corrects the spike phenomenon in Laplace learning. Note we only need $p > 2$ here since $d = 2$. Many other approaches have been proposed for correcting the degeneracy in Laplace learning. In [111], the authors proposed to reweight the edges of the graph that connect to labels more strongly to discourage spike formation and to use the ordinary graph Laplacian on the reweighted graph. The method is called the Weighted Nonlocal Laplacian (WNLL). Figure 2d shows an example of the WNLL on the two-point problem, where we can see that the method essentially just extends the labels to nearest neighbors on the graph. However, it was shown in [24] that the WNLL method remains ill-posed at arbitrarily low label rates. In [24], the authors proposed the *properly weighted* graph Laplacian, which reweights the graph in a nonlocal way that is *singular* near the labeled data, so that the continuum limit is well-posed with arbitrarily few labels. In [128], it was proposed to use a higher-order Laplace equation of the form $\mathcal{L}^m u = 0$ for problems with very few labels. The idea here is that in the continuum, the variational formulation of the equation would involve $u \in H^m(\Omega)$, and when $m > \frac{d}{2}$, the Sobolev embedding $H^m \subset C^{0,\alpha}(\Omega)$ ensures that spikes do not form and the continuum limit is well-posed. Figure 2e shows the two-point problem with higher-order Laplacian regularization. The well-posedness of higher-order Laplace learning with finite labeled data in the continuum limit was only very recently addressed in [119]. Finally, there is recent work on using Poisson equations on graphs for semi-supervised learning in [25, 27, 98], some of which connects to the earlier work on the properly weighted Laplacian [24]. In particular, the Poisson Weighted Laplace Learning (PWLL) method from [27, 98] involves reweighting the graph using the solution of a graph Poisson equation, whose singularities are sufficient to invoke the results in [24], though the theory for these methods is currently still under development. Figure 2f shows the behavior of PWLL on the two-point problem. The volume-constrained MBO method based

on auction dynamics developed in [64] has also proven to be effective at low label rates, though we are not aware of any theory to explain this. Other methods at low label rates include Hamilton-Jacobi equations on graphs [22] and the *centered kernel method* [86–88]. We also mention here the work on MBO methods[5] for graph-based semi-supervised learning [14, 48, 62, 93–95] which consider approximations of the $p = 1$ Laplacian for graph-based learning. The MBO methods perform semi-supervised learning on graphs by alternating diffusion and thresholding to label vectors until convergence, which has the effect of finding the partition of the graph with the smallest perimeter that fits the labels correctly.

3.4 Consistency Versus Well-Posedness

Much of the work described above is aimed at establishing continuum PDE or variational descriptions of graph-based machine learning algorithms, which can, for example, prove that a method like p-Laplace learning provides a well-posed method for propagating labels on graphs that does not develop spikes or other degeneracies. Many other works have also considered the continuum limit of the graph Laplacian, its spectrum, and regularity properties [23, 26, 38, 50, 59, 60]. However, this abundance of work fails to address the problem of whether the machine learning methods work well or not; that is, they say little or nothing about whether the predicted label is correct! This is the question of *consistency*, which is more important and difficult, and thus less often studied.

As an example, in Fig. 3, we show the results of graph-based semi-supervised learning applied to the same two-moons example as in Fig. 1. The p-Laplace and higher-order Laplace methods are the only methods with rigorous well-posedness results with very few labels, and Poisson learning is expected to enjoy such results, which is the subject of a forthcoming paper by the first author. Laplace learning and WNLL are ill-posed in the low-label regime. The best performing method is Poisson learning [25], followed closely by high-order Laplace, which illustrates that well-posedness does not always go hand in hand with good consistency properties, or at least that the relation between consistency and having a well-posedness continuum limit is not clear or well-understood.

To address the consistency problem, one has to make an assumption on the underlying cluster structure of the graph, or the labeling function. Then the goal is to prove that the machine learning algorithm can identify the clusters or correct labels, under reasonable assumptions on the model and parameters. This kind of consistency analysis was carried out in the context of spectral clustering in [61], in which the probability density ρ was assumed to be highly concentrated on disjoint clusters. Spectral clustering uses the eigenvectors of the graph Laplacian

[5] MBO stands for Merriman-Bence-Osher, who originally proposed threshold dynamics approaches for numerically approximating mean curvature motion [96].

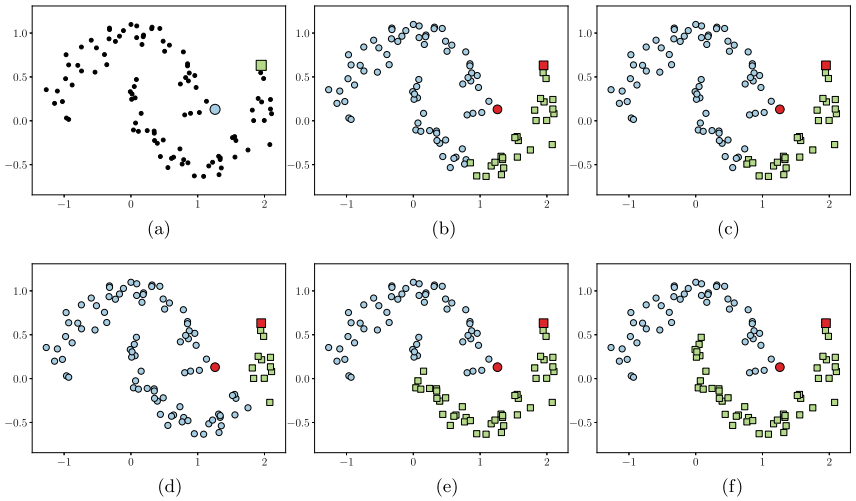

Fig. 3 Classification results using different graph-based semi-supervised learning algorithms on the two-moons data set. There are two classes, the upper moon and the lower moon, and the label function is binary $g(x) \in \{0, 1\}$. The red points are the only labeled data points. The p-Laplace method used $p = 3$, while high-order Laplace learning used order $m = 4$, both of which gave the best results over a reasonable search. (**a**) Data. (**b**) Laplace [135] (77%). (**c**) p-Laplace [47] (79%). (**d**) WNLL [111] (61%). (**e**) High-order Laplace [128] (94%). (**f**) Poisson [25] (100%)

for clustering [116] and are closely related to spectral methods in data science [9, 34, 37]. Another related work is [113], which assumes a well-separated mixture model for the data and shows that the spectral embedding reflects the clusters in the data [113]. Aside from these examples, there is a lack of consistency results for many graph-based semi-supervised learning methods, which should be viewed as an opportunity to utilize powerful PDE-continuum limit tools to establish important and interesting results about machine learning algorithms. We begin such a study in this paper.

3.5 Deep Semi-supervised Learning

There is a growing body of work on deep semi-supervised learning, which includes many methods based on graph neural networks; see [125] for a survey. The data science applications are somewhat different here; in addition to a graph, most problem formulations assume that each *node* $x \in \mathcal{X}$ in the graph has an associated feature vector $v(x) \in \mathbb{R}^k$. An example application would be a citation network, where each node in the graph is an academic paper and the edges represent citations between papers. The feature vector $v(x)$ for paper x may contain summary statistics of key words appearing in the abstract of the paper. An example task is to classify

the subject of each paper, using both the graph structure and the feature vectors, which in general provide complimentary information.[6]

Let us briefly describe how some graph neural networks work for semi-supervised learning. A standard feed-forward neural network, or multilayer perceptron, iteratively composes linear functions with nonlinear activation function, $u_{k+1} = \sigma(\Theta_k u_k + b_k)$, where $u_k \in \mathbb{R}^{n_k}$ is the input to the kth layer, $\Theta_k \in \mathbb{R}^{n_{k+1} \times n_k}$ is the weight matrix for the k^{th} layer, and $b_k \in \mathbb{R}^{n_{k+1}}$ the bias vector. A common choice for the activation σ is the rectified linear unit (ReLU) $\sigma(t) = \max\{t, 0\}$, but other choices are possible. The input to the network is u_1, and the output is u_L after L layers.

Graph neural networks incorporate the graph information into the neural network architecture. A very popular approach is the graph convolutional neural network (GCN) proposed in [68], which combines the standard feed-forward neural network with the graph structure as follows:

$$u_{k+1}(x) = \sigma\left(\frac{1}{d_x}\sum_{y\in\mathcal{X}} w_{xy}(\Theta_k u_k(y) + b_k)\right), \tag{3.19}$$

where $u_k : \mathcal{X} \to \mathbb{R}^{n_k}$ and Θ_k, b_k are as before.[7] The inputs are the node feature vectors $u_1(x) = v(x)$, and the output after L layers is, after thresholding, the network's prediction of the label for each node in the graph. This output is fed into a loss function that measures how well the network fits some training data, and the weights Θ_k and biases b_k are tuned by minimizing the loss function with gradient descent.

We note that the GCN layer (3.19) is essentially a combination of a feed-forward neural network with averaging over neighborhoods on the graph. These types of networks are also called *message passing* graph neural networks. The reader can compare this to label propagation, see (3.7), and notice that when σ is the identity $\sigma(t) = t$, $\Theta_k = I$, and $b_k = 0$, this appears to be exactly label propagation. The difference is that with graph neural networks, the operation acts on the *feature vectors* of the nodes, while label propagation acts directly on the labels, since feature vectors are not available. The graph convolutional layer (3.19) can be viewed as a special case of a more general class of of graph neural networks based on a spectral definition of convolution [35]. We also mention an important variant called graph attention networks [115], which allows the weights in (3.19) to be *learned* using an *attention mechanism*, in a similar way that transformers utilize the attention mechanism in large language models [114].

The analysis of graph neural network models like the GCN given in (3.19) would be substantially different from the analysis in this paper. The important problems

[6] A standard test data set called *PubMed* [124] uses exactly this setup.

[7] To be precise, the method in [68] divides by $\sqrt{d_x d_y}$ in the sum, instead of d_x, and adds self-loops to the graph.

concern how the *learned* weights Θ_k and biases b_k interact with the graph structure during training, especially for deeper networks, and whether the network relies more on the graph structure or the node features for classification. Although there is ostensibly a connection to label propagation (3.7), this is largely superficial, since GCNs (3.19) usually only have a handful of layers, say 2 or 3, while label propagation is often run for hundreds of steps or more, in order for the method to *converge* to the solution of Laplace's equation. Another difference with the setting of this paper is that many recent graph neural network approaches are aimed at tackling heterophily in graphs, in which adjacent nodes may very often belong to different classes [81, 122, 127, 137]. In this case, the semi-supervised smoothness assumption does not hold, and the question is how to combine information from both the feature vectors and graph structure, when neither on their own is highly informative.

There are, nevertheless, some settings where the analysis in this paper may be relevant in deep graph neural networks. Many recent works have shown that one can obtain effective methods by decoupling the graph diffusion or averaging in (3.19) from the neural network weights Θ_k and biases b_k [6, 36, 51, 63, 109]. One way to do this, proposed in [51], is to first run the feature vectors $v(x)$ for each node through a standard feed-forward neural network to make label predictions for each node x and then to diffuse those predictions over the graph before feeding them into a loss function. In the case of [51], the authors used the PageRank algorithm [52], which is essentially a graph reaction-diffusion equation [126], and the label predictions entered through the source term. The particular choice of PageRank is not essential, and other works have shown that various forms of graph diffusion can easily substitute [6, 109]. Thus, improvements in graph-based semi-supervised learning algorithms, especially theoretical guarantees, can have a direct impact on deep semi-supervised learning methods that decouple the graph from the neural network.

4 Consistency Results for the p-Laplacian

We present here a general framework for proving consistency results for graph-based semi-supervised learning with the game-theoretic p-Laplacian. The framework is based on the stochastic tug-of-war interpretation of the p-Laplacian on a graph that was introduced in Sect. 3. We illustrate how to use the framework to prove some preliminary consistency results on geometric graphs and stochastic block model graphs and then highlight some open research questions for future work.

As in Sect. 3, let $\mathcal{X} = \{x_1, \dots, x_n\}$ denote our data points, which are the vertices of the graph, and let $W = (w_{xy})_{x,y\in\mathcal{X}}$ be a symmetric (i.e., $w_{xy} = w_{yx}$) weight matrix which encodes the graph structure. We assume the graph is connected. We let $d_x = \sum_{y\in\mathcal{X}} w_{xy}$ be the degree of vertex $x \in \mathcal{X}$, and we recall that $N_x \subset \mathcal{X}$

denotes the graph neighbors of x, as defined in (3.1). Let $\ell^2(\mathcal{X})$ denote the space of functions $u : \mathcal{X} \to \mathbb{R}$, and define the game-theoretic p-Laplacian on the graph G is the operator $\mathcal{L}_p : \ell^2(\mathcal{X}) \to \ell^2(\mathcal{X})$ defined in (3.17) in Sect. 3, where $\alpha = 1/(p-1)$ and $p \geq 2$. Let us denote by $\Gamma \subset \mathcal{X}$ the labeled data set and $g : \Gamma \to \mathbb{R}$ the labels. The game-theoretic p-Laplacian regularized semi-supervised learning problem is given by the boundary value problem

$$\begin{cases} \mathcal{L}_p u = 0, & \text{in } \mathcal{X} \setminus \Gamma \\ u = g, & \text{on } \Gamma. \end{cases} \tag{4.1}$$

Since the graph is connected, the p-Laplace equation (4.1) can be shown to have a unique solution via the Perron method [20].

In this section, we address the question of *consistency* of p-Laplacian regularization. That is, supposing that g is the restriction to Γ of a label function on the whole graph $g : \mathcal{X} \to \mathbb{R}$ that is in some sense smooth with respect to the geometry of the graph, we aim to give conditions under which the solution u of (4.1) is close to g at the unlabeled vertices $\mathcal{X} \setminus \Gamma$. In other words, consistency asks the question: Under what conditions on the underlying label function g, the graph structure W, and the label set Γ does the solution of p-Laplacian regularization make the correct label predictions? In this paper, we consider the *noise-free* setting, where we observe the true labels without any noise or corruption. It would be interesting to consider the setting of noisy labels in future work; we discuss this more in Sect. 5.

In consistency of graph-based learning, it is essential that the underlying label function g is in some way regular or smooth with respect to the graph structure. In other words, there should be some degree of correlation between the labels of vertices and their nearby neighbors in the graph topology. Otherwise graph-based learning is not useful and would not be expected to yield better results than fully supervised learning that ignores the graph structure. In this section, we present results for two different types of graph structures. In Sect. 4.2, we present results for geometric graphs, where the vertices are sampled from a Euclidean domain and nearby points are connected by edges. In this case, we assume that g has some degree of regularity with respect to the underlying Euclidean geometry. In Sect. 4.3, we present results for classification on stochastic block model graphs, where there is no continuous geometric structure (SBM graphs can be thought of as having a discrete geometric structure, but this is not directly related to continuous PDEs). In this case, the label function is concentrated on the blocks, and only changes between blocks where there are relatively few edges.

4.1 Results on General Graphs

Before presenting our main results in Sects. 4.2 and 4.3, we present our general stochastic tug-of-war framework on general graphs in Sect. 4.1.1, which amounts to

selecting suboptimal strategies in the tug-of-war game to produce tractable upper and lower bounds on the solution u of the p-Laplace equation (4.1). It is interesting to remark that we do not explicitly use the symmetry assumption $w_{xy} = w_{yx}$ in our results, and our results hold with minor modifications to the quantities involved for directed (i.e., nonsymmetric) graphs.

4.1.1 The Tug-of-War Lemma

In this section, we present our main tug-of-war result (Lemma 4.3), which uses the stochastic tug-of-war game and martingale techniques to bound the difference $u(x) - g(x)$, where u solves (4.1). These techniques are later applied to geometric graphs in Sect. 4.2 and to stochastic block model (SBM) graphs in Sect. 4.3. The tug-of-war game for the p-Laplacian is based on the following dynamic programming principle associated with the p-Laplacian, which was discussed and established in Sect. 3.

Proposition 4.1 (Dynamic Programming Principle) *If $u \in \ell^2(\mathcal{X})$ and $x \in \mathcal{X}$ such that $\mathcal{L}_p u(x) = 0$ then*

$$u(x) = \frac{\alpha}{d_x} \sum_{y \in \mathcal{X}} w_{xy} u(y) + \frac{1-\alpha}{2} \left(\min_{N_x} u + \max_{N_x} u \right), \tag{4.2}$$

where we recall that $\alpha = 1/(p-1)$ and $p \geq 2$.

We call Eq. (4.2) a dynamic programming principle (DPP), because it expresses u at a point via u at nearby points and is closely related to the DPPs that appear in optimal control problems (see, e.g., [7]). In fact, the DPP arises from the stochastic two player tug-of-war game described in Sect. 3. The game is played between two players Paul and Carol. The game ends when the token of the game arrives at a point $x \in \Gamma$, and Paul pays Carol $g(x)$. Thus, Paul wants the game to end where g is smallest, while Carol wants the opposite. At each step of the game, with probability α, the token moves from its current position x to a neighbor $y \in N_x$ via a single step of a random walk on the graph (i.e., the next position is chosen randomly according to the distribution $\mathbb{P}(y) = w_{xy}/d_x$). With probability $(1-\alpha)/2$, Paul chooses the next position of the token, and with probability $(1-\alpha)/2$, Carol chooses the next position. We will use this interpretation in the proofs of our main results, without explicitly writing down the game or the appropriate spaces of strategies.

In order to use the tug-of-war game, we define a stochastic process that corresponds to a *suboptimal* strategy for Paul. We recall that $\alpha = 1/(p-1)$ and pick any $p \geq 2$. Let $u \in \ell^2(\mathcal{X})$ and $x \in \mathcal{X}$, and consider the following stochastic process defined on $\mathcal{X}$ associated with u and an initial point $x \in \mathcal{X}$. We define the sequence of random variables X_0, X_1, X_2 as follows. We first set $X_0 = x$. Then, conditioned on X_i, we choose X_{i+1} as follows: First, if $X_i \in \Gamma$, then we set $X_{i+1} = X_i$,

so the process gets *stuck* when it hits Γ. If $X_i \notin \Gamma$, then we choose X_{i+1} by the following:

1. With probability α, we take an independent random walk step, that is
$$\mathbb{P}(X_{i+1} = y \mid X_i = x) = \frac{w_{xy}}{d_X}.$$
2. With probability $\frac{1}{2}(1-\alpha)$, we choose $X_{i+1} \in \operatorname*{argmax}_{y \in N_{X_i}} u$.
3. With probability $\frac{1}{2}(1-\alpha)$, we choose $X_{i+1} \in \operatorname*{argmin}_{y \in N_{X_i} \cap \Gamma} g$, when $N_{x_i} \cap \Gamma \neq \varnothing$, and $X_{i+1} \in \operatorname*{argmin}_{y \in N_{X_i}} u$, when $N_{x_i} \cap \Gamma = \varnothing$.

All the probabilistic ingredients are chosen independently. When there are multiple choices for X_{i+1} in steps 2 and 3, the particular choice is irrelevant to the analysis, and any concrete choice will do (e.g., say, choose the vertex x_i with the smallest index). The stochastic process $X_0, X_1, X_2, \ldots$, defined above can be interpreted as a realization of the two-player game where Carol plays optimally and our choice of strategy for Paul is to move to Γ as soon as possible, which is the *suboptimal* part of his strategy.

We now establish the sub-martingale property corresponding to our stochastic process, which is a key ingredient in our analysis.

Lemma 4.2 *Let u be the solution to (4.1). Then the random variables $Z_i = u(X_i)$ form a sub-martingale with respect to X_i, that is $\mathbb{E}[Z_{i+1} \mid X_i] \geq Z_i$.*

Proof Conditioned on $X_i \in \Gamma$, we have $\mathbb{E}[u(X_{i+1}) \mid X_i] = u(X_i)$, since $X_{i+1} = X_i$. Conditioned on $X_i \notin \Gamma$, we further condition on steps 1–3 in the definition of X_{i+1} and use $u = g$ on Γ to obtain

$$\mathbb{E}[u(X_{i+1}) \mid X_i] = \frac{\alpha}{d_{X_i}} \sum_{y \in \mathcal{X}} w_{X_i,y} u(y) + \frac{1-\alpha}{2} \left(\max_{N_{X_i}} u + \min_{V} u(y) \right), \tag{4.3}$$

where $V = N_{x_i}$ when $N_{X_i} \cap \Gamma = \varnothing$ and $V = N_{X_i} \cap \Gamma$ when $N_{X_i} \cap \Gamma \neq \varnothing$. Using Proposition 4.1, we have

$$\mathbb{E}[u(X_{i+1}) \mid X_i] \geq \frac{\alpha}{d_{X_i}} \sum_{y \in \mathcal{X}} w_{X_i,y} u(y) + \frac{1-\alpha}{2} \left(\max_{N_{X_i}} u + \min_{N_{X_i}} u \right) = u(X_i),$$

which completes the proof. □

We now give the main tug-of-war lemma, which uses the sub-martingale property to bound $u(x)$.

Lemma 4.3 *Let u be the solution to (4.1). Define the stopping time*

$$\tau = \inf\{i \geq 0 \,:\, X_i \in \Gamma\}. \tag{4.4}$$

Then for any $x \in \mathcal{X}$, it holds that

$$u(x) \leq \mathbb{E}[g(X_\tau) \,|\, X_0 = x]. \tag{4.5}$$

Proof By Lemma 4.2, $Z_i = u(X_i)$ is a sub-martingale, and so by Doob's Optional Stopping Theorem for sub-martingales, see, e.g., [120]; we have

$$\begin{aligned} u(x) &= \mathbb{E}[Z_0 \,|\, X_0 = x] \leq \mathbb{E}[Z_\tau \,|\, X_0 = x] = \mathbb{E}[u(X_\tau) \,|\, X_0 = x] \\ &= \mathbb{E}[g(X_\tau) \,|\, X_0 = x], \end{aligned} \tag{4.6}$$

since $X_\tau \in \Gamma$. □

Let us give a brief preview of how Lemma 4.3 is used. We can subtract $g(x)$ from both sides of (4.5) to obtain

$$|u(x) - g(x)| \leq \mathbb{E}[|g(X_\tau) - g(X_0)| \,|\, X_0 = x].$$

If g is Lipschitz continuous, which is a reasonable assumption on a *continuous geometric* graph, then the right-hand side is $O(|X_\tau - X_0|)$, and so bounding $|u(x) - g(x)|$ amounts to estimating the stopping time τ. In this context, it is not especially important that Paul chooses the point in $N_{x_i} \cap \Gamma$ that minimizes g, and the game could be modified so Paul picks, say, a random boundary point. However, for SBM graphs, the arguments are different, and the key is to allow the *noise* to exit the game, since the noise sees the block structure. In the SBM setting, it is important for Paul to choose the minimizer of g over $N_{x_i} \cap \Gamma$, since g cannot be Lipschitz in any sense on an SBM graph. The interested reader can skip to Sect. 4.3 for the SBM analysis, which is independent of much of the geometric results below.

Throughout the rest of the paper, we make the strong, but simplifying, assumption that every vertex $x \in \mathcal{X}$ has a neighbor $y \in N_x$ with $y \in \Gamma$, meaning:

(A1) We assume that for every $x \in \mathcal{X}$

$$\Gamma \cap N_x \neq \varnothing.$$

Assumption **(A1)** is a simplifying assumption that makes much of the analysis tractable. Essentially, it allows us to estimate the stopping time τ defined in (4.4) in many different settings by allowing Paul to end the game quickly. Depending on the graph model, assumption **(A1)** is a fairly strong assumption on both the *label rate* and the *graph structure*. In Sects. 4.2 and 4.3, we will give conditions under which **(A1)** holds for geometric and SBM graphs. The results in this paper are preliminary, and we discuss the importance of relaxing assumption **(A1)** in Sect. 5. This seems

to us to be a nontrivial task that will require fine estimates on martingale stopping times.

4.1.2 General Consistency Results

Here, we use the martingale lemma, Lemma 4.3, to establish some general consistency results, which will rely on a Lipschitz or bounded derivative condition on the label function g. Thus, for $g : \mathcal{X} \to \mathbb{R}$, we need to construct a notion of gradient that is consistent with the gradient in the continuum when working with geometric graphs with a length scale ε. Given a graph described by a weight matrix W, we let $\mathrm{diam}(W)$ denote the *diameter* of the graph, which is defined as the largest number of hops required to get from any node to any other node in the graph. When the graph is a random geometric graph with bandwidth $\varepsilon > 0$, then each hop travels at most distance ε, and so we would expect to have $\mathrm{diam}(W) = O(\varepsilon^{-1})$. This motivates the definition of the *graph length scale*.

Definition 4.4 Given a weighted graph W, the *graph length scale* ε_W is defined as

$$\varepsilon_W = \frac{1}{\mathrm{diam}(W)}.$$

We can now define the gradient of g as

$$\nabla g(x, y) = \frac{g(x) - g(y)}{\varepsilon_W},$$

for any $x, y \in \mathcal{X}$ connected by an edge, so $w_{xy} > 0$. We also define

$$\|g\|_\infty = \max_{x \in \mathcal{X}} |g(x)|,$$

and

$$\|\nabla g\|_\infty = \max_{x,y \in X} \left\{ |\nabla g(x, y)| \, \mathbb{1}_{w_{xy}>0} \right\}.$$

This is to say, the norm $\|\nabla g\|_\infty$ is the maximal absolute difference of g across all edges in the graph divided by the graph length scale ε_W.

Define the smallest *weighted* ratio of labeled neighbors to neighbors to be

$$\delta := \min_{x \in X} \frac{\Sigma_{y \in \mathcal{X}} w_{xy} \mathbb{1}_{y \in \Gamma}}{\Sigma_{z \in \mathcal{X}} w_{xz}}. \tag{4.7}$$

Note that if **(A1)** holds, then $\delta > 0$.

The following theorem gives our first consistency result on general graphs.

Theorem 4.5 *Assume **(A1)** holds. Then, for any $x \in \mathcal{X}$,*

$$|u(x) - g(x)| \leq \left(\frac{2\log(\|g\|_\infty \|\nabla g\|_\infty^{-1} \varepsilon_W^{-1})}{1 - \alpha + 2\alpha\delta} + 2 \right) \|\nabla g\|_\infty \varepsilon_W. \tag{4.8}$$

Proof Throughout the proof, let us write $\zeta = \|\nabla g\|_\infty \varepsilon_W$ for simplicity. Define the stopping time τ as in (4.4). By Lemma 4.3, we have that

$$u(x) - g(x) \leq \mathbb{E}[g(X_\tau) - g(x) \,|\, X_0 = x].$$

Let us estimate the right-hand side above by fixing some k (which we will choose shortly) and conditioning on $\tau > k$ or $\tau \leq k$. For simplicity of notation, we will drop the conditioning $X_0 = x$ below. Then we have

$$\begin{aligned} u(x) - g(x) \leq{}& \mathbb{E}[g(X_\tau) - g(x) \,|\, \tau \leq k]\mathbb{P}(\tau \leq k) \\ &+ \mathbb{E}[g(X_\tau) - g(x) \,|\, \tau > k]\mathbb{P}(\tau > k). \end{aligned} \tag{4.9}$$

First, we estimate the probability $\mathbb{P}(\tau > k)$. Since each vertex has a labeled neighbor (by Assumption **(A1)**), each step has probability at least $\frac{1}{2}(1-\alpha) + \alpha\delta$ of hitting Γ and exiting, so a general estimate is

$$\mathbb{P}(\tau > k) \leq \left[1 - \tfrac{1}{2}(1-\alpha) - \alpha\delta\right]^k \leq e^{-\frac{k}{2}(1-\alpha) - \alpha\delta k}. \tag{4.10}$$

For any k, we can use a telescoping series to obtain

$$g(X_k) - g(x) = \sum_{j=0}^{k-1} (g(X_{j+1}) - g(X_j)) \leq \sum_{j=0}^{k-1} \|\nabla g\|_\infty \varepsilon_W = k\zeta.$$

It follows that

$$\mathbb{E}[g(X_\tau) - g(x) \,|\, \tau \leq k] \leq k\zeta. \tag{4.11}$$

Substituting (4.10) and (4.11) in (4.9), we have that

$$\begin{aligned} u(x) - g(x) &\leq k\zeta \mathbb{P}(\tau \leq k) + \mathbb{E}[g(X_\tau) - g(x) \,|\, \tau > k] e^{-\frac{k}{2}(1-\alpha) - \alpha\delta k} \\ &\leq k\zeta + 2\|g\|_\infty e^{-\frac{k}{2}(1-\alpha) - \alpha\delta k}. \end{aligned} \tag{4.12}$$

We want to balance the two terms in the right-hand side of (4.12), so we choose k so that

$$\zeta = \|g\|_\infty e^{-\frac{(1-\alpha)k}{2} - \alpha\delta k},$$

or equivalently

$$k = \frac{2\log(\|g\|_\infty/\zeta)}{1-\alpha+2\alpha\delta}.$$

Therefore, we have

$$u(x) - g(x) \le \zeta(k+2) = \left(\frac{2\log(\|g\|_\infty/\zeta)}{1-\alpha+2\alpha\delta} + 2\right)\zeta. \tag{4.13}$$

To estimate $g-u$, we use a similar argument, where we replace u by $-u$ and g by $-g$. This concludes the proof. □

It turns out we can use the same argument to establish a gradient estimate on the solution u of the p-Laplace equation, which shows that the gradient of u is controlled by the gradient of g.

Theorem 4.6 *Assume **(A1)** holds, and let x and y be two neighbors on the graph. Then,*

$$\|\nabla u\|_\infty \le \left(\frac{4\log(\|g\|_\infty \|\nabla g\|_\infty^{-1}\varepsilon_W^{-1})}{1-\alpha+2\alpha\delta} + 5\right)\|\nabla g\|_\infty.$$

Proof We apply the triangle inequality

$$|u(x)-u(y)| \le |u(x)-g(x)| + |u(y)-g(y)| + |g(x)-g(y)|$$

and use the bound in Theorem 4.5 twice, noting that $g(x)-g(y) = \nabla g(x,y)\varepsilon_W$. □

4.1.3 Vertex Classification Consistency Results

We now turn to vertex classification results, where the label function $g : \mathcal{X} \to \mathbb{R}$ takes constant values, and hence it will have sharp transitions across edges in the graph. Thus, $\|\nabla g\|_\infty$ may not be small, and so Theorems 4.5 and 4.6 are not applicable. We will focus on binary classification, where every point $x \in \mathcal{X}$ has a label zero or a label one, so $g : \mathcal{X} \to \{0,1\}$. To use the p-Laplace equation (4.1), we solve the equation with the binary values for the boundary condition $g : \Gamma \to \{0,1\}$ and threshold the solution $u(x)$ at $\frac{1}{2}$ to obtain a binary label prediction for each vertex $x \in \mathcal{X} \setminus \Gamma$. Thus, we make the following definition.

Definition 4.7 Given $g : \mathcal{X} \to \{0,1\}$, a point $x \in \mathcal{X}$ is classified *correctly* if $g(x) = \mathbb{1}_{u(x)\ge\frac{1}{2}}$, where u is the solution to (4.1).

In this section, we focus on identifying which points in the graph are classified correctly. This requires a few definitions.

Definition 4.8 A path on the graph is a sequence of edges which joins a sequence of vertices. The length of a path is the number of edges the path consists of.

Definition 4.9 The distance $\text{dist}(x, y)$ between two vertices $x, y \in \mathcal{X}$ is the length of the shortest path connecting the vertices. The distance between a vertex x and a set of vertices $A \subset \mathcal{X}$, denoted $\text{dist}(x, A)$, is the smallest distance between x and any of the vertices $y \in A$.

For $i = 0, 1$, we define

$$\mathcal{X}_i = \{x \in \mathcal{X} : g(x) = i\} \tag{4.14}$$

and note that $\mathcal{X} = \mathcal{X}_0 \cup \mathcal{X}_1$. For any integer $m \geq 0$ and $i = 0, 1$, we define

$$A_m^i = \{x \in \mathcal{X}_i : \text{dist}(x, \mathcal{X}_{1-i}) \leq m\},$$

and note that $A_m = A_m^0 \cup A_m^1$. That is to say, A_m^0 is the set of points from class 0 that are at most m steps away from the set of class 1 points and vice versa for A_m^1. This means that when m is small, the set A_m contains points that are sufficiently close to the boundary between the two classes that we may expect data points in A_m to be misclassified.

Observe that for any $x \in \mathcal{X}$, there is a positive probability of at least $\frac{1-\alpha}{2} + \alpha\delta = \frac{p-2+2\delta}{2(p-1)}$ of hitting the boundary Γ.. Even when $p = 2$,, this probability is positive, since it is exactly $\delta > 0$. We use this to establish the classification bound below, which shows that whenever a vertex x is sufficiently far, in terms of graph distance, away from the true decision boundary, it will be classified correctly.

Theorem 4.10 *Assume **(A1)**. Let k be the smallest integer strictly larger than*

$$\frac{2\log(2)}{1-\alpha+2\alpha\delta} = \frac{2(p-1)\log(2)}{p-2+2\delta}. \tag{4.15}$$

Then every $x \notin A_k$ is classified correctly.

Proof Let $x \in \mathcal{X} \setminus A_k$, and assume $X_0 = x$. Define the stopping time τ as in (4.4). For any point $x \in \mathcal{X}$, we have that

$$\begin{aligned} u(x) - g(x) \leq\ & \mathbb{E}[g(X_\tau) - g(x) \,|\, \tau \leq k]\mathbb{P}(\tau \leq k) \\ & + \mathbb{E}[g(X_\tau) - g(x) \,|\, \tau > k]\mathbb{P}(\tau > k). \end{aligned} \tag{4.16}$$

We want to classify x correctly whether it belongs to $\mathcal{X}_0$ or $\mathcal{X}_1$. Recall that the labels of vertices in $\mathcal{X}_0$ are zeros and in $\mathcal{X}_1$ are ones.

Since $x \notin A_k$, the labels of vertices reached by the stopping time $\tau \leq k$ can only be labels from the same class, so $g(X_\tau) = g(x)$. Thus, when $\tau \leq k$, the first term on the right-hand side of (4.16) is zero. Then, using that g is between zero and one, and that Eq. (4.10) holds, we obtain

$$u(x) - g(x) \leq \mathbb{E}[g(X_\tau) - g(x) \mid \tau > k]\mathbb{P}(\tau > k) \leq \mathbb{P}(\tau > k) \leq e^{-\frac{k}{2}(1-\alpha)-\alpha\delta k} \tag{4.17}$$

The analogous estimate on $g - u$ is similarly obtained, and so we have

$$|u(x) - g(x)| \leq e^{-\frac{k}{2}(1-\alpha)-\alpha\delta k}.$$

For *correct* classification of x, we need precisely $|u(x) - g(x)| < \frac{1}{2}$, which is equivalent to

$$e^{-\frac{k}{2}(1-\alpha)-\alpha\delta k} < \frac{1}{2},$$

and is satisfied by the choice of k in (4.15)

□

4.2 Geometric Graphs

In this section, we specialize our results to geometric graphs, which include the very commonly used random geometric ε-graphs and k-nearest neighbor (k-NN) graphs. The purpose of this section is to provide conditions under which ε-graphs and k-NN graphs, along with other graphs with geometric ε-scaling, fulfill the general assumption **(A1)** in Sect. 4.1.1, so that we can apply the theory developed in Sect. 4.1.1 to these graphs.

4.2.1 Main Assumptions

In order to construct ε-graphs and k-NN graphs and construct the labeled data set Γ, we need the following assumptions. Note that assumptions **(A2)**, **(A3)**, and **(A4)** also appear in [28].

(A2) Let $d \geq 2$ and let $\Omega \subset \mathbb{R}^d$ be an open, bounded, and connected domain with a Lipschitz boundary. Let $x_1, x_2, \ldots, x_n$ be *i.i.d.* with continuous density ρ satisfying

$$0 < \rho_{\min} := \inf_{x\in\Omega} \rho(x) \leq \sup_{x\in\Omega} p(x) =: \rho_{\max} < \infty.$$

We define $\mathcal{X} = \{x_1, x_2, \ldots, x_n\}$.

(A3) We construct Γ in the following way. Let $g \in \mathrm{Lip}(\overline{\Omega})$, where Ω is defined as in **(A2)**. Let $\beta \in (0, 1)$. For each $x \in \mathcal{X}$, define an independent random variable $z(x) \sim Bernoulli(\beta)$, and assign $x \in \Gamma$ whenever $z(x) = 1$. Each training data point $x \in \Gamma$ is assigned a label $g(x)$.

(A4) We define an interaction potential $\eta : [0, \infty) \to [0, \infty)$. The interaction potential $\eta(\cdot)$ has support on the interval $[0, 1]$, it is nonincreasing, and η restricted to $[0, 1]$ is Lipschitz continuous. For $t \geq 1$ we extend η so that $\eta(t) = 0$. Moreover, $\eta(\frac{1}{2}) > 0$ and η integrate to 1 :

$$\int_{\mathbb{R}^d} \eta(|x|)dx = 1.$$

We define $\eta_\varepsilon(t) = \frac{1}{\varepsilon^d}\eta(t/\varepsilon)$.

(A5) (Geometric graphs of length ε) There exist positive constants $c_1, c_2, \varepsilon_1, \varepsilon_2$ such that for every $x, y \in \mathcal{X}$ we have that

$$c_1\varepsilon_1^{-d}\mathbb{1}_{|x-y|<\varepsilon_1} \leq w_{xy} \leq c_2\varepsilon_2^{-d}\mathbb{1}_{|x-y|<\varepsilon_2}.$$

We note that assumptions **(A2)** and **(A4)** are used in the $\varepsilon-$graph construction (see Sect. 4.2.2). In Lemma 4.12, we prove that assumption **(A5)** holds for ε-graphs. Assumption **(A3)** has to do with how the labeled data points are chosen, and we note that the parameter $\beta \in (0, 1)$ is the *label rate*.

Assumption **(A2)** is satisfied for k-NN-graphs by construction. In Lemma 4.16, we prove that assumption **(A5)** holds with high probability for k-NN graphs. Applying **(A5)**, we prove **(A1)** holds with high probability, which means that Theorems 4.5 and 4.6 hold. For $x \in \mathcal{X}$ we define $d_n(x)$ and $p_n(x)$ as follows:

$$d_n(x) = \Sigma_{y\in\mathcal{X}} w_{xy}, \tag{4.18}$$

$$p_n(x) = \Sigma_{y\in\mathcal{X}} w_{xy}\mathbb{1}_{y\in\Gamma}. \tag{4.19}$$

Then, as before, we define

$$\delta = \min_{x\in\mathcal{X}} \frac{p_n(x)}{d_n(x)} = \min_{x\in\mathcal{X}} \frac{\Sigma_{y\in\mathcal{X}} w_{xy}\mathbb{1}_{y\in\Gamma}}{\Sigma_{z\in\mathcal{X}} w_{xz}}. \tag{4.20}$$

4.2.2 The ε-Graphs

Let $0 < \varepsilon \leq 1$. We construct our ε-graph as follows.

Definition 4.11 We define an ε-graph according to the following rules. The vertices $\mathcal{X} = \{x_1, \ldots x_n\}$ are constructed according to **(A2)**, and the labeled set Γ is chosen according to **(A3)**. We assume **(A4)**, and the symmetric edges $w_{xy} = w_{yx}$ are obtained as follows:

$$w_{xy} = \eta_\varepsilon(|x - y|) \tag{4.21}$$

Lemma 4.12 *ε-Graphs satisfy assumption (A5) with constants $c_1 = 2^{-d}\eta(\frac{1}{2})$, $c_2 = \sup \eta$, $\varepsilon_1 = \frac{\varepsilon}{2}$, and $\varepsilon_2 = \varepsilon$.*

Proof By their construction, $\varepsilon-$graphs have the following property:

$$w_{xy} = \varepsilon^{-d}\eta\left(\frac{|x-y|}{\varepsilon}\right) \in [\varepsilon^{-d}\eta(\tfrac{1}{2})\mathbb{1}_{|x-y|<\frac{\varepsilon}{2}}, \varepsilon^{-d}\sup\eta\mathbb{1}_{|x-y|<\varepsilon}] \tag{4.22}$$

$$\equiv [(\varepsilon/2)^{-d}(2^{-d}\eta(\tfrac{1}{2}))\mathbb{1}_{|x-y|<\frac{\varepsilon}{2}}, \varepsilon^{-d}\sup\eta\mathbb{1}_{|x-y|<\varepsilon}]. \tag{4.23}$$

So **(A5)** holds, with the constants stated in the Lemma. □

4.2.3 The k-Nearest Neighbors Graphs

We follow [23] in constructing a k-NN graph. We fix a number $k \in \mathbb{N}$, such that $1 \le k \le n-1$. The number of neighbors $N_\varepsilon(x)$ of x in an ε-ball is

$$N_\varepsilon(x) := \sum_{j:0<|x_j-x|\le\varepsilon} 1$$

and the distance to the k-nearest neighbor is

$$\varepsilon_k(x) := \min\{\varepsilon > 0 \ : N_\varepsilon(x) \ge k\}.$$

Definition 4.13 We define a relation $\sim_k$ on $\mathcal{X} \times \mathcal{X}$ by declaring

$$x_i \sim_k x_j$$

if x_j is among the k nearest neighbors of x_i.

Definition 4.14 For the *symmetric* k-nearest neighbor graph, we construct the vertices according to **(A2)**. If $x_i \sim_k x_j$ **or** $x_j \sim_k x_i$, we place an edge between x_i and x_j. The edges are symmetric, meaning $w_{xy} = w_{yx}$ and

$$w_{xy} = \eta_{\max\{\varepsilon_k(x),\varepsilon_k(y)\}}(|x-y|).$$

Definition 4.15 In the *mutual* k-nearest neighbor graph, we construct the vertices according to **(A2)**. If $x_i \sim_k x_j$ **and** $x_j \sim_k x_i$, we place an edge between x_i and x_j. The edges are symmetric, meaning $w_{xy} = w_{yx}$ and

$$w_{xy} = \eta_{\min\{\varepsilon_k(x),\varepsilon_k(y)\}}(|x-y|).$$

In the following lemma, we verify that **(A5)** holds for k-NN graphs.

Lemma 4.16 *There exists a $c > 0$, such that symmetric k-NN graphs and mutual k-NN graphs satisfy assumption **(A5)** with probability $1 - 4\exp(-\frac{c}{2}(k/n)^{2/d}k)$, provided k is not too large, meaning*

$$1 \leq k \leq cn\varepsilon^d$$

holds.

Proof Consider $x \in \mathcal{X}$—a vertex in the $k-$NN graph. We set

$$\gamma = \tfrac{1}{\sqrt{2}}(k/n)^{1/d}.$$

Let α_d be the volume of the d-dimensional Euclidean unit ball.

We apply Lemma 3.8 from [23] to estimate the expected value of the number of neighbors: we have that

$$\mathbb{P}(|\alpha_d \rho(x) n \varepsilon_k(x)^d - k| \geq \tfrac{1}{2}k) \leq 4\exp(-c\gamma^2 k).$$

Thus, with probability $1 - 4\exp(-c\gamma^2 k)$, we obtain

$$\alpha_d \rho(x) n \varepsilon_k^d \in \left[\tfrac{k}{2}, \tfrac{3k}{2}\right],$$

or equivalently

$$\left(\frac{k}{2\alpha_d \rho(x) n}\right)^{1/d} \leq \varepsilon_k \leq \left(\frac{3k}{2\alpha_d \rho(x) n}\right)^{1/d}.$$

Denote $m := \inf_{x\in X}(\frac{k}{2\alpha_d \rho(x)n})^{1/d}$ and $M := \sup_{x\in X}(\frac{3k}{2\alpha_d \rho(x)n})^{1/d}$. Thus, with probability at least $1 - 4\exp(-c\gamma^2 k)$, we have:

$$m \leq \varepsilon_k \leq M.$$

In order to obtain the constants in **(A5)**, we bound w_{xy} from below and above. Note that ε_k comes from the vertex x or the vertex y and that η is a nonincreasing function.

We start with the lower bound of w_{xy}:

$$\begin{aligned} w_{xy} = \frac{1}{\varepsilon_k^d}\eta\left(\frac{|x-y|}{\varepsilon_k}\right) &= \frac{1}{\varepsilon_k^d}\eta\left(\frac{|x-y|}{\varepsilon_k}\right)\mathbb{1}_{|x-y|\leq\varepsilon_k} \\ &= \frac{1}{\varepsilon_k^d}\eta\left(\frac{|x-y|}{\varepsilon_k}\right)\mathbb{1}_{|x-y|\leq m/2} \\ &\quad + \frac{1}{\varepsilon_k^d}\eta\left(\frac{|x-y|}{\varepsilon_k}\right)\mathbb{1}_{|x-y|\in[m/2,\varepsilon_k]} \end{aligned}$$

$$
\begin{aligned}
&\geq \frac{1}{\varepsilon_k^d}\eta\left(\frac{m/2}{\varepsilon_k}\right)\mathbb{1}_{|x-y|\leq m/2} \\
&\geq \frac{1}{\varepsilon_k^d}\eta\left(\frac{1}{2}\right)\mathbb{1}_{|x-y|\leq m/2} \\
&= \eta\left(\frac{1}{2}\right)\left(\frac{m}{2\varepsilon_k}\right)^d\left(\frac{m}{2}\right)^{-d}\mathbb{1}_{|x-y|\leq m/2} \\
&\geq \eta\left(\frac{1}{2}\right)\left(\frac{m}{2M}\right)^d\left(\frac{m}{2}\right)^{-d}\mathbb{1}_{|x-y|\leq m/2}.
\end{aligned}
$$

We choose $\varepsilon_1 = \frac{m}{2}$ and $c_1 = \eta(\frac{1}{2})(\frac{m}{2M})^d$.

Next, we compute the upper bound for the edge weight:

$$
\begin{aligned}
w_{xy} = \frac{1}{\varepsilon_k^d}\eta\left(\frac{|x-y|}{\varepsilon_k}\right) &= \frac{1}{\varepsilon_k^d}\eta\left(\frac{|x-y|}{\varepsilon_k}\right)\mathbb{1}_{|x-y|\leq\varepsilon_k} \\
&\leq \varepsilon_k^{-d}\max\eta\,\mathbb{1}_{|x-y|\leq\varepsilon_k} \\
&= M^{-d}\left(\frac{M}{\varepsilon_k}\right)^d\max\eta\,\mathbb{1}_{|x-y|\leq\varepsilon_k} \\
&\leq M^{-d}\left(\frac{M}{m}\right)^d\max\eta\,\mathbb{1}_{|x-y|\leq M} \\
&\leq M^{-d}\mathbb{1}_{|x-y|\leq M}\left\{\left(\frac{M}{m}\right)^d\max\eta\right\}.
\end{aligned}
$$

Now, we choose $c_2 = \left\{\left(\frac{M}{m}\right)^d\max\eta\right\}$ and $\varepsilon_2 = M$. This establishes all the constants in **(A5)**, which concludes the proof. □

4.2.4 General Geometric Graphs

Now that we have shown that ε-graphs and k-NN graphs satisfy **(A5)**, and we proceed to study general geometric graphs, which are essentially only assumed to satisfy **(A5)**. This covers a wide variety of geometrically constructed graphs, including, but not limited to, ε-graphs and k-NN graphs.

Theorem 4.17 *Assume that **(A2)**,**(A3)**, **(A4)**, **(A5)** hold. Consider the labeling rate β and δ defined as in (4.20). Then, there exist positive constants c_3, c_4, and c such that $\delta \geq c\beta$ and **(A1)** hold with probability at least*

$$1 - n\exp(-c_3 n\varepsilon_2^d) - n\exp(-c_4\beta n\varepsilon_1^d).$$

Here c_3, c_4, and c only depend on η, d, Ω, and ρ.

Proof Fix $x \in \mathcal{X}$. Because of **(A5)**, we have that:

$$d_n(x) = \sum_{y \in N_x} w_{xy} \leq \sum_{y \in \mathcal{X} \setminus x} c_2 \mathbb{1}_{|x-y| \leq \varepsilon_2} \varepsilon_2^{-d} := \Phi$$

Let us consider all the vertices in $\mathcal{X} \setminus x$ and treat them as random variables, calling them $Y_1, \ldots, Y_{n-1}$. Define $\psi_1(Y_i) = c_2 \mathbb{1}_{|x-Y_i| \leq \varepsilon_2} \varepsilon_2^{-d}$ and then it follows that

$$\Phi = \sum_{i=1}^{n-1} \psi_1(Y_i) = \sum_{i=1}^{n-1} c_2 \mathbb{1}_{|x-Y_i| \leq \varepsilon_2} \varepsilon_2^{-d}.$$

Thus,

$$\mathbb{E}[\Phi] = \sum_{Y_i \in \mathcal{X} \setminus x} \mathbb{E}[c_2 \mathbb{1}_{|x-Y_i| \leq \varepsilon_2} \varepsilon_2^{-d}] = (n-1) c_2 \int_{\Omega \cap B(x, \varepsilon_2)} \varepsilon_2^{-d} \rho(y) dy.$$

We apply Bernstein's inequality (30) in [20] to Φ obtaining that

$$\mathbb{P}(|\Phi - \mathbb{E}(\Phi)| > c_d(n-1)) \leq 2 \exp\Big(\frac{-(n-1)c_d^2}{2(\sigma^2 + \frac{1}{3} b c_d)} \Big)$$

for $\sigma^2 = Var(\psi_1(Y_i))$ and $b = \sup |\psi_1(Y_i)|$. We observe that $b = \sup |c_2 \mathbb{1}_{|x-Y_i| \leq \varepsilon_2} \varepsilon_2^{-d}| = c_2 \varepsilon_2^{-d}$ and also

$$\sigma^2 = Var(c_2 \mathbb{1}_{|x-Y_i| \leq \varepsilon_2} \varepsilon_2^{-d}) = c_2^2 \varepsilon_2^{-2d} Var(\mathbb{1}_{|x-Y_i| \leq \varepsilon_2}) \sim C \varepsilon_2^{-d},$$

because $Var(\mathbb{1}_{|x-Y_i| \leq \varepsilon_2})$ is of order ε_2^d as we integrate over a ball of radius ε_2. Thus, σ^2 and b depend on Ω, ρ, c_2, as well as on ε_2 explicitly.

We union-bound, obtaining that there exists two positive constants c_5 and c_3 for which

$$\Phi \leq c_d(n-1) + (n-1) c_2 \int_{\Omega \cap B(x, \varepsilon_2)} \varepsilon_2^{-d} \rho(y) dy = c_5(n-1)$$

holds with probability at least $1 - n \exp\{-c_3 n \varepsilon_2^d\}$. The same line of reasoning applies to $p_n(x)$ defined in (4.19). We define

$$\psi_2(Y_i) := c_1 \mathbb{1}_{|x-Y_i| \leq \varepsilon_1} \varepsilon_1^{-d} \mathbb{1}_{z(Y_i)=1}$$

and $\hat{\Phi} = \sum_{i=1}^n \psi_2(Y_i)$. We compute

$$\mathbb{E}[\hat{\Psi}] = \sum_{Y_i \in \mathcal{X}\setminus x} \mathbb{E}[c_1 \mathbb{1}_{|x-Y_i|\le \varepsilon_1} \varepsilon_1^{-d} \mathbb{1}_{z(Y_i)=1}] = (n-1)c_1\beta \int_{\Omega \cap B(x,\varepsilon_1)} \varepsilon_1^{-d} \rho(y) dy.$$

Thus there exists two positive constants c_6 and c_4 such that with probability $1 - n\exp(-c_4\beta n\varepsilon_1^d)$ we have

$$p_n(x) \ge (n-1)\beta c_6.$$

Thus, there exist constants c_5, c_6 such that

$$d_n(x) < c_5(n-1) \tag{4.24}$$

and

$$p_n(x) \ge c_6\beta(n-1) \tag{4.25}$$

hold for all all x with probability $1 - n\exp(-c_3 n\varepsilon_2^d) - n\exp(-c_4\beta n\varepsilon_1^d)$.

We use the results we just obtained for (4.24) and (4.25) to estimate that

$$\delta = \min_{x\in\mathcal{X}} \frac{p_n(x)}{d_n(x)} \ge \min \frac{c_5(n-1)\beta}{c_6(n-1)} \ge c\beta. \tag{4.26}$$

We now focus on proving **(A1)**. Pick any vertex $x \in X$. Since by definition

$$p_n(x) = \Sigma_{y\in\mathcal{X}} w_{xy} \mathbb{1}_{z(y)=1}$$

is a sum of nonnegative quantities, and by (4.25), there exists a neighbor x_ℓ of x such that

$$w_{xx_\ell} \mathbb{1}_{z(x_\ell)=1} > 0.$$

Thus, $z_\ell = 1$ for some vertex x_ℓ, therefore x_ℓ is labeled, and so x has a neighbor in Γ. This is a restatement of **(A1)**. This concludes the proof. □

In particular, with probability at least $1 - n\exp(-c_3 n\varepsilon_2^d) - n\exp(-c_4\beta n\varepsilon_1^d)$ assumption **(A1)** holds for k-NN graphs and ε-graphs. Therefore, Theorems 4.5 and 4.10 also hold (with the same probability) for ε-graphs and k-NN graphs.

Corollary 4.18 *For an ε-graph and $g \in Lip$, we have that*

$$|u(x) - g(x)| \le \varepsilon \operatorname{Lip}(g)\Big(\frac{2\log \frac{\sup g}{\varepsilon \operatorname{Lip}(g)}}{1-\alpha+2\alpha\delta} + 2\Big)$$

holds with with probability at least $1 - n\exp(-c_3 n\varepsilon^d) - n\exp(-c_4\beta n\varepsilon^d)$.

Proof The statement of Corollary 4.18 holds from Theorem 4.5, the observation $|g(z) - g(y)| \leq \text{Lip}(g)|z - y|$, and the fact that the distance between nearby nodes $|z - y|$ is of size ε. □

4.3 Stochastic Block Model Graphs

In this section, we apply our ideas and results from Sect. 4.1.1 to the stochastic block model (SBM) graph. Since SBM graphs have *discrete* geometric structure, as opposed to the *continuous* geometric structure of a random geometric graph, there is no obvious relationship to a continuum PDE on an SBM graph. Nevertheless, the tug-of-war lemma provides useful results in this setting.

We first define an SBM graph. The edge weights are binary $w_{xy} = 1$ if there is an edge between x and y and $w_{xy} = 0$ if there is no edge. We partition the vertices $\mathcal{X}$ of the SBM graph into two disjoint blocks $\mathcal{X} = \mathcal{X}_0 \cup \mathcal{X}_1$ according to an underlying binary label function $g : \mathcal{X} \to \{0, 1\}$, as in (4.14). We define $r, q \in (0, 1)$ as the two key probabilities for an SBM graph: r is the probability of an *intra-class* edge, and q is the probability of an *interclass* edge. In other words,

$$\mathbb{P}(w_{xy} = 1) = r \quad \text{if } x, y \in \mathcal{X}_i \text{ for } i = 0 \text{ or } i = 1$$
$$\mathbb{P}(w_{xy} = 1) = q \quad \text{if } x \in \mathcal{X}_i \text{ and } y \in \mathcal{X}_{1-i} \text{ for } i = 0 \text{ or } i = 1.$$

By **(A3)**, the probability that any vertex is labeled is β; thus, only a portion $\Gamma \in \mathcal{X}$ is labeled. The rest of $\mathcal{X}$ is unlabeled, and the goal is to label the points in $\mathcal{X}_0$ and $\mathcal{X}_1$ correctly. We also define $N_0 = |\mathcal{X}_0|$, $N_1 = |\mathcal{X}_1|$ and $n = |\mathcal{X}| = N_0 + N_1$. Let us set $\Gamma_i = \Gamma \cap \mathcal{X}_i$ so that $\Gamma = \Gamma_0 \cup \Gamma_1$.

Now, let u be the solution of the p-Laplace equation (4.1), and define the two sets $\mathcal{Y}_0$ and $\mathcal{Y}_1$ as follows: We assign x to $\mathcal{Y}_0$ whenever $u(x) \leq \frac{1}{2}$,, and we assign x to $\mathcal{Y}_1$ whenever $u(x) > \frac{1}{2}$. That is, $\mathcal{Y}_0$ is the set of vertices classified as class 0 by p-Laplace learning, and $\mathcal{Y}_1$ is the set classified as class 1. In order to classify correctly, we need $\mathcal{Y}_0 = \mathcal{X}_0$ and $\mathcal{Y}_1 = \mathcal{X}_1$.

Consider a vertex $x_i \in \mathcal{X}$. Let γ_i be the ratio of the number of other-class neighbors to the number of neighbors of vertex x_i, that is, if $x_i \in \mathcal{X}_j$, $j \in \{0, 1\}$, then

$$\gamma_i = \frac{\sum_{x \in \mathcal{X}_{1-j}} w_{x_i, x}}{\sum_{x \in \mathcal{X}} w_{x_i, x}}. \tag{4.27}$$

For a vertex x_i, the quantity γ_i is exactly the probability that a random walk step will jump to the opposite block. Let β_i be the fraction of neighbors from the same class that are labeled. Thus, if $x_i \in \mathcal{X}_j$ with $j \in \{0, 1\}$, then

$$\beta_i = \frac{\sum_{x \in \mathcal{X}_j \cap \Gamma} w_{x_i,x}}{\sum_{x \in \mathcal{X}_j} w_{x_i,x}}. \tag{4.28}$$

The term β_i is the probability that the random walk exits at at labeled node in the same class on the next step, conditioned on the event that the random walk remains in the same class.

We now derive upper and lower bounds for γ_i and β_i. The lower bound for β_i implies that **(A1)** holds with high probability, but we do not use **(A1)** directly in this section.

Lemma 4.19 *Let us consider an SBM graph with parameters N_0, N_1, r, q, and label rate β. Let $\sigma_1 \geq 0$ and $0 \leq \sigma_2 < 1$. Then, with probability of at least*

$$1 - \sum_{j=0}^{1} N_j \left\{ \exp\left(-\frac{q N_{1-j} \sigma_1^2}{2(1 + \frac{1}{3}\sigma_1)} \right) - 3 \exp\left(-\frac{1}{8} \beta r N_j \sigma_2^2 \right) \right\}. \tag{4.29}$$

for any $x_i \in \mathcal{X}_j$, $j \in \{0, 1\}$, we have

$$\gamma_i \leq \frac{(1+\sigma_1) q N_1}{(1+\sigma_1) q N_1 + (1-\sigma_2) r (N_0 - 1)}, \tag{4.30}$$

and

$$\beta_i \geq \frac{1-\sigma_2}{1+\sigma_2} \beta. \tag{4.31}$$

Proof Pick a vertex $x_i \in \mathcal{X}_0$. Define the following notation:

$$Y_j = \begin{cases} 1, & \text{if } x_i \text{ is connected to } x_j \in \mathcal{X}_1 \\ 0, & \text{otherwise.} \end{cases}$$

$$Z_j = \begin{cases} 1, & \text{if } x_i \text{ is connected to } x_j \in \mathcal{X}_0 \\ 0, & \text{otherwise.} \end{cases}$$

Denote $A = \sum_{j=1}^{N_1} Y_j$, $B = \sum_{j=1}^{N_0} Z_j$, and note that by (4.27), we have

$$\gamma_i = \frac{A}{A+B} = \frac{1}{1+C}, \quad C = \frac{B}{A}.$$

We now apply the Chernoff bounds (see, e.g., [13]) for A and B to obtain

$$\mathbb{P}(A \geq (1+\sigma_1)qN_1) \leq \exp\left(-\frac{qN_1\sigma_1^2}{2(1+\frac{1}{3}\sigma_1)}\right)$$

$$\mathbb{P}\left(|B - r(N_0-1)| \leq \sigma_2 r(N_0-1)\right) \leq 2\exp\left(-\frac{3}{8}r(N_0-1)\sigma_2^2\right),$$

for any $\sigma_1 > 0$ and $0 \leq \sigma_2 < 1$. Therefore,

$$C \geq \frac{(1-\sigma_2)r(N_0-1)}{(1+\sigma_1)qN_1}$$

holds with probability at least

$$1 - \exp\left(-\frac{qN_1\sigma_1^2}{2(1+\frac{1}{3}\sigma_1)}\right) - 2\exp\left(-\frac{3}{8}r(N_0-1)\sigma_2^2\right).$$

Assuming this event holds, we then have

$$\gamma_i \leq \frac{1}{1+C} \leq \frac{(1+\sigma_1)qN_1}{(1+\sigma_1)qN_1 + (1-\sigma_2)r(N_0-1)} = \frac{(1+\sigma)qN_1}{(1+\sigma)qN_1 + r(N_0-1)},$$

which establishes (4.30), upon union bounding over $\mathcal{X}_0$.

Now, let V_i be *i.i.d.* Bernoulli(β) for $i = 1, \ldots, n$ so that $x_i \in \Gamma$ if and only if $V_i = 1$. Then

$$\beta_i = \frac{1}{B}\sum_{j=1}^{n} Z_j V_j.$$

Since $Z_j V_j$ is Bernoulli(βr), we have

$$\mathbb{P}\left(\sum_{j=1}^{n} Z_j V_j \leq (1-\sigma_2)\beta r(N_0-1)\right) \leq \exp\left(-\frac{3}{8}\beta r(N_0-1)\sigma_2^2\right).$$

Assuming this event holds as well yields

$$\beta_i \geq \frac{(1-\sigma_2)\beta r(N_0-1)}{(1+\sigma_2)r(N_0-1)} = \frac{1-\sigma_2}{1+\sigma_2}\beta,$$

which establishes, with a union bound, (4.31). The argument is similar for $x_i \in \mathcal{X}_1$. The probabilities are simplified by combining like terms and using that $N_i - 1 \geq N_i/2$ since $N_i \geq 2$. □

Now we prove the main result for SBM graphs.

Theorem 4.20 *Let $2 \le p < \infty$, and consider an SBM graph with parameters r, q, N_0, N_1, and label rate β. Let $\sigma_1 \ge 0$, $0 \le \sigma_2 < 1$, and assume that*

$$\frac{r}{q}\beta > (1+\sigma)\max\left\{\tfrac{N_0}{N_1-1}, \tfrac{N_1}{N_0-1}\right\}, \tag{4.32}$$

where

$$1+\sigma = \frac{(1+\sigma_1)(1+\sigma_2)}{(1-\sigma_2)^2}. \tag{4.33}$$

Then the solution u to (4.1) classifies all vertices correctly with probability at least (4.29).

Remark 4.21 Note that the condition $p < \infty$ ensures that $\alpha > 0$ in (4.2), and thus there is always a positive probability of taking a random walk step. The proof of Theorem 4.20 relies entirely on the random walk and allows the two players to balance each other out. We expect this is suboptimal and is why the value of p does not explicitly enter the condition on $\frac{r}{q}$.

Proof of Theorem 4.20 We fix $\sigma_1 \ge 0$ and $0 \le \sigma_2 \le \frac{1}{2}$ and assume that the conclusions of Lemma 4.19 hold. We consider the stochastic tug-of-war process $X_0, X_1, \dots$, starting from $X_0 = x \in \mathcal{X}_0$. We define the associated random variables W_k by

$$W_k = \begin{cases} 0, & \text{if } X_k \in \Gamma_0 \\ 1, & \text{if } X_k \in \mathcal{X}_1 \\ 2, & \text{if } X_k \in \mathcal{X}_0 \setminus \Gamma_0. \end{cases}$$

The random variables W_k indicate whether the process stops at Γ_0, moves to $\mathcal{X}_1$ or stays in $\mathcal{X}_0$ and does not exit. The probabilities of each of these transitions depend on the vertex the process is currently at. If $X_{k-1} = x_i \in \mathcal{X}_0$, then

$$\begin{aligned} p_{0,i} &:= \mathbb{P}(W_k = 0) = \frac{1-\alpha}{2} + \alpha(1-\gamma_i)\beta_i, \\ p_{1,i} &:= \mathbb{P}(W_k = 1) \le \frac{1-\alpha}{2} + \alpha\gamma_i, \\ p_{2,i} &:= \mathbb{P}(W_k = 2) \ge \alpha(1-\gamma_i)(1-\beta_i), \end{aligned}$$

where γ_i is defined in (4.27), and $\beta_i > 0$ is fraction of same-class neighbors of x_i that are labeled. Let τ be the stopping time defined by

$$\tau = \inf\{k : W_k = 0 \text{ or } W_k = 1\}.$$

Let us define

$$p_0 = \min_{1\leq i\leq n} p_{0,i} = \frac{1-\alpha}{2} + \alpha(1-\gamma_{max})\beta_{min} \quad \text{and}$$
$$p_1 = \max_{1\leq i\leq n} p_{1,i} = \frac{1-\alpha}{2} + \alpha\gamma_{max},$$

where we write

$$\gamma_{max} = \max_{1\leq i\leq n} \gamma_i \quad \text{and} \quad \beta_{min} = \min_{1\leq i\leq n} \beta_i.$$

By Doob's optional stopping theorem and Lemma 4.2, we have $u(x) \leq \mathbb{E}[u(X_\tau)]$. Note that the expected value of τ is finite, since the probability of stopping at each step is $p_{0,i} + p_{1,i} \geq p_0 > 0$.

Let i_k denote the vertex index of the stochastic tug-of-war process on the k^{th} step, so that $X_k = x_{i_k}$. Then conditioned on $i_{\tau-1} = i$, the probability of $W_\tau = 1$ is exactly $p_{1,i}/(p_{0,i} + p_{1,i})$. Therefore by the law of conditional probability, and the fact that $x_{i_{\tau-1}} \in \mathcal{X}_0$, we have

$$\begin{aligned} u(x) \leq \mathbb{E}[u(X_\tau)] \leq \mathbb{P}(X_\tau \in \mathcal{X}_1) &= \sum_{i=1}^{n} \frac{p_{1,i}}{p_{0,i}+p_{1,i}} \mathbb{P}(i_{\tau-1} = i) \\ &= \sum_{x_i\in\mathcal{X}_0} \frac{p_{1,i}}{p_{0,i}+p_{1,i}} \mathbb{P}(i_{\tau-1} = i) \leq \frac{p_1}{p_0+p_1}. \end{aligned}$$

Since $x \in \mathcal{X}_0$, its correct label is 0, so in order to classify x correctly, we need to show that $u(x) < \frac{1}{2}$, which holds whenever $p_0/p_1 > 1$, or rather, $p_0 > p_1$. This is equivalent to

$$\beta_{min} > \frac{\gamma_{max}}{1-\gamma_{max}}. \tag{4.34}$$

By Lemma 4.19, we have

$$\gamma_{max} \leq \frac{(1+\sigma_1)qN_1}{(1+\sigma_1)qN_1 + (1-\sigma_2)r(N_0-1)},$$

and so

$$1-\gamma_{max} \geq \frac{(1-\sigma_2)r(N_0-1)}{(1+\sigma_1)qN_1 + (1-\sigma_2)r(N_0-1)}.$$

Therefore,

$$\frac{\gamma_{max}}{1-\gamma_{max}} \leq \frac{(1+\sigma_1)qN_1}{(1-\sigma_2)r(N_0-1)}.$$

Also, by Lemma 4.19, we have

$$\beta_{min} \geq \frac{1-\sigma_2}{1+\sigma_2}\beta.$$

Thus, a sufficient condition for (4.34) to hold is

$$\frac{1-\sigma_2}{1+\sigma_2}\beta > \frac{(1+\sigma_1)qN_1}{(1-\sigma_2)r(N_0-1)},$$

which is equivalent to

$$\frac{r}{q}\beta > (1+\sigma)\frac{N_1}{N_0-1},$$

where σ is defined in (4.33). A similar argument holds for $x \in \mathcal{X}_1$, except with N_0 and N_1 reversed. This concludes its proof. □

4.4 Numerical Results

We present here some numerical results with synthetic and real data, in order to illustrate the main results established in this section, and the room for improvement in future work. The Python code to reproduce the results in this section is available on GitHub.[8]

Our first experiment is with a stochastic block model graph to illustrate Theorem 4.20. We take a graph with $n = 3000$ vertices and consider the case of equal block sizes $N_0 = N_1 = 1500$ and unbalanced block sizes $N_0 = 2000$, $N_1 = 1000$. We take the intraclass probability to be $r = 0.5$, and we vary the interclass probability $q \leq r$. In Fig. 4, we show the error rates for binary classification using a labeling rate of $\beta = 0.2$ as a function of the ratio r/q. For each value of r/q, we ran 100 randomized trials and averaged the error rates. For equal size blocks $N_0 = N_1$, we see that the error rate decreases rapidly when $r/q > 1$. Theorem 4.20 guarantees that all vertices are classified correctly when $r/q > \beta^{-1} = 5$, which is a pessimistic bound in this case, since all values of p classify correctly when $r/q = 2$. For unequal block sizes, $N_0/N_1 = 2$ in Fig. 4b, we see that for larger values of p, the error rate decreases quickly when $r/q > 2$, while $p = 2$ requires $r/q > 8$ to see a similar decrease and $r/q = 10$ to classify correctly. In this case, Theorem 4.20 guarantees correct classification with high probability when $r/q > 2\beta^{-1} = 10$, which agrees very closely with the $p = 2$ result but is pessimistic for $p > 2$. Thus, our results may be tight for $p = 2$, but there is clearly much room for improvement in the range $p > 2$. In particular, we currently cannot explain why the results in Fig. 4b

[8] https://github.com/jwcalder/p-Laplace-consistency

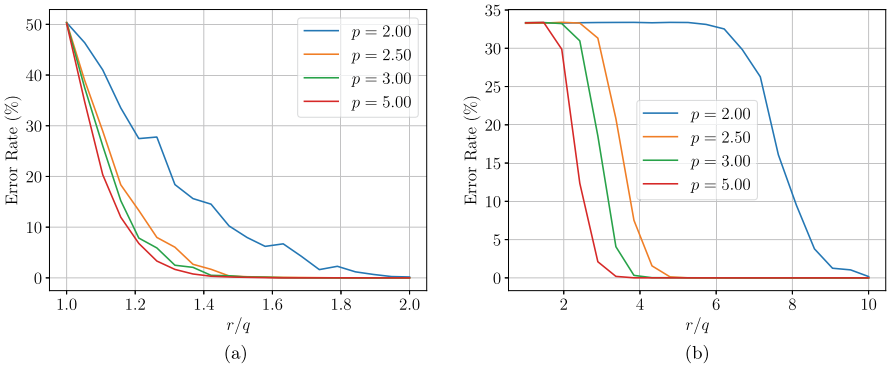

Fig. 4 Example of the effect of r/q on the classification performance on stochastic block model graphs for various values of p. (**a**) $N_0 = N_1 = 1500$. (**b**) $N_0 = 2000$, $N_1 = 1000$

Fig. 5 Sample of MNIST 4s and 9s, and Cifar-10 deer and dogs. (**a**) MNIST. (**b**) Cifar-10

are so dramatically better as p increases. An analysis of this sort would presumably require a different approach to Theorem 4.20 that exploited the tug-of-war game as well.

To illustrate our results on geometric graphs, we turn to real data. We consider the MNIST data set which consists of 70,000 images of handwritten digits between 0 and 9 [72], with each image a 28×28 grayscale image. We also consider the Cifar-10 data set [70] which contains 60,000 natural images from ten classes (airplane, automobile, bird, cat, deer, dog, frog, horse, ship, truck). Each Cifar-10 image is a 32×32 pixel color image. To make the problem more similar to our setting, we restrict the data sets from ten classes down to two, so we have a binary classification problem. For MNIST, we use the 4s and 9s, which are one of the harder pairs of digits to separate, while for Cifar-10, we use the deer and dogs. Figure 5 shows some example images from each data set.

In order to obtain a good feature embedding for constructing the graph, we used a variational autoencoder [67] for MNIST and the contrastive learning SimCLR method [33] for Cifar-10. Both methods are unsupervised deep learning algorithm that learn feature embeddings, or representations, of image data sets that maps the

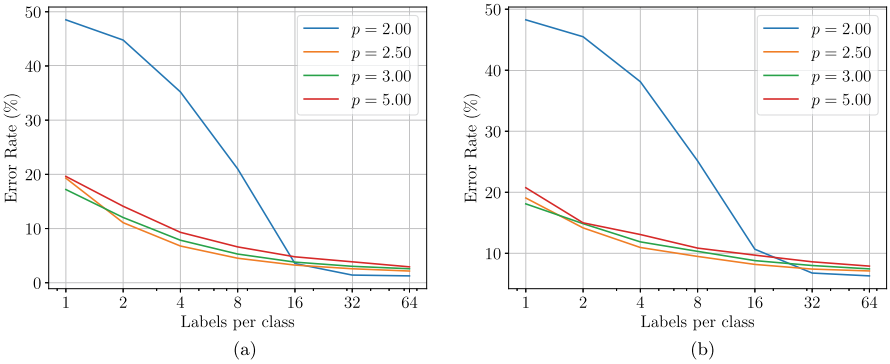

Fig. 6 Accuracy for binary classification of two classes in the MNIST and Cifar-10 data sets using p-Laplacian regularization with different values of p. We used the 4s and 9s from MNIST, which are one of the more difficult pairs to separate, and the deer and trucks from Cifar-10. (**a**) MNIST. (**b**) Cifar-10

images into a latent space identified with $\mathbb{R}^k$ where the similarity in latent features is far more informative of image similarity than pixel-wise similarity. After embedding the images into the feature space, we constructed a 10 nearest neighbor graph using the angular similarity, as in [25].

For our experiment, we used 1 up to 64 labeled examples per class for each data set, ran 100 trials at each label rate, and reported the average error rate. Figure 6 shows the error rates for different values of p as a function of label rate. We can see that larger values of p give better classification performance at very low label rates, as expected from previous work [47]. Once the label rate is sufficiently large, the error rate becomes small. As predicted in Theorem 4.10, the only points that are misclassified are those that are close to the boundary between classes in terms of graph distance, which are a small fraction of the data points—based on Fig. 6 about 2–4% for MNIST and 5–10% for Cifar-10.

5 Conclusions and Future Work

In this paper, we gave a thorough overview of the intersection between graph-based semi-supervised learning and PDEs and highlighted problems focused on consistency of classification that have not received significant attention yet in the community. We presented some preliminary results on consistency of p-Laplacian-based semi-supervised learning. Our results use the stochastic tug-of-war interpretation of the p-Laplacian on a graph, and we also provided a brief overview of this field. We proved consistency results for general graphs, geometric graphs, and stochastic block model graphs, the latter of which are not usually covered by continuum PDE-based arguments. One of our findings is that the tug-of-war

game transfers nicely between different graph structures and does not require the geometric structure of the graph. We also presented numerical experiments on synthetic and real data that illustrated our results and suggested directions for future work.

We highlight below some open problems for future work.

1. **Relaxing the assumption (A1)**. The assumption **(A1)** is a strong simplifying assumption we made in order to obtain preliminary results. It asks that every vertex on the graph has a labeled neighbor and is used to make our results tractable. In practice, this means that, in the geometric graph setting, labels do not propagate very far on the graph. It would be very interesting, and more practically relevant, to extend these results to settings where **(A1)** does not hold. This would require a far more delicate martingale analysis than we provided in this paper.
2. **Stochastic block models**. Our proof technique in Theorem 4.20 for the stochastic block model graph only exploited the random walk and ignored the tug-of-war component of the game. The numerical results in Sect. 4.4 suggest that the classification accuracy improves dramatically as $p > 2$ increases, even at moderately large label rates—20% in this case. In order to explain this, it would seem necessary to improve Theorem 4.20 by utilizing the tug-of-war game as well, so that the condition (4.32) on r/q depends on p.
3. **Noise and corruption:** In real-world applications, we may observe noisy or corrupted labels $g = g^\dagger + \xi$, where $\xi(y)$ for $y \in \Gamma$ are independent and identically distributed random variables. For simplicity, assume they have mean zero and variance $\sigma^2 > 0$. The results in this paper would establish that we recover g, while the true goal is to recover the clean uncorrupted labels $g^\dagger$. Applying the sub-martingale estimate Lemma 4.3 in this case and subtracting $g^\dagger$ from both sides, we obtain

$$|u(x) - g^\dagger(x)| \leq \mathbb{E}[|g^\dagger(X_\tau) - g^\dagger(x) \,|\, X_0 = x] + |\mathbb{E}[\xi(X_\tau)| \; X_0 = x]|\,,$$

where the expectation is over the random walk, and not the independent noise ξ. To recover $g^\dagger$, we need to be able to show that the last term above is small, either in expectation or with high probability. Without the absolute values, it is given by

$$\mathbb{E}[\xi(X_\tau) \,|\, X_0 = x] = \sum_{y \in \Gamma} \mathbb{P}(X_\tau = y)\xi(y).$$

If we assume the distribution of the stopping vertex, X_τ is independent of the noise ξ, then the variance of this term with respect to the noise is given by

$$\mathrm{Var}\, \mathbb{E}[\xi(X_\tau) \,|\, X_0 = x] = \sigma^2 \sum_{y \in \Gamma} \mathbb{P}(X_\tau = y)^2. \tag{5.1}$$

Thus, understanding the noisy setting requires a finer analysis of the stochastic process, so that we can estimate the ℓ^2 norm of the probability distribution of X_τ. If X_τ is well-spread out on Γ, say $\mathbb{P}(X_\tau = y) = \frac{1}{\#\Gamma}$, then the variance (5.1) is small: $\sigma^2/\#\Gamma$. On the other hand, if the distribution concentrates on a single node, $\mathbb{P}(X_\tau = y) = \delta_{y=z}$ for some $z \in \Gamma$, then the variance is σ^2; the same as that of the noise $\xi(z)$. Essentially, the more labels we average over to compute $u(x)$, the smaller the effect of noise. Estimating the variance in (5.1) appears to be a nontrivial undertaking that we leave for future work. In particular, X_τ will generally only be independent of ξ when $p = 2$; for $p > 2$, the variance computation in (5.1) will also contain covariance terms that may be difficult to control.

On the other hand, we expect the SBM analysis in Theorem 4.20 to extend in a fairly direct way to the setting where some fraction $\theta \in (0, 1)$ of the labels are corrupted by flipping to the opposite class. In this case, we expect a similar result to hold where β_i decreases and γ_i increases, by an amount proportional to θ, and then the left-hand side of (4.32) is decreased by a factor like $1 - \theta$. We leave this to a future study as well.

4. **Extension to other random-walk models**. It would be interesting to extend these results to other models that have random walk interpretations, including Poisson learning [25], PWLL [27, 98], and the properly weighted Laplacian [24]. Some of the same high level ideas may work, but we expect many of the ingredients to be different. The case of Laplace learning was essentially already studied in [28].
5. **Similar results for other models**. There are a number of models that do not have random walk interpretations, such as the variational p-Laplacian [39, 112] and the MBO methods [14, 48, 62, 93–95]. It would be interesting to prove similar consistency results for these methods, though the techniques would be substantially different, since as far as we are aware, there are no representation formulas that express the solutions through stochastic processes in these works.

Acknowledgments

Funding: Calder was supported by NSF-DMS grant 1944925, the Alfred P. Sloan foundation, the McKnight foundation, and an Albert and Dorothy Marden Professorship.

Source Code: https://github.com/jwcalder/p-Laplace-consistency.

References

1. M. Alamgir, U.V. Luxburg, Phase transition in the family of p-resistances. in *Advances in Neural Information Processing Systems* (2011), pp. 379–387
2. R.K. Ando, T. Zhang, Learning on graph with Laplacian regularization, in *Advances in Neural Information Processing Systems* (2007), pp. 25–32
3. S. Armstrong, C. Smart, A finite difference approach to the infinity Laplace equation and tug-of-war games. Trans. Amer. Math. Soc. **364**(2), 595–636 (2012)

4. G. Aronsson, M. Crandall, P. Juutinen, A tour of the theory of absolutely minimizing functions. Bull. Amer. Math. Soc. **41**(4), 439–505 (2004)
5. A. Attouchi, H. Luiro, M. Parviainen, Gradient and Lipschitz estimates for tug-of-war type games. SIAM J. Math. Analy. **53**(2), 1295–1319 (2021)
6. A. Azad, Learning Label Initialization for Time-Dependent Harmonic Extension (2022). arXiv preprint arXiv:2205.01358
7. M. Bardi, I.C. Dolcetta, et al., *Optimal Control and Viscosity Solutions of Hamilton-Jacobi-Bellman Equations*, vol. 12 (Springer, Berlin, 1997)
8. M. Belkin, P. Niyogi, Using manifold stucture for partially labeled classification, in *Advances in Neural Information Processing Systems*, 15 (2002)
9. M. Belkin, P. Niyogi, Laplacian eigenmaps for dimensionality reduction and data representation. Neural Comput. **15**(6), 1373–1396 (2003)
10. M. Belkin, P. Niyogi, Semi-supervised learning on Riemannian manifolds. Mach. Learn. **56**, 209–239 (2004)
11. M. Belkin, I. Matveeva, P. Niyogi, Regularization and semi-supervised learning on large graphs, in *Learning Theory: 17th Annual Conference on Learning Theory, COLT 2004, Banff, July 1–4, 2004. Proceedings 17* (Springer, Berlin, 2004), pp 624–638
12. Y. Bengio, O. Delalleau, N. Le Roux, *Label Propagation and Quadratic Criterion*, Semi-supervised Learning edition (MIT Press, Cambridge, 2006), pp. 193–216
13. S. Boucheron, G. Lugosi, P. Massart, *Concentration Inequalities: A Nonasymptotic Theory of Independence* (Universities Press, Oxford , 2013)
14. Z.M. Boyd, E. Bae, X.-C. Tai, A.L. Bertozzi, Simplified energy landscape for modularity using total variation. SIAM J. Appl. Math. **78**(5), 2439–2464 (2018)
15. F. Bozorgnia, M. Fotouhi, A. Arakelyan, A. Elmoataz, Graph based semi-supervised learning using spatial segregation theory. J. Comput. Sci. **74**, 102153 (2023)
16. N. Bridle, X. Zhu, p-voltages: Laplacian regularization for semi-supervised learning on high-dimensional data, in *Eleventh Workshop on Mining and Learning with Graphs (MLG2013)* (2013)
17. J. Brown, R. O'Neill, J. Calder, A.L. Bertozzi, Utilizing contrastive learning for graph-based active learning of SAR data, in *SPIE Defense and Commercial Sensing: Algorithms for Synthetic Aperture Radar Imagery XXX* (2023)
18. L. Bungert, J. Calder, T. Roith, Uniform convergence rates for lipschitz learning on graphs. IMA J. Numer. Analy. **43**, 2445–2495 (2022)
19. L. Bungert, J. Calder, T. Roith, Ratio convergence rates for Euclidean first-passage percolation: applications to the graph infinity Laplacian. Ann. Appl. Probab. **34**(4), 3870–3910 (2024)
20. J. Calder, The game theoretic p-Laplacian and semi-supervised learning with few labels. Nonlinearity **32**(1), 301 (2018)
21. J. Calder, Consistency of Lipschitz learning with infinite unlabeled data and finite labeled data. SIAM J. Math. Data Sci. **1**(4), 780–812 (2019)
22. J. Calder, M. Ettehad, Hamilton-Jacobi equations on graphs with applications to semi-supervised learning and data depth. J. Mach. Learn. Res. **23**(318), 1–62 (2022)
23. J. Calder, N. García Trillos, Improved spectral convergence rates for graph Laplacians on ε-graphs and k-NN graphs. Appl. Comput. Harmonic Analy. **60**, 123–175 (2022)
24. J. Calder, D. Slepčev, Properly-weighted graph Laplacian for semi-supervised learning. Appl. Math. Optim. **82**, 1111–1159 (2020)
25. J. Calder, B. Cook, M. Thorpe, D. Slepčev, Poisson learning: graph based semi-supervised learning at very low label rates, in *Proceedings of the 37th International Conference on Machine Learning, PMLR*, vol. 119 (2020), pp. 1306–1316
26. J. Calder, N. García Trillos, M. Lewicka, Lipschitz regularity of graph Laplacians on random data clouds. SIAM J. Math. Analy. **54**(1), 1169–1222 (2022)
27. J. Calder, B. Cook, M. Thorpe, D. Slepčev, Y. Zhang, S. Ke, Graph-based semi-supervised learning with Poisson equations. In Preparation (2024)

28. J. Calder, D. Slepčev, M. Thorpe, Rates of convergence for Laplacian semi-supervised learning with low labeling rates. Res. Math. Sci. Special issue on PDE Methodsr Mach. Learn. **10**(1), 10 (2023)
29. J. Cervino, L.F. Chamon, B.D. Haeffele, R. Vidal, A. Ribeiro, Learning globally smooth functions on manifolds, in *International Conference on Machine Learning* (PMLR, London, 2023), pp. 3815–3854
30. O. Chapelle, B. Scholkopf, A. Zien, *Semi-supervised Learning* (MIT, Cambridge, 2006)
31. J. Chapman, B. Chen, Z. Tan, J. Calder, K. Miller, A.L. Bertozzi, Novel batch active learning approach and its application on the synthetic aperture radar datasets, in *SPIE Defense and Commercial Sensing: Algorithms for Synthetic Aperture Radar Imagery XXX (Best Student Paper)* (2023)
32. F. Charro, J. García Azorero, J.D. Rossi, A mixed problem for the infinity Laplacian via tug-of-war games. Calcul. Variat. Part. Differ. Equ. **34**(3), 307–320 (2009)
33. T. Chen, S. Kornblith, M. Norouzi, G. Hinton, A simple framework for contrastive learning of visual representations, in *International Conference on Machine Learning* (PMLR, London, 2020), pp. 1597–1607
34. R.R. Coifman, S. Lafon, Diffusion maps. Appl. Comput. Harmonic Analy. **21**(1), 5–30 (2006)
35. M. Defferrard, X. Bresson, P. Vandergheynst, Convolutional neural networks on graphs with fast localized spectral filtering, in *Advances in Neural Information Processing Systems*, 29 (2016)
36. H. Dong, J. Chen, F. Feng, X. He, S. Bi, Z. Ding, P. Cui, On the equivalence of decoupled graph convolution network and label propagation. Proc. Web Confer. **2021**, 3651–3662 (2021)
37. D.L. Donoho, C. Grimes, Hessian eigenmaps: locally linear embedding techniques for high-dimensional data. Proc. Natl. Acad. Sci. **100**(10), 5591–5596 (2003)
38. M.M. Dunlop, D. Slepčev, A.M. Stuart, M. Thorpe, Large data and zero noise limits of graph-based semi-supervised learning algorithms. Appl. Comput. Harmonic Analy. **49**(2), 655–697 (2020)
39. A. El Alaoui, X. Cheng, A. Ramdas, M.J. Wainwright, M.I. Jordan, Asymptotic behavior of ℓ_p-based Laplacian regularization in semi-supervised learning, in *Conference on Learning Theory* (2016), pp. 879–906
40. A. Elmoataz, M. Toutain, D. Tenbrinck, On the p-Laplacian and ∞-Laplacian on graphs with applications in image and data processing. SIAM J. Imag. Sci. **8**(4), 2412–2451 (2015)
41. A. Elmoataz, X. Desquesnes, M. Toutain, On the game p-Laplacian on weighted graphs with applications in image processing and data clustering. Eur. J. Appl. Math. **28**(6), 922–948 (2017)
42. A. Elmoataz, F. Lozes, M. Toutain, Nonlocal PDEs on graphs: from tug-of-war games to unified interpolation on images and point clouds. J. Math. Imag. Vis. **57**(3), 381–401 (2017)
43. H. Ennaji, Y. Quéau, A. Elmoataz, Tug of war games and PDEs on graphs with applications in image and high dimensional data processing. Sci. Rep. **13**(1), 6045 (2023)
44. J. Enwright, H. Hardiman-Mostow, J. Calder, A.L. Bertozzi, Deep semisupervised label propagation for SAR image classification, in *SPIE Defense and Commercial Sensing: Algorithms for Synthetic Aperture Radar Imagery XXX* (2023)
45. L.C. Evans, A new proof of local C1, α regularity for solutions of certain degenerate elliptic pde. J. Differ. Equ. **45**(3), 356–373 (1982)
46. L. Evans, *Partial Differential Equations (Graduate Studies in Mathematics, V. 19) GSM/19* (American Mathematical Society, Providence, 1998)
47. M. Flores, J. Calder, G. Lerman, Analysis and algorithms for Lp-based semi-supervised learning on graphs. Appl. Comput. Harmonic Analy. **60**, 77–122 (2022)
48. C. Garcia-Cardona, E. Merkurjev, A.L. Bertozzi, A. Flenner, A.G. Percus, Multiclass data segmentation using diffuse interface methods on graphs. IEEE Trans. Pattern Analy. Mach. Intell. **36**(8), 1600–1613 (2014)
49. N. García Trillos, D. Slepčev, Continuum limit of total variation on point clouds. Arch. Ration. Mech. Analy. **220**, 193–241 (2016)

50. N. García Trillos, M. Gerlach, M. Hein, D. Slepčev, Error estimates for spectral convergence of the graph Laplacian on random geometric graphs toward the Laplace–Beltrami operator. Foundat. Comput. Math. **20**(4), 827–887 (2020)
51. J. Gasteiger, A. Bojchevski, S. Günnemann, Predict then propagate: graph neural networks meet personalized PageRank, in *International Conference on Learning Representations* (2018)
52. D.F. Gleich, PageRank beyond the Web. SIAM Rev. **57**(3), 321–363 (2015)
53. I. Goodfellow, Y. Bengio, A. Courville, *Deep Learning* (MIT Press, Cambridge, 2016)
54. Y. Hafiene, J. Fadili, A. Elmoataz, Nonlocal p-Laplacian Variational problems on graphs (2018). arXiv:1810.12817
55. J. Ham, D. Lee, L. Saul, Semisupervised alignment of manifolds, in *International Workshop on Artificial Intelligence and Statistics* (PMLR, London, 2005), pp. 120–127
56. J. Han, Time-dependent tug-of-war games and normalized parabolic p-Laplace equations. Nonlinear Analy. **214**, 112542 (2022)
57. J. He, M. Li, H.-J. Zhang, H. Tong, C. Zhang, Manifold-ranking based image retrieval, in *Proceedings of the 12th Annual ACM International Conference on Multimedia* (ACM, New York, 2004), pp. 9–16
58. J. He, M. Li, H.-J. Zhang, H. Tong, C. Zhang, Generalized manifold-ranking-based image retrieval. IEEE Trans. Image Process. **15**(10), 3170–3177 (2006)
59. M. Hein, J.-Y. Audibert, U. Von Luxburg, From graphs to manifolds-weak and strong pointwise consistency of graph Laplacians, in *COLT*, vol. 3559 (Springer, Berlin, 2005), pp. 470–485
60. M. Hein, J.-Y. Audibert, U.V. Luxburg, Graph Laplacians and their convergence on random neighborhood graphs. J. Mach. Learn. Res. **8**(6), 1325–1368 (2007)
61. F. Hoffmann, B. Hosseini, A.A. Oberai, A.M. Stuart, Spectral analysis of weighted Laplacians arising in data clustering. Appl. Comput. Harmonic Analy. **56**, 189–249 (2022)
62. H. Hu, T. Laurent, M.A. Porter, A.L. Bertozzi, A method based on total variation for network modularity optimization using the MBO scheme. SIAM J. Appl. Math. **73**(6), 2224–2246 (2013)
63. Q. Huang, H. He, A. Singh, S.-N. Lim, A.R. Benson, Combining label propagation and simple models out-performs graph neural networks (2020). arXiv preprint arXiv:2010.13993
64. M. Jacobs, E. Merkurjev, S. Esedoḡlu, Auction dynamics: a volume constrained MBO scheme. J. Comput. Phys. **354**, 288–310 (2018)
65. M. Ji, J. Han, A variance minimization criterion to active learning on graphs, in *Artificial Intelligence and Statistics* (PMLR, London, 2012), pp. 556–564
66. A. Jung, A.O. Hero III, A. Mara, S. Jahromi, Semi-supervised learning via sparse label propagation (2016). arXiv preprint arXiv:1612.01414
67. D.P. Kingma, M. Welling, Auto-encoding variational bayes (2013). arXiv preprint arXiv:1312.6114
68. T.N. Kipf, M. Welling, Semi-supervised classification with graph convolutional networks (2016). arXiv:1609.02907
69. R. Kohn, S. Serfaty, A deterministic-control-based approach motion by curvature. Commun. Pure Appl. Math. J. Iss. Courant Instit. Math. Sci. **59**(3), 344–407 (2006)
70. A. Krizhevsky, G. Hinton, et al., Learning multiple layers of features from tiny images. *University of Toronto* (2009)
71. R. Kyng, A. Rao, S. Sachdeva, D.A. Spielman, Algorithms for Lipschitz learning on graphs, in *Conference on Learning Theory* (2015), pp. 1190–1223
72. Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition. Proc. IEEE **86**(11), 2278–2324 (1998)
73. W.-Y. Lee, L.-C. Hsieh, G.-L. Wu, W. Hsu, Graph-based semi-supervised learning with multi-modality propagation for large-scale image datasets. J. Visual Commun. Image Represent. **24**(3), 295–302 (2013)
74. M. Lewicka, Random Tug of War games for the p-Laplacian: $1 < p < \infty$ (2018). arXiv preprint arXiv:1810.03413

75. M. Lewicka, *A Course on Tug-of-War Games with Random Noise* (Springer, Berlin, 2020)
76. M. Lewicka, Non-local Tug-of-War with noise for the geometric fractional p-Laplacian. Adv. Differ. Equ. **27**(1–2), 31–76 (2022)
77. M. Lewicka, J.J. Manfredi, Game theoretical methods in PDEs. Bollettino dell'Unione Matematica Italiana **7**(3), 211–216 (2014)
78. M. Lewicka, J.J. Manfredi, The obstacle problem for the p-laplacian via optimal stopping of tug-of-war games. Probab. Theory Relat. Fields **167**, 349–378 (2017)
79. M. Lewicka, Y. Peres, The Robin mean value equation I: a random walk approach to the third boundary value problem. Potent. Analy. **59**, 1–32 (2022)
80. M. Lewicka, Y. Peres, The Robin mean value equation II: asymptotic Hölder regularity. Potent. Analy. **59**, 1–35 (2022)
81. X. Li, R. Zhu, Y. Cheng, C. Shan, S. Luo, D. Li, W. Qian, Finding global homophily in graph neural networks when meeting heterophily, in *International Conference on Machine Learning* (PMLR, London, 2022), pp. 13242–13256
82. P. Lindqvist, *Notes on the Stationary p-Laplace Equation* (Springer, Berlin, 2019)
83. H. Luiro, M. Parviainen, E. Saksman, Harnack's inequality for p-harmonic functions via stochastic games. Commun. Part. Differ. Equ. **38**(11), 1985–2003 (2013)
84. U.V. Luxburg, O. Bousquet, Distance-based classification with Lipschitz functions. J. Mach. Learn. Res. **5**(Jun), 669–695 (2004)
85. Y. Ma, R. Garnett, J. Schneider, Σ-optimality for active learning on gaussian random fields, in ed. by C.J.C. Burges, L. Bottou, M. Welling, Z. Ghahramani, K.Q. Weinberger, *Advances in Neural Information Processing Systems 26* (Curran Associates, San Jose, 2013), pp. 2751–2759
86. X. Mai, A random matrix analysis and improvement of semi-supervised learning for large dimensional data. J. Mach. Learn. Res. **19**(79), 1–27 (2018)
87. X. Mai, R. Couillet, Random matrix-inspired improved semi-supervised learning on graphs, in *International Conference on Machine Learning* (2018)
88. X. Mai, R. Couillet, Consistent semi-supervised graph regularization for high dimensional data. J. Mach. Learn. Res. **22**(1), 4181–4228 (2021)
89. J. Manfredi, M. Parviainen, J. Rossi, An asymptotic mean value characterization for p-harmonic functions. Proc. Am. Math. Soc. **138**(3), 881–889 (2010)
90. J.J. Manfredi, M. Parviainen, J.D. Rossi, Dynamic programming principle for tug-of-war games with noise. ESAIM Control Optim. Calcul. Variat. **18**(1), 81–90 (2012)
91. J.J. Manfredi, M. Parviainen, J.D. Rossi, On the definition and properties of p-harmonious functions. Annali della Scuola Normale Superiore di Pisa-Classe di Scienze **11**(2), 215–241 (2012)
92. J.J. Manfredi, A.M. Oberman, A.P. Sviridov, Nonlinear elliptic partial differential equations and p-harmonic functions on graphs. Differ. Integr. Equ. **28**(1–2), 79–102 (2015)
93. E. Merkurjev, T. Kostic, A.L. Bertozzi, An MBO scheme on graphs for classification and image processing. SIAM J. Imag. Sci. **6**(4), 1903–1930 (2013)
94. E. Merkurjev, C. Garcia-Cardona, A.L. Bertozzi, A. Flenner, A.G. Percus, Diffuse interface methods for multiclass segmentation of high-dimensional data. Appl. Math. Lett. **33**, 29–34 (2014)
95. E. Merkurjev, A.L. Bertozzi, F. Chung, A semi-supervised heat kernel pagerank MBO algorithm for data classification. Commun. Math. Sci. **16**(5), 1241–1265 (2018)
96. B. Merriman, J.K. Bence, S.J. Osher, Motion of multiple junctions: a level set approach. J. Comput. Phys. **112**(2), 334–363 (1994)
97. K. Miller, A.L. Bertozzi, Model-change active learning in graph-based semi-supervised learning (2021). arXiv preprint arXiv:2110.07739
98. K. Miller, J. Calder, Poisson reweighted Laplacian uncertainty sampling for graph-based active learning. SIAM J. Math. Data Sci. **5**, 1160–1190 (2023)
99. K. Miller, X. Baca, J. Mauro, J. Setiadi, Z. Shi, J. Calder, A. Bertozzi, Graph-based active learning for semi-supervised classification of SAR data, in *SPIE Defense and Commercial Sensing: Algorithms for Synthetic Aperture Radar Imagery XXIX* (2022), p. 12095

100. J.M. Murphy, M. Maggioni, Unsupervised clustering and active learning of hyperspectral images with nonlinear diffusion. IEEE Trans. Geosci. Remote Sens. **57**(3), 1829–1845 (2019)
101. B. Nadler, N. Srebro, X. Zhou, Semi-supervised learning with the graph Laplacian: the limit of infinite unlabelled data. Adv. Neural Inf. Process. Syst. **22**, 1330–1338 (2009)
102. A. Oberman, A convergent difference scheme for the infinity Laplacian: construction of absolutely minimizing Lipschitz extensions. Math. Comput. **74**(251), 1217–1230 (2005)
103. A.M. Oberman, Finite difference methods for the infinity Laplace and p-Laplace equations. J. Comput. Appl. Math. **254**, 65–80 (2013)
104. M. Parviainen, *Notes on Tug-of-War Games and the p-Laplace Equation*. Springer Briefs on PDEs and Data Science (Springer, Berlin, 2024)
105. Y. Peres, S. Sheffield et al., Tug-of-war with noise: a game-theoretic view of the p-Laplacian. Duke Math. J. **145**(1), 91–120 (2008)
106. Y. Peres, O. Schramm, S. Sheffield, D. Wilson, Tug-of-war and the infinity Laplacian. J. Amer. Math. Soc. **22**(1), 167–210 (2009)
107. Y. Peres, G. Pete, S. Somersille, Biased tug-of-war, the biased infinity Laplacian, and comparison with exponential cones. Calc. Var. Partial Differ. Equ. **38**(3–4), 541–564 (2010)
108. Y.-L. Qiao, C.X. Shi, C. Wang, H. Li, M. Haberland, X. Luo, A.M. Stuart, A.L. Bertozzi, Uncertainty quantification for semi-supervised multi-class classification in image processing and ego-motion analysis of body-worn videos, in *Image Processing: Algorithms and Systems* (2019)
109. P. Sellars, A.I. Aviles-Rivero, C.-B. Schönlieb, Laplacenet: a hybrid graph-energy neural network for deep semisupervised classification. IEEE Trans. Neural Netw. Learn. Syst. **35**(4), 5306–5318 (2022)
110. B. Settles, Active learning literature survey. University of Wisconsin-Madison Department of Computer Sciences (2009)
111. Z. Shi, S. Osher, W. Zhu, Weighted nonlocal Laplacian on interpolation from sparse data. J. Sci. Comput. **73**(2–3), 1164–1177 (2017)
112. D. Slepcev, M. Thorpe, Analysis of p-laplacian regularization in semisupervised learning. SIAM J. Math. Analy. **51**(3), 2085–2120 (2019)
113. N.G. Trillos, F. Hoffmann, B. Hosseini, Geometric structure of graph Laplacian embeddings. J. Mach. Learn. Res. **22**(1), 2934–2988 (2021)
114. A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, L. Kaiser, I. Polosukhin, Attention is all you need, in *Advances in Neural Information Processing Systems, 30* (2017)
115. P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, Y. Bengio, Graph attention networks (2017). arXiv preprint arXiv:1710.10903
116. U. von Luxburg, A tutorial on spectral clustering. Stat. Comput. **17**(4), 395–416 (2007)
117. B. Wang, Z. Tu, J.K. Tsotsos, Dynamic label propagation for semi-supervised multi-class multi-label classification, in *Proceedings of the IEEE International Conference on Computer Vision* (2013), pp. 425–432
118. Y. Wang, M.A. Cheema, X. Lin, Q. Zhang, Multi-manifold ranking: using multiple features for better image retrieval, in *Pacific-Asia Conference on Knowledge Discovery and Data Mining* (Springer, Berlin, 2013), pp. 449–460
119. A. Weihs, M. Thorpe, Consistency of Fractional Graph-Laplacian Regularization in Semi-Supervised Learning with Finite Labels (2023). arXiv preprint arXiv:2303.07818
120. D. Williams, *Probability with Martingales* (Cambridge University Press, Cambridge, 1991)
121. B. Xu, J. Bu, C. Chen, D. Cai, X. He, W. Liu, J. Luo, Efficient manifold ranking for image retrieval, in *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval* (ACM, New York, 2011), pp. 525–534
122. Y. Yan, M. Hashemi, K. Swersky, Y. Yang, D. Koutra, Two sides of the same coin: Heterophily and oversmoothing in graph convolutional neural networks, in *2022 IEEE International Conference on Data Mining (ICDM)* (IEEE, Piscataway, 2022), pp. 1287–1292
123. C. Yang, L. Zhang, H. Lu, X. Ruan, M.-H. Yang, Saliency detection via graph-based manifold ranking, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2013), pp. 3166–3173

124. Z. Yang, W. Cohen, R. Salakhudinov, Revisiting semi-supervised learning with graph embeddings, in *International Conference on Machine Learning* (PMLR, London, 2016), pp. 40–48
125. X. Yang, Z. Song, I. King, Z. Xu, A survey on deep semi-supervised learning. IEEE Trans. Knowl. Data Eng. **35**, 8934–8954 (2022)
126. A. Yuan, J. Calder, B. Osting, A continuum limit for the PageRank algorithm. Eur. J. Appl. Math. **33**, 472–504 (2022)
127. X. Zheng, Y. Liu, S. Pan, M. Zhang, D. Jin, P.S. Yu, Graph neural networks for graphs with heterophily: A survey (2022). arXiv preprint arXiv:2202.07082
128. X. Zhou, M. Belkin, Semi-supervised learning by higher order regularization, in *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*. JMLR Workshop and Conference Proceedings (2011), pp. 892–900
129. D. Zhou, B. Schölkopf, Learning from labeled and unlabeled data using random walks, in *Joint Pattern Recognition Symposium* (Springer, Berlin, 2004), pp. 237–244
130. D. Zhou, B. Schölkopf, Regularization on discrete spaces, in *Joint Pattern Recognition Symposium* (Springer, Berlin, 2005), pp. 361–368
131. D. Zhou, O. Bousquet, T.N. Lal, J. Weston, B. Schölkopf, Learning with local and global consistency, in *Advances in Neural Information Processing Systems* (2004), pp. 321–328
132. D. Zhou, J. Weston, A. Gretton, O. Bousquet, B. Schölkopf, Ranking on data manifolds, in *Advances in Neural Information Processing Systems* (2004), pp. 169–176
133. D. Zhou, J. Huang, B. Schölkopf, Learning from labeled and unlabeled data on a directed graph, in *Proceedings of the 22nd International Conference on Machine Learning* (ACM, New York, 2005), pp. 1036–1043
134. X. Zhu, Z. Ghahramani, Learning from labeled and unlabeled data with label propagation. *ProQuest number: information to all users* (2002)
135. X. Zhu, Z. Ghahramani, J.D. Lafferty, Semi-supervised learning using Gaussian fields and harmonic functions, in *Proceedings of the 20th International Conference on Machine Learning (ICML-03)* (2003), pp. 912–919
136. X. Zhu, J. Lafferty, Z. Ghahramani, Combining active learning and semi-supervised learning using Gaussian fields and harmonic functions, in *International Conference on Machine Learning (ICML) 2003 Workshop on The Continuum from Labeled to Unlabeled Data in Machine Learning and Data Mining* (2003), pp. 58–65
137. J. Zhu, Y. Yan, L. Zhao, M. Heimann, L. Akoglu, D. Koutra, Beyond homophily in graph neural networks: current limitations and effective designs. Adv. Neural Inf. Process. Syst. **33**, 7793–7804 (2020)