Adaptive Softassign via Hadamard-Equipped Sinkhorn

Binrui Shen
Department of Applied Mathematics, School of Mathematics and Physics,
Xi'an Jiaotong-Liverpool University, Suzhou 215123, P.R. China
Department of Mathematical Sciences, School of Physical Sciences,
University of Liverpool, Liverpool, United Kingdom
binrui.shen19@student.xjtlu.edu.cn

Qiang Niu
Department of Applied Mathematics, School of Mathematics and Physics,
Xi'an Jiaotong-Liverpool University, Suzhou 215123, P.R. China
qiang.niu@xjtlu.edu.cn

Shengxin Zhu†
Research Centers for Mathematics, Advanced Institute of Natural Science,
Beijing Normal University, Zhuhai 519087, P.R.China
Guangdong Provincial Key Laboratory of Interdisciplinary Research and Application for Data Science,
BNU-HKBU United International College, Zhuhai 519087, P.R. China
Shengxin.Zhu@bnu.edu.cn

Abstract

Softassign is a pivotal method in graph matching and other learning tasks. Many softassign-based algorithms exhibit performance sensitivity to a parameter in the softassign. However, tuning the parameter is challenging and almost done empirically. This paper proposes an adaptive softassign method for graph matching by analyzing the relationship between the objective score and the parameter. This method can automatically tune the parameter based on a given error bound to guarantee accuracy. The Hadamard-Equipped Sinkhorn formulas introduced in this study significantly enhance the efficiency and stability of the adaptive softassign. Moreover, these formulas can also be used in optimal transport problems. The resulting adaptive softassign graph matching algorithm enjoys significantly higher accuracy than previous state-of-the-art large graph matching algorithms while maintaining comparable efficiency.

1. Introduction

Graph matching aims to find a correspondence between two graphs. As a fundamental problem in computer vision and pattern recognition, it is widely used in shape matching [28, 39], detection of similar pictures[31], medical imaging [9], graph similarity computation [15, 16] and face authentication [13, 38]. It can even be used in activity analysis [2] and recently in bioinformatics [41].

The general graph matching is an NP-hard problem, because of its combinatorial nature [17]. Therefore, recent works on graph matching mainly focus on continuous relaxation to obtain a sub-optimal solution with an acceptable cost by constructing approximate optimization methods. Popular approaches include, but are not limited to, spectral-based methods [18, 25, 29, 35], continuous path optimization [27, 37, 43], random walk [3] and probabilistic modeling [7] and optimal transport methods [41, 42].

Among recently proposed graph matching algorithms, projected gradient-based algorithms [5, 8, 19, 23, 32] have drawn a lot of attention due to their competitive performances in large graph matching problems. These algorithms iteratively update the solution by projecting gradient matrices into a feasible region, typically addressing a *linear assignment problem*. The performance of these algorithms mainly depends on the underlying projection methods. Among projections, the discrete projection may lead the matching algorithm [8] to converge to a circular sequence [34]; the doubly stochastic projection used in [23] suffers from poor convergence when the numerical values of the input matrix are large [30]. Softassign is a more flex-

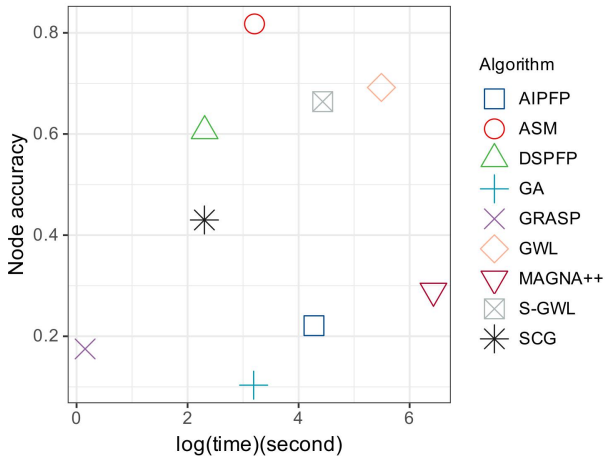

Figure 1. Mean matching accuracy and running time of different algorithms on protein network matching (25% noise level).

Table 1. Symbols and Notations.

Symbol	Definition
$\mathbb{G}, \tilde{\mathbb{G}}$	matching graphs
$A, \tilde{A}$	edge attribute matrices of $\mathbb{G}$ and $\tilde{\mathbb{G}}$
$F, \tilde{F}$	node attribute matrices of $\mathbb{G}$ and $\tilde{\mathbb{G}}$
$n, \tilde{n}$	number of nodes of $\mathbb{G}$ and $\tilde{\mathbb{G}}$
M	matching matrix
$\Pi_{n\times n}$	set of $n \times n$ permutation matrices
$\Sigma_{n\times n}$	set of $n \times n$ doubly stochastic matrices
$\mathbf{1}, \mathbf{0}$	a column vector of all 1s,0s
$D_{(\mathbf{x})}$	diagonal matrix of a vector $\mathbf{x}$
$\mathrm{tr}(\cdot)$	trace
$\langle \cdot, \cdot \rangle$	inner product
$\|\cdot\|_{Fro}$	Frobenius norm
exp	element-wise exponential
$\oslash$	element-wise division
$\circ$	Hadamard product
$.^{\circ}$	Hadamard power
β	the parameter in softassign
$\mathcal{P}_{sk}(\cdot)$	Sinkhorn method
S_X^{β}	a matrix from applying softassign with β on a matrix X

ible method that allows for a trade-off between efficiency and accuracy. It is proposed to solve linear assignment problems in [12] and is later used as an approximate projection method in graph matching [8]. It consists of an exponential operator and the Sinkhorn method [33] to achieve inflation and bistochastic normalization, respectively. The inflation step can effectively attenuate unreliable correspondences while simultaneously amplifying reliable ones [3].

The performance of the softassign-based graph matching algorithms depends largely on the inflation parameter in the inflation step [32]. Previous algorithms tune this parameter empirically [3, 8, 32, 44]. To address such an inconvenience and improve accuracy, this paper proposes an adaptive softassign method. The main contributions of this paper are summarized as follows:

- **Adaptive softassign.** We propose an adaptive softassign method for large graph matching problems. It is designed to automatically tune the parameter according to a given error bound, which can be interpreted as the distance from optimal performance.
- **Sinkhorn operation rules**. Several introduced convenient operation rules for the Sinkhorn method significantly accelerate the adaptive softassign and increase the stability in Sinkhorn iterations. Furthermore, all theoretical results regarding softassign can be readily extended to the optimal transport problems [6].
- **Graph matching algorithm.** By combining the adaptive softassign method with a project fixed-point approach, we propose a novel adaptive softassign matching algorithm (ASM). It enjoys significantly higher accuracy than previous state-of-the-art large matching algorithms. See Figure 1 for comparison.

The structure of this paper is as follows. Section 2 introduces the graph matching problem, the projected fixed-point method, and softassign. Section 3 showcases adaptive softassign, Sinkhorn formulas, and the potential impact of Sinkhorn formulas on the optimal transport problem. Section 4 discusses algorithmic details of adaptive softassign matching. An empirical study is conducted in Section 5 before concluding. Theoretical proofs are shown in the Appendix.

2. Preliminaries

Table 1 summarizes the main symbols and notations used in this paper. We use lowercase letters for scalars (e.g., β), bold lowercase letters for vectors (e.g., $\mathbf{1}$), and uppercase letters for matrices (e.g., A).

2.1. Background

A graph $\mathbb{G} = \{\mathbb{V}, \mathbb{E}, A, F\}$ consists of a node set $\mathbb{V}$ and an edge set $\mathbb{E}$. Further, we can use a symmetric matrix A to denote the attributes of edges and F to store the attributes of each node.

Matching matrix The matching correspondence of two graphs with the same number of nodes is usually represented by a *permutation matrix* $M = (M_{ij})$

$$M_{ij} = \begin{cases} 1 & \text{if } \mathbb{V}_i \text{ corresponds to } \tilde{\mathbb{V}}_j, \\ 0 & \text{otherwise,} \end{cases} \tag{1}$$

where $M \in \Pi_{n\times n}$.

The graph matching problem can be formulated as a quadratic assignment problem minimizing the dissimilarity of two graphs [43]:

$$\min_{M\in\Pi_{n\times n}} \frac{1}{4}\left\|A - M\tilde{A}M^T\right\|_{Fro}^2 + \lambda\left\|F - M\tilde{F}\right\|_{Fro}^2, \tag{2}$$

where the left term presents the dissimilarity between edges and the right term presents the dissimilarity between nodes. Since $\|X\|_{Fro}^2 = \operatorname{tr}\left(XX^T\right)$, problem (2) can be rewritten as

$$\max_{M\in\Pi_{n\times n}} \quad \frac{1}{2}\operatorname{tr}\left(M^T A M \widetilde{A}\right) + \lambda \operatorname{tr}\left(M^T K\right), \quad (3)$$

where $K = F\tilde{F}^T$, see [23] for more details.

Relaxation method Due to the discrete constraints, (3) is an NP-hard problem [17]. A common trick for solving such discrete problems is relaxation: one first finds a solution X on a continuous domain $\Sigma_{n\times n}$,

$$N^* = \arg\max_{N\in\Sigma_{n\times n}} \frac{1}{2}\operatorname{tr}\left(N^T A N \widetilde{A}\right) + \lambda \operatorname{tr}\left(N^T K\right), \quad (4)$$

and N^* is transformed back to the original discrete domain $\Pi_{n\times n}$ by solving a *linear assignment problem* of the following form

$$M^* = \arg\min_{M\in\Pi_{n\times n}} \|M - N^*\|_{Fro}. \quad (5)$$

The matrix M^* is the final solution for graph matching, which is commonly obtained by the Hungarian method [14] or the greedy method (efficient but not exact) [25].

2.2. Adaptive projected fixed-point method

Consider the objective function

$$\mathcal{Z}(M) = \frac{1}{2}\operatorname{tr}\left(M^T A M \widetilde{A}\right) + \lambda \operatorname{tr}\left(M^T K\right). \quad (6)$$

With the help of matrix differential [10], one can obtain the 'gradient' of the objective function with respect to M

$$\nabla\mathcal{Z}(M) = \frac{\partial \mathcal{Z}(M)}{\partial M} = A M \widetilde{A} + \lambda K. \quad (7)$$

The adaptive projected fixed-point method is

$$\begin{aligned} M^{(k)} &= (1-\alpha)M^{(k-1)} + \alpha D^{(k)}, \\ D^{(k)} &= \mathcal{P}(\nabla\mathcal{Z}(M^{(k-1)})), \end{aligned} \quad (8)$$

where $\alpha \in [0,1]$ is a step size parameter and $\mathcal{P}(\cdot)$ is a projection operator used to project the gradient matrix to a feasible region.

An adaptive strategy of the step size parameter proposed in [32] can guarantee the convergence of (8) with any projection type. The optimal step size parameter α^* is determined according to a 'linear search' type technique:

$$(\alpha^*)^{(k)} = \arg\max_{\alpha} \ \mathcal{Z}((1-\alpha)M^{(k-1)} + \alpha D^{(k)}). \quad (9)$$

According to underlying constraints, projections include the discrete projection used in the integer projected fixed-point method [19] and the doubly stochastic projection used in the doubly stochastic projected fixed-point method (DSPFP) [23]. The discrete projection solves problem (5) and the doubly stochastic projection aims to find the closet doubly stochastic matrix to a given matrix X by solving

$$Y^* = \arg\min_{Y\in\Sigma_{n\times n}} \|X - Y\|_{Fro}, \quad (10)$$

which equals

$$Y^* = \arg\max_{Y\in\Sigma_{n\times n}} \langle X, Y\rangle. \quad (11)$$

In essence, the projected fixed point method solves a series of linear assignment problems to approximate the solution of problem (4). The performance of algorithms depends on the quality of solutions to linear problems (projections).

2.3. Softassign

Among projection methods, the discrete projection suffers from information loss when the linear assignment problem with discrete constraints has multiple solutions; the doubly stochastic projection suffers from poor convergence when the numerical value of the input matrix is large [30]. To address these issues, an entropic regularization term is added to smooth the problem (11):

$$\begin{aligned} S_X^\beta &= \arg\max_{S\in\Sigma_{n\times n}} \langle S, X\rangle + \frac{1}{\beta}\mathcal{H}(S), \\ \mathcal{H}(S) &= -\sum S_{ij}\ln S_{ij}, \end{aligned} \quad (12)$$

where $X = \nabla\mathcal{Z}(M^{(k)})$ in the projected fixed-point method for graph matching. As the inflation parameter β increases, S_X^β approaches the optimal solution of the linear assignment problem (11).

Softassign solves (12) to approximate the solution of (11) [12]. It has been widely used in graph matching [3, 8, 31]; its general form has been widely used in optimal transport [6]. The solution S_X^β is unique of form [6]

$$(S_X^\beta)_{ij} = \mathbf{r}_i J_{ij} \mathbf{c}_j, \quad J = \exp(\beta X), \quad \mathbf{r}, \mathbf{c} \in \mathbb{R}_+^n. \quad (13)$$

In matrix form, the solution reads as

$$S_X^\beta = D_{(\mathbf{r})} J D_{(\mathbf{c})}. \quad (14)$$

To improve numerical stability, we perform a preprocessing on J according to [32]:

$$\hat{J} = \exp(\beta(X/\max(X))), \quad (15)$$

where $\max(X)$ is the maximum element of X. The two balancing vectors $\mathbf{r}$ and $\mathbf{c}$ can be computed by Sinkhorn iterations

$$\mathbf{r}^{(\ell+1)} = \mathbf{1} \oslash \hat{J}\mathbf{c}^{(\ell)} \quad \text{and} \quad \mathbf{c}^{(\ell+1)} = \mathbf{1} \oslash \hat{J}^{\mathrm{T}}\mathbf{r}^{(\ell+1)}. \quad (16)$$

To summarize, the softassign algorithm consists of two components: inflation by matrix element-wise exponential in (15) and doubly stochastic normalization by the Sinkhorn method in (16). The inflation step magnifies large values and diminishes small ones to reduce the effect of unreliable correspondences. Figure 2 illustrates the effect of the β.

$$X = \begin{pmatrix} 1 & 0.9 & 0.9 \\ 0.9 & 1 & 0.5 \\ 0.6 & 0.25 & 1 \end{pmatrix}$$

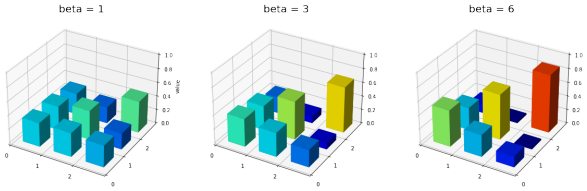

Figure 2. The heights of histograms represent values of corresponding elements in S_X^β. As β increases, S_X^β gradually converges towards the solution of the assignment problem, namely, the identity matrix.

3. Adaptive softassign

This section introduces an adaptive softassign algorithm and some nice Sinkhorn operation rules.

3.1. Adaptive softassign

The performance of the softassign depends on the parameter β: a larger β leads to a better result but more Sinkhorn iterations [6]. Theoretically, S_X^∞ is the optimal solution for the problem (11) [4], while the corresponding time cost is exorbitantly high. Therefore, we aim to design an adaptive softassign that can automatically select a moderately sized β while still yielding promising results for various large graph matching problems. Inspired by the analysis of optimal transport problems [24], we analyze the relation between β and optimal score to provide feasibility for the aim.

Proposition 1 *For a square matrix X and $\beta > 0$, we have*

$$\begin{aligned} |\langle S_X^\beta, X\rangle - \langle S_X^\infty, X\rangle| &\le \|S_X^\beta - S_X^\infty\|\|X\| \\ \|S_X^\beta - S_X^\infty\| &\le \frac{c}{\mu}(e^{(-\mu\beta)}), \end{aligned} \tag{17}$$

where c and $\mu > 0$ are constants independent of β.

Proposition 1 illustrates an exponential decay of $|\langle S_X^\beta, X\rangle - \langle S_X^\infty, X\rangle|$ with respect to β. This Proposition supports that a moderately sized β can yield favorable outcomes. Such a β can be determined by setting a threshold of $\|S_X^\beta - S_X^\infty\|$, which is a trade-off between accuracy and efficiency. However, S_X^∞ is unknown, so we consider utilizing $\|S_X^\beta - S_X^{\beta+\Delta\beta}\|$ to determine β. Then we analyze the convergence of $\|S_X^\beta - S_X^{\beta+\Delta\beta}\|$.

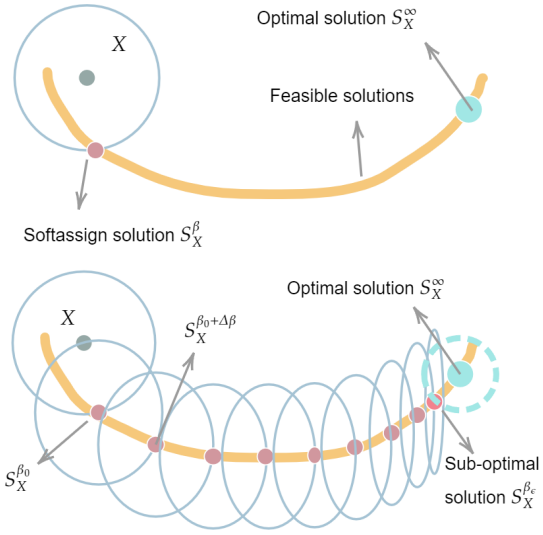

Figure 3. Softassign and adaptive softassign process.

Proposition 2 *For a square matrix X and β, $\Delta\beta > 0$, we have*

$$\|S_X^\beta - S_X^{\beta+\Delta\beta}\| \le (1 - e^{(-\mu\Delta\beta)})\frac{c}{\mu}e^{(-\mu\beta)}, \tag{18}$$

where c and $\mu > 0$ are constants independent of β.

Proposition 2 indicates that $\|S_X^\beta - S_X^{\beta+\Delta\beta}\|$ and $\|S_X^\beta - S_X^\infty\|$ decay at similar order as β increases. This allows us to use $\beta + \Delta\beta$ instead of ∞ to choose a suboptimal β_ϵ:

$$\beta_\epsilon = \arg\min_\beta \beta, \quad s.t.\ \|S_X^\beta - S_X^{\beta+\Delta\beta}\| \le \epsilon. \tag{19}$$

The pseudocode for adaptive softassign appears in the Algorithm 1.

On choosing of $\Delta\beta$ Altschuler et al. [1] and Shen et al. [32] demonstrate that softassign can be robust to the nodes' cardinality n by setting $\beta = \gamma\ln(n)$, where γ is a constant related with the type of matching graphs. Enlightened by this, we also set $\Delta\beta = \ln(n)$ so that the adaptive softassign is robust to n.

On choosing of β_0 Empirical evidence suggests that the computational time required for adaptive softassign positively correlates with $|\beta_\epsilon - \beta_0|$. Therefore, choosing a β_0, close to β_ϵ, can enhance the algorithm's efficiency. The choice of β_0 for graph matching is discussed in detail in the subsequent section, as β_ϵ varies across different problems.

Error analysis Since the adaptive softassign has the same accuracy as softassign with β_ϵ, the performance of adaptive softassign is guaranteed by [32, Proposition 2]:

$$\frac{1}{n}|\langle S_X^{\beta_\epsilon}, X\rangle - \langle S_X^{\infty}, X\rangle| \leq \frac{\ln(n)}{\beta_\epsilon} = \frac{1}{\gamma_\epsilon}, \tag{20}$$

where the left term, an *average assignment error*, quantifying the distance between $S_X^{\beta_\epsilon}$ and the optimal solution S_X^{∞}.

Algorithm 1 Adaptive softassign

Require: X, β_0, ϵ
1: Compute $S_X^{\beta_0}$ by softassign in (15) and (16)
2: **for** $k = 0, 1, 2 \ldots,$ until $r < \epsilon$ **do**
3: $\beta_k = \beta_{k-1} + \Delta\beta$
4: Compute $S_X^{\beta_k}$ (Accelerated by Alg. 2)
5: $r = \|S_X^{\beta_k} - S_X^{\beta_{k-1}}\|_1$
6: **end for**
7: **Return** $S_X^{\beta_k}$, β_k

Algorithm 2 Softassign Transition

Require: $S^{\beta_{k-1}}, \beta_{k-1}, \beta_k$
1: $\hat{S} = (S^{\beta_{k-1}})^{\circ(\frac{\beta_k}{\beta_{k-1}})}$
2: **for** $\ell = 0, 1, 2 \ldots,$ until convergence **do**
3: $\mathbf{r}^{(\ell+1)} = \mathbf{1} \oslash \hat{S}\mathbf{c}^{(\ell)}$
4: $\mathbf{c}^{(\ell+1)} = \mathbf{1} \oslash \hat{S}^T\mathbf{r}^{(\ell+1)}$
5: **end for**
6: **Return** $S^{\beta_k} = D_{(\mathbf{r})}\hat{S}D_{(\mathbf{c})}$

3.2. Softassign Transition

Since the adaptive softassign inevitably compute $S_X^{\beta+\Delta\beta}$ for different β repeatedly, we propose a delicate strategy to compute $S_X^{\beta+\Delta\beta}$ from S_X^{β} instead of $X \in \mathbb{R}^{n\times n}$. This recursive computation is much easier than direct computation. The process is shown in Figure 3.

To achieve the recursive computation, we first propose some nice Sinkhorn formulas. For convenience, we use $\mathcal{P}_{sk}(X)$ to represent $Sinkhorn(X) = D_{(\mathbf{r})}XD_{(\mathbf{c})}$ where $\mathbf{r}$ and $\mathbf{c} \in \mathbb{R}_+^n$ are balancing vectors resulting from (16).

Proposition 3 *Hadamard-Equipped Sinkhorn*
Let $X \in \mathbb{R}_+^{n\times n}$, then

$$\mathcal{P}_{sk}(X) = X \circ SK^{(X)} = X \circ (\mathbf{r}^T \otimes \mathbf{c}) \tag{21}$$

where $SK^{(X)} \in \mathbb{R}^{n\times n}$ is unique, $\mathbf{r}$ and $\mathbf{c} \in \mathbb{R}_+^n$ are balancing vectors so that $D_{(\mathbf{r})}XD_{(\mathbf{c})}$ is doubly stochastic.

This Proposition builds a bridge between the Hadamard product and the Sinkhorn method. The connection yields some convenient Sinkhorn operation rules.

Lemma 1 *Let $X \in \mathbb{R}_+^{n\times n}$, $\mathbf{u}$ and $\mathbf{v} \in \mathbb{R}_+^n$, then*

$$\mathcal{P}_{sk}(X) = \mathcal{P}_{sk}(X \circ (\mathbf{u}^T \otimes \mathbf{v})). \tag{22}$$

Lemma 2 *Sinkhorn-Hadamard product*
Let $X_1, X_2 \in \mathbb{R}_+^{n\times n}$, then

$$\mathcal{P}_{sk}(X_1 \circ X_2) = \mathcal{P}_{sk}(\mathcal{P}_{sk}(X_1) \circ X_2). \tag{23}$$

Lemma 3 *Sinkhorn-Hadamard power*
Let $X \in \mathbb{R}_+^{n\times n}$, then

$$\mathcal{P}_{sk}(X^{\circ(ab)}) = \mathcal{P}_{sk}(\mathcal{P}_{sk}(X^{\circ a})^{\circ b}), \tag{24}$$

where a and b are two constants not equal to zero.

According to the Lemma 1 and Lemma 2, we have

Theorem *Softassign Transition*
Let $X \in \mathbb{R}_+^{n\times n}$, then

$$S_X^{\beta_2} = \mathcal{P}_{sk}((S_X^{\beta_1})^{\circ(\frac{\beta_2}{\beta_1})}), \ where\ \beta_1, \beta_2 > 0. \tag{25}$$

The softassign transition enables us to compute $S_X^{\beta+\Delta\beta}$ from S_X^{β}, which significantly reduces the computational cost. The strategy is detailed in Algorithm 2. Its performance is displayed in Figure 4. When the matrix size is 2000, the speedup ratio of the strategy is 6.7x.

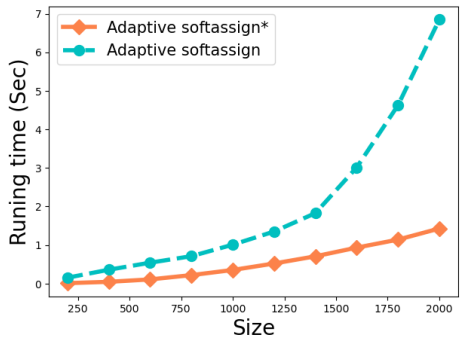

Figure 4. The orange solid line represents the performance of adaptive softassign; the blue dashed line represents the performance of adaptive softassign* (adaptive softassign with the softassign transition). These two methods are evaluated on random matrices over 20 runs.

3.3. Stability

For a large β, the computation of softassign may cause numerical instability. The instability includes (1) overflow: the elements of J in (13) are too large to handle, and (2) underflow: a row/column sum of $\hat{J}$ approaches to 0 in (15), then a denominator of zero occurs in the Sinkhorn process (16) [40]. Adaptive softassign can significantly reduce such a risk since it calculates the $S_X^{\beta_\epsilon}$ by $S_X^{\beta_0}$ and a series of softassign transitions. It is akin to dividing a vast distance into smaller segments, thereby enabling one to traverse the distance step by step.

Example *How adaptive softassign avoids instability when finding* S_X^8 *for*

$$X = \begin{pmatrix} -99 & -100 \\ -100 & -99 \end{pmatrix}. \tag{26}$$

Calculating the S_X^8 *directly will cause instability:* $\exp(-99 \times 8)$ *and* $\exp(-100 \times 8)$ *are smaller than the smallest number that a program can handle, so the program rounds down* $\exp(8X)$ *to a zero matrix.*

A stable choice is computing it by a two-step computation:

$$S_X^8 = \mathcal{P}_{sk}((S_X^2)^{\circ 4}) \text{ or } \mathcal{P}_{sk}((S_X^4)^{\circ 2}). \tag{27}$$

We show the results as follows:

$$S_X^4 = \begin{pmatrix} 0.98 & 0.02 \\ 0.02 & 0.98 \end{pmatrix}, \mathcal{P}_{sk}((S_X^4)^{\circ 2}) = \begin{pmatrix} 0.997 & 0.003 \\ 0.003 & 0.997 \end{pmatrix}$$

$$S_X^2 = \begin{pmatrix} 0.88 & 0.12 \\ 0.12 & 0.88 \end{pmatrix}, \mathcal{P}_{sk}((S_X^2)^{\circ 4}) = \begin{pmatrix} 0.997 & 0.003 \\ 0.003 & 0.997 \end{pmatrix}$$

The results also validate the softassign transition that $\mathcal{P}_{sk}((S_X^2)^{\circ 4}) = \mathcal{P}_{sk}((S_X^4)^{\circ 2})$. *The risk of overflow can also be addressed by this method.*

3.4. Connection with the optimal transport problem

The Sinkhorn formulas, introduced in Section 3.2, are closely related to the optimal transport problem. Cuturi [6] formulates the regularized optimal transport problem as

$$\mathrm{T}_C^{\beta}(\mathbf{a}, \mathbf{b}) = \arg\min_{T \in \mathcal{U}(\mathbf{a}, \mathbf{b})} \langle T, C \rangle - \frac{1}{\beta} \mathcal{H}(T), \tag{28}$$

where $\mathcal{U}_{(\mathbf{a},\mathbf{b})} := \{T \in \mathbb{R}_+^{n \times n} : T\mathbf{1} = \mathbf{a}, T^T\mathbf{1} = \mathbf{b}\}$, $C \in \mathbb{R}_+^{n \times n}$ is a given cost matrix, and $\mathbf{a}$, $\mathbf{b} \in \mathbb{R}_+^n$ are given vectors with positive entries with the sum being one. The regularized linear assignment problem (12) is a special case of the regularized optimal transport problem where $\mathbf{a}$ and $\mathbf{b}$ are vectors of ones. The solution of (28) has the form

$$\mathrm{T}_C^{\beta}(\mathbf{a}, \mathbf{b}) = D_{(\mathbf{u})} \exp(-\beta \mathbf{C}) D_{(\mathbf{v})}, \tag{29}$$

where $\mathbf{v}$ and $\mathbf{u}$ can be computed by the Sinkhorn iteration. The form of (29) is very similar to the solution of the regularized assignment problem in (14). According to Proposition 3, we have $\mathrm{T}_C^{\beta}(\mathbf{a}, \mathbf{b}) = \exp(-\beta C) \circ (\mathbf{u} \otimes \mathbf{v}^T)$, and the $\mathbf{u}$ and $\mathbf{v}$ are unique [6]. This property makes it easy to prove Lemma 1, Lemma 2, Lemma 3, and the transition theorem for optimal transport problems. Such theoretical results will provide more flexibility for computation and shed light on optimal transport problems. For instance, Liao et al. [21, 22] enhance the Sinkhorn method in special optimal transport problems by leveraging Hadamard operations (which differs from our Sinkhorn formulas). Another interesting finding based on the Sinkhorn formulas is that adaptive softassign is a variant of the proximal point method for optimal transport problems (described in the Appendix).

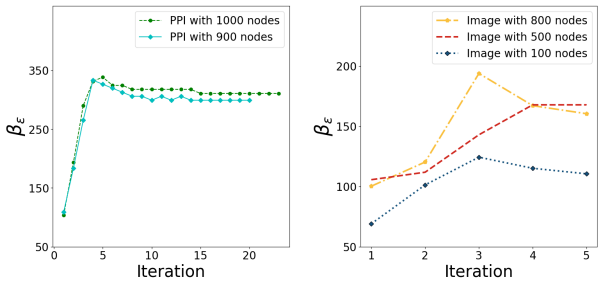

Figure 5. The change of β_ϵ in ASM when β_0 is $\ln n$ in adaptive softassign. PPI and image are two kinds of graph matching tasks introduced in experiments.

4. The adaptive softassign matching algorithm

The adaptive softassign matching algorithm[1] is shown in Algorithm 3. In step 1, a uniform initialization approach is adopted when no prior information is available. For the problem of matching graphs with different numbers of nodes (we assume that $\tilde{n} \leq n$), Gold and Rangarajan [8] introduce a square *slack matrix* like $\hat{D}$ in step 5: $\hat{D}_{(1:n,1:\tilde{n})} = AN\tilde{A} + \lambda K$ and rest elements of $\hat{D}$ are zero. Discussion in [23] indicates that matching quality is not sensitive to the parameter λ, and we set $\lambda = 1$ follows [23]. In step 7, we utilize $\beta_\epsilon^{(k)} - \Delta\beta$ as the $\beta_0^{(k+1)}$ to reduce the computational cost of the adaptive softassign in the next iterate: such a $\beta_0^{(k+1)}$ is close to $\beta_\epsilon^{(k+1)}$, since β_ϵ typically increases in early iterations of the algorithm before leveling off at a certain point with minor fluctuations (see Figure 5). It should be noted that $\beta_\epsilon^{(k)} \geq \beta_0^{(k)} + \Delta\beta$ according to Algorithm 1, which indicates that $\{\beta^{(k)}\}$ will inevitably be an increasing series if $\beta_0^{(k+1)} = \beta_\epsilon^{(k)}$. The discretization in Step 10 is completed by the Hungarian method [14].

Regardless of fast and sparse matrix computation, step 4 and step 5 entail $O(n^3)$ operations per iteration. In the matching process, β_ϵ in the adaptive softassign (step 6) includes an increasing and stable state, shown in Figure 5. In stable state, the cost of adaptive softassign is close to that of softassign with β_ϵ^m, where β_ϵ^m is the maximum of β_ϵ in the matching process. In the increasing state, the cost of adaptive softassign is less than that of softassign with β_ϵ^m. Therefore, the average cost of adaptive softassign in a matching process is close to the softassign with β_ϵ^m: $O(n^2 \beta_\epsilon^m \|X\|_\infty)$ where the maximum of X is 1 [26]. Step 8 requires $O(n^2)$ operations per iteration. The Hungarian algorithm completes the final discretization step [14] with complexities of $O(n^3)$. Thus, the algorithm has time complexity $O(n^3) + O(n^2 \beta_\epsilon^m \|X\|_\infty)$ per iteration and space complexity $O(n^2)$.

[1]Our codes are available at https://github.com/BinruiShen/Adaptive-Softassign-Matching.

Algorithm 3 Adaptive softassign matching (ASM)

Require: $A, \tilde{A}, K, \lambda$
Ensure: M
Initial $\Delta\beta = \ln n, N^{(0)} = (\frac{\mathbf{1}}{\mathbf{n}})_{n\times\tilde{n}}, \hat{D}^{(0)} = \mathbf{0}_{n\times n}$
$\beta_0^{(1)} = \Delta\beta$
for $k = 1, 2\ldots,$ until N converge **do**
  Compute optimal α by (9)
  $\hat{D}^{(k)}_{(1:n,1:\tilde{n})} = AN^{(k-1)}\tilde{A} + \lambda K$
  $[D^{(k)}, \beta_\epsilon^{(k)}] =$ Adaptive softassign$(\hat{D}^{(k)}, \beta_0^{(k)})$
  $\beta_0^{(k+1)} = \beta_\epsilon^{(k)} - \Delta\beta$
  $N^{(k)} = (1-\alpha)N^{(k-1)} + \alpha D^{(k)}_{(1:n,1:\tilde{n})}$
end for
Discretize N to M
return M

5. Experiments

Baselines We compare ASM against the following baselines: DSPFP [23], GA [8], AIPFP [19, 23], SCG [32], GWL[2] [42] , S-GWL[3] [41], MAGNA++[4] [36], and GRASP[5] [11].

Benchmarks We perform algorithms in three benchmarks: the protein-protein interaction network (PPI), Facebook social networks, and real images. Unweighted graphs represent the first two networks. Weighted graphs with attributed nodes are extracted from real images.

Evaluations The evaluation in PPI and the social network is node accuracy $\frac{n_c}{n}$ where n_c represents the number of correct matching nodes. Since the ground truth of matching on real images is unknown, we evaluate the algorithms by matching error

$$\frac{1}{4}\left\|A - M\tilde{A}M^T\right\|_{Fro}^2 + \left\|F - M\tilde{F}\right\|_{Fro}^2. \tag{30}$$

The first four baselines can adapt the (30) as the objective function. Other algorithms are not compared in real image experiments since they are not designed to solve matching problems with attributed nodes.

5.1. Protein network and Social network

The yeast's protein-protein interaction (PPI) networks contains 1,004 proteins and 4,920 high-confidence interactions[4]. The social network comprising 'circles' (or 'friends lists') from Facebook [20] contains 4039 users (nodes) and 88234 relations (edges). Following the experimental protocol of [41], we compare different methods on matching networks with 5%, 15% and 25% noisy versions. Table 2 and Table 3 list the performance of various methods. The ASM consistently attains the highest accuracy across all scenarios, demonstrating its robustness. Notably, it yields an approximate 20% enhancement in accuracy amidst a 25% noise level, further accentuating its efficacy. Compared to the suboptimal algorithm GWL, ASM showcases an efficiency improvement of approximately tenfold.

Table 2. Comparisons on yeast PPI

Yeast network	5% noise		15% noise		25% noise	
Methods	Node Acc	time	Node Acc	time	Node Acc	time
MAGNA++ [36]	48.3%	603.3s	25.0%	630.6s	13.6%	624.2s
S-GWL [41]	81.3%	82.3s	62.4%	82.1s	55.5%	88.4s
GWL[42]	83.7%	226.4s	66.3%	254.7s	57.6%	246.5s
DSPFP [23]	78.1%	10.2s	60.8%	10.14s	42.9%	9.8s
GA [8]	14.0%	24.4s	9.6%	24.5s	7.4%	24.0s
GRASP [11]	38.6%	**1.1s**	8.3%	**1.2s**	5.6%	**1.2s**
SCG [32]	73.1%	10.7s	53.1%	10.3s	43.0%	10.0s
AIPFP [19, 23]	43.1%	105.4s	27.1%	75.2s	22.1%	73.8s
ASM	**89.0%**	28.7s	**81.2%**	22.7s	**75.1%**	22.6s

Table 3. Comparisons on Facebook network

Social network	5% noise		15% noise		25% noise	
Methods	Node Acc	time	Node Acc	time	Node Acc	time
S-GWL [41]	26.4%	1204.1s	18.3%	1268.2s	17.9%	1295.8s
GWL[42]	78.1%	3721.6s	68.4%	4271.3s	60.8%	4453.9s
DSPFP [23]	79.7%	151.3s	68.3%	154.2s	62.2%	156.9s
GA [8]	35.5%	793.2s	21.4%	761.7s	16.0%	832.6s
GRASP[11]	37.9%	**63.6s**	20.3%	**67.4s**	15.7%	**71.3s**
SCG [32]	58.2%	211.7s	43.1%	221.3s	43.1%	211.0s
AIPFP [19, 23]	68.6%	2705.5s	55.1%	2552.7s	47.8%	2513.8s
ASM	**91.1%**	387.2s	**88.4%**	391.7s	**85.7%**	393.1s

5.2. Real images

In this set of experiments, we construct attributed weighted graphs from a public dataset[6], which covers five common picture transformations: viewpoint changes, scale changes, image blur, JPEG compression, and illumination.

Following the experimental protocol of [23], the construction includes extraction of nodes, selection of nodes, and calculation of edge weight. We extract key points by scale-invariant feature transform (SIFT) as candidates of nodes, and corresponding feature vectors are also obtained in this step. Nodes are selected if the node candidates have high similarity (inner product of feature vectors) with all candidate nodes from another graph. Then, all chosen nodes are connected, and the weights of edges are measured by the Euclidean distance between two corresponding nodes.

The running time and matching error are calculated by the average results of five matching pairs (1 vs. 2, 2 vs. 3, 3 vs. 4, 4 vs. 5, 5 vs. 6) from the same picture set. The results are shown in Figure 6. More details on experiments with 1000 nodes are shown in Table 4 for further comparison. The ASM method consistently attains the lowest error across all cases while maintaining comparable efficiency.

[2]https://github.com/HongtengXu/gwl
[3]https://github.com/HongtengXu/s-gwl
[4]https://www3.nd.edu/ cone/MAGNA++/
[5]https://github.com/AU-DIS/GRASP
[6]http://www.robots.ox.ac.uk/ vgg/research/affine/

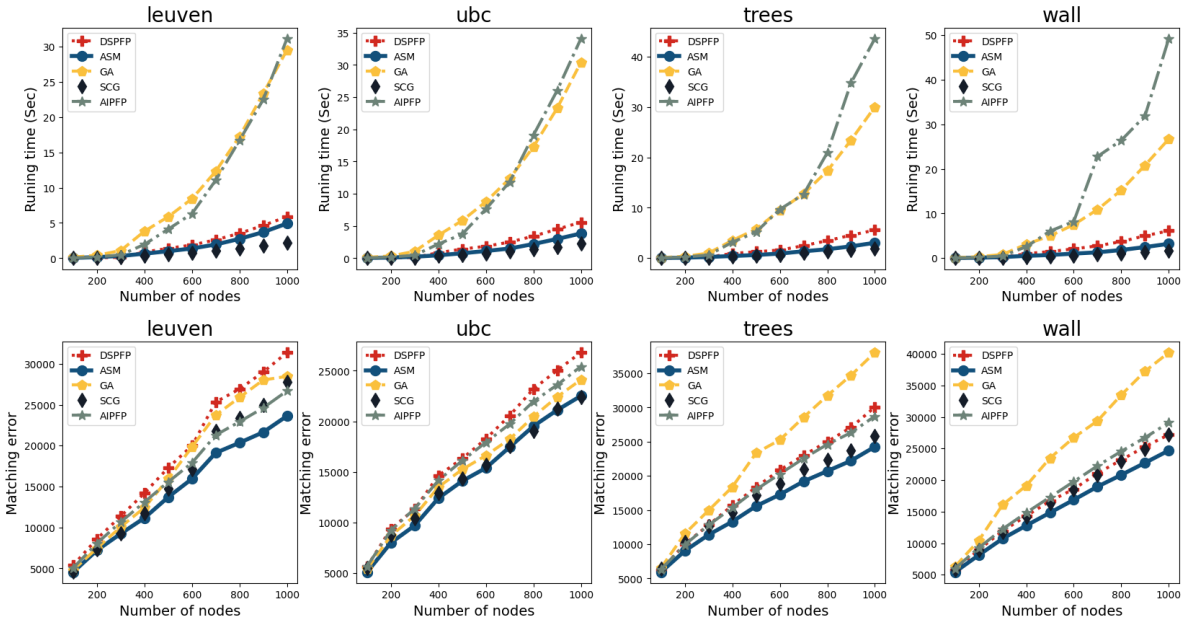

Figure 6. Comparision between algorithms in four graph pairs.

Table 4. Comparisons for graph matching methods on real images with 1000 nodes

Image set	Leuven		ubc		trees		wall	
Methods	Error ($\times 10^4$)	time (s)	Error ($\times 10^4$)	time	Error ($\times 10^4$)	time	Error ($\times 10^4$)	time
DSPFP [23]	3.1	5.9s	2.7	5.6s	3.0	5.6s	2.7	6.1s
GA [8]	2.8	29.5s	2.4	30.3s	3.8	30.0s	4.0	26.7s
AIPFP [19, 23]	2.7	31.1s	2.5	34.1s	2.9	43.6s	2.9	49.2s
SCG [32]	2.7	**1.7**s	2.3	**1.8**s	2.5	**1.5**s	2.6	**1.3**s
ASM	**2.3**	4.9s	**2.2**	3.8s	**2.4**	3.0s	**2.5**	3.2s

6. Conclusion

This paper proposes an adaptive softassign method for large graph matching problems. It can automatically tune the parameter according to a given error bound, which is convenient and robust. The resulting matching algorithm enjoys significantly higher accuracy than previous state-of-the-art large graph matching algorithms.

The proposed Hadamard-Equipped Sinkhorn formulas significantly accelerate the adaptive softassign process and avoid numerical instability in Sinkhorn. These formulas provide a new perspective on operations related to Sinkhorn and optimal transport problems. The Hadamard-Equipped Sinkhorn formulas seem to have some nice properties of group, which might be a promising research direction.

Experiments show that ASM has comparable efficiency in attributed graph matching tasks while the efficiency is not in the first tier in plain graph matching. Therefore, increasing the efficiency in plain graph matching is one of the future works.

Acknowledgement

The authors would like to appreciate the support from the Interdisciplinary Intelligence Super Computer Center of Beijing Normal University at Zhuhai. This work was partially supported by the Natural Science Foundation of China (12271047); UIC research grant (R0400001-22; UICR0400008-21; UICR04202405-21); Guangdong College Enhancement and Innovation Program (2021ZDZX1046); Key Programme Special Fund in XJTLU (KSF-E-32), Research Enhancement Fund of XJTLU (REF-18-01-04); Guangdong Provincial Key Laboratory of Interdisciplinary Research and Application for Data Science, BNU-HKBU United International College (2022B1212010006).

References

[1] Jason Altschuler, Jonathan Niles-Weed, and Philippe Rigollet. Near-linear time approximation algorithms for optimal transport via sinkhorn iteration. *Advances in neural information processing systems*, 30, 2017. 4

[2] Chao-Yeh Chen and Kristen Grauman. Efficient activity detection with max-subgraph search. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1274–1281. IEEE, 2012. 1

[3] Minsu Cho, Jungmin Lee, and Kyoung Mu Lee. Reweighted random walks for graph matching. In *European Conference on Computer Vision*, 2010. 1, 2, 3

[4] Roberto Cominetti and J San Martín. Asymptotic analysis of the exponential penalty trajectory in linear programming. *Mathematical Programming*, 67:169–187, 1994. 4, 11

[5] Timothee Cour, Praveen Srinivasan, and Jianbo Shi. Balanced graph matching. In *Advances in Neural Information Processing Systems*, 2006. 1

[6] Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. In *Advances in Neural Information Processing Systems*, pages 2292–2300, 2013. 2, 3, 4, 6

[7] Amir Egozi, Yosi Keller, and Hugo Guterman. A probabilistic approach to spectral graph matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(1):18–27, 2012. 1

[8] Steven Gold and Anand Rangarajan. A graduated assignment algorithm for graph matching. *IEEE Transactions on pattern analysis and machine intelligence*, 18(4):377–388, 1996. 1, 2, 3, 6, 7, 8, 11, 13

[9] Yanrong Guo, Guorong Wu, Jianguo Jiang, and Dinggang Shen. Robust anatomical correspondence detection by hierarchical sparse graph matching. *IEEE Transactions on Medical Imaging*, 32, 2012. 1

[10] Harville and A. David. *Matrix Algebra From a Statistician's Perspective*. Springer Science & Business Media, 2008. 3

[11] Judith Hermanns, Konstantinos Skitsas, Anton Tsitsulin, Marina Munkhoeva, Alexander Kyster, Simon Nielsen, Alexander M Bronstein, Davide Mottin, and Panagiotis Karras. Grasp: Scalable graph alignment by spectral corresponding functions. *ACM Transactions on Knowledge Discovery from Data*, 17(4):1–26, 2023. 7, 13

[12] Jeffrey J Kosowsky and Alan L Yuille. The invisible hand algorithm: Solving the assignment problem with statistical physics. *Neural Networks*, 7(3):477–490, 1994. 2, 3

[13] Constatine Kotropoulos, Anastasios Tefas, and Ioannis Pitas. Frontal face authentication using morphological elastic graph matching. *IEEE Transactions on Image Processing*, 9(4):555–560, 2000. 1

[14] Harold W Kuhn. The hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2(1-2):83–97, 1955. 3, 6

[15] Zixun Lan, Binjie Hong, Ye Ma, and Fei Ma. More interpretable graph similarity computation via maximum common subgraph inference. *arXiv preprint arXiv:2208.04580*, 2022. 1

[16] Zixun Lan, Ye Ma, Limin Yu, Linglong Yuan, and Fei Ma. Aednet: Adaptive edge-deleting network for subgraph matching. *Pattern Recognition*, page 109033, 2022. 1

[17] E. L. Lawler. The quadratic assignment problem. *Management Science*, 9(4):586–599, 1963. 1, 3

[18] Marius Leordeanu and Martial Hebert. A spectral technique for correspondence problems using pairwise constraints. In *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*, pages 1482–1489. IEEE, 2005. 1

[19] Marius Leordeanu, Martial Hebert, and Rahul Sukthankar. An integer projected fixed point method for graph matching and map inference. In *Advances in Neural Information Processing systems*, pages 1114–1122, 2009. 1, 3, 7, 8, 11, 13

[20] Jure Leskovec and Andrej Krevl. SNAP Datasets: Stanford large network dataset collection. `http://snap.stanford.edu/data`, 2014. 7

[21] Qichen Liao, Jing Chen, Zihao Wang, Bo Bai, Shi Jin, and Hao Wu. Fast sinkhorn i: An o (n) algorithm for the wasserstein-1 metric. *arXiv preprint arXiv:2202.10042*, 2022. 6

[22] Qichen Liao, Zihao Wang, Jing Chen, Bo Bai, Shi Jin, and Hao Wu. Fast sinkhorn ii: Collinear triangular matrix and linear time accurate computation of optimal transport. *arXiv preprint arXiv:2206.09049*, 2022. 6

[23] Yao Lu, Kaizhu Huang, and Cheng-Lin Liu. A fast projected fixed-point algorithm for large graph matching. *Pattern Recognition*, 60:971–982, 2016. 1, 3, 6, 7, 8, 11, 13

[24] Giulia Luise, Alessandro Rudi, Massimiliano Pontil, and Carlo Ciliberto. Differential properties of sinkhorn approximation for learning with wasserstein distance. *Advances in Neural Information Processing Systems*, 31, 2018. 4

[25] Bin Luo, Richard C Wilson, and Edwin R Hancock. Spectral embedding of graphs. *Pattern Recognition*, 36(10):2213–2230, 2003. 1, 3

[26] Jianzhou Luo, Dingchuan Yang, and Ke Wei. Improved complexity analysis of the sinkhorn and greenkhorn algorithms for optimal transport. *arXiv preprint arXiv:2305.14939*, 2023. 6

[27] Haggai Maron and Yaron Lipman. (probably) concave graph matching. In *Advances in Neural Information Processing Systems*, 2018. 1

[28] D. Michel, I. Oikonomidis, and A. Argyros. Scale invariant and deformation tolerant partial shape matching. *Image and Vision Computing*, 29(7):459–469, 2011. 1

[29] Antonio Robles-Kelly and Edwin R Hancock. A riemannian approach to graph embedding. *Pattern Recognition*, 40(3): 1042–1056, 2007. 1

[30] Nikitas Rontsis and Paul Goulart. Optimal approximation of doubly stochastic matrices. In *International Conference on Artificial Intelligence and Statistics*, pages 3589–3598. PMLR, 2020. 1, 3

[31] Binrui Shen, Qiang Niu, and Shengxin Zhu. Fabricated pictures detection with graph matching. In *Proceedings of the 2020 2nd Asia Pacific Information Technology Conference*, pages 46–51, 2020. 1, 3

[32] Binrui Shen, Qiang Niu, and Shengxin Zhu. Dynamical softassign and adaptive parameter tuning for graph matching. *arXiv preprint arXiv:2208.08233*, 2022. 1, 2, 3, 4, 5, 7, 8, 13

[33] Richard Sinkhorn. A relationship between arbitrary positive matrices and doubly stochastic matrices. *The Annals of Mathematical Statistics*, 35(2):876–879, 1964. 2

[34] Yu Tian, Junchi Yan, Hequan Zhang, Ya Zhang, Xiaokang Yang, and Hongyuan Zha. On the convergence of graph matching: Graduated assignment revisited. In *Computer Vision–ECCV 2012: 12th European Conference on Computer Vision, Florence, Italy, October 7-13, 2012, Proceedings, Part III 12*, pages 821–835. Springer, 2012. 1

[35] Shinji Umeyama. An eigendecomposition approach to weighted graph matching problems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10(5):695–703, 1988. 1

[36] Vipin Vijayan, Vikram Saraph, and Tijana Milenković. Magna++: maximizing accuracy in global network alignment via both node and edge conservation. *Bioinformatics*, 31(14):2409–2411, 2015. 7, 13

[37] Tao Wang, Haibin Ling, Congyan Lang, and Songhe Feng. Graph matching with adaptive and branching path following. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(12):2853–2867, 2017. 1

[38] Laurenz Wiskott, Jean-Marc Fellous, Norbert Kruger, and Christoph von der Malsburg. Face recognition by elastic bunch graph matching. *Intelligent Biometric Techniques in Fingerprint and Face Recognition*, 11(5):355–396, 1999. 1

[39] B. Xiang, X. Yang, L. J. Latecki, W. Liu, and E. Al. Learning context-sensitive shape similarity by graph transduction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(5):861, 2010. 1

[40] Yujia Xie, Xiangfeng Wang, Ruijia Wang, and Hongyuan Zha. A fast proximal point method for computing exact wasserstein distance. In *Uncertainty in artificial intelligence*, pages 433–453. PMLR, 2020. 5, 12

[41] Hongteng Xu, Dixin Luo, and Lawrence Carin. Scalable gromov-wasserstein learning for graph partitioning and matching. In *Advances in Neural Information Processing Systems*, 2019. 1, 7, 13

[42] Hongteng Xu, Dixin Luo, Hongyuan Zha, and Lawrence Carin Duke. Gromov-wasserstein learning for graph matching and node embedding. In *International conference on machine learning*, pages 6932–6941. PMLR, 2019. 1, 7, 13

[43] Mikhail Zaslavskiy, Francis Bach, and Jean-Philippe Vert. A path following algorithm for the graph matching problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(12):2227–2242, 2008. 1, 2

[44] Yali Zheng, Lili Pan, Jiye Qian, and Hongliang Guo. Fast matching via ergodic markov chain for super-large graphs. *Pattern Recognition*, 106:107418, 2020. 2