Title

A spectral algorithm for polynomial-time graph isomorphism testing.

Corresponding author

Robert J. O'Shea
King's College London
5th floor, Becket House, 1 Lambeth Palace Rd, London, SE1 7EU, United Kingdom
robert.1.oshea@kcl.ac.uk

Abstract

Polynomial time solutions to the graph isomorphism problem have remained elusive to date, presenting one of the most important issues in computer science. This study describes a spectral algorithm to infer quadratic permutations mapping tuples of isomorphic graphs in $\mathcal{O}(n^4)$ time. Robustness to degeneracy and multiple isomorphisms are achieved through low dimensional eigenspace projections and iterative perturbations respectively. An analytic proof is provided and experimental performance is demonstrated in synthetic and biological data.

Background

The graph isomorphism problem requires the determination of structural equivalence of two graphs, which may be demonstrated by inference of a quadratic permutation function mapping one graph adjacency matrix to the other. It is considered one of the most important unresolved problems in computer science [1] and arises frequently in applications such as image recognition, systems biology and cheminformatics [1]. Although the problem's computational complexity has remained an open problem since its description in 1972 [1, 2], it has been demonstrated that it is in the low hierarchy of the class NP [3]. Polynomial time algorithms are available for some special cases of the problem. Babai proposed an algorithm for graphs with bounded eigenvalue multiplicity [4]. Luks proposed an algorithm for graphs of bounded degree [5] which was developed further by Babai to provide a quasipolynomial-time solution for the general case, running in at $\exp(\log n)^{\mathcal{O}(1)}$ time [6–8]. However, to date, no polynomial time algorithm has been found for the general case [1, 2].

This study presents a $\mathcal{O}(n^4)$ algorithm for the symmetric graph isomorphism problem. Theoretical analysis is provided to support this claim and experimental performance is demonstrated in real and synthetic data. Lemma proofs are included in **Supplementary Information 1**.

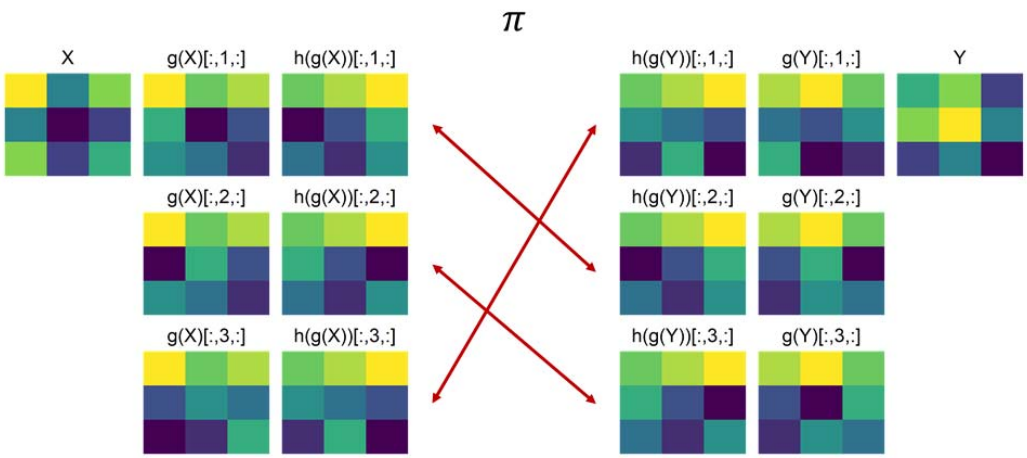

Figure 1. Visualisation of the permutation inference algorithm solving a non-degenerate toy problem. Function $g()$ returns the outer product of each eigenspace, scaled by its eigenvalue. Function $h()$ permutes matrix rows to sort by each column in turn from left to right, resolving earlier ties by later columns. Red arrows indicate analogous vertices. As each vertex has a single analogous vertex in this case, the set of analogies is the permutation function.

Algorithm

Let $X \in \mathbb{R}^{\{n \times n\}}$ be a symmetric matrix, with eigenvalues $\lambda \in \mathbb{R}^{\{n\}}$ such that $\lambda_1 \geq \lambda_2 \ldots \geq \lambda_n$. As X may be considered a graph adjacency matrix of a graph, we refer to the ith row and column of X as the "vertex" X_i. Let $\Lambda \in \mathbb{R}^{\{n \times n\}}$ denote the diagonal matrix of λ. Let $Q \in \mathbb{R}^{\{n \times n\}}$ denote an orthonormal eigenbasis of X, such that:

$$X = Q \Lambda Q^T$$

Let $\dot{\lambda}$ denote the set of $m \leq n$ unique eigenvalues in λ. Let C denote the indexes of each unique eigenvalue, such that:

$$c_l = \{i | \lambda_i = \dot{\lambda}_l\}$$

Let $\dot{\lambda}$ denote the set of $m \leq n$ unique eigenvalues in λ. Let C denote the set of sets indexing each unique eigenvalue, such that:

$$c_l = \{i | \lambda_i = \dot{\lambda}_l, i \in \mathbb{N}_{\leq n}\}$$

Let $Y \in \mathbb{R}^{\{n \times n\}}, \pi \in \Pi^{\{n\}}$ such that:

$$Y = X_{\pi,\pi}$$

Let $\mathbb{I}$ denote the indicator function. Let $P \in \mathbb{N}_{\leq 1}^{\{n \times n\}}$ denote the matrix representation of π such that:

$$P_{i,j} = \mathbb{I}\left(i = \pi_j\right)$$

$$\therefore Y = P X P^T$$

P and π will be used interchangeably as convenient.

Let $\phi(X,Y)\colon \left(\mathbb{R}^{\{n\times n\}}, \mathbb{R}^{\{n\times n\}}\right) \mapsto \mathbb{N}_{\leq n!}$ denote the number of unique solutions $\hat{\pi} \in \Pi^{\{n\}}$ satisfying $Y = X_{\hat{\pi},\hat{\pi}}$. Thus, $\phi(X,Y) > 0$ implies isomorphism of X and Y. Let $X_a \to Y_b$ denote "analogy" of vertices X_a and Y_b, such that there exists some $\hat{\pi}$ satisfying $\hat{\pi}_a = b$ and $Y = X_{\hat{\pi},\hat{\pi}}$. If only one vertex Y_b is analogous to X_a, the analogy is described as unique. If a vertex X_a has multiple analogous vertices Y_B the analogy is described as non-unique. Let $X_A \to Y_B$ denote $X_{a_i} \to Y_{b_j}$ for all $a_i \in A$ and $b_j \in B$.

Let $g(X)\colon \mathbb{R}^{\{n\times n\}} \mapsto \mathbb{R}^{\{n\times n\times m\}}$ be a function transforming X to an array of the outer products of its eigenspaces scaled by their eigenvalues, such that:

$$g(X)_{:,:,l} = \dot{\lambda}_l Q_{:,c_l}\left(Q_{:,c_l}\right)^T$$

Lemma 1 demonstrates that $g(Y) = g(X)_{\pi,\pi,:}$, or ,equivalently, $g\left(X_{\pi,\pi}\right) = g(X)_{\pi,\pi,:}$. Thus, g is a deterministic function, which is consitent under quadratic permutation. X is fullly described by $g(X)$ such that:

$$X = \sum_{l=1}^{m} \dot{\lambda}_l Q_{:,c_l}\left(Q_{:,c_l}\right)^T$$

$$\therefore X = \sum_{l=1}^{|C|} g(X)_{:,:,l}$$

Lemma 2 demonstrates that $Pg(X)_{:,\pi_i,:} = g(Y)_{:,i,:}$ for any permutation $P \in \Pi^{\{n\}}$. Thus, slices of $g(X)$ corresponding to vertices are row permutations of the analougous slices in $g(Y)$. Equivalence is induced between analgous slices of $h\left(g(X)\right)$ and $h\left(g(X)\right)$ by applying a row ordering function h. Let $h(Z)\colon \mathbb{R}^{\{n\times m\}} \mapsto \mathbb{R}^{\{n\times m\}}$ be a hierarchical row ordering function such that:

$$H^{(Z)} \in \Pi^{\{n\}}\colon \left(H^{(Z)}Z\right)_{a,c} \geq \left(H^{(Z)}Z\right)_{b,d} \ \forall \ \{a,b,c,d \mid a < b, c \leq d\}$$

$$h(Z) = H^{(Z)}Z$$

Where $Z \in \mathbb{R}^{\{n\times m\}}$. **Lemma 3** demonstrates that $h(Z) = h(PZ)$ for all $Z \in \mathbb{R}^{\{n\times m\}}$ and $P \in \Pi^{\{n\}}$. Thus, the function h induces equivalence between a matrix subjected to any two row permutations.

Lemma 4a demonstrates that $h\left(g(Y)_{:,i,:}\right) = h\left(g(X)_{:,\pi_i,:}\right)$. Thus, analougous slices of $h\left(g(X)\right)$ and $h\left(g(Y)\right)$ are equal. **Lemma 4b** demonstrates that if $h\left(g(X)_{:,i,:}\right) = h\left(g(Y)_{:,j,:}\right)$ then $X_i \to Y_j$. **Lemma 4c** demonstrates that if $h\left(g(X)_{:,i,:}\right) \neq h\left(g(Y)_{:,j,:}\right)$ then $X_i \nrightarrow Y_j$. Thus, equality of slices of $h\left(g(X)\right)$ and $h\left(g(Y)\right)$ determines analogy between the corresponding vertices. Accordingly, a precursor to the permutation matrix, $\check{P}$, may be generated generated indicating the pairwise equality of the slices of $h\left(g(X)\right)$ and $h\left(g(Y)\right)$, such that:

$$\check{P}_{j,i} \coloneqq \mathbb{I}\left(h\left(g(X)\right)_{:,i,:} = h\left(g(Y)\right)_{:,j,:}\right)$$

$$\therefore \check{P}_{j,i} = \mathbb{I}\left(X_i \to Y_j\right)$$

Lemma 5a demonstrates that $\check{P} = P$ if $\phi(X,Y) = 1$. Thus, if the solution to π is unique, the permutation matrix inferred at this point. **Lemma 5b** demonstrates that there exists some $i \in \mathbb{N}_{\leq n}$ such that $\sum \check{P}_{i,:} > 1$ if $\phi(X,Y) > 1$. Thus, However, if multiple isomorphisms exist, some vertices will have multiple analogies and $\check{P} \neq P$. To resolve this issue, analogous perturbations are applied to X

and Y to induce unique analogies. Let $\{a, b\} \subseteq \mathbb{N}_{\leq n}$ such that $\check{P}_{b,a} = 1$. X and Y may be perturbed to generate X^{*1} and Y^{*1}, with unique analogy $X_a^{*1} \to Y_b^{*1}$.

Let $X^{*1} \in \mathbb{R}^{\{n \times n\}}$ such that:

$$X_{i,j}^{*1} \coloneqq \begin{cases} \max(X) + 1, & i = j = a \\ X_{i,j}, & otherwise \end{cases}$$

Let $Y^{*1} \in \mathbb{R}^{\{n \times n\}}$ such that:

$$Y_{i,j}^{*1} \coloneqq \begin{cases} \max(Y) + 1, & i = j = b \\ Y_{i,j}, & otherwise \end{cases}$$

Lemma 6 demonstrates that the correspondence $X_{a_{*k}}^{*k} \to Y_{b_{*k}}^{*k}$ is unique. **Lemma 7a** demonstrates that if $\hat{P}$ satisfies the perturbed equation $Y^{*1} = \hat{P}X^{*1}\hat{P}^T$ then it satisfies the original equation $Y = \hat{P}X\hat{P}^T$. Thus, a solution for π may be identified in the perturbed equality. If a non-uniquely analogous vertex pair is perturbed in this way, a unique analogy is induced and the number of isomorphisms is thereby reduced. Vertex analogies may now be checked in the perturbed matrices X^{*1} and Y^{*1}, such that:

$$\check{P}_{j,i}^{*1} \coloneqq \mathbb{I}\left(h\left(g\left(X^{*1}\right)_{:,i,:}\right) = h\left(g\left(Y^{*1}\right)_{:,j,:}\right)\right)$$

If there again exists some $a_{*1} \in \mathbb{N}_{\leq n}$ such that $\sum \check{P}_{a_{*1},:}^{*1} > 1$ then the steps of perturbation and analogy checking may be reiterated such that:

$$X_{i,j}^{*k+1} \coloneqq \begin{cases} \max\left(X^{*k}\right) + 1, & i = j = a_{*k} \\ X_{i,j}^{*k}, & otherwise \end{cases}$$

$$Y_{i,j}^{*k+1} \coloneqq \begin{cases} \max\left(Y^{*k}\right) + 1, & i = j = b_{*k} \\ Y_{i,j}^{*k}, & otherwise \end{cases}$$

$$\check{P}_{j,i}^{*k+1} \coloneqq \mathbb{I}\left(h\left(g\left(X^{*k+1}\right)_{:,i,:}\right) = h\left(g\left(Y^{*k+1}\right)_{:,j,:}\right)\right)$$

Each perturbation iteration reduces the number of solutions to $\hat{\pi}$ by inducing a unique analogy between some non-uniquely analogous vertex pair. When all analogies are unique $\check{P} = P$ by Lemma 5a. **Lemma 8** demonstrates that this procedure will return a unique solution $\hat{P} \in \Pi^{\{n\}}$ satisfying $Y = \hat{P}X\hat{P}$ within $n - 1$ iterations. **Figure 1** provides a visualisation for the algorithm in a toy problem with $\phi(X, Y) = 1$ and $n = 3$.

Algorithm pseudocode

Inputs

$X \in \mathbb{R}^{\{n \times n\}}$

$Y \in \mathbb{R}^{\{n \times n\}}$

Outputs

$\mathbb{I}(\phi(X, Y) > 0)$

Algorithm

$\check{P} \coloneqq 1^{\{n \times n\}}$

$V \coloneqq \left\{ v \middle| \sum \check{P}_{:,v} > 1 \right\}$

$X^{*k} \coloneqq X$

$Y^{*k} \coloneqq Y$

WHILE $(V \neq \emptyset)$:

Compute $g\left(X^{*k}\right)$

Compute $g\left(Y^{*k}\right)$

FOR i in $\{1, \dots n\}$:

Compute $h\left(g\left(X^{*k}\right)_{:,i,:}\right)$

Compute $h\left(g\left(Y^{*k}\right)_{:,i,:}\right)$

FOR i in $\{1, \dots n\}$:

FOR j in $\{1, \dots n\}$:

$$\check{P}_{j,i} \coloneqq \mathbb{I}\left(h\left(g\left(X^{*k}\right)_{:,i,:}\right) = h\left(g\left(Y^{*k}\right)_{:,j,:}\right)\right)$$

$V \coloneqq \left\{ v \middle| \sum \check{P}_{:,v} > 1 \right\}$

IF $(V \neq \emptyset)$:

$a_{*k} \coloneqq a \,|\, \sum \check{P}_{a,:} > 1$

$b_{*k} \coloneqq b \,|\, \check{P}_{a,b} = 1$

$X^{*k+1} \coloneqq X^{*k}$

$X^{*k+1}_{a_{*k},a_{*k}} \coloneqq \max\left(X^{*k}\right) + 1$

$Y^{*k+1} \coloneqq Y^{*k}$

$Y^{*k+1}_{b_{*k},b_{*k}} \coloneqq \max\left(Y^{*k}\right) + 1$

$k \coloneqq k + 1$

ELSE:

$\hat{P} \leftarrow \check{P}$

$\mathbb{I}(\phi(X,Y) > 0) \coloneqq \mathbb{I}\left(Y = \hat{P} X \hat{P}^T\right)$

RETURN $\mathbb{I}(\phi(X,Y) > 0)$

Experiments

Algorithm performance was tested on real-word and synthetic graph dataset. Real world datasets consisted of the C-elegans network [9, 10] and the first 100 enzyme structure graphs in a chemoinformatics repository [10]. As regularly structured synthetic graphs present some of the most challenging problems for traditional algorithms [4], Heawood graph [11] and Johnson graphs [12] with parameters (n=7, k=3) and (n=8, k=4) were also included. Graphs were represented as symmetric adjacency matrices. In each experiment, a random quadratic permutation was applied to one of the graph datasets, generating a tuple of isomorphic graphs. The algorithm was subsequently applied to infer a quadratic permutation function mapping the graphs. Five experiments were generated from different random permutation of each graph dataset. Analysis was performed with R and Rstudio [13, 14]. Numerical tolerances of 10^{-8} and 10^{-6}were applied when testing equality of eigenvalues and eigenvectors, respectively. All code required to reproduce the findings of this study is provided in **Supplementary Information 2**.

Results

The graph isomorphism algortihm identified a correct solution in each experiment. Algorithmic vulnerability to numerical instability was identified in some experiments, necessitating the imposition of numerical tolerances during equality checking operations.

Discussion

The algorithm described in this study solves all real-valued symmetric graph isomorphism problems. Neither positive definiteness nor eigenvalue multiplicity bounds are required conditions, differentiating this work from previous approaches. Where $\phi(X, Y) > 1$, The algorithm returns a single solution to $\hat{\pi}$, however the full set of solutions may be returned by changing the order of perturbations.

The complexity of the algorithm is dominated by the eigendecomposition procedure, which runs in $\mathcal{O}(n^3)$ time. As up to $n - 1$ runs of eigendecomposition may be required, the algorithm will run in $\mathcal{O}(n^4)$ time in the worst case scenario (multiples of the identity matrix).

As the algorithm requires testing equality of eigenvalues and outer products of eigenspaces, issues may arise with regards to numerical precision. These problems may be addressed with appropriate rounding or numerical tolerance.

Schoning demonstrated that the polynomial hierarchy collapses to a finite level if the graph isomorphism problem is NP-complete [3]. The algorithm presented here demonstrates that any problem which may be reduced to a graph isomorphism problem may be solved in polynomial time. Thus, If the graph isomorphism problem is found to be NP-complete this algorithm may be used to solve all NP problems in polynomial time. Further research may investigate developments to the algorithm to solve the subgraph isomorphism problem, which is NP-complete [15], in polynomial time.

Funding

Authors acknowledge funding support from the UK Research & Innovation London Medical Imaging and Artificial Intelligence Centre; Wellcome/Engineering and Physical Sciences Research Council Centre for Medical Engineering at King's College London (WT 203148/Z/16/Z); National Institute for Health Research Biomedical Research Centre at Guy's & St Thomas' Hospitals and King's College London; Cancer Research UK National Cancer Imaging Translational Accelerator (A27066).

References

1. Grohe M, Schweitzer P (2020) The graph isomorphism problem. Commun ACM 63:128–134. https://doi.org/10.1145/3372123

2. Karp RM (1972) Reducibility among Combinatorial Problems

3. Schöning U (1988) Graph isomorphism is in the low hierarchy. J Comput Syst Sci 37:312–323. https://doi.org/10.1016/0022-0000(88)90010-4

4. Babai L, Grigoryev DY, Mount DM (1982) Isomorphism of graphs with bounded eigenvalue multiplicity. Proc Annu ACM Symp Theory Comput 310–324. https://doi.org/10.1145/800070.802206

5. Luks EM (1982) Isomorphism of graphs of bounded valence can be tested in polynomial time. J Comput Syst Sci 25:42–65. https://doi.org/10.1016/0022-0000(82)90009-5

6. Babai L (2016) Graph isomorphism in quasipolynomial time: [Extended abstract]. Proc Annu ACM Symp Theory Comput 19-21-June:684–697. https://doi.org/10.1145/2897518.2897542

7. Babai L (2015) Graph Isomorphism in Quasipolynomial Time

8. Helfgott HA, Bajpai J, Dona D (2017) Graph isomorphisms in quasi-polynomial time

9. Kaiser M, Hilgetag CC (2005) Non-Optimal Component Placement, but Short Processing Paths, due to Long-Distance Projections in Neural Systems. PLoS Comput Biol preprint:e95. https://doi.org/10.1371/journal.pcbi.0020095.eor

10. Rossi RA, Ahmed NK (2015) The network data repository with interactive graph analytics and visualization. Proc Natl Conf Artif Intell 6:4292–4293

11. Heawood PJ (1890) Map-Colour Lemma. Q J Math First Seri:332–338

12. Holton DA, Sheehan J (1993) The Petersen Graph. The Petersen Graph. https://doi.org/10.1017/cbo9780511662058

13. R Core Team (2021) R: A Language and Environment for Statistical Computing. R Found. Stat. Comput.

14. RStudio Team (2021) RStudio: Integrated Development for R

15. Wegener I (2005) Complexity theory: Exploring the limits of efficient algorithms. Complex Theory Explor Limits Effic Algorithms 1–308. https://doi.org/10.1007/3-540-27477-4

Supplementary information 1: Lemma proofs.

Lemma 1. $g(X)_{\pi,\pi,:} = g(Y)$.

We have:

$$Y = PXP^T$$

$$\therefore Y = PQ\Lambda Q^T P^T$$

$$\therefore Y = PQ\Lambda(PQ)^T$$

As PQ is an orthogonal matrix, this factorisation is an eigendecomposition. Therefore, B shares eigenvalues with A. Thus, we denote the eigendecomposition of B such that:

$$Y = U\Lambda U^T$$

We have:

$$X = Q\Lambda Q^T$$

$$\therefore Q^T X Q = \Lambda$$

Therefore, for all $l \in \mathbb{N}_{\leq m}$, we have:

$$\therefore \left(Q_{:,c_l}\right)^T X Q_{:,c_l} = \Lambda_{c_l,c_l}$$

Also:

$$\left(U_{:,c_l}\right)^T Y U_{:,c_l} = \Lambda_{c_l,c_l}$$

$$\therefore \left(Q_{:,c_l}\right)^T X Q_{:,c_l} = \left(U_{:,c_l}\right)^T Y U_{:,c_l}$$

$$\therefore \left(Q_{:,c_l}\right)^T X Q_{:,c_l} = \left(U_{:,c_l}\right)^T P X P^T U_{:,c_l}$$

$$\therefore \left(Q_{:,c_l}\right)^T X Q_{:,c_l} = \left(P^T U_{:,c_l}\right)^T X P^T U_{:,c_l}$$

As X is symmetric we have $Q_{:,c_l} \in \mathbb{R}^{\{n \times |C_l|\}}$ and $U_{:,c_l} \in \mathbb{R}^{\{n \times |C_l|\}}$.

$$\therefore \left(Q_{:,c_l}\right)^T Q_{:,c_l} = \left(P^T U_{:,c_l}\right)^T P^T U_{:,c_l}$$

$$\therefore P Q_{:,c_l}\left(Q_{:,c_l}\right)^T P^T = U_{:,c_l}\left(U_{:,c_l}\right)^T$$

$$\therefore P \dot{\lambda}_l \, Q_{:,c_l}\left(Q_{:,c_l}\right)^T P^T = \dot{\lambda}_l U_{:,c_l}\left(U_{:,c_l}\right)^T$$

$$\therefore P g(X)_{:,:,l} P^T = g(Y)_{:,:,l} \; \forall \, l \in \mathbb{N}_{\leq m}$$

$$\therefore g(X)_{\pi,\pi,:} = g(Y)$$

Lemma 2. $P g(X)_{:,\pi_i,:} = g(Y)_{:,i,:}$.

From Lemma 1 we have:

$$g(X)_{\pi,\pi,:} = g(Y)$$

$$\therefore P g(X)_{:,\pi,:} = g(Y)$$

$$\therefore \; P g(X)_{:,\pi_i,:} = g(Y)_{:,i,:}$$

Lemma 3. $h(Z) = h(PZ)$ for all $Z \in \mathbb{R}^{\{n \times m\}}$ and $P \in \Pi^{\{n\}}$.

We have:

$$h(Z) = H^{(Z)} Z$$

$$h(PZ) = H^{(PZ)} PZ$$

$$H^{(PZ)} = H^{(Z)} P^T$$

$$\therefore H^{(PZ)} PZ = H^{(Z)} P^T PZ$$

$$\therefore H^{(PZ)} PZ = H^{(Z)} Z$$

Lemma 4a. $h\left(g(Y)_{:,i,:}\right) = h\left(g(X)_{:,\pi_i,:}\right)$.

$$h\big(g(Y)_{:,i,:}\big) = h\big(g(PXP^T)_{:,i,:}\big)$$
$$= h\big(g(PX)_{:,\pi_i,:}\big)$$
$$= h\big(Pg(X)_{:,\pi_i,:}\big)$$

By Lemma 3 we have:

$$h\big(Pg(X)_{:,\pi_i,:}\big) = h\big(g(X)_{:,\pi_i,:}\big)$$
$$\therefore h\big(g(Y)_{:,i,:}\big) = h\big(g(X)_{:,\pi_i,:}\big)$$

Lemma 4b. If $h\big(g(X)_{:,i,:}\big) = h\big(g(Y)_{:,j,:}\big)$ then $X_i \to Y_j$.

Let $g(Y)_{:,j,:} = \hat{P}g(X)_{:,i,:}$

$$\therefore g(Y)_{:,j,:} = g(X)_{\hat{\pi},i,:}$$
$$\therefore \dot{\lambda}_l U_{:,c_l}\big(U_{:,c_l}\big)^T_{:,j} = \hat{P}\dot{\lambda}_l Q_{:,c_l}\big(Q_{:,c_l}\big)^T_{:,i} \ \forall\ l \le m$$
$$\therefore \dot{\lambda}_l U_{:,c_l}\big(U_{:,c_l}\big)^T_{:,j} = \hat{P}\dot{\lambda}_l Q_{:,c_l}\big(Q_{:,c_l}\big)^T \hat{P}^T_{:,\hat{\pi}_i}$$
$$\therefore \sum_{k=1}^{m} \dot{\lambda}_l U_{:,c_l}\big(U_{:,c_l}\big)^T_{:,j} = \sum_{k=1}^{m} \hat{P}\dot{\lambda}_l Q_{:,c_l}\big(Q_{:,c_l}\big)^T \hat{P}^T_{:,\hat{\pi}_i}$$
$$\therefore Y_{:,j} = \hat{P}X\hat{P}^T_{:,\hat{\pi}_i}$$

Likewise, we have:

$$\sum_{k=1}^{m} \dot{\lambda}_l U_{:,c_l}\big(U_{:,c_l}\big)^T_{:,\backslash j} = \sum_{k=1}^{m} \hat{P}\dot{\lambda}_l Q_{:,c_l}\big(Q_{:,c_l}\big)^T \hat{P}^T_{:,\backslash i}$$
$$\therefore Y_{:,\backslash j} = \hat{P}X\hat{P}^T_{:,\backslash \hat{\pi}_i}$$

Thus, if $\hat{\pi}_i = j$ we have:

$$\therefore Y = \hat{P}X\hat{P}^T$$
$$\therefore X_i \to Y_j$$

Lemma 4c. If $X_i \to Y_j$ then $h\big(g(X)_{:,i,:}\big) = h\big(g(Y)_{:,j,:}\big)$.

Let $X_i \to Y_j$.

$$\therefore \pi_i = j$$
$$\therefore h\big(g(X)_{:,i,:}\big) = h\big(g(PX)_{:,\pi_i,:}\big)$$
$$\therefore h\big(g(X)_{:,i,:}\big) = h\big(g(PX)_{:,\pi_i,:}\big)$$

$$h\big(g(Y)_{:,j,:}\big) = h\big(g(PXP^T)_{:,j,:}\big)$$
$$= h\big(g(PX)_{:,i,:}\big)$$

$$= h(Pg(X)_{:,i,:})$$

By Lemma 2 we have:

$$h\big(Pg(X)_{:,i,:}\big) = h\big(g(X)_{:,i,:}\big)$$

$$\therefore h\big(g(Y)_{:,j,:}\big) = h\big(g(X)_{:,i,:}\big)$$

Lemma 5a. If $\phi(X,Y) = 1$ then $\check{P} = P$.

If $\phi(X,Y) = 1$ then each X_i has one analogous vertex Y_{π_i} and each correspondence is unique. Thus we have:

$$\check{P}_{i,j} = \mathbb{I}\big(X_i \to Y_j\big)$$

$$\therefore \check{P}_{i,j} = \mathbb{I}(\pi_i = j)$$

$$\therefore \check{P} = P$$

Lemma 5b If $\phi(X,Y) > 1$ then there exists some $i \in \mathbb{N}_{\leq n}$ such that $\sum \check{P}_{i,:} > 1$.

If $\phi(X,Y) = 1$ then there exists at least one vertex X_i with multiple analogous target vertices Y_B such that $X_i \to Y_{b_i}$ for all $b_i \in B$. Therefore, there exists some row in $\check{P}$ with multiple positive entries.

Lemma 6. The correspondence $X^{*k}_{a_{*k}} \to Y^{*k}_{b_{*k}}$ is unique.

Let $\hat{\pi}$ such that $\hat{\pi}_{a_{*k}} = b_{*k}$ and that $Y^{*k} = \hat{P}X^{*k}\hat{P}^T$.

$$diag\big(Y^{*k}\big) = \hat{P} \cdot diag\big(X^{*k}\big)$$

$$diag\big(X^{*k}\big)_{a_{*k}} = diag\big(Y^{*k}\big)_{b_{*k}} = \max\big(X^{*(k-1)}\big) + 1$$

$$diag\big(X^{*k}\big)_{\setminus a_{*k}} \leq \max\big(X^{*(k-1)}\big)$$

$$\therefore diag\big(X^{*k}\big)_{\setminus a_{*k}} < diag\big(Y^{*k}\big)_{b_{*k}}$$

Therefore b_{*k} is the only solution to $\hat{\pi}_{a_{*k}}$.

Lemma 7a. If $\hat{\pi}$ satisfies Y satisfies $Y^{*1} = \hat{P}X^{*1}\hat{P}^T$ then it satisfies $Y = \hat{P}X\hat{P}^T$.

Let $X_A \to X_B$ and $Y = \hat{P}X\hat{P}^T$.

$$\therefore X_{a_i} \to X_{b_j} \; \forall \; a_i \in A, b_j \in B$$

$$\therefore diag(X)_{a_i} = diag(Y)_{b_j} \forall \; a_i \in A, b_j \in B$$

$$\therefore diag(X)_{a_i} = diag(X)_{a_j} \forall \; a_i \in A, a_j \in A$$

Let $E_a \in \mathbb{N}^{\{n \times n\}}_{\leq 1}$ be a matrix with a single unit diagonal entry and zeros elsewhere, such that:

$$E_a = \mathbb{I}(i = a, j = a)$$

Let X^{*1} and Y^{*1} be the perturbed matrices at $a_* \in A$ and $b_* \in B$ respectively.

$$\therefore X^{*1} = X + E_{a*}$$

Also:

$$\therefore Y^{*1} = Y + E_{b*}$$

$$\therefore \hat{P}X^{*1}\hat{P}^T = Y + E_{b*}$$

$$\therefore \hat{P}X\hat{P}^T + \hat{P}E_{a*}\hat{P}^T = Y + E_{b*}$$

$$\therefore \hat{P}X\hat{P}^T + E_{b*} = Y + E_{b*}$$

$$\therefore \hat{P}X\hat{P}^T = Y$$

Lemma 7b. If $\hat{\pi}$ satisfies Y satisfies $Y^{*k+1} = \hat{P}X^{*1}\hat{P}^T$ then it satisfies $Y = \hat{P}X\hat{P}^T$.

Following Lemma 7a, we have:

$$\hat{P}X^{*k+1}\hat{P}^T = Y^{*k} + E_{b*}$$

$$\therefore \hat{P}X^{*k}\hat{P}^T = Y^{*k}$$

By induction, it is demonstrated that $\hat{P}X^{*k}\hat{P}^T = Y^{*k}$ for all $k \geq 0$.

Lemma 8. If $\phi(X, Y) > 1$ then this procedure will return a unique solution $\hat{\pi}$ within $n - 1$ iterations.

Each perturbation iteration induces a unique analogy between a pair of vertices by assigning a unique value to the corresponding entries on the diagonal. After $n - 1$ iterations, the matrix diagonals contain n distinct values, so a unique solution to π must have been reached at or before this iteration.