A CONVERGENT SINGLE-LOOP ALGORITHM FOR GROMOV-WASSERSTEIN IN GRAPH DATA

Anonymous authors
Paper under double-blind review

ABSTRACT

We propose the first provable single-loop algorithm for computing the Gromov-Wasserstein (GW) distance. We call it Bregman Alternating Projected Gradient (BAPG) method. Our analysis is based on a surprising observation that the GW problem satisfies a so-called Luo-Tseng error bound condition (Luo & Tseng, 1992), which relates to estimating the distance of a point to the critical point set of the GW problem based on the optimality residual. Armed with this error bound condition, we are able to show that BAPG converges to a critical point of the GW problem asymptotically (as step size goes to infinity), and further give an approximation bound for the distance between the fixed-point set of BAPG and the critical point set of GW explicitly. We conduct comprehensive numerical experiments to validate the effectiveness of BAPG on graph alignment and partition tasks. In terms of both wall-clock time and quality of solutions, the proposed method achieves state-of-the-art results.

1 INTRODUCTION

The Gromov-Wasserstein (GW) distance provides a flexible way to compare and couple probability distributions supported on different metric spaces. As such, we have witnessed a fast-growing body of literature that applies the GW distance to various structural data analysis tasks, e.g., 2D/3D shape matching (Peyré et al., 2016; Mémoli & Sapiro, 2004; Mémoli, 2009), molecule analysis (Vayer et al., 2018; 2019a), graph alignment and partition (Chowdhury & Mémoli, 2019; Xu et al., 2019b;a; Chowdhury & Needham, 2021; Gao et al., 2021), graph embedding and classification (Vincent-Cuaz et al., 2021b; Xu et al., 2022), generative modeling (Bunne et al., 2019; Xu et al., 2021), to name a few.

Although the GW distance has received much attention in both machine learning and data science communities, there are still few results that focus on the design of practically efficient algorithms with provable convergence guarantees. Existing algorithms either are double-loop (i.e., requiring another iterative algorithm as a subroutine at each iteration) or do not have any theoretical justification. Recently, Solomon et al. (2016) has proposed an entropy-regularized iterative sinkhorn projection algorithm called eBPG, which is proven to converge under the Kurdyka-Łojasiewicz framework (Attouch et al., 2010; 2013). However, eBPG has several crucial drawbacks that prevent its widespread adoption. First, instead of tackling the GW problem directly, eBPG addresses an entropic regularized GW objective, whose regularization parameter affects the model performance dramatically. Second, eBPG has to solve the entropic optimal transport problem (Peyré et al., 2019; Benamou et al., 2015) at each iteration via another iterative algorithm (Cuturi, 2013). Thus, eBPG is neither computationally efficient nor practically robust. To solve the GW problem directly, Xu et al. (2019b) proposes the Bregman projected gradient (BPG). Unfortunately, BPG is still a double-loop algorithm that also invokes the sinkhorn as its subroutine. Notably, due to the lack of the entropic regularizer, the inner solver suffered from the numerical instability issue. On another front, Vayer et al. (2019a); Mémoli (2011) have introduced the Frank-Wolfe (FW) method (see (Jaggi, 2013; Lacoste-Julien, 2016) for recent treatments) to solve the GW problem. Nevertheless, in their implementation, they still rely on off-the-shelf linear programming solvers and line-search schemes, which are not well-suited for even medium size tasks. it is only recently that Xu et al. (2019b) has further developed a simple heuristic single-loop method (BPG-S) based on BPG, which presented an attractive empirical performance on the node correspondence task. However, such a heuristic still lacks theoretical support, and thus it is not necessarily guaranteed to perform well (or even converge) under naturally noisy observations.

Table 1: Comparison of various algorithms. "Exactly solve " means that algorithms can reach a critical point of the original GW problem instead of the approximation problem.

	Single-loop?	Provable?	Exactly solve (1)?
eBPG (Solomon et al., 2016)	No	Yes	No
BPG (Xu et al., 2019b)	No	Yes	Yes
BPG-S (Xu et al., 2019a)	Yes	No	No
FW (Vayer et al., 2019a)	No	Yes	Yes
BAPG (**our method**)	Yes	Yes	No

To bridge the above theoretical-computational gap, we propose the first provable single-loop algorithm for the GW distance computation, which we call Bregman Alternating Projected Gradient (BAPG). The computational stumbling block here is the Birkhoff polytope constraint (i.e., polytope of doubly stochastic matrices) for the coupling matrix, as the associated Bregman projection is either computationally expensive or practically sensitive to hyperparameters. To address this issue, we are inspired to decouple the Birkhoff polytope as simplex constraints for rows and columns separately, and then execute the projected gradient descent in an alternating fashion. By leveraging the closed-form Bregman projection of the simplex constraint, BAPG only involves the matrix-vector/matrix-matrix multiplications and element-wise matrix operations at each iteration. In short, the proposed single-loop algorithm (BAPG) enjoys a variety of convenient properties: GPU-friendly implementation, robustness concerning the step size (i.e., the only hyper-parameter), and low memory cost. Nonetheless, the iterates generated by BAPG do not necessarily satisfy the Birkhoff polytope constraint and can only reach a critical point of the original GW problem asymptotically. Fortunately, some learning tasks, e.g., graph alignment and partition, only care about the matching accuracy (i.e., performance measure) but not the sharpness of the matching coupling. Thus, we can *sacrifice* some feasibility to gain both matching accuracy and computational efficiency. For these practically relevant tasks, BAPG is already able to obtain the desirable performance and achieve a great balance between accuracy and efficiency. The comparison of various algorithms is summarized in Table 1.

Next, we investigate the approximation bound and convergence behavior of the proposed BAPG. A key fact here is that the GW problem is a symmetric nonconvex quadratic program with polytope constraints. By fully exploring this problem-specific structure, it is surprisingly interesting to note that the GW problem satisfies the Luo-Tseng error bound condition (Luo & Tseng, 1992), which acts as a main technical tool to bound the distance between the fixed-point set of BAPG and the critical point set of the GW problem. This is a notable departure from the usual approach to utilizing the Luo-Tseng error bound condition in establishing the local linear convergence rate for structured convex problems (Zhou & So, 2017). As a result, we are able to quantify the approximation bound for the fixed-point set of BAPG explicitly and further establish the subsequent convergence result.

Lastly, we provide extensive experiment results to validate the effectiveness of BAPG on graph alignment and graph partition. In terms of both wall-clock time and modeling performance (i.e., matching accuracy), we have shown that BAPG consistently achieves superior performance compared with other heuristic single-loop or theoretically sound double-loop methods. We also conduct the sensitivity analysis of BAPG and justify the effectiveness of its GPU acceleration on both synthetic and real-world datasets. All theoretical insights and results have been well-corroborated in the experiments.

2 Proposed Algorithm

In this section, we first formulate the GW distance as a nonconvex quadratic problem with Birkhoff polytope constraints. Subsequently, we provide the theoretical insights and computational properties of the proposed BAPG.

2.1 Problem Setup

The Gromov-Wasserstein (GW) distance was originally proposed in (Mémoli, 2011; 2014; Peyré et al., 2019) for quantifying the distance between two probability measures supported on heterogeneous metric spaces. More precisely:

Definition 2.1 (GW distance)**.** *Suppose that we are given two unregistered compact metric spaces* $(\mathcal{X}, d_X)$*,* $(\mathcal{Y}, d_Y)$ *accompanied with Borel probability measures* μ, ν *respectively. The GW distance*

between μ and ν is defined as

$$\inf_{\pi \in \Pi(\mu,\nu)} \iint |d_X(x,x') - d_Y(y,y')|^2 d\pi(x,y) d\pi(x',y'),$$

where $\Pi(\mu,\nu)$ is the set of all probability measures on $\mathcal{X} \times \mathcal{Y}$ with μ and ν as marginals.

Intuitively, the GW distance is trying to preserve the isometric structure between two probability measures under optimal derivation. If a map pairs $x \to y$ and $x' \to y'$, then the distance between x and x' is supposed to be close to the distance between y and y'. In view of these nice properties, the GW distance acts as a powerful modeling tool in structural data analysis, especially in graph learning; see, e.g., (Vayer et al., 2019b; Xu et al., 2019b;a; Solomon et al., 2016; Peyré et al., 2016) and the references therein.

To start with our algorithmic developments, we consider the discrete case for simplicity and practicality, where μ and ν are two empirical distributions, i.e., $\mu = \sum_{i=1}^n \mu_i \delta_{x_i}$ and $\nu = \sum_{j=1}^m \nu_j \delta_{y_j}$. Then, the GW-distance admits the following reformulation:

$$\begin{aligned} \min_{\pi \in \mathbb{R}^{n \times m}} \; & -\mathrm{Tr}(D_X \pi D_Y \pi^T) \\ \text{s.t.} \quad & \pi 1_m = \mu, \; \pi^T 1_n = \nu, \; \pi \geq 0, \end{aligned} \tag{1}$$

where D_X and D_Y are two symmetric distance matrices.

2.2 Bregman Alternating Projected Gradient (BAPG)

Next, we present the proposed Bregman alternating projected gradient (BAPG) method, which is the first provable single-loop algorithm tailored to the GW distance computation. It is easy to observe that Problem (1) is a nonconvex quadratic program with polytope constraints. Naturally, if we want to invoke (Bregman) projected gradient descent type algorithms to address (1), the key difficulty lies in the Birkhoff constraint with respect to the coupling matrix π. In fact, all existing theoretically sound algorithms rely on an inner iterative algorithm to tackle the regularized optimal transport problem at each iteration, such as Sinkhorn (Cuturi, 2013) or semi-smooth Newton method (Li et al., 2020). Arguably, one of the main drawbacks of the double-loop scheme is its computational burden, which is not GPU friendly. To circumvent this drawback, we are motivated to handle the row and columns constraints separately (i.e., $C_1 = \{\pi \geq 0 : \pi 1_m = \mu\}$ and $C_2 = \{\pi \geq 0 : \pi^T 1_n = \nu\}$). The crux of BPAG is to take the alternating projected gradient descent step between C_1 and C_2. Moreover, by fully detecting the hidden structures, we are able to further benefit from the fact that the Bregman projection with respect to the negative entropy for the simplex constraint (e.g., C_1 and C_2) can be extremely efficient computed (Krichene et al., 2015). The above considerations lead to the closed-form updates in each iteration of BAPG:

BAPG

$$\begin{aligned} \pi &\leftarrow \pi \odot \exp(D_X \pi D_Y / \rho), \quad \pi \leftarrow \mathrm{diag}(\mu./\pi 1_m)\pi, \\ \pi &\leftarrow \pi \odot \exp(D_X \pi D_Y / \rho), \quad \pi \leftarrow \pi \mathrm{diag}(\nu./\pi^T 1_n), \end{aligned} \tag{2}$$

where ρ is the step size and $\odot$ denotes element-wise (Hadamard) matrix multiplication. BAPG enjoys several nice properties that are extremely attractive for medium or large-scale graph learning tasks. First, BAPG is a single-loop algorithm that only involves matrix-vector/matrix-matrix multiplications and element-wise matrix operations. All these operations are GPU-friendly. Second, different from the entropic regularization parameter in eBPG, BAPG is more robust to the step size ρ in terms of solution performance. Third, BAPG only involves one memory operation of a large matrix with size nm. Notably, even for large-scale optimal transport problems, the main bottleneck is not floating-point computations, but rather time-consuming memory operations (Mai et al., 2021).

Theoretical Insights for BAPG Now, we start to give the theoretical intuition of why BAPG will work well in practice. For simplicity, we consider the compact form for illustration:

$$\min_{\pi} f(\pi) + g_1(\pi) + g_2(\pi). \tag{3}$$

Here, $f(\pi) = -\mathrm{Tr}(D_X \pi D_Y \pi^T)$ is a nonconvex quadratic function; $g_1(\pi) = \mathbb{I}_{\{\pi \in C_1\}}$ and $g_2(\pi) = \mathbb{I}_{\{\pi \in C_2\}}$ are two indicator functions over closed convex polyhedral sets. To better understand the alternating projected scheme developed in (2), we adopt the operator splitting strategy to reformulate (3) as

$$\min_{\pi = w} f(\pi, w) + g_1(\pi) + g_2(w) \tag{4}$$

where $f(\pi, w) = -\mathrm{Tr}(D_X \pi D_Y w^T)$. Then, the BAPG update (2) can be interpreted as processing the alternating minimization scheme on the constructed penalized function, i.e.,

$$F_\rho(\pi, w) = f(\pi, w) + g_1(\pi) + g_2(w) + \rho D_h(\pi, w).$$

Here, $D_h(\cdot, \cdot)$ is the so-called Bregman divergence, i.e., $D_h(x, y) := h(x) - h(y) - \langle \nabla h(y), x - y \rangle$, where $h(\cdot)$ is the Legendre function, e.g., $\frac{1}{2}\|x\|^2$, relative entropy $x \log x$, etc. For the k-th iteration, the BAPG update takes the form

$$\begin{aligned} \pi^{k+1} &= \arg\min_{\pi \in C_1} \left\{ f(\pi, w^k) + \rho D_h(\pi, w^k) \right\}, \\ w^{k+1} &= \arg\min_{w \in C_2} \left\{ f(\pi^{k+1}, w) + \rho D_h(w, \pi^{k+1}) \right\}. \end{aligned} \tag{5}$$

When h is the relative entropy and we use the same variable to update π and π, (2) can be derived from (5).

It is worth noting that the way to invoke the operator splitting strategy is not for the algorithm design as usual but instead to provide an intuitive way to explain why BAPG will work. As we shall see later, it plays an important role in deriving the approximation bound of the fix-point set of BAPG instead. Moreover, similar to the quadratic penalty method (Nocedal & Wright, 2006), BAPG is an infeasible method that can only converge to a critical point of (1) in an asymptotic sense. In other words, if we choose the parameter ρ as a constant, there is always an infeasibility gap. Fortunately, BAPG is already able to achieve the desired performance for some graph learning tasks, especially those that care more about implementation efficiency and matching accuracy. Recently, (Vincent-Cuaz et al., 2021a) has proposed a relaxed version of GW distance for the graph partition task, which further corroborates our investigation that it is acceptable and promising to sacrifice some feasibility to gain other benefits. Moreover, Séjourné et al. (2021) also proposed a closely related marginal relaxation. That is, we make $\pi = w$ and $F_\rho(\pi, \pi)$ is the objective introduced in (Séjourné et al., 2021). Unfortunately, they did not develop any provable algorithms for the unbalanced GW distance. In Sections 4.2 and 4.3, extensive experiments have been conducted to demonstrate that BAPG has superior performance compared with other existing baselines on graph alignment and partition tasks.

3 Theoretical Results

In this section, we present all theoretical results conducted in this paper, including the approximation bound of the fix-point set of BAPG and its convergence analysis. At the heart of our analysis is that the following regularity condition holds for the GW problem (1).

Proposition 3.1 (Luo-Tseng Error Bound Condition for (1))**.** *There exist scalars $\epsilon > 0$ and $\tau > 0$ such that*

$$\mathrm{dist}(\pi, \mathcal{X}) \le \tau \left\| \pi - \mathrm{proj}_{C_1 \cap C_2}(\pi + D_X \pi D_Y) \right\|, \tag{6}$$

whenever $\|\pi - \mathrm{proj}_{C_1 \cap C_2}(\pi + D_X \pi D_Y)\| \le \epsilon$, where $\mathcal{X}$ is the critical point set of (3) defined by

$$\mathcal{X} = \{\pi \in C_1 \cap C_2 : 0 \in \nabla f(\pi) + \mathcal{N}_{C_1}(\pi) + \mathcal{N}_{C_2}(\pi)\} \tag{7}$$

and $\mathcal{N}_C(\pi)$ is the normal cone to C at π.

As the GW problem is a nonconvex quadratic program with polytope constraint, we can invoke Theorem 2.3 in (Luo & Tseng, 1992) to conclude that the error bound condition (6) holds on the whole feasible set $C_1 \cap C_2$. Proposition 3.1 extends (6) to the whole space $\mathbb{R}^{n \times m}$. This regularity condition is trying to bound the distance of any coupling matrix to the critical point set of the GW problem by its optimality residual, which is characterized by the difference for one step projected gradient descent. It turns out that this error bound condition plays an important role in quantifying the approximation bound for the fixed points set of BAPG explicitly.

3.1 Approximation Bound for the Fix-Point Set of BAPG

To start, we present one key lemma that shall be used in studying the approximation bound of BAPG.

Lemma 3.2. *Let C_1 and C_2 be convex polyhedral sets. There exists a constant $M > 0$ such that*

$$\left\|\mathrm{proj}_{C_1}(x) + \mathrm{proj}_{C_2}(y) - 2\,\mathrm{proj}_{C_1\cap C_2}\left(\frac{x+y}{2}\right)\right\| \leqslant M\left\|\mathrm{proj}_{C_1}(x) - \mathrm{proj}_{C_2}(y)\right\|, \quad \forall x \in C_1, y \in C_2.$$

The proof idea follows essentially from the observation that the inequality can be regarded as the stability of the optimal solution for a linear-quadratic problem, i.e.,

$$(p(r), q(r)) = \begin{array}{ll} \arg\min\limits_{p,q} & \frac{1}{2}\|x-p\|^2 + \frac{1}{2}\|y-q\|^2 \\ \text{s.t.} & p - q = r,\ p \in C_1, q \in C_2. \end{array}$$

The parameter r is indeed the perturbation quantity. If $r = 0$, we have $p(0) = q(0) = \mathrm{proj}_{C_1\cap C_2}(\frac{x+y}{2})$; by choosing $r = \mathrm{proj}_{C_1}(x) - \mathrm{proj}_{C_2}(y)$, it is easy to see that $(p(r), q(r)) = (\mathrm{proj}_{C_1}(x), \mathrm{proj}_{C_2}(y))$. Together with Theorem 4.1 in (Zhang & Luo, 2022), the desired result is obtained. All the proof details are given in Appendix.

Equipped with Lemma 3.2 and Proposition 3.1, it is not hard to obtain the following approximation result.

Proposition 3.3 (Approximation Bound of the Fix-point Set of BAPG)**.** *The point $(\pi^\star, w^\star)$ belongs to the fixed-point set $\mathcal{X}_{BAPG}$ of BAPG if it satisfies*

$$\begin{aligned} &\nabla f(w^\star) + \rho(\nabla h(\pi^\star) - \nabla h(w^\star)) + p = 0, \\ &\nabla f(\pi^\star) + \rho(\nabla h(w^\star) - \nabla h(\pi^\star)) + q = 0, \end{aligned} \tag{8}$$

where $p \in \mathcal{N}_{C_1}(\pi^\star)$ and $q \in \mathcal{N}_{C_2}(w^\star)$. Then, the infeasibility error satisfies $\|\pi^\star - w^\star\| \leq \frac{\tau_1}{\rho}$ and the gap between $\mathcal{X}_{BAPG}$ and $\mathcal{X}$ satisfies

$$\mathrm{dist}\left(\frac{\pi^\star + w^\star}{2}, \mathcal{X}\right) \leq \frac{\tau_2}{\rho},$$

where τ_1 and τ_2 are two constants.

Remark 3.4. *If $\pi^\star = w^\star$, then $\mathcal{X}_{BAPG}$ ((8)) is identical to $\mathcal{X}$ and BAPG can reach a critical point of the GW problem (1). Proposition 3.3 indicates that as $\rho \to +\infty$, the infeasibility error term $\|\pi^\star - w^\star\|$ shrinks to zero and thus BAPG converges to a critical point of (1) in an asymptotic way. Furthermore, it explicitly quantifies the approximation gap when we select the parameter ρ as a constant. The proof can be found in Appendix. The explicit form of τ_1 and τ_2 only depend on the problem itself, including $\sigma_{\max}(D_X)\sigma_{\max}(D_Y)$, the constant for the Luo-Tseng error bound condition in Proposition 3.1 and so on.*

3.2 Convergence Analysis of BAPG

A further natural question is whether BAPG will converge or not. We answer the question in the affirmative. Specifically, we show that under several canonical assumptions, any limit point of BAPG belongs to $\mathcal{X}_{\mathrm{BAPG}}$. Towards that end, let us first establish the sufficient decrease property of BAPG based on the potential function $F_\rho(\cdot)$,

Proposition 3.5. *Let $\{(\pi^k, w^k)\}_{k\geq 0}$ be the sequence generated by BAPG. Suppose that $D_h(\cdot,\cdot)$ is symmetric on the whole sequence. Then, we have*

$$F_\rho(\pi^{k+1}, w^{k+1}) - F_\rho(\pi^k, w^k) \leq -\rho D_h(\pi^k, \pi^{k+1}) - \rho D_h(w^k, w^{k+1}). \tag{9}$$

As $F_\rho(\cdot)$ is coercive, we have $\sum_{k=0}^{\infty} D_h(\pi^k, \pi^{k+1}) + D_h(w^k, w^{k+1}) < +\infty$. Both $\{D_h(\pi^k, \pi^{k+1})\}_{k\geq 0}$ and $\{D_h(w^k, w^{k+1})\}_{k\geq 0}$ converge to zero. Thus, the following convergence result holds.

Theorem 3.6 (Subsequent Convergence of BAPG)**.** *Any limit point of the sequence $\{(\pi^k, w^k)\}_{k\geq 0}$ generated by BAPG belongs to $\mathcal{X}_{BAPG}$.*

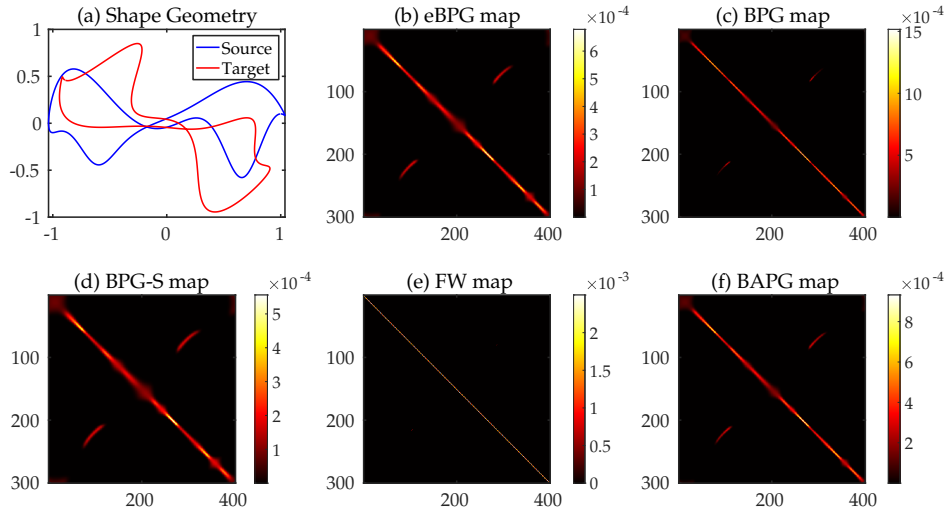


Figure 1: (a): 2D shape geometry of the source and target; (b)-(f): visualization of coupling matrix.

Remark 3.7. *The symmetric assumption in Proposition 3.5 does not hold for the KL divergence in general. However, if we add a mild condition that the asymmetric ratio is bounded for iterations, similar results can be achieved. Moreover, when h is a quadratic function, we can further obtain the global convergence result under the Kurdyka-Lojasiewicz analysis framework developed in (Attouch et al., 2010; 2013).*

To the best of our knowledge, the convergence analysis of alternating projected gradient descent methods has only been given under the convex setting, see (Wang & Bertsekas, 2016; Nedić, 2011) for details. In this paper, by heavily exploiting the error bound condition of the GW problem, we take the first step and provide a new path to conduct the analysis of alternating projected descent method for nonconvex problems, which could be of independent interest.

4 Experiment Results

In this section, we provide extensive experiment results to validate the effectiveness of the proposed BAPG on various representative graph learning tasks, including graph alignment and graph partition. All simulations are implemented using Python 3.9 on a high-performance computing server running Ubuntu 20.04 with an Intel(R) Xeon(R) Gold 6226R CPU and an NVIDIA GeForce RTX 3090 GPU. For all methods conducted in the experiment part, we use the relative error $\|\pi^{k+1}-\pi^k\|_2/\|\pi^k\|_2 \leq 1e^{-6}$ and the maximum iteration as the stopping criterion, i.e., $\min\{k \in \mathbb{Z} : \|\pi^{k+1}-\pi^k\|_2/\|\pi^k\|_2 \leq 1e^{-6} \text{ and } k \leq 2000\}$.

4.1 Toy 2D Matching Problem

In this subsection, we study a toy matching problem in 2D to corroborate our theoretical insights and results in Sec 2 and 3. Fig. 1 (a) shows an example of mapping a two-dimensional shape without any symmetries to a rotated version of the same shape. Here, we sample 300 and 400 points from source and target shapes respectively and use the Euclidean distance to construct the distance matrices D_X and D_Y.

Figs.1 (b)-(f) provide all color maps of coupling matrices to visualize the matching results. Here, the sparser coupling matrices indicate sharper mapping. All experiment results are consistent with Table 1. We can observe that both BPG and FW give us satisfactory solution performance as they aim at solving the GW problem exactly. However, FW and BPG will suffer from a huge computation burden which will be further justified in Sec 4.2 and 4.3. On the other hand, the performance of BPG-S and eBPG are obviously harmed by the infeasibility issue. The sharpness of BAPG's coupling matrix is relatively not effected by its infeasibility issue too much although its coupling matrix is denser than BPG and FW ones. As we shall see later, the effect of the infeasibility issue is minor

Table 2: Comparison of the matching accuracy (%) and wall-clock time (seconds) on graph alignment. For BAPG, we also report the time of GPU implementation.

Method	Synthetic		Proteins			Enzymes			Reddit		
	Acc	Time	Raw	Noisy	Time	Raw	Noisy	Time	Raw	Noisy	Time
FW	24.50	8182	29.96	20.24	54.2	32.17	22.80	10.8	21.51	17.17	1121
ScalaGW	17.93	12002	16.37	16.05	372.2	12.72	11.46	213.0	0.54	0.70	1109
SpecGW	13.27	1462	78.11	19.31	**30.7**	79.07	21.14	**6.7**	50.71	19.66	1074
eBPG	34.33	9502	67.48	45.85	208.2	78.25	60.46	499.7	3.76	3.34	1234
BPG	57.56	22600	71.99	52.46	130.4	79.19	62.32	73.1	39.04	36.68	1907
BPG-S	61.48	18587	71.74	52.74	40.4	79.25	62.21	13.4	39.04	36.68	1431
BAPG	**99.79**	9024	**78.18**	**57.16**	59.1	**79.66**	**62.85**	14.8	**50.93**	**49.45**	780
BAPG-GPU	-	**1253**	-	-	75.4	-	-	21.8	-	-	**115**

when the penalty parameter ρ is not too small and will not even result in even a real cost for graph alignment and partition tasks, which only care about the matching accuracy instead of the sharpness of the coupling.

4.2 Graph Alignment

Graph alignment aims to identify the node correspondence between two graphs possibly with different topology structures (Zhang et al., 2021; Chen et al., 2020). Instead of solving the restricted quadratic assignment problem (Lawler, 1963; Lacoste-Julien et al., 2006), the GW distance provides the optimal probabilistic correspondence relationship via preservation of the isometric property. Here, we compare the proposed BAPG with all existing baselines: FW (Vayer et al., 2019a), BPG (Xu et al., 2019b), BPG-S (Xu et al., 2019b) (i.e., the only difference between BPG and BPG-S is that the number of inner iterations for BPG-S is just one), ScalaGW (Xu et al., 2019a), SpecGW (Chowdhury & Needham, 2021), and eBPG (Solomon et al., 2016). Except for BPG and eBPG, others are pure heuristic methods without any theoretical guarantee.

Parameters Setup We utilize the unweighted symmetric adjacency matrices as our input distance matrices, i.e., D_X and D_Y. Alternatively, SpecGW uses the heat kernel $\exp(-L)$ where L is the normalized graph Laplacian matrix. We set both μ and ν to be the uniform distribution. For three heuristic methods — BPG-S, ScalaGW, and SpecGW, we follow the same setup reported in their papers. As mentioned, eBPG is very sensitive to the entropic regularization parameter. To get comparable results, we report the best result among the set $\{0.1, 0.01, 0.001\}$ of the regularization parameter. For BPG and BAPG, we use the constant step size 5 and $\rho = 0.1$ respectively. For FW, we use the default implementation in the PythonOT package (see Appendix D). All the experiment results reported here were the average of 5 independent trials over different random seeds and the standard deviation is collected in Appendix.

Database Statistics We test all methods on both synthetic and real-world databases. Our setup for the synthetic database is the same as in (Xu et al., 2019b). The source graph $\mathcal{G}_s = \{\mathcal{V}_s, \mathcal{E}_s\}$ is generated by two ideal random models, Gaussian random partition and Barabasi-Albert models, with different scales, i.e., $|\mathcal{V}_s| \in \{500, 1000, 1500, 2000, 2500\}$. Then, we generate the target graph $\mathcal{G}_t = \{\mathcal{V}_t, \mathcal{E}_t\}$ by first adding $q\%$ noisy nodes to the source graph, and then generating $q\%$ noisy edges between the nodes in V_t, i.e., $|\mathcal{V}_t| = (1+q\%)|\mathcal{V}_s|, |\mathcal{E}_t| = (1+q\%)|\mathcal{E}_s|$, where $q \in \{0, 10, 20, 30, 40, 50\}$. For each setup, we generate five synthetic graph pairs over different random seeds. To sum up, the synthetic database contains 300 different graph pairs. We also validate our proposed methods on other three real-world databases from (Chowdhury & Needham, 2021), including two biological graph databases *Proteins* and *Enzymes*, and a social network database *Reddit*. Furthermore, to demonstrate the robustness of our method regarding the noise level, we follow the noise generating process (i.e., $q = 10\%$) conducted for the synthesis case to create new databases on top of the three real-world databases. Towards that end, the statistics of all databases used for the graph alignment task have been summarized in Appendix D. We match each node in $\mathcal{G}_s$ with the most likely node in $\mathcal{G}_t$ according to the optimized $\pi^\star$. Given the predicted correspondence set $\mathcal{S}_{\text{pred}}$ and the ground-truth correspondence set $\mathcal{S}_{\text{gt}}$, we calculate the matching accuracy by Acc $= |\mathcal{S}_{\text{gt}} \cap \mathcal{S}_{\text{pred}}|/|\mathcal{S}_{\text{gt}}| \times 100\%$.

Table 3: Comparison of AMI scores on graph partition datasets using the adjacency matrices and the heat kernel matrices.

Category	Method	Wikipedia		EU-email		Amazon		Village	
		Raw	Noisy	Raw	Noisy	Raw	Noisy	Raw	Noisy
Non-GW	FastGreedy	0.382	0.341	0.312	0.251	0.637	0.573	**0.881**	0.778
	Louvain	0.377	0.329	0.447	0.382	0.622	**0.584**	**0.881**	**0.827**
	Infomap	0.332	0.329	0.374	0.379	**0.940**	0.463	**0.881**	0.190
Adjacency	FW	0.341	0.323	0.440	0.409	0.374	0.338	0.684	0.539
	eBPG	0.461	**0.413**	**0.517**	0.422	0.429	0.387	0.703	0.658
	BPG	0.367	0.333	0.478	0.414	0.412	0.368	0.642	0.575
	BPG-S	0.357	0.285	0.451	0.404	**0.443**	0.352	0.606	0.560
	BAPG	**0.468**	0.385	0.508	**0.428**	0.436	**0.426**	**0.709**	**0.681**
Heat Kernel	SpecGW	0.442	0.395	0.487	0.425	0.565	0.487	0.758	0.707
	eBPG	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
	BPG	0.405	0.373	0.473	0.253	0.492	0.436	0.705	0.619
	BPG-S	0.411	0.373	0.475	0.253	0.483	0.425	0.642	0.619
	BAPG	**0.529**	**0.397**	**0.533**	**0.436**	**0.609**	**0.505**	**0.797**	**0.711**

Results of Our Methods Table 2 shows the comparison of matching accuracy and wall-clock time on four databases. We observe that BAPG works exceptionally well both in terms of computational time and accuracy, especially for two large-scale noisy graph databases *Synthetic* and *Reddit*. Notably, BAPG is robust enough so that it is not necessary to perform parameter tuning. As we mentioned in Sec 2, the effectiveness of GPU acceleration for BAPG is also well corroborated on *Synthetic* and *Reddit*. GPU cannot further speed up the training time of *Proteins* and *Reddit* as graphs in these two databases are small-scale. Additional experiment results to demonstrate the robustness of BAPG and its GPU acceleration will be given in Sec 4.4.

Comparison with Other Methods The performance of eBPG and ScalaGW are influenced by the entropic regularization parameter and approximation error respectively, which accounts for their poor performance. Moreover, it is easy to observe that SpecGW works pretty well on the small dataset but the performance degrades dramatically on the large one, e.g., *synthetic*. The reason is that SpecGW relies on a linear programming solver as its subroutine, which is not well-suited for large-scale settings. Besides, although ScalaGW has the lowest per-iteration computational complexity, the recursive K-partition mechanism developed in (Xu et al., 2019a) is not friendly to parallel computing. Therefore, ScalaGW does not demonstrate attractive performance on multi-core processors.

4.3 Graph Partition

The GW distance can also be potentially applied to the graph partition task. That is, we are trying to match the source graph with a disconnected target graph having K isolated and self-connected super nodes, where K is the number of clusters (Abrishami et al., 2020). Similarly, we compare the proposed BAPG with other baselines described in Section 4.2 on four real-world graph partitioning datasets. Following (Chowdhury & Needham, 2021), we also add three non-GW methods specialized in graph alignment, including FastGreedy (Clauset et al., 2004), Louvain (Blondel et al., 2008), and Infomap (Rosvall & Bergstrom, 2008).

Parameters Setup For the input distance matrices D_X and D_Y, we test our methods on both the adjacency matrices and the heat kernel matrices proposed in (Chowdhury & Needham, 2021). For BAPG, we pick up the lowest converged function value among $\rho \in \{0.1, 0.05, 0.01\}$ for adjacency matrices and $\rho \in \{0.01, 0.005, 0.001\}$ for heat kernel matrices. The quality of graph partition results is quantified by computing the adjusted mutual information (AMI) score (Vinh et al., 2010) against the ground-truth partition.

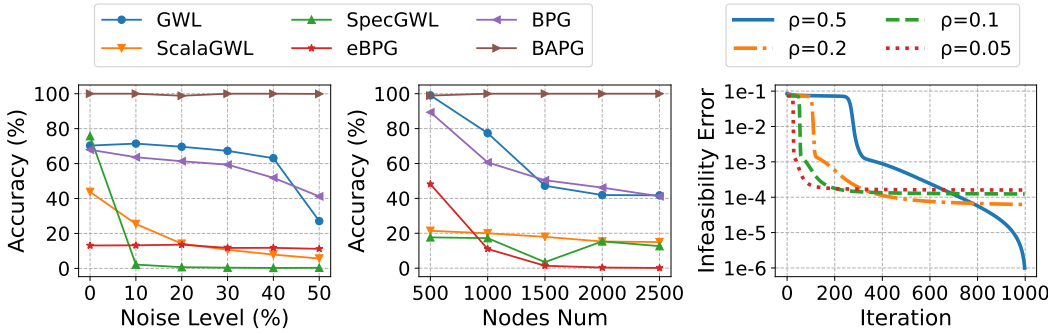
Results of All Methods Table 3 shows the comparison of AMI scores among all methods on graph partition. BAPG outperforms other GW-based methods in most cases and is more robust to the noisy setting. Specifically, BAPG is consistently better than FW and SpecGW which both rely on the Frank-Wolfe method to solve the problem. eBPG has comparable results using the adjacency matrices but is unable to process the spectral matrices. The possible reason is that the adjacency matrix and the heat kernel matrix admit quite different structures, e.g., the former is sparse while

Table 4: The sensitivity analysis of BAPG with different step sizes ρ on graph alignment.

Method	Synthetic		Proteins			Enzymes			Reddit		
	Acc	Time	Clean	Noise	Time	Clean	Noise	Time	Clean	Noise	Time
BAPG (ρ=0.5)	94.58	4610	77.21	57.69	70.8	79.20	63.18	21.8	50.04	49.19	257
BAPG (ρ=0.2)	98.87	2074	78.27	57.96	67.4	79.50	63.06	17.6	50.94	50.46	144
BAPG (ρ=0.1)	99.79	1253	78.18	57.16	59.1	79.66	62.85	14.8	50.93	49.45	115
BAPG (ρ=0.05)	99.20	675	77.59	56.54	32.6	79.27	62.13	11.8	50.96	50.30	98
BAPG (ρ=0.01)	93.63	616	60.08	34.47	22.5	66.45	42.38	8.9	28.74	16.14	85

Table 5: GPU & CPU wall-clock time comparison of BAPG, BPG., and eBPG on graph alignment.

	Reddit Dataset			Synthetic Dataset		
	BAPG	BPG	eBPG	BAPG	BPG	eBPG
CPU Time(s)	780	1907	1234	9024	22600	9502
GPU Time(s)	115	1013	2274	1253	4458	2709
Acceleration Ratio	**6.78**	1.88	0.54	**7.20**	5.07	3.51



(a) (b)

Figure 2: (a) Sensitivity of the noise level and graph scale on the synthetic graph alignment database. (b) Sensitivity of BAPG's step size ρ. The infeasibility error is defined as $\|\pi^k - w^k\|_2$.

the latter is dense. BPG and BPG-S enjoy similar performances in most cases, but both are not as good as our proposed BAPG on all datasets. BAPG also shows competitive performance compared to the specialized non-GW graph partition methods. For example, BAPG outperforms Infomap and Louvain in 6 out of 8 scores, respectively.

4.4 The Effectiveness and Robustness of BAPG

At first, we target at demonstrating the robustness of BAPG on graph alignment, as it would be more reasonable to test the robustness of a method on a database (e.g., graph alignment) rather than a single point (e.g., graph partition).

Noise Level and Graph Scale At the beginning, we report the sensitivity analysis of BAPG regarding the noise level $q\%$ and the graph scale $|\mathcal{V}_s|$ in Fig. 2 on the Synthetic Database. Surprisingly, the solution performance of BAPG is robust to both the noise level and graph scale. In contrast, the accuracy of other methods degrades dramatically as the noise level or the graph scale increases.

Step Size ρ Subsequently, we conduct the sensitivity analysis of BAPG with respect to the step size ρ on four real-world databases. Table 4 showcases that the matching performance of BAPG is robust (not sensitive) regarding the step size. Moreover, convergence curves of different ρ in Fig. 2 corroborate Proposition 3.3 empirically and further guide us on how to choose the step size. The larger ρ means the smaller infeasibility error but slower convergence speed. Together with Table 4, we can conclude that if the resulting infeasibility error is not quite large (i.e., the step size ρ is not so small), the final matching accuracy is robust to the step size. Under this situation, the step size only affects the convergence speed.

GPU Acceleration of BAPG. We conduct experiment results to further justify that BAPG is GPU-friendly. In Table 5, We compare the acceleration ratio (i.e., CPU wall-clock time divided by GPU wall-clock time) of BAPG, eBPG, and BPG on two large-scale graph alignment datasets using the same computing server. For eBPG, we use the official implementation in the PythonOT package, which supports running on GPU. For BPG, we implement the GPU version by ourselves using Pytorch. We can find that BAPG has a much higher acceleration ratio on the GPU compared with BPG and eBPG.

5 Closing Remark

In this paper, we conduct a systematic investigation on developing the provable single-loop algorithm (i.e., BAPG) for the GW distance computation. By exploiting the error bound condition (i.e., never investigated in the GW literature before), we are able to conduct the convergence analysis of BAPG. For a broader viewpoint, our work also takes the first step and provides a rather new path to study the alternating projected descent method for general nonconvex problems. To the best of our knowledge, it is still an open question even in the optimization community. On the computational side, the proposed single-loop algorithm BAPG still suffers from the cubic per-iteration computational complexity, which will further limit its applications in real-world large-scale settings. A natural future direction is to consider sparse and low-rank structures of the matching matrix to decrease the per-iteration cost and further speed up our method (Scetbon et al., 2021). Moreover, for simplicity, we assume that D_X and D_Y are two symmetric distance matrices. However, the proposed BAPG can be potentially applied to the non-symmetric case as the Luo-tseng error bound condition still holds and algorithmic steps keep the same.

References

Tara Abrishami, Nestor Guillen, Parker Rule, Zachary Schutzman, Justin Solomon, Thomas Weighill, and Si Wu. Geometry of graph partitions via optimal transport. *SIAM Journal on Scientific Computing*, 42(5):A3340–A3366, 2020.

Hédy Attouch, Jérôme Bolte, Patrick Redont, and Antoine Soubeyran. Proximal alternating minimization and projection methods for nonconvex problems: An approach based on the kurdyka-łojasiewicz inequality. *Mathematics of operations research*, 35(2):438–457, 2010.

Hedy Attouch, Jérôme Bolte, and Benar Fux Svaiter. Convergence of descent methods for semi-algebraic and tame problems: proximal algorithms, forward–backward splitting, and regularized gauss–seidel methods. *Mathematical Programming*, 137(1):91–129, 2013.

Heinz H. Bauschke. *Projection Algorithms and Monotone Operators*. PhD thesis, Simon Fraser University, 1996.

Jean-David Benamou, Guillaume Carlier, Marco Cuturi, Luca Nenna, and Gabriel Peyré. Iterative bregman projections for regularized transportation problems. *SIAM Journal on Scientific Computing*, 37(2):A1111–A1138, 2015.

Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment*, 2008 (10):P10008, 2008.

Charlotte Bunne, David Alvarez-Melis, Andreas Krause, and Stefanie Jegelka. Learning generative models across incomparable spaces. In *International Conference on Machine Learning*, pp. 851–861. PMLR, 2019.

Xiyuan Chen, Mark Heimann, Fatemeh Vahedian, and Danai Koutra. Consistent network alignment via proximity-preserving node embedding. *arXiv preprint arXiv:2005.04725*, 2020.

Samir Chowdhury and Facundo Mémoli. The gromov–wasserstein distance between networks and stable network invariants. *Information and Inference: A Journal of the IMA*, 8(4):757–787, 2019.

Samir Chowdhury and Tom Needham. Generalized spectral clustering via gromov-wasserstein learning. In *International Conference on Artificial Intelligence and Statistics*, pp. 712–720. PMLR, 2021.

Aaron Clauset, Mark EJ Newman, and Cristopher Moore. Finding community structure in very large networks. *Physical review E*, 70(6):066111, 2004.

Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. *Advances in neural information processing systems*, 26:2292–2300, 2013.

Rémi Flamary, Nicolas Courty, Alexandre Gramfort, Mokhtar Z. Alaya, Aurélie Boisbunon, Stanislas Chambon, Laetitia Chapel, Adrien Corenflos, Kilian Fatras, Nemo Fournier, Léo Gautheron, Nathalie T.H. Gayraud, Hicham Janati, Alain Rakotomamonjy, Ievgen Redko, Antoine Rolet, Antony Schutz, Vivien Seguy, Danica J. Sutherland, Romain Tavenard, Alexander Tong, and Titouan Vayer. Pot: Python optimal transport. *Journal of Machine Learning Research*, 22(78):1–8, 2021. URL http://jmlr.org/papers/v22/20-451.html.

Ji Gao, Xiao Huang, and Jundong Li. Unsupervised graph alignment with wasserstein distance discriminator. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pp. 426–435, 2021.

Martin Jaggi. Revisiting frank-wolfe: Projection-free sparse convex optimization. In *International Conference on Machine Learning*, pp. 427–435. PMLR, 2013.

Walid Krichene, Syrine Krichene, and Alexandre Bayen. Efficient bregman projections onto the simplex. In *2015 54th IEEE Conference on Decision and Control (CDC)*, pp. 3291–3298. IEEE, 2015.

Simon Lacoste-Julien. Convergence rate of frank-wolfe for non-convex objectives. *arXiv preprint arXiv:1607.00345*, 2016.

Simon Lacoste-Julien, Ben Taskar, Dan Klein, and Michael Jordan. Word alignment via quadratic assignment. 2006.

Eugene L Lawler. The quadratic assignment problem. *Management science*, 9(4):586–599, 1963.

Xudong Li, Defeng Sun, and Kim-Chuan Toh. On the efficient computation of a generalized jacobian of the projector over the birkhoff polytope. *Mathematical Programming*, 179(1-2):419–446, 2020.

Zhi-Quan Luo and Paul Tseng. Error bound and convergence analysis of matrix splitting algorithms for the affine variational inequality problem. *SIAM Journal on Optimization*, 2(1):43–54, 1992.

Vien V Mai, Jacob Lindbäck, and Mikael Johansson. A fast and accurate splitting method for optimal transport: Analysis and implementation. *arXiv preprint arXiv:2110.11738*, 2021.

Facundo Mémoli. Spectral gromov-wasserstein distances for shape matching. In *2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops*, pp. 256–263. IEEE, 2009.

Facundo Mémoli. Gromov–wasserstein distances and the metric approach to object matching. *Foundations of computational mathematics*, 11(4):417–487, 2011.

Facundo Mémoli. The gromov–wasserstein distance: A brief overview. *Axioms*, 3(3):335–341, 2014.

Facundo Mémoli and Guillermo Sapiro. Comparing point clouds. In *Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing*, pp. 32–40, 2004.

Angelia Nedić. Random algorithms for convex minimization problems. *Mathematical programming*, 129(2):225–253, 2011.

Jorge Nocedal and Stephen Wright. *Numerical optimization*. Springer Science & Business Media, 2006.

Gabriel Peyré, Marco Cuturi, and Justin Solomon. Gromov-wasserstein averaging of kernel and distance matrices. In *International Conference on Machine Learning*, pp. 2664–2672. PMLR, 2016.

Gabriel Peyré, Marco Cuturi, et al. Computational optimal transport: With applications to data science. *Foundations and Trends® in Machine Learning*, 11(5-6):355–607, 2019.

Martin Rosvall and Carl T Bergstrom. Maps of random walks on complex networks reveal community structure. *Proceedings of the national academy of sciences*, 105(4):1118–1123, 2008.

Meyer Scetbon, Gabriel Peyré, and Marco Cuturi. Linear-time gromov wasserstein distances using low rank couplings and costs. *arXiv preprint arXiv:2106.01128*, 2021.

Thibault Séjourné, François-Xavier Vialard, and Gabriel Peyré. The unbalanced gromov wasserstein distance: Conic formulation and relaxation. *Advances in Neural Information Processing Systems*, 34:8766–8779, 2021.

Justin Solomon, Gabriel Peyré, Vladimir G Kim, and Suvrit Sra. Entropic metric alignment for correspondence problems. *ACM Transactions on Graphics (TOG)*, 35(4):1–13, 2016.

Titouan Vayer, Laetita Chapel, Rémi Flamary, Romain Tavenard, and Nicolas Courty. Fused gromov-wasserstein distance for structured objects: theoretical foundations and mathematical properties. *arXiv preprint arXiv:1811.02834*, 2018.

Titouan Vayer, Nicolas Courty, Romain Tavenard, and Rémi Flamary. Optimal transport for structured data with application on graphs. In *International Conference on Machine Learning*, pp. 6275–6284. PMLR, 2019a.

Titouan Vayer, Rémi Flamary, Romain Tavenard, Laetitia Chapel, and Nicolas Courty. Sliced gromov-wasserstein. In *NeurIPS 2019-Thirty-third Conference on Neural Information Processing Systems*, volume 32, 2019b.

Cédric Vincent-Cuaz, Rémi Flamary, Marco Corneli, Titouan Vayer, and Nicolas Courty. Semi-relaxed gromov wasserstein divergence with applications on graphs. *arXiv preprint arXiv:2110.02753*, 2021a.

Cédric Vincent-Cuaz, Titouan Vayer, Rémi Flamary, Marco Corneli, and Nicolas Courty. Online graph dictionary learning. *arXiv preprint arXiv:2102.06555*, 2021b.

Nguyen Xuan Vinh, Julien Epps, and James Bailey. Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance. *The Journal of Machine Learning Research*, 11:2837–2854, 2010.

Mengdi Wang and Dimitri P Bertsekas. Stochastic first-order methods with random constraint projection. *SIAM Journal on Optimization*, 26(1):681–717, 2016.

Hongteng Xu, Dixin Luo, and Lawrence Carin. Scalable gromov-wasserstein learning for graph partitioning and matching. *Advances in neural information processing systems*, 32:3052–3062, 2019a.

Hongteng Xu, Dixin Luo, Hongyuan Zha, and Lawrence Carin. Gromov-wasserstein learning for graph matching and node embedding. In *International conference on machine learning*, pp. 6932–6941. PMLR, 2019b.

Hongteng Xu, Dixin Luo, Lawrence Carin, and Hongyuan Zha. Learning graphons via structured gromov-wasserstein barycenters. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 10505–10513, 2021.

Hongteng Xu, Jiachang Liu, Dixin Luo, and Lawrence Carin. Representing graphs via gromov-wasserstein factorization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

Jiawei Zhang and Zhi-Quan Luo. A global dual error bound and its application to the analysis of linearly constrained nonconvex optimization. *SIAM Journal on Optimization*, 32(3):2319–2346, 2022.

Si Zhang, Hanghang Tong, Long Jin, Yinglong Xia, and Yunsong Guo. Balancing consistency and disparity in network alignment. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pp. 2212–2222, 2021.

Zirui Zhou and Anthony Man-Cho So. A unified approach to error bounds for structured convex optimization problems. *Mathematical Programming*, 165(2):689–728, 2017.

A APPENDIX

Overall, the norm $\|\cdot\|$ means ℓ_2 norm. Both dist and proj are defined w.r.t. ℓ_2 norm, i.e., for any convex set C and a point x,

$$\mathrm{dist}(x, C) = \min_{y \in C} \|x - y\|_2, \tag{10}$$

$$\mathrm{proj}_C(x) = \arg\min_{y \in C} \|x - y\|_2. \tag{11}$$

B PROBLEM PROPERTIES OF (1)

In this paper, we consider the discrete case, where only m, n samples are drawn from real distributions, i.e., $\mu = \sum_{i=1}^{n} \mu_i \delta_{x_i}$ and $\nu = \sum_{j=1}^{m} \nu_j \delta_{y_j}$. The Gromov-Wasserstein distance between μ and ν can be recast into the following minimization problem:

$$\begin{aligned} &\min_{\pi \in \Pi(\mu,\nu)} \iint |d_X(x, x') - d_Y(y, y')|^2 d\pi(x, y) d\pi(x', y') \\ \Rightarrow &\min_{\pi \in \Pi(\mu,\nu)} \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{k=1}^{m} \sum_{l=1}^{m} |d_X(x_i, x_j) - d_Y(y_k, y_l)|^2 \pi_{ik} \pi_{jl}, \end{aligned}$$

where $\Pi(\mu, \nu)$ is a generalized Birkhoff polytope, i.e., $\Pi(\mu, \nu) = \{\pi \in \mathbb{R}_+^{n \times m} | \pi 1_m = \mu, \pi^T 1 = \nu\}$. To further simplify the problem and detect the hidden structures, we reformulate it as a compact form,

$$\begin{aligned} &\sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{k=1}^{m} \sum_{l=1}^{m} (D_X(i, j) - D_Y(k, l))^2 \pi_{ik} \pi_{jl} \\ =&\sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{k=1}^{m} \sum_{l=1}^{m} (D_X^2(i, j) + D_Y^2(k, l)) \pi_{ik} \pi_{jl} - 2 \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{k=1}^{m} \sum_{l=1}^{m} D_X(i, j) D_Y(k, l) \pi_{ik} \pi_{jl} \\ =&\underbrace{\sum_{i=1}^{n} \sum_{j=1}^{n} D_X^2(i, j) \mu_i \mu_j + \sum_{k=1}^{m} \sum_{l=1}^{m} D_Y^2(k, l) \nu_k \nu_l}_{\text{constant}} - 2\mathrm{Tr}(D_X \pi D_Y \pi^T). \end{aligned}$$

where $D_X(i, j) = d_X(x_i, x_j)$ and $D_Y(k, l) = d_Y(y_k, y_l)$ for $i, j \in [n]$ and $k, l \in [m]$. Thus, (1) is obtained.

Proof of Proposition 3.1 — Luo-Tseng Error Bound Condition for the GW Problem (1)

Proof. D_X and D_Y are two symmetric matrices and $C_1 \cap C_2$ is a convex polyhedral set. By invoking Theorem 2.3 in [24], the Luo-Tseng local error bound condition (6) just holds for the feasible set $C_1 \cap C_2$. That is,

$$\mathrm{dist}(\pi, \mathcal{X}) \leq \tau \left\| \pi - \mathrm{proj}_{C_1 \cap C_2}(\pi + D_X \pi D_Y) \right\|, \tag{12}$$

where $\pi \in C_1 \cap C_2$. Then, we aim at extending (12) to the whole space. Define $\tilde{\pi} = \mathrm{proj}_{C_1 \cap C_2}(\pi)$, we have

$$\mathrm{dist}(\pi, \mathcal{X}) \leq \|\pi - \tilde{\pi}\| + d(\tilde{\pi}, \mathcal{X}) \leq \|\pi - \tilde{\pi}\| + \tau \|\tilde{\pi} - \mathrm{proj}_{C1 \cap C_2}(\tilde{\pi} + D_X \tilde{\pi} D_Y)\| \tag{13}$$

where the first inequality holds because of the triangle inequality, and the second one holds because of the fact $\tilde{\pi} \in C_1 \cap C_2$ and (12). Apply the triangle inequality again, we have

$$\begin{aligned} &\|\tilde{\pi} - \mathrm{proj}_{C1 \cap C_2}(\tilde{\pi} + D_X \tilde{\pi} D_Y)\| \\ \leq &\|\tilde{\pi} - \mathrm{proj}_{C1 \cap C_2}(\pi + D_X \pi D_Y)\| + \|\mathrm{proj}_{C1 \cap C_2}(\pi + D_X \pi D_Y) - \mathrm{proj}_{C1 \cap C_2}(\tilde{\pi} + D_X \tilde{\pi} D_Y)\| \end{aligned} \tag{14}$$

Since $\tilde{\pi} = \mathrm{proj}_{C_1 \cap C_2}(\pi)$, owing to the fact that the projection operator onto a convex set $\mathrm{proj}_{C_1 \cap C_2}(\cdot)$ is non-expensive, i.e., $\|\mathrm{proj}_{C_1 \cap C_2}(x) - \mathrm{proj}_{C_1 \cap C_2}(y)\| \leq \|x - y\|$ holds for any x and y, we have

$$\begin{aligned} \|\tilde{\pi} - \mathrm{proj}_{C1 \cap C_2}(\pi + D_X \pi D_Y)\| &= \|\mathrm{proj}_{C_1 \cap C_2}(\pi) - \mathrm{proj}_{C1 \cap C_2}(\mathrm{proj}_{C1 \cap C_2}(\pi + D_X \pi D_Y))\| \\ &\leq \|\pi - \mathrm{proj}_{C1 \cap C_2}(\pi + D_X \pi D_Y)\|, \end{aligned} \tag{15}$$

and

$$\begin{aligned}\|\operatorname{proj}_{C1\cap C_2}(\pi + D_X\pi D_Y) - \operatorname{proj}_{C1\cap C_2}(\tilde{\pi} + D_X\tilde{\pi}D_Y)\| &\le \|(\pi + D_X\pi D_Y) - (\tilde{\pi} + D_X\tilde{\pi}D_Y)\| \\ &\le (\sigma_{\max}(D_X)\sigma_{\max}(D_Y) + 1)\|\pi - \tilde{\pi}\|\end{aligned} \tag{16}$$

where $\sigma_{\max}(D_X)$ and $\sigma_{\max}(D_Y)$ denote the maximum singular value of D_X and D_Y, respectively. By substituting (14) and (16) into (13), we get

$$\begin{aligned}\operatorname{dist}(\pi, \mathcal{X}) &\le (\tau\sigma_{\max}(D_X)\sigma_{\max}(D_Y) + \tau + 1)\|\pi - \tilde{\pi}\| + \tau\|\pi - \operatorname{proj}_{C_1\cap C_2}(\pi + D_X\pi D_Y)\| \\ &\le (\tau\sigma_{\max}(D_X)\sigma_{\max}(D_Y) + 2\tau + 1)\|\pi - \operatorname{proj}_{C_1\cap C_2}(\pi + D_X\pi D_Y)\|\end{aligned} \tag{17}$$

where the last inequality holds because of the fact that

$$\|\pi - \tilde{\pi}\| = \|\pi - \operatorname{proj}_{C_1\cap C_2}(\pi)\| = \min_{x\in C_1\cap C_2}\|\pi - x\| \le \|\pi - \operatorname{proj}_{C_1\cap C_2}(\pi + D_X\pi D_Y)\|.$$

By letting $\tau = \tau\sigma_{\max}(D_X)\sigma_{\max}(D_Y) + 2\tau + 1$, we get the desired result. □

C Convergence Analysis of BAPG

Assumption C.1. *The critical point set $\mathcal{X}$ is non-empty.*

Definition C.2 (Bregman Divergence)**.** *We define the proximity measure $D_h : \operatorname{dom}(h) \times \operatorname{int}(\operatorname{dom}(h)) \to \mathbb{R}_+$*

$$D_h(x, y) := h(x) - h(y) - \langle \nabla h(y), x - y\rangle$$

The proximity measure D_h is the so-called Bregman Distance. It measures the proximity between x and y. Indeed, thanks to the gradient inequality, one has h is convex if and only if $D_h(x, y) \ge 0, \forall x \in \operatorname{dom} h, y \in \operatorname{int}\operatorname{dom} h$.

Concrete Examples

- Relative entropy, or Kullback–Leibler divergence, $h(x) = \sum_i x_i \log x_i$ and the Bregman divergence is given as
$$D_h(x, y) = \sum_i x_i \log \frac{x_i}{y_i}.$$
- Quadratic function: $h(x) = \frac{1}{2}\|x\|^2$. As such, the Bregman divergence $D_h(x, y)$ will reduce to the Euclidean distance.

For the completeness of our algorithmic development, we also give the details how the general BAPG (5) update implies (2) if we choose the Legendre function as relative entropy.

$$\begin{aligned}\pi^{k+1} &= \arg\min_{\pi\in C_1}\left\{-\langle \pi, D_Y w^k D_Y\rangle + \rho D_h(\pi, w^k)\right\} \\ &= \arg\min_{\pi\in C_1} D_h\left(\pi, w^k \odot \exp\left(\frac{D_X w^k D_Y}{\rho}\right)\right), \\ &= \operatorname{diag}(\mu./P^k 1_m)P^k,\end{aligned}$$

where $P^k = w^k \odot \exp\left(\frac{D_X w^k D_Y}{\rho}\right)$.

$$\begin{aligned}w^{k+1} &= \arg\min_{w\in C_2}\left\{-\langle w, D_X \pi^{k+1} D_Y\rangle + \rho D_h(w, \pi^{k+1})\right\} \\ &= \arg\min_{w\in C_2} D_h\left(w, \pi^{k+1} \odot \exp\left(\frac{D_X \pi^{k+1} D_Y}{\rho}\right)\right), \\ &= Z^k \operatorname{diag}(\nu./{Z^k}^T 1_n),\end{aligned}$$

where $Z^k = \pi^{k+1} \odot \exp\left(\frac{D_X \pi^{k+1} D_Y}{\rho}\right)$. The optimality conditions for each iteration follow,

$$\begin{cases}0 \in -D_X w^k D_Y + \rho(\nabla h(\pi^{k+1}) - \nabla h(w^k)) + \mathcal{N}_{C_1}(\pi^{k+1}), \\ 0 \in -D_X \pi^{k+1} D_Y + \rho(\nabla h(w^{k+1}) - \nabla h(\pi^{k+1})) + \mathcal{N}_{C_2}(w^{k+1}).\end{cases} \tag{18}$$

Proof of Lemma 3.2

Proof. To prove

$$\left\|\mathrm{proj}_{C_1}(x)+\mathrm{proj}_{C_2}(y)-2\,\mathrm{proj}_{C_1\cap C_2}\left(\frac{x+y}{2}\right)\right\| \leqslant M\left\|\mathrm{proj}_{C_1}(x)-\mathrm{proj}_{C_2}(y)\right\|, \tag{19}$$

we first convert the left-hand side of (19) to

$$\begin{aligned}&\left\|\mathrm{proj}_{C_1}(x)-\mathrm{proj}_{C_1\cap C_2}\left(\frac{x+y}{2}\right)+\mathrm{proj}_{C_2}(y)-\mathrm{proj}_{C_1\cap C_2}\left(\frac{x+y}{2}\right)\right\|\\ \leq\ &\sqrt{2}\left\|\left(\mathrm{proj}_{C_1\cap C_2}\left(\frac{x+y}{2}\right),\mathrm{proj}_{C_1\cap C_2}\left(\frac{x+y}{2}\right)\right)-\left(\mathrm{proj}_{C_1}(x),\mathrm{proj}_{C_2}(y)\right)\right\|.\end{aligned}$$

Then, the inequality can be regarded as the stability of the optimal solution for a linear-quadratic problem, i.e.,

$$(p(r),q(r))=\begin{array}{ll}\arg\min\limits_{p,q} & \frac{1}{2}\|x-p\|^2+\frac{1}{2}\|y-q\|^2\\ \text{s.t.} & p-q=r,\\ & p\in C_1, q\in C_2.\end{array} \tag{20}$$

When $r=0$, the pair $(p(0),q(0))$ satisfies $p(0)=q(0)=\mathrm{proj}_{C_1\cap C_2}\left(\frac{x+y}{2}\right)$. Moreover, the parameter r itself can be viewed as the perturbation quantity, which is indeed the right-hand side of (19). By invoking Theorem 4.1 in (Zhang & Luo, 2022), we can bound the distance between two optimal solutions by the perturbation quantity r, i.e.,

$$\|(p(0),q(0))-(p(r),q(r))\|\leq M\|r\|.$$

Remark C.3. *For any vector r, let $(p(r),q(r))$ be the optimal solution to (20). On the one hand, choosing $r=0$, it is easy to see that $p(0)=q(0)=\mathrm{proj}_{C_1\cap C_2}(\frac{x+y}{2})$. On the other hand, by choosing $r=\mathrm{proj}_{C_1}(x)-\mathrm{proj}_{C_2}(y)$, it is easy to see that $(p(r),q(r))=(\mathrm{proj}_{C_1}(x),\mathrm{proj}_{C_2}(y))$. Since Theorem 4.1 in (Zhang & Luo, 2022) implies that for any r, we have*

$$\|(p(0),q(0))-(p(r),q(r))\|_2\leq M\|r\|_2$$

where M only depends on C_1 and C_2. By choosing $r=\mathrm{proj}_{C_1}(x)-\mathrm{proj}_{C_2}(y)$, we have

$$\|(\mathrm{proj}_{C_1\cap C_2}(\frac{x+y}{2}),\mathrm{proj}_{C_1\cap C_2}(\frac{x+y}{2}))-(\mathrm{proj}_{C_1}(x),\mathrm{proj}_{C_2}(y))\leq M\|\mathrm{proj}_{C_1}(x)-\mathrm{proj}_{C_2}(y)\|_2.$$

□

Proof of Proposition 3.3 — Approximation bound of fixed points of BAPG

Definition C.4 (**Bounded Linear Regularity** (cf. Definition 5.6 in Bauschke (1996))). *Let $C_1,\ldots,C_N$ be closed convex subsets of $\mathbb{R}^n$ with a non-empty intersection C. We say that the collection $\{C_1,\ldots,C_N\}$ is* bounded linearly regular (BLR) *if for every bounded subset $\mathbb{B}$ of $\mathbb{R}^n$, there exists a constant $\kappa>0$ such that*

$$d(x,C)\leq\kappa\max_{i\in\{1,\ldots,N\}}d(x,C_i),\ \textit{for all } x\in\mathbb{B}.$$

If all of C_i are polyhedral sets, the BLR condition will automatically hold.

Proof. Recall the fixed-point set of BAPG:

$$\mathcal{X}_{\text{BAPG}}=\left\{(\pi^\star,w^\star):\begin{array}{l}\nabla f(w^\star)+\rho(\nabla h(\pi^\star)-\nabla h(w^\star))+p=0, p\in\mathcal{N}_{C_1}(\pi^\star)\\ \nabla f(\pi^\star)+\rho(\nabla h(w^\star)-\nabla h(\pi^\star))+q=0, q\in\mathcal{N}_{C_2}(w^\star)\end{array}\right\}. \tag{21}$$

Define $\hat{\pi}=\mathrm{proj}_{C_1\cap C_2}(\pi^\star)$, we first want to argue the following inequality holds,

$$\|\hat{\pi}-\pi^\star\|+\|\hat{\pi}-w^\star\|\leq(2\kappa+1)\|\pi^\star-w^\star\|.$$

As the BLR condition is satisfied for the polyhedral constraint, see Definition C.4 for details, we have

$$\begin{aligned}\|\hat{\pi}-\pi^\star\|+\|\hat{\pi}-w^\star\| &\le 2\|\hat{\pi}-\pi^\star\|+\|\pi^\star-w^\star\| \\ &= 2\,\mathrm{dist}(\pi^\star, C_1\cap C_2)+\|\pi^\star-w^\star\| \\ &\le (2\kappa+1)\|\pi^\star-w^\star\|.\end{aligned} \tag{22}$$

Based on the stationary points defined in (8), we have,

$$\nabla f(w^\star)^T(\hat{\pi}-\pi^\star)+\rho(\nabla h(\pi^\star)-\nabla h(w^\star))^T(\hat{\pi}-\pi^\star)+p^T(\hat{\pi}-\pi^\star)=0, p\in\mathcal{N}_{C_1}(\pi^\star),$$
$$\nabla f(\pi^\star)^T(\hat{\pi}-w^\star)+\rho(\nabla h(w^\star)-\nabla h(\pi^\star))^T(\hat{\pi}-w^\star)+q^T(\hat{\pi}-w^\star)=0, q\in\mathcal{N}_{C_2}(w^\star).$$

Summing up the above two equations,

$$\begin{aligned}&\nabla f(w^\star)^T(\hat{\pi}-\pi^\star)+\nabla f(\pi^\star)^T(\hat{\pi}-w^\star)+\rho(\nabla h(\pi^\star)-\nabla h(w^\star))^T(w^\star-\pi^\star)+p^T(\hat{\pi}-\pi^\star)+q^T(\hat{\pi}-w^\star)=0\\ \overset{(\clubsuit)}{\Rightarrow}&\nabla f(w^\star)^T(\hat{\pi}-\pi^\star)+\nabla f(\pi^\star)^T(\hat{\pi}-w^\star)-\rho(D_h(\pi^\star,w^\star)+D_h(w^\star,\pi^\star))+p^T(\hat{\pi}-\pi^\star)+q^T(\hat{\pi}-w^\star)=0\\ \overset{(\spadesuit)}{\Rightarrow}&\nabla f(w^\star)^T(\hat{\pi}-\pi^\star)+\nabla f(\pi^\star)^T(\hat{\pi}-w^\star)-\rho(D_h(\pi^\star,w^\star)+D_h(w^\star,\pi^\star))\ge 0,\end{aligned}$$

where $(\clubsuit)$ holds as $(\nabla h(x)-\nabla h(y))^T(x-y)=D_h(x,y)+D_h(y,x)$ and $(\spadesuit)$ follows from the property of the normal cone. For instance, since $q\in\mathcal{N}_{C_1}(\pi^\star)$, we have $q^T(\hat{\pi}-\pi^\star)\le 0$ (i.e., $\hat{\pi}\in C_1$). Therefore,

$$\begin{aligned}\rho(D_h(\pi^\star,w^\star)+D_h(w^\star,\pi^\star)) &\le \nabla f(w^\star)^T(\hat{\pi}-\pi^\star)+\nabla f(\pi^\star)^T(\hat{\pi}-w^\star)\\ &\le \|\nabla f(w^\star)\|\|\hat{\pi}-\pi^\star\|+\|\nabla f(\pi^\star)\|\|\hat{\pi}-w^\star\|\\ &\le L_f(\|\hat{\pi}-\pi^\star\|+\|\hat{\pi}-w^\star\|)\\ &\overset{(22)}{\le}(2\kappa+1)L_f\|\pi^\star-w^\star\|.\end{aligned}$$

The next to last inequality holds as $f(\cdot)$ is a quadratic function and the effective domain $C_1\cap C_2$ is bounded. Thus, the norm of its gradient will naturally have a constant upper bound, i.e., $L_f=\sigma_{\max}(D_X)\sigma_{\max}(D_Y)$. As h is a σ-strongly convex, we have

$$D_h(\pi^\star,w^\star)\ge\frac{\sigma}{2}\|\pi^\star-w^\star\|^2.$$

Together with this property, we can quantify the infeasibility error $\|\pi^\star-w^\star\|$,

$$\|\pi^\star-w^\star\|\le\frac{(2\kappa+1)L_f}{\sigma\rho}. \tag{23}$$

When $\rho\to+\infty$, it is easy to observe that the infeasibility error term $\|\pi^\star-w^\star\|$ will shrink to zero. More importantly, if $\pi^\star=w^\star$, then X_{BAPG} will be identical to $\mathcal{X}$. Next, we target at quantifying the approximation gap between the fixed-point set of BAPG and the critical point set of the original problem (1). Upon (21), we have

$$\nabla f\left(\frac{\pi^\star+w^\star}{2}\right)+\frac{p+q}{2}=0,$$

where $\nabla f(\cdot)$ is a linear operator. By applying the Luo-Tseng local error bound condition of (1), i.e., Proposition 3.1, we have

$$\begin{aligned}\mathrm{dist}\left(\frac{\pi^\star+w^\star}{2},\mathcal{X}\right) &\le \tau\left\|\frac{\pi^\star+w^\star}{2}-\mathrm{proj}_{C_1\cap C_2}\left(\frac{\pi^\star+w^\star}{2}-\nabla f\left(\frac{\pi^\star+w^\star}{2}\right)\right)\right\|\\ &=\tau\left\|\frac{\pi^\star+w^\star}{2}-\mathrm{proj}_{C_1\cap C_2}\left(\frac{\pi^\star+p+w^\star+q}{2}\right)\right\|\\ &\overset{(\clubsuit)}{=}\frac{\tau}{2}\left\|\mathrm{proj}_{C_1}(\pi^\star+p)+\mathrm{proj}_{C_1}(w^\star+q)-2\,\mathrm{proj}_{C_1\cap C_2}\left(\frac{\pi^\star+p+w^\star+q}{2}\right)\right\|\\ &\overset{(\spadesuit)}{\le}\frac{M\tau}{2}\|\pi^\star-w^\star\|,\end{aligned}$$

where $(\clubsuit)$ holds due to the normal cone property, that is, $\mathrm{proj}_C(x+z)=x$ for any $x\in C$ and $z\in\mathcal{N}_C(x)$, and $(\spadesuit)$ follows from Lemma 3.2. Incorporating with (23), the approximation bound for BAPG has been characterized quantitatively, i.e.,

$$\mathrm{dist}\left(\frac{\pi^\star+w^\star}{2},\mathcal{X}\right)\le\frac{(2\kappa+1)L_fM}{2\sigma\rho}.$$

□

Proof of Proposition 3.5 — Sufficient decrease property of BAPG

Proof. We first observe from the optimality conditions of main updates, i.e.,

$$0 \in \nabla_\pi f(\pi^k, w^k) + \rho(\nabla h(\pi^{k+1}) - \nabla h(w^k)) + \partial g_1(\pi^{k+1}) \tag{24}$$

$$0 \in \nabla_w f(\pi^{k+1}, w^k) + \rho(\nabla h(w^{k+1}) - \nabla h(\pi^{k+1})) + \partial g_2(w^{k+1}) \tag{25}$$

where $g_1(\pi) = I_{\{\pi \in C_1\}}$ and $g_2(w) = I_{\{w \in C_2\}}$. Due to the convexity of $g_1(\cdot)$, it is natural to obtain,

$$\begin{aligned} g_1(\pi^k) - g_1(\pi^{k+1}) &\geq -\langle \nabla_\pi f(\pi^k, w^k) + \rho(\nabla h(\pi^{k+1}) - \nabla h(w^k)), \pi^k - \pi^{k+1} \rangle \\ &= -\langle \nabla_\pi f(\pi^k, w^k), \pi^k - \pi^{k+1} \rangle + \langle \rho(\nabla h(\pi^{k+1}) - \nabla h(w^k)), \pi^{k+1} - \pi^k \rangle \end{aligned}$$

As $f(\pi, w) = -\mathrm{tr}(D_X \pi D_Y w^T)$ is a bilinear function, we have,

$$f(\pi^k, w^k) - f(\pi^{k+1}, w^k) - \langle \nabla_\pi f(\pi^k, w^k), \pi^k - \pi^{k+1} \rangle = 0.$$

Consequently, we get

$$f(\pi^k, w^k) + g_1(\pi^k) - f(\pi^{k+1}, w^k) - g_1(\pi^{k+1}) \geq \rho \langle \nabla h(\pi^{k+1}) - \nabla h(w^k), \pi^{k+1} - \pi^k \rangle. \tag{26}$$

Similarly, based on the w-update, we obtain

$$f(\pi^{k+1}, w^k) + g_2(w^k) - f(\pi^{k+1}, w^{k+1}) - g_2(w^{k+1}) \geq \rho \langle \nabla h(w^{k+1}) - \nabla h(\pi^{k+1}), w^{k+1} - w^k \rangle. \tag{27}$$

Combine with (26) and (27), we obtain

$$\begin{aligned} & f(\pi^k, w^k) + g_1(\pi^k) + g_2(w^k) - f(\pi^{k+1}, w^{k+1}) - g_1(\pi^{k+1}) - g_2(w^{k+1}) \\ \geq\ & \rho \langle \nabla h(\pi^{k+1}) - \nabla h(w^k), \pi^{k+1} - \pi^k \rangle + \rho \langle \nabla h(w^{k+1}) - \nabla h(\pi^{k+1}), w^{k+1} - w^k \rangle. \end{aligned} \tag{28}$$

The right-hand side can be further simplified,

$$\begin{aligned} & \rho \langle \nabla h(\pi^{k+1}) - \nabla h(w^k), \pi^{k+1} - \pi^k \rangle + \rho \langle \nabla h(w^{k+1}) - \nabla h(\pi^{k+1}), w^{k+1} - w^k \rangle \\ \overset{(\clubsuit)}{=}\ & \rho(D_h(\pi^k, \pi^{k+1}) + D_h(\pi^{k+1}, w^k) - D_h(\pi^k, w^k)) + \rho(D_h(w^k, w^{k+1}) + \\ & D_h(w^{k+1}, \pi^{k+1}) - D_h(w^k, \pi^{k+1})) \\ \overset{(\spadesuit)}{=}\ & \rho D_h(\pi^k, \pi^{k+1}) + \rho D_h(w^k, w^{k+1}) - \rho D_h(\pi^k, w^k) + \rho D_h(\pi^{k+1}, w^{k+1}). \end{aligned}$$

Here, $(\clubsuit)$ uses the fact that the three-point property of Bregman divergence holds, i.e., For any $y, z \in \mathrm{int}\, \mathrm{dom}\, h$ and $x \in \mathrm{dom}\, h$,

$$D_h(x, z) - D_h(x, y) - D_h(y, z) = \langle \nabla h(y) - \nabla h(z), x - y \rangle.$$

Moreover, $(\spadesuit)$ holds as $D_h(\cdot, \cdot)$ is symmetric on the whole sequence. To proceed, this together with (28) implies

$$F_\rho(\pi^{k+1}, w^{k+1}) - F_\rho(\pi^k, w^k) \leq -\rho D_h(\pi^k, \pi^{k+1}) - \rho D_h(w^k, w^{k+1}). \tag{29}$$

Summing up (29) from $k = 0$ to $+\infty$, we obtain

$$F_\rho(\pi^\infty, w^\infty) - F_\rho(\pi^0, w^0) \leq -\kappa_1 \sum_{k=0}^{\infty} \left(D_h(\pi^k, \pi^{k+1}) + D_h(w^k, w^{k+1}) \right).$$

As the potential function $F_\rho(\cdot, \cdot)$ is coercive and $\{(\pi^k, w^k)\}_{k \geq 0}$ is a bounded sequence, it means the left-hand side is bounded, which implies

$$\sum_{k=0}^{\infty} \left(D_h(\pi^k, \pi^{k+1}) + D_h(w^k, w^{k+1}) \right) < +\infty.$$

□

Proof of Theorem 3.6 — Subsequence convergence result of BAPG

Proof. Let (π^∞, w^∞) be a limit point of the sequence $\{(\pi^k, w^k)\}_{k\geq 0}$. Then, there exists a sequence $\{n_k\}_{k\geq 0}$ such that $\{(\pi^{n_k}, w^{n_k})\}_{k\geq 0}$ converges to (π^∞, w^∞). Replacing k by n_k in (24) and (25), taking limits on both sides as $k \to \infty$

$$0 \in \nabla_\pi f(\pi^\infty, w^\infty) + \rho(\nabla h(\pi^\infty) - \nabla h(w^\infty)) + \partial g_1(\pi^\infty)$$
$$0 \in \nabla_w f(\pi^\infty, w^\infty) + \rho(\nabla h(w^\infty) - \nabla h(\pi^\infty)) + \partial g_2(w^\infty).$$

Based on the fact that $\nabla_\pi f(\pi^\infty, w^\infty) = \nabla f(w^\infty)$ and $\nabla_w f(\pi^\infty, w^\infty) = \nabla f(\pi^\infty)$, it can be easily concluded that $(\pi^\infty, w^\infty) \in \mathcal{X}_{\text{BAPG}}$. □

More importantly, the above subsequence convergence result can be extended to the global convergence if the Legendre function $h(\cdot)$ is quadratic.

Theorem C.5 (Global Convergence of BAPG — Quadratic Case)**.** *The sequence* $\{(\pi^k, w^k)\}_{k\geq 0}$ *converges to a critical point of* $F_\rho(\pi, w)$.

Proof. To invoke the Kurdyka-Lojasiewicz analysis framework developed in (Attouch et al., 2010; 2013), we have to establish two crucial properties — sufficient decrease and safeguard condition. The first one has already been proven in Proposition 3.5. Then, we would like to show that there exists a constant κ_2 such that,

$$\text{dist}(0, \partial F_\rho(\pi^{k+1}, w^{k+1})) \leq \kappa_2(\|\pi^{k+1} - \pi^k\| + \|w^{k+1} - w^k\|).$$

At first, noting that

$$\partial F_\rho(\pi^{k+1}, w^{k+1}) = \begin{bmatrix} \nabla_\pi f(\pi^{k+1}, w^{k+1}) + \rho(\pi^{k+1} - w^{k+1}) + \partial g_1(\pi^{k+1}) \\ \nabla_w f(\pi^{k+1}, w^{k+1}) + \rho(w^{k+1} - \pi^{k+1}) + \partial g_2(w^{k+1}) \end{bmatrix}.$$

Again, together with the updates (24) and (25), this implies

$$\begin{aligned} \text{dist}^2(0, \partial F_\rho(\pi^{k+1}, w^{k+1})) &\leq \|\nabla_\pi f(\pi^{k+1}, w^{k+1}) - \nabla_\pi f(\pi^k, w^k)\|^2 + \rho\|w^k - w^{k+1}\|^2 \\ &\quad + \|\nabla_w f(\pi^{k+1}, w^k) - \nabla_w f(\pi^{k+1}, w^{k+1})\|^2 \\ &\leq (\sigma_{\max}^2(D_X)\sigma_{\max}^2(D_Y) + \rho)\|w^k - w^{k+1}\|^2, \end{aligned}$$

where $\sigma_{\max}(\cdot)$ denotes the maximum singular value. By letting $\kappa_2 = \sqrt{\sigma_{\max}^2(D_X)\sigma_{\max}^2(D_Y) + \rho}$, we get the desired result. □

D Additional Experiment Details

Table 6: Statistics of databases for graph alignment.

Dataset	# Samples	Ave. Nodes	Ave. Edges
Synthetic	300	1500	56579
Proteins	1113	39.06	72.82
Enzymes	600	32.63	62.14
Reddit	500	375.9	449.3

Source codes of all baselines used in this paper and their licenses

- BPG-S (GPL-3.0 license) (Xu et al., 2019b): https://github.com/HongtengXu/gwl
- ScalaGW (No license) (Xu et al., 2019a): https://github.com/HongtengXu/s-gwl
- SpecGW (MIT license) (Chowdhury & Needham, 2021): https://github.com/trneedham/Spectral-Gromov-Wasserstein
- eBPG (MIT license) (Flamary et al., 2021): https://github.com/PythonOT/POT
- FW (MIT license) (Flamary et al., 2021) https://github.com/PythonOT/POT

Table 7: Graph alignment results (Mean ± Std.) of compared methods in 5 independent trials over different random seeds in the noise generating process.

Method	Synthetic	Proteins-Noisy	Enzymes-Noisy	Reddit-Noisy
FW	24.50±0.95	20.24±0.38	22.80±1.15	18.34±0.22
ScalaGW	17.93±0.27	16.05±0.06	11.46±0.22	0.70±0.05
SpecGW	13.27±0.49	19.31±0.70	21.14±0.45	19.66±0.16
eBPG	34.33±0.61	45.85±0.73	60.46±0.57	3.34±0.03
BPG	57.56±0.45	52.46±0.25	62.32±0.51	36.68±0.02
BPG-S	61.48±0.34	52.74±0.13	62.21±0.55	36.68±0.02
BAPG	**99.79±0.21**	**57.16±0.23**	**62.85±0.48**	**49.45±0.02**