Title

Exact polynomial-time graph canonisation and isomorphism testing through comparison of ordered vertex eigenprojections

Author

Robert John O'Shea, Clinical Research Fellow, Department of Cancer Imaging, King's College London.

Corresponding author

Robert John O'Shea

Clinical Research Fellow

Department of Cancer Imaging

King's College London

5th floor, Becket House, 1 Lambeth Palace Rd, London, SE1 7EU, United Kingdom

robert.1.oshea@kcl.ac.uk

Abstract

Motivation

Graph canonisation and isomorphism testing representation are fundamental computational problems, whose complexity has remained unsolved to date. This study examines graph eigenprojections, demonstrating that linear-ordering transformations induce canonical properties therein to yield polynomial-time canonisation and isomorphism testing in all undirected graphs.

Results

This study presents an exact method to identify analogous vertices in isomorphic graphs, through comparison of vertices' eigenprojection matrices, which are shown to be related by a linear permutation. Systematic perturbation strategies are developed to reduce degeneracy whilst

conserving isomorphism, through the addition of characteristically weighted self-loops to analogous vertices. Repeated iterations of analogy testing and perturbation deliver canonical vertex labelling and recovery of isomorphic mappings in $\mathcal{O}(n^4)$ time in all graphs. Analytical proofs are provided to support claims and experimental performance is demonstrated in biological and synthetic data, with comparison to a commonly used heuristic algorithm.

Availability and Implementation

Source code is provided at github.com/robertoshea/graph_isomorphism.

Contact

robert.1.oshea@kcl.ac.uk

Supplementary Data.

Not applicable.

Introduction

The graph isomorphism problem requires the determination of structural equivalence of two graphs. It is considered one of the most important unresolved problems in computer science (Grohe and Schweitzer, 2020) and arises in applications across the domains of image recognition, systems biology and cheminformatics (Grohe and Schweitzer, 2020). Although the problem's computational complexity has remained an open problem since its description in 1972 (Karp, 1972; Grohe and Schweitzer, 2020), it has been demonstrated that it is in the low hierarchy of the class NP (Schöning, 1988; Arvind and Torán, 2005). Graph canonisation is a closely related task, which requires graph representation in a form which is invariant under permutation and characteristic of the graph automorphism group. Thus, isomorphism testing may also be achieved by representing both graphs in canonical form and checking equality.

Exact polynomial time algorithms are available for some special cases of the graph isomorphism problem. Babai proposed a spectral assignment algorithm for graphs with bounded eigenvalue multiplicity (Babai *et al.*, 1982). Luks proposed an algorithm for graphs of bounded degree (Luks, 1982) which was developed further by Babai to provide a quasipolynomial-time solution for the general case (Babai, 2016; Helfgott *et al.*, 2017).

Klus and Sahai developed a heuristic spectral assignment algorithm for degenerate graphs, in which vertex analogy is estimated by comparing vertex eigenpolytopes (Klus and Sahai, 2018). A graph perturbation strategy was introduced in which pairs of analogous vertices were sought and perturbations applied to induce unique analogies. Analogous vertices were estimated by sorting individual rows of vertices' eigenprojection matrices and subsequently testing equality. As the algorithm was not constrained to apply the same sorting permutation to each row, equality could be induced between some non-analogous vertex pairs, yielding false positives. Subsequently, when perturbations were erroneously applied to non-analogous vertex pairs, "backtracking" was required (Klus and Sahai, 2018). No exact polynomial-time algorithm has been identified for the general case graph isomorphism problem to date (Grohe and Schweitzer, 2020; Klus and Sahai, 2018).

Hypothesis and Objectives

It was hypothesised that Klus' heuristic could be developed into an exact method by modifying the ir analogy testing and perturbation strategies. It was hypothesised that a linear permutation would induce equality between analogous vertices' eigenprojection matrices, which would suffice to demonstrate analogy. The objective of this study is to prove this result and employ it to develop exact polynomial-time algorithms for graph canonisation and isomorphism testing. Accordingly, an exact method is developed to identify analogous vertex pairs through comparison of vertices' eigenprojection matrices. Klus' perturbation strategy is developed to reduce degeneracy in a canonically reproducible manner, conserving isomorphism of perturbed graph tuples by perturbing

pairs of analogous vertices. Theoretical analysis is provided to support these claims and experimental performance is demonstrated in real and synthetic data.

Systems and Methods

Preliminaries and notation

Let $\mathcal{S}_n$ denote the symmetric group of degree n and $\mathcal{P}_n$ denote the set of $n \times n$ dimensional permutation. Let $\pi \in \mathcal{S}_n$ and $P \in \mathcal{P}_n$ denote corresponding vector and matrix representations of the same permutation, such that:

$$p_{ij} = \begin{cases} 1, & \pi(i) = j \\ 0, & \pi(i) \neq j \end{cases}$$

Throughout the article, vector and matrix representation of permutation functions will be used interchangeably.

Let $\mathcal{G}_A(\mathcal{V}, \mathcal{E}_A)$ and $\mathcal{G}_B(\mathcal{V}, \mathcal{E}_B)$ be two weighted undirected graphs, with adjacency matrices $A, B \in \mathbb{R}^{n \times n}$, where $\mathcal{V} = \{v_1, \dots, v_n\}$ is the vertex set and $\mathcal{E}_A$ and $\mathcal{E}_B$ are edge sets.

Definition . Graphs $\mathcal{G}_A$ and $\mathcal{G}_B$ are termed "isomorphic" if there exists some permutation $\pi \in \mathcal{S}_n$ such that:

$$\left(v_i, v_j\right) \in \mathcal{E}_A \Leftrightarrow \left(v_{\pi(i)}, v_{\pi(j)}\right) \in \mathcal{E}_B$$

Equivalently, $\mathcal{G}_A$ and $\mathcal{G}_B$ are isomorphic if there exists some $P \in \mathcal{P}_n$, such that:

$$B = PAP^T$$

Isomorphism of $\mathcal{G}_A$ and $\mathcal{G}_B$ is denoted $\mathcal{G}_A \cong \mathcal{G}_B$.

Definition. A permutation $\pi \in \mathcal{S}_n$ is termed an "isomorphic mapping" from $\mathcal{G}_A$ to $\mathcal{G}_B$ if it satisfies:

$$\left(v_{\pi(i)}, v_{\pi(j)}\right) \in \mathcal{E}_A \Leftrightarrow \left(v_i, v_j\right) \in \mathcal{E}_B$$

Equivalently, $P \in \mathcal{P}_n$ is termed an isomorphic mapping from $\mathcal{G}_A$ to $\mathcal{G}_B$ if it satisfies $B = PAP^T$. This is denoted $\pi: \mathcal{G}_A \to \mathcal{G}_B$. Likewise, a permutation $\pi \in \mathcal{S}_n$ is termed an isomorphic mapping from $\mathcal{G}_A$ to $\mathcal{G}_A$ if it satisfies:

$$\left(v_{\pi(i)}, v_{\pi(j)}\right) \in \mathcal{E}_A \Leftrightarrow \left(v_i, v_j\right) \in \mathcal{E}_A$$

It is emphasised here that the double right arrow "⇒" will be used to denote implication of the right statement by the left., which is distinct from the single left-right arrow "→" used to denote function mappings.

Definition. "Analogy" of the ith vertex in $\mathcal{G}_A$ to the jth vertex in $\mathcal{G}_B$, implies that there exists some $\hat{\pi} \in \mathcal{S}_n$ such that $\hat{\pi}: \mathcal{G}_A \to \mathcal{G}_B$ and $\hat{\pi}(i) = j$. This is denoted $\mathcal{G}_A(v_i) \leftrightarrow \mathcal{G}_B(v_j)$. Likewise, $\mathcal{G}_A(v_i) \leftrightarrow \mathcal{G}_A(v_j)$ implies that there exists some $\hat{\pi} \in \mathcal{S}_n$ such that $\hat{\pi}: \mathcal{G}_A \to \mathcal{G}_A$ and $\hat{\pi}(i) = j$. It is emphasised here that the double left-right arrow "⇔" will be used to denote equivalence of statements, which is distinct from the single left-right arrow "↔" used to denote analogy.

Definition. A "unique analogy" of the ith vertex in $\mathcal{G}_A$ to the jth vertex in $\mathcal{G}_B$ implies that:

If $x = i$ and $y = j$ then:

$$\mathcal{G}_A(v_x) \leftrightarrow \mathcal{G}_B(v_y)$$

Otherwise:

$$\mathcal{G}_A(v_x) \nleftrightarrow \mathcal{G}_B(v_y)$$

Unique analogy of the ith vertex in $\mathcal{G}_A$ to the jth vertex in $\mathcal{G}_B$ is denoted $\mathcal{G}_A(v_i) \dot{\leftrightarrow} \mathcal{G}_B(v_j)$.

Definition. The "automorphism group" of $\mathcal{G}_A$, denoted $\mathrm{Aut}(\mathcal{G}_A)$ is the set of all graphs $\mathcal{G}_B$ satisfying $\mathcal{G}_B \cong \mathcal{G}_A$.

Definition. A "canonical labelling function" is a function mapping $\mathbb{R}^{n \times n} \to S_n$ such that for any $\mathcal{G}_B \in \mathrm{Aut}(\mathcal{G}_A)$, labellings γ_A and γ_B satisfy:

$$\left(v_{\gamma_A(i)}, v_{\gamma_A(j)}\right) \in \mathcal{E}_A \Leftrightarrow \left(v_{\gamma_B(i)}, v_{\gamma_B(j)}\right) \in \mathcal{E}_B$$

Let $\lambda_A = \left[\lambda_A^{(1)}, \dots, \lambda_A^{(n)}\right]$ denote the eigenvalues of $A \in \mathbb{R}^{n\times n}$ such that $\lambda_A^{(1)} \geq \dots \geq \lambda_A^{(n)}$. Let $\Lambda_A \in \mathbb{R}^{n\times n}$ denote $\mathrm{diag}(\lambda_A)$.

Let $\dot{\lambda}_A = \left[\dot{\lambda}_A^{(1)}, \dots, \dot{\lambda}_A^{(m)}\right]$ denote the $m \leq n$ distinct eigenvalues of A, such that $\dot{\lambda}_A^{(1)} > \dots > \dot{\lambda}_A^{(m)}$.

Let $\mu_A = \left[\mu_A^{(1)}, \dots, \mu_A^{(m)}\right]$ denote the eigenvalue multiplicities of A, such that:

$$\sum_{k=1}^{m} \mu_A^{(k)} = n$$

Let $V_A \in \mathbb{R}^{n\times n}$ be an eigenbasis of A. Let V_A be partitioned by eigenvalue, such that $V_A^{(k)} = \left[V_A^{(1)}, \dots, V_A^{(m)}\right]$, where $V_A^{(k)} \in \mathbb{R}^{n\times \mu_A^{(k)}}$.

Definition. Let $E_A^{(k)} \in \mathbb{R}^{n\times n}$ be the orthogonal projection of A onto its kth eigenspace, such that:

$$E_A^{(k)} = V_A^{(k)} \left(V_A^{(k)}\right)^T$$

$E_A \in \mathbb{R}^{n\times n\times m}$ is termed the "tensor of eigenspace projections", such that $E_A = \left[E_A^{(1)}, \dots, E_A^{(m)}\right]$. Thus, we have:

$$A = \sum_{k=1}^{m} \dot{\lambda}_A^{(k)} E_A^{(k)}$$

Definition. Let $E_A^{(k)}(v_i) \in \mathbb{R}^n$ denote the ith column of $E_A^{(k)}$, representing the projection of the ith vertex onto the kth eigenspace. The matrix of a vertex's projections onto each eigenspace is termed the "vertex eigenprojection". $E_A(v_i) \in \mathbb{R}^{n\times m}$ denotes the ith vertex eigenprojection in A, such that:

$$E_A(v_i) = \left[E_A^{(1)}(v_i), \dots, E_A^{(m)}(v_i)\right]$$

Definition. For some matrix $X \in \mathbb{R}^{n\times n}$, let $Q_X \in \mathcal{P}_n$ be a row-permutation matrix satisfying:

$$(Q_X X)_{ik} < (Q_X X)_{jk} \; \forall \{i,j,k,l \in \mathbb{N}_{\leq n} | i < j, k < l, (Q_X X)_{il} > (Q_X X)_{jl}\}$$

Thus, pre-multiplication by Q_X orders rows in ascension by the first column, then the second , then the third etc, resolving ties in former columns by elements of latter columns. It is noted that Q_X may have multiple solutions if some rows X are identical. Let $q_X \in \mathcal{S}_n$ denote the permutation vector corresponding to Q_X.

Definition. Let $s: \mathbb{R}^{n\times m} \to \mathbb{R}^{n\times m}$ denote the row ordering function $s(X) = Q_X X$.

Definition. The row-permutation of a vertex eigenprojection under s is termed the "ordered vertex eigenprojection". The ith ordered vertex eigenprojection in A is given by $s\big(E_A(v_i)\big)$.

Definition. Let $H_A \in \mathbb{R}^{n\times nm}$ denote the matrix of ordered vertex eigenprojections in vectorised form, such that the ith row, $H_A(v_i) \in \mathbb{R}^{nm}$, is given by:

$$H_A(v_i) = \mathrm{vec}\Big(s\big(E_A(v_i)\big)\Big)$$

Let I_n denote the n-dimensional identity matrix. Let 1_n denote an $n \times 1$ column of ones and let $1_{n\times n}$ denote a matrix of ones.

Definition. Let $K(A,x) \in \mathbb{R}^{n\times(n+1)}$ denote the Krylov matrix (Dax, 2017) generated by the product of $x \in \mathbb{R}^n$ and exponents of $A \in \mathbb{R}^{n\times n}$ such that

$$K(A,x) = [A^0 x \quad \cdots \quad A^n x]$$

Algorithm Overview

This study presents an exact method to identify analogous vertices in isomorphic graphs, through modification of Klus' algorithm. It is demonstrated that the ordered eigenprojections of analogous vertices are equal. Furthermore, it is demonstrated that equality of ordered vertex eigenprojections implies analogy of vertices. These results may be exploited to test analogy of any vertex pair. An isomorphism-preserving perturbation strategy is introduced, in which pairs of analogous vertices are perturbed with characteristically weighted self-loops, inducing uniqueness of their analogy. The

steps of analogy-testing and perturbation may be iterated until all analogies in the perturbed graphs are unique. It is demonstrated that fewer than n iterations are required to infer an isomorphic mapping in this way. Thus, isomorphic mappings may be extracted in $\mathcal{O}(n^4)$ time. Two algorithms are presented to infer mappings between isomorphic graph tuples. In the first algorithm, graphs are perturbed simultaneously. In the second algorithm, the procedure is optimised by iteratively generating canonical orderings for each graph, reducing the number of vertex analogy tests required.

Analogy Testing

When searching for an isomorphic mapping between two graphs, it is appropriate to presume graphs are isomorphic, and test this assumption when a candidate isomorphic mapping has been identified. Accordingly, we let $\mathcal{G}_A \cong \mathcal{G}_B$. **Lemma 1** demonstrates that E_A is invariant to choice of eigenbasis, even when spectra are degenerate.

Lemma 1. If $B = PAP^T$ then $E_B^{(k)} = PE_A^{(k)}P^T$ for all $k \in \{1, \dots, m\}$.

Proof

We have:

$$V_A^T A V_A = V_B^T B V_B = \Lambda$$

As Λ is diagonal, it may be decomposed by eigenspace, such that:

$$\left(V_A{}^{(k)}\right)^T A V_A^{(k)} = \left(V_B{}^{(k)}\right)^T B V_B^{(k)} = \dot{\lambda}^{(k)} I_{\mu^{(k)}}$$

Substituting $B = PAP^T$, we have

$$\left(V_A{}^{(k)}\right)^T A V_A = \left(P^T V_B{}^{(k)}\right)^T A \left(P^T V_B{}^{(k)}\right)$$

Removing A, we have:

$$P\left(V_A{}^{(k)}\right)^T V_A P^T = V_B{}^{(k)} \left(V_B{}^{(k)}\right)^T$$

Substituting the definition of $E_A^{(k)}$ and $E_B^{(k)}$, we have:

$$PE_A^{(k)}P^T = E_B^{(k)}$$

End of lemma 1

Hence, the projections $E_A^{(k)}$ and $E_B^{(k)}$ are related by the same isomorphic mappings as $\mathcal{G}_A$ and $\mathcal{G}_B$, such that:

$$B = \hat{P}A\hat{P}^T \Leftrightarrow E_B^{(k)} = \hat{P}E_A^{(k)}\hat{P}^T$$

We have $E_B^{(k)} = PE_A^{(k)}P^T$ for all $k \in \{1, \dots, m\}$. Thus, we have:

$$E_B^{(k)}(v_i) = PE_A^{(k)}P^T(v_i)$$

$$E_B^{(k)}P(v_i) = PE_A^{(k)}(v_i)$$

$$E_B^{(k)}\left(v_{\pi(i)}\right) = PE_B^{(k)}(v_i)$$

Likewise, we have:

$$E_B\left(v_{\pi(i)}\right) = PE_A(v_i)$$

Lemma 2 demonstrates that the eigenprojection matrices of analogous vertices are related by a row-permutation.

Lemma 2. $PE_A(v_i) = E_B\left(v_{\pi(i)}\right)$.

Proof

From Lemma 1 we have:

$$PE_A^{(k)}P^T = E_B^{(k)}$$

Extracting v_i from each eigenspace we have:

$$PE_A^{(k)}P^T(v_i) = E_B^{(k)}(v_i)$$

$$PE_A^{(k)}(v_i) = E_B^{(k)}P(v_i)$$

Substituting the vertex permutation, we have:

$$PE_A^{(k)}(v_i) = E_B^{(k)}\left(v_{\pi(i)}\right)$$

Applying to each eigenspace $k \in \{1, \dots, m\}$:

$$PE_A(v_i) = E_B\left(v_{\pi(i)}\right)$$

End of lemma 2

The linearly permuted relationship between analogous vertices' eigenprojections allows reframing of the analogy detection task as a linear assignment problem. However, as P may be underdefined, it may be not deducible from this equation alone. This issue may be circumvented by applying the row-sorting function s to the vertex eigenprojections, neutralising the row-permutation effect of P. **Lemma 3** demonstrates that s induces equality between any two row-permutations of a matrix.

Lemma 3. $s(X) = s(PX)$ for all $X \in \mathbb{R}^{\{n \times m\}}$ and $P \in \mathcal{P}_n$.

Proof

We have:

$$(Q_X X)_{ik} < (Q_X X)_{jk} \; \forall \left\{i, j, k, l \in \mathbb{N}_{\leq n} \middle| i < j, k < l, (Q_X X)_{il} > (Q_X X)_{jl}\right\}$$

Multiplying internally by $P^T P$, we have:

$$(Q_X P^T PX)_{ik} < (Q_X P^T PX)_{jk} \; \forall \left\{i, j, k, l \in \mathbb{N}_{\leq n} \middle| i < j, k < l, (Q_X P^T PX)_{il} > (Q_X P^T PX)_{jl}\right\}$$

Factorising by $(Q_X P^T)$, it is evident that $Q_X P^T$ is a solution to Q_{PX}. Letting $Q_{PX} = Q_X P^T$, and multiplying from the right by PX, we have:

$$Q_{PX} PX = Q_X P^T PX$$

Thus:

$$Q_{PX}PX = Q_X X$$

Substituting for the definitions of $s(PX)$ and $s(X)$, we have:

$$s(PX) = s(X)$$

End of lemma 3

As analogous vertices eigenprojections are row-permutations of one another, s induces equality between them. Applying s to a vertex eigenprojection $E_A(v_i)$, we attain the "ordered vertex eigenprojection", $s\big(E_A(v_i)\big)$. **Lemma 4a** demonstrates that analogous vertices have equal ordered eigenprojections, such that:

$$\mathcal{G}_A(v_i) \leftrightarrow \mathcal{G}_B(v_j) \Rightarrow\ s\big(E_A(v_i)\big) = s\left(E_B(v_j)\right)$$

Lemma 4a. $s\big(E_A(v_i)\big) = s\left(E_B\big(v_{\pi(i)}\big)\right)$.

Proof

From Lemma 2 we have:

$$PE_A(v_i) = E_B\big(v_{\pi(i)}\big)$$

Therefore

$$s\big(PE_A(v_i)\big) = s\left(E_B\big(v_{\pi(i)}\big)\right)$$

By lemma 3 we may substitute $s\big(PE_A(v_i)\big)$ for $s\big(E_A(v_i)\big)$.

Thus

$$s\big(E_A(v_i)\big) = s\left(E_B\big(v_{\pi(i)}\big)\right)$$

End of lemma 4a

Lemma 4b demonstrates that if $\mathcal{G}_A \cong \mathcal{G}_B$, then equality of ordered vertex eigenprojections implies analogy of the corresponding vertices, such that:

$$s\big(E_A(v_i)\big) = s\left(E_B\big(v_j\big)\right) \Rightarrow \mathcal{G}_A(v_i) \leftrightarrow \mathcal{G}_B\big(v_j\big)$$

Lemma 4b. If $\mathcal{G}_A \cong \mathcal{G}_B$ and $s\big(E_A(v_i)\big) = s\left(E_B\big(v_j\big)\right)$ then $\mathcal{G}_A(v_i) \leftrightarrow \mathcal{G}_B\big(v_j\big)$.

Proof

By lemma 3, there exists some $\hat{P} \in \mathcal{P}_n$ such that:

$$\hat{P}E_A(v_i) = E_B\big(v_j\big)$$

For every $k \in \{1, \dots, m\}$ we have:

$$\hat{P}E_A^{(k)}(v_i) = E_B^{(k)}\big(v_j\big)$$

Substituting (v_i) for $\hat{P}^T\big(v_{\hat{\pi}(i)}\big)$, we have:

$$\hat{P}E_A^{(k)}\hat{P}^T\big(v_{\hat{\pi}(i)}\big) = E_B^{(k)}\big(v_j\big)$$

Therefore, for all $l \in \mathbb{R}$ we have

$$\big(\lambda^{(k)}\big)^l\hat{P}E_A^{(k)}\hat{P}^T\big(v_{\hat{\pi}(i)}\big) = \big(\lambda^{(k)}\big)^l E_B^{(k)}\big(v_j\big)$$

As $A^l = V_A\Lambda^l(V_A)^T$, we have:

$$\hat{P}A^l\hat{P}^T\big(v_{\hat{\pi}(i)}\big) = \sum_{k=1}^{m}\big(\lambda^{(k)}\big)^l\hat{P}E_A^{(k)}\hat{P}^T\big(v_{\hat{\pi}(i)}\big)$$

Likewise:

$$B^l(v_j) = \sum_{k=1}^{m}\big(\lambda^{(k)}\big)^l E_B^{(k)}\big(v_j\big)$$

Thus, we have $\hat{P}A^l\hat{P}^T(v_{\hat{\pi}(i)}) = B^l(v_j)$ for all $l \in \mathbb{R}$. Equal Krylov matrices $K\left(\hat{P}A\hat{P}^T, \hat{P}A\hat{P}^T(v_{\hat{\pi}(i)})\right)$ and $K\left(B, B(v_j)\right)$ may now be generated such that:

$$\left[\hat{P}A^0\hat{P}^T(v_{\hat{\pi}(i)}) \quad \cdots \quad \hat{P}A^n\hat{P}^T(v_{\hat{\pi}(i)})\right] = \left[B^0(v_j) \quad \cdots \quad B^n(v_j)\right]$$

It is noted that these Krylov matrices may be rank deficient if $m < n$ (Dax, 2017). Deleting the final column, we have:

$$\left[\hat{P}A^0\hat{P}^T(v_{\hat{\pi}(i)}) \quad \cdots \quad \hat{P}A^{n-1}\hat{P}^T(v_{\hat{\pi}(i)})\right] = \left[B^0(v_j) \quad \cdots \quad B^{n-1}(v_j)\right]$$

Likewise, deleting the first column we have:

$$\left[\hat{P}A^1\hat{P}^T(v_{\hat{\pi}(i)}) \quad \cdots \quad \hat{P}A^n\hat{P}^T(v_{\hat{\pi}(i)})\right] = \left[B^1(v_j) \quad \cdots \quad B^n(v_j)\right]$$

Factorising, we have

$$\hat{P}A\hat{P}^T\left[\hat{P}A^0\hat{P}^T(v_{\hat{\pi}(i)}) \quad \cdots \quad \hat{P}A^{n-1}\hat{P}^T(v_{\hat{\pi}(i)})\right] = B\left[B^0(v_j) \quad \cdots \quad B^{n-1}(v_j)\right]$$

The remaining Krylov submatrices are observed to be equal. Deleting them we observe that $\hat{P}A\hat{P}^T$ is a solution for B. Thus, there exists some $\hat{\pi}$ satisfying $\hat{\pi}: \mathcal{G}_A \to \mathcal{G}_B$ if $\hat{\pi}(i) = j$, fulfilling the conditions required to demonstrate that $\mathcal{G}_A(v_i) \leftrightarrow \mathcal{G}_B(v_j)$. It is noted that the solution to $\hat{P}$ may not be unique, as the Krylov matrices may be rank deficient.

End of lemma 4b

Thus, equality of $s(E_A(v_i))$ and $s\left(E_B(v_j)\right)$ determines analogy of the corresponding vertices, such that:

$$s(E_A(v_i)) = s\left(E_B(v_{\pi(i)})\right) \Leftrightarrow \mathcal{G}_A(v_i) \leftrightarrow \mathcal{G}_B(v_i)$$

This result implies that the row-ordered vertex eigenprojections may be used to identify analogous vertex pairs, without inferring P. Thus, the vertex analogy testing problem is linearised exactly, providing a polynomial time route to extract an isomorphic mapping $\hat{\pi}: \mathcal{G}_A \to \mathcal{G}_B$.

At this point, a critical distinction from Klus' method must be noted (Klus and Sahai, 2018). Klus sorted the columns of $E_A(v_i)$ separately, such that the ordered vertex eigenprojection was given by:

$$s_{(Klus)}\left(E_A(v_i)\right) := \left[s\left(E_A^{(1)}(v_i)\right), \dots, s\left(E_A^{(k)}(v_i)\right)\right]$$

Consequently, different permutations may be applied to each column, so that the vertex eigenprojection and its ordered image may not be related by a linear permutation. Clearly, Klus' ordering transformation maps analogous vertices to equal ordered vertex eigenprojections, such that:

$$\mathcal{G}_A(v_i) \leftrightarrow \mathcal{G}_B(v_i) \Rightarrow s_{(Klus)}\left(E_A(v_i)\right) = s_{(Klus)}\left(E_B\left(v_{\pi(i)}\right)\right)$$

However, the converse does not hold, as Klus' method may also map some non-analogous vertices to the same ordered vertex eigenprojection if the columns of their eigenprojections can be sorted equally. Therefore:

$$s_{(Klus)}\left(E_A(v_i)\right) = s_{(Klus)}\left(E_B\left(v_{\pi(i)}\right)\right) \not\Rightarrow \mathcal{G}_A(v_i) \leftrightarrow \mathcal{G}_B(v_i)$$

We now return to the description of our method. As $H_A(v_i) = vec\left(s\left(E_A(v_i)\right)\right)$, equality of $H_A(v_i)$ and $H_B(v_j)$ may also be employed to test analogy of $\mathcal{G}_A(v_i)$ and $\mathcal{G}_B(v_j)$. By applying a second row-ordering transformation to H_A and H_B it is observed that they are brought into a matrix form, $s(H_A)$, which is invariant to graph permutation. **Lemma 5** demonstrates that $s(H_A)$ is a permutation invariant representation of $\mathrm{Aut}(\mathcal{G}_A)$.

Lemma 5. $\mathcal{G}_A \cong \mathcal{G}_B \Rightarrow s(H_A) = s(H_B)$.

From lemmas 4a and 4b we have:

$$s\left(E_A(v_i)\right) = s\left(E_B(v_j)\right) \Leftrightarrow \mathcal{G}_A(v_i) \leftrightarrow \mathcal{G}_B(v_j)$$

As $H_A(v_i) = \mathrm{vec}\left(s\left(E_A(v_i)\right)\right)$ and $H_B(v_j) = \mathrm{vec}\left(s\left(E_B(v_j)\right)\right)$, we have:

$$H_A(v_i) = H_B(v_j) \Leftrightarrow \mathcal{G}_A(v_i) \leftrightarrow \mathcal{G}_B(v_j)$$

Thus

$$H_A(v_i) = H_B(v_{\pi(i)})$$

Therefore $PH_A = H_B$ if $\mathcal{G}_A \cong \mathcal{G}_B$. From lemma 3 we have $s(PH_A) = s(H_A)$. Therefore

$$\mathcal{G}_A \cong \mathcal{G}_B \Rightarrow s(H_A) = s(H_B)$$

Therefore, application of the row-ordering function s to H_A returns a such that $s(H_A) = s(H_B)$ if $\mathcal{G}_A \cong \mathcal{G}_B$.

End of lemma 5

Constructing the isomorphic mapping

Now that individual vertex analogies may be identified, the problem of constructing the isomorphic mapping from individual analogies is addressed. A precursor to the permutation matrix, $\check{P} \in \{0,1\}^{n\times n}$, may be generated by testing each vertex pair for analogy, such that:

$$\check{p}_{ji} \coloneqq \mathbb{I}\left(H_A(v_i) = H_B(v_j)\right)$$

Equivalently,

$$\check{p}_{ji} \coloneqq \mathbb{I}\left(\mathcal{G}_A(v_i) \leftrightarrow \mathcal{G}_B(v_j)\right)$$

If the solution to π is unique, each vertex has only one analogy and $\check{P} = P$. However, if multiple isomorphisms exist, some vertices will have multiple analogies and $\check{P} \notin \mathcal{P}_n$. To resolve this issue, analogy-preserving perturbations must be applied to A and B to induce uniqueness to the isomorphic mapping. Let $A^*, B^* \in \mathbb{R}^{n\times n}$ denote the perturbed matrices. Let $\mathcal{G}_A(v_i) \leftrightarrow \mathcal{G}_B(v_j)$. A and B may be perturbed at a_{ii} and b_{jj} to generate isomorphic matrices A^* and B^* with a unique isomorphism $\mathcal{G}_{A^*}(v_i) \dot{\leftrightarrow} \mathcal{G}_{B^*}(v_j)$.

Let $W^{(x)} \in \mathbb{R}^{n\times n}$ such that:

$$w_{ij}^{(x)} \coloneqq \mathbb{I}(i = j = x)$$

The perturbed matrices A^* and B^* are declared such that:

$$A^* \coloneqq A + W^{(i)} \cdot (1 + \max(A) - a_{ii})$$

$$B^* \coloneqq B + W^{(j)} \cdot \left(1 + \max(B) - b_{jj}\right)$$

Thus:

$$a_{ii}^* = 1 + \max(A)$$

$$b_{jj}^* = 1 + \max(B)$$

Thus, characteristically weighted self-loops are added to analogous vertices, thereby inducing a unique analogy between them. The perturbation scheme is depicted graphically in **Figure 1**.

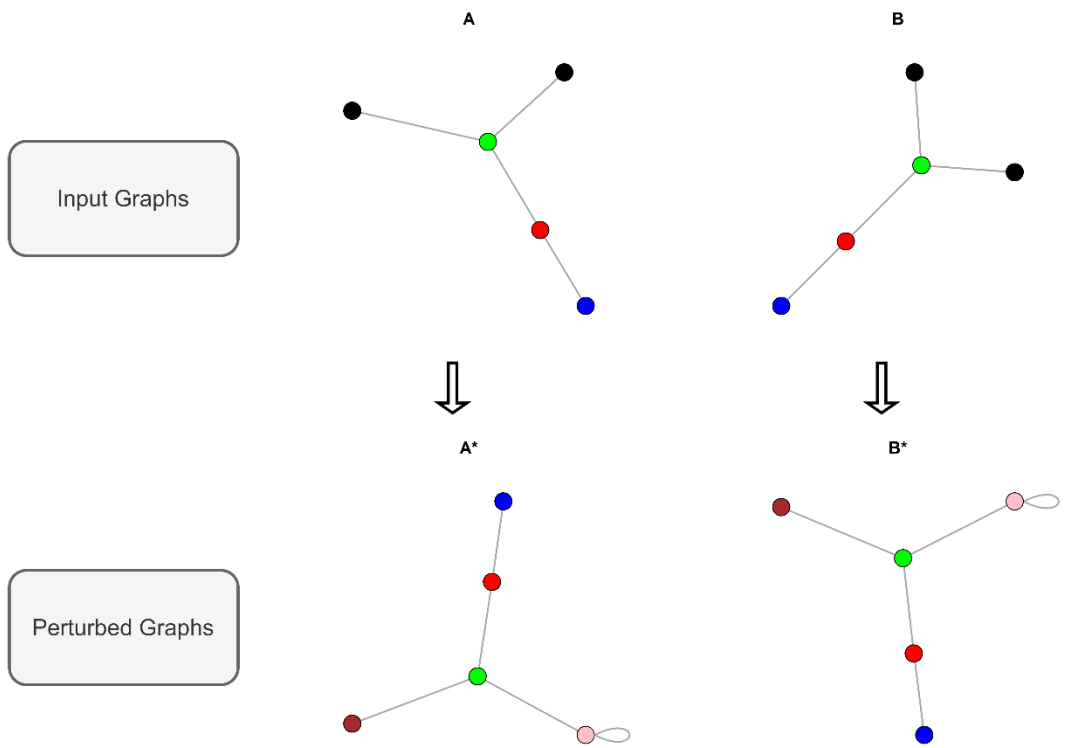

Figure 1. Graphical illustration of analogy testing and perturbation in an isomorphic tuple of degenerate graphs. A and B represent input graphs, and vertex colours indicate their analogy to one another. As multiple analogies exist between the black vertices, perturbation is required. A* and B* demonstrate the perturbed graphs in which self-loops have been added to one of the black vertices in each graph, removing any instances of multiple analogy.

Lemma 6 demonstrates that perturbation of analogous vertices $\mathcal{G}_A(v_i)$ and $\mathcal{G}_B(v_j)$ induces a unique analogy $\mathcal{G}_{A^*}(v_i) \dot{\leftrightarrow} \mathcal{G}_{B^*}(v_j)$. Thus, if $\mathcal{G}_A(v_i)$ and $\mathcal{G}_B(v_j)$ had other analogies, these are broken in the perturbed graphs. Thus, perturbation may be targeted to break graph symmetries and remove degeneracy.

Lemma 6. Let $\mathcal{G}_A \cong \mathcal{G}_B$. Let A^* and B^* denote perturbations of A and B such that:

$$A^* \coloneqq A + W^{(i)} \cdot (1 + \max(A) - a_{ii})$$

$$B^* \coloneqq B + W^{(j)} \cdot \left(1 + \max(B) - b_{jj}\right)$$

If $v_i, v_j \in \mathcal{V}$ such that $\mathcal{G}_A(v_i) \leftrightarrow \mathcal{G}_B(v_j)$ then $\mathcal{G}_{A^*}(v_i) \dot{\leftrightarrow} \mathcal{G}_{B^*}(v_j)$

Proof

We have $\max(A) = \max(B)$. Hence:

$$\mathrm{diag}(A^*)_i = \mathrm{diag}(B^*)_j = \max(A) + 1$$

Hence, for all $v_x \in \mathcal{V} \setminus v_j$ we have:

$$\mathrm{diag}(A^*)_i > \mathrm{diag}(B^*)_x$$

Therefore $\hat{\pi}$ cannot satisfy $B^* = \hat{P}A^*\hat{P}^T$ unless $\hat{\pi}(i) = j$. Therefore $\mathcal{G}_{A^*}(v_i) \dot{\leftrightarrow} \mathcal{G}_{B^*}(v_j)$.

End of lemma 6

Lemma 7 demonstrates that if $\hat{\pi}$ satisfies the perturbed equality $B^* = \hat{P}A^*\hat{P}^T$ then it satisfies the original equality $B = \hat{P}A\hat{P}^T$. Thus, if $\hat{\pi}: \mathcal{G}_{A^*} \to \mathcal{G}_{B^*}$ then $\hat{\pi}: \mathcal{G}_A \to \mathcal{G}_B$. Consequently, the search for $\hat{\pi}$ may continue in the perturbed graphs.

Lemma 7. If $\mathcal{G}_{A^*}(v_i) \leftrightarrow \mathcal{G}_{B^*}(v_j)$ and $\hat{\pi}: \mathcal{G}_{A^*} \to \mathcal{G}_{B^*}$ then $\hat{\pi}: \mathcal{G}_A \to \mathcal{G}_B$.

We have:

$$B^* = B + W^{(j)} \cdot \left(1 + \max(B) - b_{jj}\right)$$

Factorising the left side of the equality we have:

$$\hat{P}A^*\hat{P}^T = B + W^{(j)} \cdot \left(1 + \max(B) - b_{jj}\right)$$

$$\hat{P}A\hat{P}^T + \hat{P}W^{(i)}\hat{P}^T \cdot \left(1 + \max(A) - a_{jj}\right) = B + W^{(j)} \cdot \left(1 + \max(B) - b_{jj}\right)$$

As $\mathcal{G}_{A^*}(v_i) \leftrightarrow \mathcal{G}_{B^*}(v_j)$ we can substitute $\hat{P}W^{(i)}\hat{P}^T$ for $W^{(j)}$, thus:

$$\hat{P}A\hat{P}^T + W^{(j)} \cdot \left(1 + \max(A) - a_{jj}\right) = B + W^{(j)}\left(1 + \max(B) - b_{jj}\right)$$

As $\max(A) = \max(B)$ and $a_{ii} = b_{jj}$, we have:

$$\hat{P}A\hat{P}^T = B$$

Therefore $\hat{\pi}: \mathcal{G}_A \to \mathcal{G}_B$.

End of lemma 7

Here, a subtle distinction from Klus' perturbation strategy is noted. Klus also perturbed graphs through the addition of self-loops, which were weighted $i \in \mathbb{N}$ at the ith iteration. This strategy does not guarantee induction of a unique analogy as identical self-loops may already exist on other vertices. By weighting the new self-loop as $(1 + \max(A))$ uniqueness is guaranteed.

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 A^* and B^*, such that:

$$\check{p}_{ji} \coloneqq \mathbb{I}\left(H_{A^*}(v_i) = H_{B^*}(v_j)\right)$$

If $\mathcal{G}_{A^*}$ and $\mathcal{G}_{B^*}$ remain degenerate, other non-uniquely analogous vertex pairs may be sought in the perturbed graphs. Thus, perturbations may be applied iteratively, such that:

$$A^{*\eta+1} \coloneqq A^{*\eta} + W^{(i)} \cdot \left(1 + \max(A^{*\eta}) - a_{ii}^{*\eta}\right)$$
$$B^{*\eta+1} \coloneqq B^{*\eta} + W^{(j)} \cdot \left(1 + \max(A^{*\eta}) - b_{jj}^{*\eta}\right)$$

where $A^{*\eta}$ and $B^{*\eta}$ denote the graphs after $\eta \in \mathbb{N}$ perturbation iterations and $v_i, v_j \in \mathcal{V}$ such that $\mathcal{G}_{A^{*\eta}}(v_i) \leftrightarrow \mathcal{G}_{A^{*\eta}}(v_j)$. Naively, analogies could be sought, and perturbations applied for each vertex in turn. After $n-1$ iterative perturbations, each entry on the perturbed matrix diagonals would be distinct, implying that any solution to $\hat{\pi}: \mathcal{G}_{A^*} \to \mathcal{G}_{B^*}$ would be unique. However, if some vertices already have unique analogies, these need not be perturbed. Thus, fewer than n iterations are required to induce a unique isomorphism between $\mathcal{G}_{A^{*\eta}}$ and $\mathcal{G}_{B^{*\eta}}$. This procedure constitutes **Algorithm 1**, a naïve approach which is included here to aid understanding.

Now the task of graph canonical labelling is addressed. From lemma 1 it is known that E_A is invariant to A. Likewise, for all $v_i \in \mathcal{V}$ the ordered vertex eigenprojections $H_A(v_i)$ remain invariant across the permutation group of $\mathcal{G}_A$. Thus, vertices $\mathcal{G}_A(v_i)$ and $\mathcal{G}_A(v_j)$ may be ordered canonically by $H_A(v_i)$ and $H_A(v_j)$ if $\mathcal{G}_A(v_i) \nleftrightarrow \mathcal{G}_A(v_j)$. However, analogous vertices cannot be ordered canonically in this manner, as $H_A(v_i) = H_A(v_j)$ iff $\mathcal{G}_A(v_i) \leftrightarrow \mathcal{G}_A(v_j)$. Therefore, using the ordered eigenprojections, vertices may only be ordered up to analogy. This result is consistent with our definition of analogy, which implies equivalence of analogous vertices under permutation. Hence, uniqueness of the canonical ordering requires that no analogous vertices exist within the graph. Thus, a perturbation strategy must be implemented to order vertices canonically. This is achieved by partially ordering

the vertices according to the row-ordering of H_A, then applying a perturbation to the first vertex in the ordering with an analogy. It is observed that, as analogous vertices are identified by identical rows in H_A, that any non-random row-ordering operation will assign adjacent order positions to analogous vertices. Thus, it is guaranteed that no analogy will be missed if only rows with adjacent order positions are tested. This procedure constitutes **Algorithm 2**, which dominates Algorithm 1 in terms of computational efficiency, as fewer analogy tests are required.

Algorithms

Algorithm 1

In algorithm 1, the ordered vertex eigenprojection matrices are extracted for both graphs and every vertex pair is compared. If vertices have multiple analogies, one of the non-uniquely analogous pairs is selected and a characteristic perturbation applied to both vertices. This concludes the first iteration. Ordered vertex eigenprojection matrices may now be extracted from the perturbed graphs and further perturbation are applied in this manner until no vertices remain with multiple analogies.

Algorithm 1 pseudocode

Inputs

Symmetric matrices $A, B \in \mathbb{R}^{\{n \times n\}}$

Outputs

$\hat{P} \in \mathcal{S}_n$ such that $B = \hat{P} A \hat{P}^T$

Algorithm

$P \coloneqq 1_{n \times n}$

$A^*, B^* \coloneqq A, B$

$MultipleAnalogies \coloneqq 1$

WHILE $(MultipleAnalogies = 1)$:

Compute E_{A^*}, E_{B^*}

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

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

$$\check{p}_{ji} \coloneqq \mathbb{I}\left(H_{A^*}(v_i) = H_{B^*}(v_j)\right)$$

$MultipleAnalogies \coloneqq \mathbb{I}\left(\max\left(\check{P}1\right) > 1\right)$

IF $(MultipleAnalogies = 1)$:

$i \coloneqq i | \left(\check{P}1\right)_i > 1$

$j \coloneqq j | \check{p}_{ji} = 1$

$a^*_{ii} \coloneqq \max(A^*) + 1$

$b^*_{jj} \coloneqq \max(B^*) + 1$

ELSE:

$\hat{P} \coloneqq \check{P}$

RETURN $\hat{P}$

Algorithm 2

In algorithm 2, canonical labellings are inferred for each graph separately. For each graph, the ordered vertex eigenprojections are extracted. Then, vertices themselves are ordered according to their ordered vertex eigenprojections. This results in a partial ordering, such that non-analogous pairs are ordered whilst analogous pairs remain unordered. Following the ordering, the first $n - 1$ vertices are tested for analogy with the subsequent vertex. The first vertex identified to have an

analogy is perturbed by adding a characteristically weighted self-loop. This concludes the first iteration. Ordered vertex eigenprojections are extracted now from the perturbed graph and further perturbations applied until no vertex analogies remain.

Algorithm 2 pseudocode

Inputs

Symmetric matrices $A, B \in \mathbb{R}^{\{n \times n\}}$

Outputs

$\gamma_A, \gamma_B \in \mathcal{S}_n$ satisfying $\left(v_{\gamma_A(i)}, v_{\gamma_A(j)}\right) \in \mathcal{E}_A \Leftrightarrow \left(v_{\gamma_B(i)}, v_{\gamma_B(j)}\right) \in \mathcal{E}_B$

Algorithm

$A^*, B^* \coloneqq A, B$

FOR X^* in $\{A^*, B^*\}$:

    $Analogy \coloneqq 1$

    WHILE $(Analogy = 1)$:

        Compute H_{X^*}

        $\check{\gamma}_X \coloneqq q_{(H_{X^*})}$

        FOR i in $\{1, \ldots n-1\}$:

            $j = \hat{\gamma}_X(i)$

            $k = \hat{\gamma}_X(i+1)$

            $Analogy \coloneqq \mathbb{I}\left(H_{X^*}(v_j) = H_{X^*}(v_k)\right)$

            IF $(Analogy = 1)$:

$x_{jj}^{*} \coloneqq \max(X^{*}) + 1$

BREAK

$\gamma_X \coloneqq \check{\gamma}_X$

RETURN γ_A, γ_B

Implementation

The performance of algorithm 2 was tested on real-word and synthetic graphs. Real-world graphs were extracted from a public repository of brain networks (Amunts *et al.*, 2013; Rossi and Ahmed, 2015). As regularly structured synthetic graphs present some of the most challenging problems for traditional algorithms (Babai *et al.*, 1982), a 100-vertex ring graph and Johnson graphs (Holton and Sheehan, 1993) with parameters (n=9, k=4) and (n=10, k=3) were also included. In each experiment, a permutation was applied to one of the graphs to generating a tuple of isomorphic graphs. The algorithm was subsequently applied to infer an isomorphic mapping. Computation was performed with R and RStudio using iGraph and abind libraries (R Core Team, 2021; RStudio Team, 2021; Csardi and Nepusz, 2006; Plate and Heiburger, 2016). Algorithm 2 was employed to infer isomorphic mappings. To minimise discrepancies between matrix eigenvalues due to numerical fuzz, the presumption of isomorphism was invoked, and eigenvalues of both matrices were averaged. This procedure is permissible if isomorphism is formally tested later through validation of the isomorphic mapping. A numerical tolerance of 10^{-10} was applied when testing equality of eigenvalues and eigenvectors, respectively. All code required to reproduce the findings of this study is provided at github.com/robertoshea/graph_isomorphism. The BLISS canonical labelling algorithm (Junttila and Kaski, 2007) was included for comparison, via iGraph. Experiments were conducted on a standard laptop with 16Gb RAM and an Intel Core i7 CPU.

Results

Algorithm 2 identified a correct isomorphic mapping in each experiment (**Table 1**). No experiment required more than n perturbations. The longest running experiment "bn_mouse_visual_cortex_2", with 193 vertices and 23 distinct eigenvalues, required 159 iterations over 245 seconds to infer an isomorphic mapping. The eigenprojection ordering algorithm inferred correct isomorphic mappings in each experiment. BLISS failed to infer correct mappings in any of the experiments involving degenerate graphs.

Table 1. Experiment characteristics and results.

Graph	**Vertices**	**Distinct eigenvalues**	**Edges**	**Algorithm**	**Correct solution**	**Iterations**	**Time** *(secs)*
cat mixed species brain 1	65	65	1139	Eigenprojection ordering	TRUE	1	0.25
cat mixed species brain 1	65	65	1139	BLISS	TRUE	NA	0
macaque rhesus brain 1	242	234	4090	Eigenprojection ordering	TRUE	9	47.62
macaque rhesus brain 1	242	234	4090	BLISS	FALSE	NA	0.01
macaque rhesus brain 2	91	23	628	Eigenprojection ordering	TRUE	33	5.8

macaque rhesus brain 2	91	23	628	BLISS	FALSE	NA	0
macaque rhesus cerebral cortex 1	91	59	1615	Eigenprojection ordering	TRUE	1	0.3
macaque rhesus cerebral cortex 1	91	59	1615	BLISS	TRUE	NA	0
macaque rhesus interareal cortical network 2	93	6	2667	Eigenprojection ordering	TRUE	91	18.55
macaque rhesus interareal cortical network 2	93	6	2667	BLISS	FALSE	NA	0.03
mouse brain 1	213	213	21807	Eigenprojection ordering	TRUE	1	4.86

mouse brain 1	213	213	21807	BLISS	TRUE	NA	0.02
mouse visual cortex 1	29	23	44	Eigenprojection ordering	TRUE	5	0.15
mouse visual cortex 1	29	23	44	BLISS	FALSE	NA	0
mouse visual cortex 2	193	23	214	Eigenprojection ordering	TRUE	159	267.29
mouse visual cortex 2	193	23	214	BLISS	FALSE	NA	0.01
ring 100	100	51	100	Eigenprojection ordering	TRUE	3	1.22
ring 100	100	51	100	BLISS	FALSE	NA	0
johnson 9 4	126	5	1260	Eigenprojection ordering	TRUE	7	0.95
johnson 9 4	126	5	1260	BLISS	FALSE	NA	0.01
johnson 10 3	120	4	1260	Eigenprojection ordering	TRUE	7	0.78
johnson 10 3	120	4	1260	BLISS	FALSE	NA	0

Table 1. Experiment characteristics and results. In each experiment a random permutation was applied to the input graph and each algorithm was applied to infer an isomorphic mappings. Algorithm 2 was employed for Eigenprojection Ordering. BLISS iterations were not provided by the implementation used.

Discussion

This analysis unifies the problems of graph canonisation and isomorphism testing via the tensor of eigenprojections. It is demonstrated that the ordered vertex eigenprojections provide a signature by which vertex analogy may be tested. By reshaping the ordered vertex eigenprojections into a matrix structure and applying a second ordering, a permutation invariant graph representation $s(H_A)$ is achieved. This result is consistent with the quadratic nature of the problem – that two ordering operations should be required to recover the equivalence broken by two permutation operations. It is conjectured that this representation is unique to $Aut(\mathcal{G}_A)$, and is therefore a canonical form. This hypothesis is supported by the observation that equality of $s(H_A)$ and $s(H_B)$ implies that every vertex in $\mathcal{G}_A$ has an analogy in $\mathcal{G}_B$ and vice versa. Correctness of this hypothesis would imply that $s(H_A)$ is invariant over $\mathrm{Aut}(\mathcal{G}_A)$ and unique to $\mathrm{Aut}(\mathcal{G}_A)$, fulfilling the requirements of a canonical representation. Therefore, if this conjecture was proven correct, isomorphism testing could proceed in $\mathcal{O}(n^3)$ time through equality testing of $s(H_A)$ and $s(H_B)$. Analysis of this conjecture is left for further research.

This study demonstrates that a graph with analogous vertices may be labelled canonically through repeated iterations of partial ordering and perturbation. As n such iterations may be required, this algorithm guarantees canonical vertex labelling in $\mathcal{O}(n^4)$ time. Algorithm 2 dominates algorithm 1, by ordering vertices such that analogous vertices are assigned adjacent positions in the ordering. Consequently, analogy tests are only required between vertices with adjacent positions in the ordering. Thus, $n-1$ analogy tests are required to demonstrate uniqueness of each vertex in algorithm 2, compared with n^2-n in algorithm 1.

The algorithms described in this paper assume that the graph is undirected. If the graph is directed, the adjacency matrix may be asymmetric and uniqueness of the tensor of eigenprojections is not guaranteed. The extension of these algorithms to directed graphs is left for further research. The algorithms do not require any spectral conditions, such as positive definiteness or bounded eigenvalue multiplicity. Where multiple isomorphic mappings exist, a single solution to $\hat{\pi}$ is returned, however the full set of solutions may also be computed by changing the order of perturbations.

As the algorithm tests equality of eigenvalues and elements of eigenprojections, numerical stability issues are expected (Klus and Sahai, 2018). This is a critical limitation of the above algorithms to practical graph isomorphism problems. In some cases, numerical stability may be achieved through appropriate rounding or numerical tolerance. However, errors are likely in large graphs with similar eigenvalues. To assess the suitability of the algorithm for a particular use case it is recommended to perform simulations by duplicating one of the graphs, permuting it and attempting to recover an isomorphic mapping. This procedure may also be used to optimise numerical tolerance settings.

The algorithm presented here demonstrates that any problem which may be reduced to a graph isomorphism problem may be solved in polynomial time. Further research will investigate developments to the algorithm to solve the subgraph isomorphism problem and the general case-quadratic assignment problem in polynomial time.

Acknowledgements

The author would like to acknowledge the advice of Reimer Kuhn and Stefan Izaak at the Department of Theoretical Physics, King's College London. The author would like to acknowledge the support of Vicky Goh and Gary Cook at the Department of Cancer Imaging, King's College London; and Sophia Tsoka at the Department of Informatics, King's College London.

Conflicts of Interest

The author has no conflicts of interest to declare.

Availability of Code and Materials

All code required to reproduce the results of this analysis is implemented in the R language and provided at github.com/robertoshea/graph_isomorphism.

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

Amunts,K. *et al.* (2013) BigBrain: An ultrahigh-resolution 3D human brain model. *Science (80-.).*, **340**, 1472–1475.

Arvind,V. and Torán,J. (2005) Isomorphism testing: Perspective and open problems. *Bull. Eur. Assoc. Theor. Comput. Sci.*, **86**, 66–84.

Babai,L. (2016) Graph isomorphism in quasipolynomial time: [Extended abstract]. *Proc. Annu. ACM Symp. Theory Comput.*, **19-21-June**, 684–697.

Babai,L. *et al.* (1982) Isomorphism of graphs with bounded eigenvalue multiplicity. *Proc. Annu. ACM Symp. Theory Comput.*, 310–324.

Csardi,G. and Nepusz,T. (2006) The igraph software package for complex network research. *InterJournal Complex Syst.*, **Complex Sy**, 1695.

Dax,A. (2017) The numerical rank of Krylov matrices. *Linear Algebra Appl.*, **528**, 185–205.

Grohe,M. and Schweitzer,P. (2020) The graph isomorphism problem. *Commun. ACM*, **63**, 128–134.

Helfgott,H.A. *et al.* (2017) Graph isomorphisms in quasi-polynomial time.

Holton,D.A. and Sheehan,J. (1993) The Petersen Graph. *The Petersen Graph*.

Junttila,T. and Kaski,P. (2007) Engineering an efficient canonical labeling tool for large and sparse graphs. *Proc. 9th Work. Algorithm Eng. Exp. 4th Work. Anal. Algorithms Comb.*, 135–149.

Karp,R.M. (1972) Reducibility among Combinatorial Problems.

Klus,S. and Sahai,T. (2018) A spectral assignment approach for the graph isomorphism problem. *Inf. Inference*, **7**, 689–706.

Luks,E.M. (1982) Isomorphism of graphs of bounded valence can be tested in polynomial time. *J. Comput. Syst. Sci.*, **25**, 42–65.

Plate,T. and Heiburger,R. (2016) abind: Combine Multidimensional Arrays.

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

Rossi,R.A. and Ahmed,N.K. (2015) The network data repository with interactive graph analytics and visualization. *Proc. Natl. Conf. Artif. Intell.*, **6**, 4292–4293.

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

Schöning,U. (1988) Graph isomorphism is in the low hierarchy. *J. Comput. Syst. Sci.*, **37**, 312–323.