UNIVERSITY OF CALIFORNIA SAN DIEGO

Exploiting Network Boundaries for Spectral Clustering and Tensor Network Generative Modeling

A dissertation submitted in partial satisfaction of the
requirements for the degree Doctor of Philosophy

in

Physics

by

Sami Joseph Ortoleva

2023

EPIGRAPH

You talk when you cease to be at peace with your thoughts;
And when you can no longer dwell in the solitude of your heart
you live in your lips, and sound is a diversion and a pastime.
And in much of your talking, thinking is half murdered.

Gibran Kahlil Gibran

The real cycle you're working on is a cycle called yourself.
The machine that appears to be "out there" and the person
that appears to be "in here" are not two separate things.
They grow toward Quality or fall away from Quality together.

Robert M. Pirsig

TABLE OF CONTENTS

LIST OF FIGURES

ACKNOWLEDGEMENTS

Chapters 1 and 2, in part, are currently being prepared for submission for publication of the material. The dissertation author was the primary investigator and author of this material. Professor Alexander Cloninger is the sole co-author of the material being prepared for submission for publication.

My thanks to Professor Alexander Cloninger, whose kind and persistent support over the past several difficult years is the reason this work was completed.

I would also like to acknowledge the diverse and ever-supportive members of my chosen family, among them Shaun, Rob, Barbara, Cliff, Leilani, Fabian, Adam, Ashley, Chuck, Ben, Mejgan, John, Bill, Chris, Jay, Brandon, Eli, Jeff, Todd, Nelson, Thomas, Drew, and Jarrod. They have aided me in ways too many to number throughout the years-long process of completing this work.

Thank you to to both my Palestinian family, and to the Palestinian people at large, for teaching me so much about resilience, perseverance, and joy in the face of adversity.

And most of all, thank you to my mother, Amal, who raised me to be the man that I am. Would that I could share this day with you.

VITA

2011	Bachelor of Science in Electrical Engineering Purdue University
2014	Master of Science in Electrical Engineering University of California, Santa Barbara
2023	Doctor of Philosophy in Physics University of California San Diego

ABSTRACT OF THE DISSERTATION

Exploiting Network Boundaries for Spectral Clustering and Tensor Network Generative Modeling

by

Sami Joseph Ortoleva

Doctor of Philosophy in Physics

University of California San Diego, 2023

Professor Alexander Cloninger, Chair
Professor Yi-Zhuang You, Co-Chair

In many systems, both physical and mathematical, internal boundaries play an outsized role in dictating the large-scale behavior of the system. This theme is explored here in two parts: by identifying natural boundaries to diffusion on undirected graphs for the improvement of spectral graph methods for clustering and classification, and by imposing boundaries in two dimensional PEPS tensor networks to reduce their computational complexity for image generation. In the graph diffusion work, an algorithm is developed that identifies and removes vertices serving as bridges between well-connected clusters. With the use of this algorithm, the performance of unsupervised spectral clustering on graphs derived from synthetic point cloud data shows excellent cluster separation down to approximately two-thirds the point cloud gap size that standard spectral clustering alone tolerates. The same vertex-removal scheme applied to a diffusion-informed active learning classification algorithm shows an approximately order-of-magnitude best-case reduction in the classification error rate on benchmark hyperspectral imagery data in the low-label-query-limit versus the same active learning algorithm applied without vertex removal. The PEPS work proposes a scheme whereby such networks are cut into overlapping patch networks, whose individual contraction complexity is comparatively low. Feasibility testing is performed showing that in principle, this scheme could be used for image in-painting, and has intuitively appealing model features that are directly reflective of the data.

Chapter 1

Graph Clustering via Bridge Identification and Removal

Consider an undirected weighted graph $G = (V, E, w)$ with nonnegative weights $w : V \times V \to [0, +\infty)_{\mathbb{R}}$. If G is a disconnected graph made of $N_{\text{clusters}} \equiv N_C$ connected components, then random walks starting in one of the clusters will fully explore that cluster in the infinite time limit, and never jump to other clusters. Indeed, a well-known theorem of graph theory guarantees that there are N_C eigenvectors of the various graph Laplacians with eigenvalue 0 in this case (Chung 1997).

Now loosen the strong assumption of a disconnected graph slightly by adding weak connections between the N_C (formerly) connected components, which we now denote as *clusters*. Assuming the connections between clusters are weak enough, there will be intermediate times where the intra-cluster random walk properties are approximately those of the formerly isolated clusters. That is to say, at short to intermediate times, the behavior of random walks that begin in cluster i are approximately those of a random walk of the induced subgraph of cluster i alone.

This behavior can be used to identify clusters on graphs which are connected, but only weakly so. The field of spectral graph theory addresses precisely this question. While there is much work on spectral clustering on graphs, much of it focuses on examining the specific properties of eigenvectors of the graph Laplacian with small eigenvalues (the so-called *low-*

frequency[1], or *first* eigenvectors) and attempting to extract relevant information from these vectors (Schiebinger, Wainwright, and Yu 2015; X. Cheng and Mishne 2020; Nadler and Galun 2006).

In the sufficiently weak inter-cluster connection regime, spectral clustering is spectacularly successful. However, when there are stronger connections between clusters, or even if the inter-cluster connections are weak, but the cluster sizes are disparate, the information needed to separate clusters from one another can be buried in Laplacian eigenvectors with larger eigenvalues (Alexander Cloninger and Czaja 2015). This has led to a concerted effort to find ways to extract information from "deeper in the spectrum" of the graph Laplacian in order to be able to extend the regime of effectiveness of spectral clustering.

In this work, we propose using time-domain information from $|V|$ independent diffusion processes on an undirected graph $G = (V, E, w)$ to weaken the inter-cluster connections prior to employing techniques, such as spectral clustering, whose performance improves in the weak-inter-cluster-coupling limit. Despite being rooted in time-domain intuitions and definitions, the algorithm proposed herein is efficiently calculable from eigenvectors of the graph Laplacian itself, and so may be naturally incorporated into the analysis of any data set that is amenable to spectral clustering or other techniques which rely on at-least-partial eigendecompositions of graph Laplacians.

1.1 Score-based Clustering

Score-based methods are one way to identify vertices of interest within a larger graph. The general methodology is:

1. Design a *score function* $S_w : V \to \mathbb{R}$, which using only the weights w of the graph G, maps each vertex to a real number that will be large on the vertices of interest $v \in V_{\text{interest}}$, and small on all other vertices.

[1]This may be a misnomer, as argued by (Saito 2018).

2. Evaluate this function on every vertex of the graph G.

3. Identify some critical value $S_{\text{threshold}}$ which separates vertices of interest (where $S(v \in V_{\text{interest}}) > S_{\text{threshold}}$) from the rest of the graph (where $S(v \notin V_{\text{interest}}) < S_{\text{threshold}}$).

If our goal is to perform general clustering, this methodology suffers from a critical flaw, however: because it separates the vertices into only two subsets, it effectively tries to two-color the clusters of the graph (see Formanowicz and Tanaś 3912 for an overview of graph coloring). This is doomed to failure, as generically we cannot avoid distinct clusters that are directly connected both falling on the same side of our scoring threshold $S_{\text{threshold}}$. Even in the case of planar graphs, the celebrated four-color theorem shows that up to four colors are needed in general (Appel and Haken 1989).

Fortunately, the analogy with graph coloring (and by extension, map coloring) suggests a way to extend score-based methods to general clustering problems: rather than attempting to identify all the vertices in a cluster, we can instead identify vertices[2] that *connect* different clusters. Drawing an analogy to the map coloring problem, this is the equivalent of defining our score to identify the *borders* of countries, rather than their interiors.

1.2 The Utility of Boundary Vertex Identification

Consider a weighted graph $G = (V, E, w)$ with the vertex set partitioned into disjoint subsets as:

$$V = \left(\bigsqcup_{n=1}^{N_C} V_n \right) \sqcup V_B \tag{1.1}$$

where V_n is the set of vertices in cluster n, and V_B is the set of boundary vertices which connect the clusters to one another. G has an associated symmetric weight matrix $\underline{\underline{W}} \in \mathbb{R}^{|V| \times |V|}$ with

[2]The astute reader may object that it makes more sense to identify *edges* instead of *vertices* that connect clusters. In some contexts this is true. However, when using graph methods to perform data clustering/labeling, typically each vertex corresponds to a data point in the data set. There is then a natural notion of boundary *vertices* as data points which share some features of prototypical vertices from multiple clusters, and do not clearly belong to any one cluster.

non-negative elements $W_{ij} = w(v_i, v_j) \geq 0$. Noting that $\mathbb{R}^{|V|\times|V|} = \mathbb{R}^{|V|} \otimes \mathbb{R}^{|V|}$, we can split $\mathbb{R}^{|V|}$ analogously to the vertex set:

$$\mathbb{R}^{|V|} = \left(\bigoplus_{n=1}^{N_C} \mathbb{R}^{|V_n|} \right) \oplus \mathbb{R}^{|V_b|}$$

which induces a block structure on the weight matrix by ordering its elements according to the vertex partition:

$$\underline{\underline{W}}^G = \left[\begin{array}{ccc:c} \underline{\underline{W}}^{(1,1)} & \cdots & \underline{\underline{W}}^{(1,N_C)} & \underline{\underline{W}}^{(1,B)} \\ \vdots & \ddots & \vdots & \vdots \\ \underline{\underline{W}}^{(N_C,1)} & \cdots & \underline{\underline{W}}^{(N_C,N_C)} & \underline{\underline{W}}^{(N_C,B)} \\ \hdashline \underline{\underline{W}}^{(B,1)} & \cdots & \underline{\underline{W}}^{(B,N_C)} & \underline{\underline{W}}^{(B,B)} \end{array} \right]$$

where the elements of block (p,q) correspond to the weight function evaluated on a vertex from V_p and a vertex from V_q, i.e. $W^{(p,q)}_{*,*} = w(u,v)\big|_{u\in V_p,\, v\in V_q}$. Note that the symmetry of $\underline{\underline{W}}^G$ implies $\underline{\underline{W}}^{(B,n)} = \left(\underline{\underline{W}}^{(n,B)}\right)^{\dagger}$.

It is insightful to omit the off-diagonal cluster-to-cluster blocks, as vertices which strongly connect two different clusters would be identified as bridge vertices under any reasonable analysis. Denoting the diagonal blocks $\underline{\underline{W}}^{(n,n)} \equiv \underline{\underline{W}}^{(n)}$ for simplicity, the weight matrix becomes

$$\underline{\underline{W}}^G = \left[\begin{array}{ccc:c} \underline{\underline{W}}^{(1)} & & & \underline{\underline{W}}^{(1,B)} \\ & \ddots & & \vdots \\ & & \underline{\underline{W}}^{(N_C)} & \underline{\underline{W}}^{(N_C,B)} \\ \hdashline \underline{\underline{W}}^{(B,1)} & \cdots & \underline{\underline{W}}^{(B,N_C)} & \underline{\underline{W}}^{(B)} \end{array} \right]$$

This makes visually apparent the fact that, were we to define a subgraph without the boundary vertices, the clustering problem would become much simpler. Defining the induced subgraph with only vertices in the clusters, which we call the *volume* vertices, as

$$G_{\text{vol.}} \equiv G[V_1 \sqcup V_2 \sqcup \cdots V_{N_C}] = G[V \backslash V_B]$$

its associated weight matrix $W^{G_{\text{vol.}}}$ is:

$$\underline{\underline{W}}^{G_C} = \begin{bmatrix} \underline{\underline{W}}^{(1)} & & \\ & \ddots & \\ & & \underline{\underline{W}}^{(N_C)} \end{bmatrix}$$

With only (visually omitted) very weak connections between the clusters, $G_{\text{vol.}}$ is much easier to cluster by any traditional graph clustering method than the original graph G precisely because of the removal of the boundary vertices. Thus, our program becomes identifying the boundary vertices to enable this dramatic simplification.

After clustering is applied to this *pruned* subgraph G_C, the question of what to do with the pruned boundary vertices in V_B remains. This is ultimately context-dependent, as in some graphs, they may represent data points where the label is fundamentally uncertain. In cases where a label is expected on every data point, any approach to spreading known labels on part of a graph to labels on the full graph may be used, e.g. label diffusion (see section 2.3.2).

1.3 Designing a Boundary-Identifying Score: A Narrative

To identify boundary vertices (or *bridge* vertices; the two terms are interchangeable in this text), consider the following continuous-time process:

$$\partial_t \underline{f}^\dagger(t) = -\underline{f}^\dagger(t) \underline{\underline{L}}^{\text{r.w.}} \tag{1.2}$$

where $\underline{f}^\dagger \in \left(\mathbb{R}^{|V|}\right)^\dagger$ is a row vector, $\left(\mathbb{R}^{|V|}\right)^\dagger$ is the dual vector space to $\mathbb{R}^{|V|}$, and $\underline{\underline{L}}^{\text{r.w.}}$ is the random walk-normalized graph Laplacian of the graph $G = (V, E, w)$:

$$\underline{\underline{L}}^{\text{r.w.}} \equiv \underline{\underline{\mathbb{I}}} - \underline{\underline{\bcancel{d}}}^{-1}\underline{\underline{W}} \qquad [\underline{d}]_i \equiv \sum_{j=1}^{|V|} W_{ij} \qquad [\underline{\underline{\bcancel{d}}}]_{ij} = \delta_{ij} d_i, \tag{1.3}$$

where for any vector $\underline{v} \in \mathbb{R}^n$, $\underline{\underline{\bcancel{v}}} \in \mathbb{R}^{n \times n}$ is diagonal square matrix with the elements of $\underline{v}$ on the main diagonal[3]. Throughout this document, $\underline{\underline{L}}$ without qualification should be assumed to refer to this random walk-normalized Laplacian unless specifically stated otherwise. See Chung 1997 for a classic overview of spectral graph theory, or Liu and Han 2018 for a contemporary overview including the various normalizations of the graph Laplacian.

Since $\underline{\underline{L}}$ is independent of t, the formal solution to (1.2) is

$$\underline{f}^\dagger(t) = \underline{f}^\dagger(0) e^{-\underline{\underline{L}}t} \tag{1.4}$$

which is well-behaved for all $t \geq 0$ due to the eigenvalues of $\underline{\underline{L}}$ being non-negative. In particular, if we wish to consider what the probability of a random walk starting at vertex i returning back to that same vertex after time t, the relevant quantity is:

$$R_i(t) \equiv \underline{e}^{(i)^\dagger} \cdot e^{-\underline{\underline{L}}t} \cdot \underline{e}^{(i)} = \left[e^{-\underline{\underline{L}}t}\right]_{ii} \tag{1.5}$$

where $\underline{e}^{(i)}$ is the ith standard basis vector. We shall henceforth refer to $\underline{R}(t)$ as the *return probability*.

The qualitative behavior of the return probability $\underline{R}(t)$ for connected[4] graphs with strong intracluster connections, but weak intercluster connections can be deduced by recognizing that each element of $\underline{R}(t)$ corresponds to the probability that a random walk *started on* vertex v_i

[3]The direction of the slash for this vector-to-diagonal-matrix notation is intended to be evocative of the main diagonal of the matrix.

[4]For clustering tasks, we may assume that G is connected. If G is not connected, we can split it into its connected components in linear time in the size of the graph and cluster each connected component separately (Pearce 2005).

returns to that same vertex after time t, and thus each element corresponds to an *independent* diffusion process on the same graph. Since this graph is connected, the diffusion process has a unique infinite time limit[5], the so-called *equilibrium distribution* $\underline{\pi}$, proportional to the degree vector:

$$\underline{\pi} \equiv \frac{\underline{d}}{\underline{\mathbb{1}}^\dagger \underline{d}} \qquad \underline{\pi}^\dagger \underline{\underline{L}}^{\text{r.w.}} = \underline{0}^\dagger \qquad \forall \underline{v}^\dagger \in \left(\mathbb{R}^{|V|}\right)^\dagger, \quad \underline{v}^\dagger e^{-t\underline{\underline{L}}^{\text{r.w.}}} \xrightarrow[t\to+\infty]{} \left(\underline{v}^\dagger \underline{\mathbb{1}}\right) \underline{\pi}^\dagger \tag{1.6}$$

Since every diffusion process on G converges to $\underline{\pi}$, this implies that

$$\lim_{t\to+\infty} \underline{R}(t) = \underline{\pi}. \tag{1.7}$$

Understanding the finite time behavior of $\underline{R}(t)$ in the weak coupling limit is facilitated by using the partition of the vertex set $V = \left(\bigsqcup_{n=1}^{N_C} V_n\right) \sqcup V_B$ to create the corresponding induced subgraphs $\left\{G_{V_1}, \cdots, G_{V_{N_C}}, G_{V_B}\right\}$. We then draw a notational distinction between the *restriction* of $\underline{R}(t)$ to a subset of the vertices of G, denoted $\overset{\circ\!-\!\circ}{\underline{R}}{}^{(*)}(t) \in \mathbb{R}^{|V_*|}$ for the restriction to the vertex subset $V_* \subset V$, and the return probability computed on the corresponding *induced subgraph*, denoted $\mathring{\underline{R}}^{(*)}(t) \in \mathbb{R}^{|V_*|}$ for the return probability on G_{V_*}[6]. The use of $\circ\!-\!\circ$ is intended to be evocative of a graph with two clusters (the circles) connected by a bridge/boundary vertex set (the line)[7]; the use of $\circ$ to denote the return probability on the induced subgraph of a cluster is then natural.

With this notation, it is trivially true that $\underline{R}(t) = \left(\bigoplus_{n=1}^{N_C} \overset{\circ\!-\!\circ}{\underline{R}}{}^{(n)}(t)\right) \oplus \overset{\circ\!-\!\circ}{\underline{R}}{}^{(B)}(t)$. Indeed, such an equality is exactly true for *any* partition of the vertices V, regardless of whether that partition respects the cluster structure of the graph. The key insight is that in the weak inter-cluster

[5]The fact that this global equilibrium distribution always exists is a consequence of our assumption that the graph in question is connected and undirected, and that the Laplacian of any connected undirected weighted graph has a single unique equilibrium distribution (Chung 1997).

[6]In practice, $\mathring{\underline{R}}^{(*)}(t)$ is constructed by using the principal submatrix of G's weight matrix $\underline{\underline{W}}$ corresponding to the vertex subset V_*, and forming the corresponding subgraph Laplacian from it, $\mathring{\underline{\underline{L}}}^{(*)}$. The subgraph return probability is then $\mathring{\underline{R}}^{(*)}(t) \equiv \operatorname{diag}\left(e^{-t\mathring{\underline{\underline{L}}}^{(*)}}\right) \in \mathbb{R}^{|V_*|}$.

[7]Indeed, $\circ\!-\!\circ$ is visually strikingly similar the so-called *dumbbell* point cloud, which is a common toy example.

coupling limit *for a vertex partition that respects the cluster structure*,

$$\overset{\circ\!-\!\circ}{\underline{R}}^{(n)}(t) \approx a_n(t)\mathring{\underline{R}}^{(n)}(t), \tag{1.8}$$

i.e. the return probability in cluster n of the full graph G is approximately the same as the return probability of the corresponding subgraph G_{V_n}, up to a time-dependent (positive) proportionality constant. This is a consequence of the fact that any reasonable definition of a well-defined cluster V_n from a graph diffusion perspective requires that the corresponding induced subgraph G_{V_n} be connected and have a short mixing time $T_n^{\text{mix}} \ll T_{\text{global}}^{\text{mix}}$, where $T_{\text{global}}^{\text{mix}}$ is the mixing time of the full graph G. Note that this is *not* true of the boundary/bridge portion of the return probability generically, even in the weak-coupling limit. Thus:

$$\underline{R}(t) \approx \left(\bigoplus_{n=1}^{N_C} a_n(t)\mathring{\underline{R}}^{(n)}(t)\right) \oplus \overset{\circ\!-\!\circ}{\underline{R}}^{(B)}(t). \tag{1.9}$$

For short times $t < T_n^{\text{mix}}$, the behavior of $\mathring{\underline{R}}^{(n)}(t)$ depends on the details of the cluster's induced subgraph G_{V_n}. But for $t > T_n^{\text{mix}}$, $\mathring{\underline{R}}^{(n)}(t)$ is well-approximated by the equilibrium distribution of G_{V_n}, which we denote $\mathring{\underline{\pi}}^{(n)}$. Thus, in the intermediate time regime where $t > \max_n(T_n^{\text{mix}})$,

$$\underline{R}\left(t > \max_n(T_n^{\text{mix}})\right) \approx \left(\bigoplus_{n=1}^{N_C} a_n(t)\mathring{\underline{\pi}}^{(n)}\right) \oplus \overset{\circ\!-\!\circ}{\underline{R}}^{(B)}(t). \tag{1.10}$$

Recognizing that in the weak inter-cluster coupling limit, the global equilibrium distribution should satisfy

$$\underline{\pi} = \left(\bigoplus_{n=1}^{N_C} \overset{\circ\!-\!\circ}{\underline{\pi}}^{(n)}\right) \oplus \overset{\circ\!-\!\circ}{\underline{\pi}}^{(B)} \approx \left(\bigoplus_{n=1}^{N_C} \mu_\pi^{(n)}\mathring{\underline{\pi}}^{(n)}\right) \oplus \overset{\circ\!-\!\circ}{\underline{\pi}}^{(B)} \tag{1.11}$$

where $\mu_\pi^{(n)} \in (0,1)_{\mathbb{R}}$ represents the equilibrium probability mass in cluster n, we define the *rescaled return probability* as:

$$\underline{\mathscr{R}}(t) \equiv \underline{R}(t) \oslash \underline{\pi} \tag{1.12}$$

where $\oslash$ is the element-wise division operator between vectors: $\underline{z} = \underline{x} \oslash \underline{y} \implies z_i = x_i/y_i$. Then our expected behavior for this rescaled return probability for intermediate times is:

$$\underline{\mathscr{R}}\left(t > \max_n(T_n^{\text{mix}})\right) \approx \left(\bigoplus_{n=1}^{N_C} \alpha_n(t)\underline{\mathbb{1}}^{(n)}\right) \oplus \overset{\circ\!-\!\circ}{\underline{\mathscr{R}}}^{(B)}(t) \tag{1.13}$$

where $\alpha_n(t) \equiv \frac{a_n(t)}{\mu_\pi^{(n)}} \in (0,+\infty)_{\mathbb{R}}$, and we leave $\overset{\circ\!-\!\circ}{\underline{\mathscr{R}}}^{(b)}(t) \equiv \overset{\circ\!-\!\circ}{\underline{R}}^{(B)}(t) \oslash \overset{\circ\!-\!\circ}{\underline{\pi}}^{(B)}$ unsimplified due to its complex behavior that depends on the specific structure of the boundary vertices and their coupling to the clusters.

In the weak coupling limit, $\underline{\mathscr{R}}$'s simple behavior is suggestive. While using the values of $\underline{\mathscr{R}}$ directly to try to cluster vertices based on the values of the coefficients $\alpha_n(t)$ may have immediate appeal due to its simplicity, it suffers from simple failure modes[8]. Instead, note that at intermediate times for two vertices in the same cluster $u, v \in V_n$, $\mathscr{R}_u - \mathscr{R}_v \approx 0$, while generically $\mathscr{R}_u - \mathscr{R}_w \not\approx 0$ for $w \in V_B$.[9] Thus we propose the *instantaneous boundary identification score*

$$s_i(t) \equiv \left(\sum_{j=1}^{|V|} \left| \underbrace{\left[e^{-T\underline{\underline{L}}^{\text{r.w.}}}\right]_{ij}}_{\equiv K_{ij}(T)} \left(\mathscr{R}_i(t) - \mathscr{R}_j(t)\right)\right|^p\right)^{\frac{1}{p}} \equiv \left[\underline{\mathscr{RN}}_p\left(\underline{\underline{\nabla}}_{\underline{\underline{K}}(T)}\underline{\mathscr{R}}(t)\right)\right]_i \tag{1.14}$$

where for any matrix $\underline{\underline{M}} \in \mathbb{R}^{N\times M}$, $\underline{\mathscr{RN}}_p(\underline{\underline{M}}) \in \mathbb{R}^N$ is the column vector of p-norms of the rows of $\underline{\underline{M}}$, and we use the diffusion kernel $\underline{\underline{K}}(T) \equiv e^{-T\underline{\underline{L}}^{\text{r.w.}}}$ for some fixed time T to define the distance on the graph differences in $\underline{\mathscr{R}}(t)$ will be compared across. In particular, choosing a sufficiently small value of T suppresses contributions to the instantaneous score from $\mathscr{R}_u - \mathscr{R}_v$ for u, v in different clusters. The *boundary identification score* is then defined as:

$$S_i \equiv \int_0^{+\infty} dt\, w(t)\, s_i(t) = \int_0^{+\infty} dt\, w(t)\, \left[\underline{\mathscr{RN}}_p\left(\underline{\underline{\nabla}}_{\underline{\underline{K}}(T)}\underline{\mathscr{R}}(t)\right)\right]_i \tag{1.15}$$

[8]One example is a graph with two identical clusters. One would expect the $\alpha_1(t) \approx \alpha_2(t)$ in this case.

[9]Note that the elements of $\underline{\mathscr{R}}(t)$ are being indexed with vertices here. The meaning is clear despite the (arguable) abuse of notation, and we will not draw a distinction between the function $\mathscr{R} : \mathbb{R}_+ \times V \mapsto \mathbb{R}$ and the elements of the time-dependent column vector $\underline{\mathscr{R}}(t)$.

where $w(t) : \mathbb{R} \to [0, +\infty)_{\mathbb{R}}$ is a weighting function allowing the score to focus on specific timescales.

This score in equation (1.15) was constructed based on a qualitative description of the expected behavior of diffusion on graphs with well-defined, weakly coupled clusters. Based on this exposition, S_i should be *large* on vertices that connect clusters together, and *small* in the interiors of clusters.

1.4 Connection to the Spectral Embedding Norm

The boundary identification score $\underline{S}$ explored in this work is defined in terms of local differences of the rescaled return probability $\underline{\mathscr{R}}(t)$. Later, in equations (2.11) and (2.12), $\underline{\mathscr{R}}(t)$ will be expressed in terms of the right eigenvectors of the random walk Laplacian:

$$\underline{\mathscr{R}}(t) \propto \sum_{k=1}^{|V|} e^{-\lambda_k^{(L)} t} (\underline{r}_k)^{\odot 2} \qquad \underline{\underline{L}}^{\text{r.w.}} \underline{r}_k = \lambda_k^{(L)} \underline{r}_k \tag{1.16}$$

The work of X. Cheng and Mishne 2020 defined a so-called *spectral embedding norm* in terms of a similar sum of element-wise squares of the same Laplacian right eigenvectors:

$$\underline{S}^{\text{s.e.n.}} \equiv \sum_{i=1}^{I} (\underline{r}_i)^{\odot 2} \qquad I \in [1, |V|]_{\mathbb{Z}} \tag{1.17}$$

where the eigenvectors $\underline{r}_i$ are assumed to be sorted in order of *increasing* eigenvalues $\lambda_i^{(L)}$, and I controls how deep into the spectrum $\underline{S}^{\text{s.e.n.}}$ uses information from.

There is a clear quantitative connection between these quantities. Define the timescale associated with an eigenvalue as $T_k \equiv 1/\lambda_k^{(L)}$. If there is a large timescale gap such that $\frac{T_I}{T_{I+1}} \gg 1$, then splitting the sum over eigenvectors in equation (1.16) and evaluating $\underline{\mathscr{R}}(t)$ between these two timescales such that $T_{I+1} \ll t \ll T_I$ reveals:

$$\underline{\mathscr{R}}(T_{I+1} \ll t \ll T_I) \propto \underbrace{\sum_{k=1}^{I} e^{-t/T_k} (\underline{r}_k)^{\odot 2}}_{\approx \underline{S}^{\text{s.e.n.}} \text{ for } t \ll T_I} + \underbrace{\sum_{k=I+1}^{|V|} e^{-t/T_k} (\underline{r}_k)^{\odot 2}}_{\approx 0 \text{ for } t \gg T_{I+1}} \approx \underline{S}^{\text{s.e.n.}}. \tag{1.18}$$

In the absence of such a large timescale gap, there is no clean correspondence between $\underline{\mathscr{R}}$ and $\underline{S}^{\text{s.e.n.}}$, though a qualitative connection remains.

In light of this relationship, the success of the spectral embedding norm at identifying small anomalous clusters of data points has an intuitive explanation. Small clusters will have an enormous rescaled return probability at intermediate times before a random walk starting in the anomaly is likely to leave it. On the other hand, random walks in large clusters have a much larger portion of the graph to explore, and the return probability drops more quickly at intermediate times. Thus we expect $\mathscr{R}_{\text{anomaly}}(t) \gg \mathscr{R}_{\text{large cluster}}(t)$ for a broad range of appropriate intermediate times t for well-defined anomalies, and by extension, $S^{\text{s.e.n.}}_{\text{anomaly}} \gg S^{\text{s.e.n.}}_{\text{large cluster}}$ for a broad range of values of I, since the value of I controls the time t at which $\underline{S}^{\text{s.e.n.}}$ is approximating the rescaled return probability $\underline{\mathscr{R}}(t)$.

That being said, the spectral embedding norm relies on a hard truncation of the sum with no eigenvalue-based weightings of the various vectors, which is a deeply unnatural quantity. Indeed, a large portion of X. Cheng and Mishne 2020 is devoted to such concerns and to what degree they are or are not mitigated by summing over a particular fraction of the eigenvectors. This can all be sidestepped by instead considering a quantity rooted in the natural time-domain dynamics of the diffusion process, such as $\underline{\mathscr{R}}(t)$, from which the boundary-identifying score $\underline{S}$ of this work is derived.

1.5 Tractable Toy Models

With our boundary identifying score $\underline{S}$ in hand, let us understand its behavior in some simple, analytically tractable cases. The general score in equation (1.15) contains a few tunable parameters: the time weighting function $w(t) : \mathbb{R}_+ \to \mathbb{R}_+$, the row p-norm with $p \in [1, \infty)$, and the timescale T used in the weighted graph gradient $\underline{\nabla}_{\underline{\underline{K}}(T)}$ to define the distance on the graph across which values of $\underline{\mathscr{R}}(t)$ are compared. This setting is too general for our purpose here, however, and so we first simplify the weighted graph difference $\underline{\nabla}_{\underline{\underline{K}}(T)}$ by focusing on the small-T regime:

Lemma 1. *For a vertex function $f : V \to \mathbb{R}$, the weighted graph gradient $\underline{\nabla}_{\underline{\underline{K}}(T)}\underline{f}$ obeys:*

$$\underline{\nabla}_{\underline{\underline{K}}(T)}\underline{f} = T\left(\underline{\nabla}_{\underline{\underline{P}}^{\text{r.w.}}}\underline{f}\right) + O(T^2)$$

consequently,

$$\lim_{T\to 0^+} \frac{1}{T}\underline{\nabla}_{\underline{\underline{K}}(T)}\underline{f} = \underline{\nabla}_{\underline{\underline{P}}^{\text{r.w.}}}\underline{f}$$

Proof.

Using the power series definition of the matrix exponential yields

$$\left[\underline{\nabla}_{\underline{\underline{K}}(T)}\underline{f}\right]_{ij} = \left[e^{-T\underline{\underline{L}}^{\text{r.w.}}}\right]_{ij}(f_i - f_j) = \left[\underline{\underline{\mathbb{I}}} + \sum_{n=1}^{+\infty}\left(-T\underline{\underline{L}}^{\text{r.w.}}\right)^n\right]_{ij}(f_i - f_j). \tag{1.19}$$

Since $(f_i - f_j)\big|_{i=j} = 0$, the $\underline{\underline{\mathbb{I}}}$ term drops out, leaving

$$\left[\underline{\nabla}_{\underline{\underline{K}}(T)}\underline{f}\right]_{ij} = -TL_{ij}^{\text{r.w.}}(f_i - f_j) + O(T^2) = -T\left(\underline{\underline{\mathbb{I}}} - P_{ij}^{\text{r.w.}}\right)(f_i - f_j) + O(T^2). \tag{1.20}$$

Once again, the $\underline{\underline{\mathbb{I}}}$ term drops out, leaving

$$\underline{\nabla}_{\underline{\underline{K}}(T)} = \underline{\nabla}_{\underline{\underline{K}}(T)} = T\underline{\nabla}_{\underline{\underline{P}}^{\text{r.w.}}}\underline{f} + O(T^2) \tag{1.21}$$

The limit then follows trivially.

□

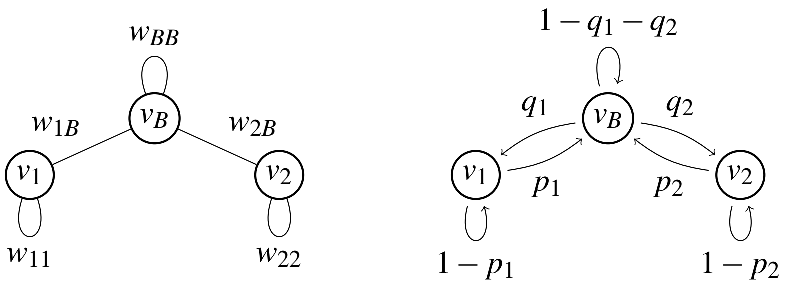


Figure 1.1. A toy model with two 'clusters' represented by vertices v_1, v_2, and a bridge vertex v_B connecting them. On the left is the undirected weighted graph, and at the right is the corresponding random walk state transition diagram induced by the weighted graph.

Then in the $T \to 0^+$ limit, the score becomes:

$$\frac{1}{T} S_i \xrightarrow[T\to 0^+]{} \int_0^{+\infty} dt\; w(t)\; \underline{\mathscr{RN}}_p \left(\underline{\nabla}_{\underline{\underline{P}}^{\text{r.w.}}} \underline{\mathscr{R}}(t) \right), \tag{1.22}$$

which has a number of advantages analytically, not the least of which being that the structure of $\underline{\underline{P}}^{\text{r.w.}}$ is sparse and simple in the examples that follow, greatly simplifying computations. The simplest version of the score that we will use furthermore fixes $w(t) = 1$ and $p = 1$, which gives what we term the *effective* score as:

$$S_i^{\text{eff.}} \equiv \int_0^{+\infty} dt \sum_{j=1}^{|V|} P_{ij}^{\text{r.w.}} \left| \mathscr{R}_i(t) - \mathscr{R}_j(t) \right| \tag{1.23}$$

1.5.1 2-Cluster Toy Model

Consider the toy model weighted graph shown in figure (1.1). This is the simplest non-trivial model of two clusters, represented by vertices v_1 and v_2, connected by a set of bridge points, represented by the vertex v_B. Its weights, degrees, and row-normalized transition matrix are:

$$\underline{\underline{W}} = \begin{bmatrix} w_{11} & 0 & w_{1B} \\ 0 & w_{22} & w_{2B} \\ w_{B1} & w_{B2} & w_{BB} \end{bmatrix} \Longrightarrow \underline{d} = \begin{bmatrix} w_{11} + w_{1B} \\ w_{22} + w_{2B} \\ w_{1B} + w_{2B} + w_{BB} \end{bmatrix} \equiv \begin{bmatrix} d_1 \\ d_2 \\ d_B \end{bmatrix} \tag{1.24}$$

$$\underline{\underline{P}} = \underline{\underline{\bcancel{d}}}^{-1}\underline{\underline{W}} = \begin{bmatrix} 1 - p_1 & 0 & p_1 \\ 0 & 1 - p_2 & p_2 \\ q_1 & q_2 & 1 - q_1 - q_2 \end{bmatrix} \tag{1.25}$$

Note that symmetry of the weight matrix requires $w_{ij} = w_{ji}$, which has been used in various places above for simplicity or consistency, as desired. The transition matrix in equation (1.25) corresponds to the state transition diagram in figure (1.1), where the definitions of $p_1, p_2, q_1, q_2 \in [0,1]$ are implied by the matrix equations. In particular, in the regime of interest where v_1, v_2 represent clusters, we expect $(0,1) \ni p_1, p_2 \ll 1$ since p_1, p_2 represent the probabilities of *leaving* the clusters to go to the bridge vertex. However, while the probabilities of leaving the bridge vertex to go to the clusters $q_1, q_2 > 0$ and $q_1 + q_2 < 1$, they are not generically close to 0 in the domain of interest.

We can exploit the structure of our score function and the simplicity of this model to extract an exact relationship between the scores of these vertices with minimal computation:

Theorem 1. *The instantaneous score $\underline{s}(t)$ of the three-vertex graph shown in figure (1.1) obeys:*

$$s_B(t) = \left(\left(\frac{q_1}{p_1} s_1(t) \right)^p + \left(\frac{q_2}{p_2} s_2(t) \right)^p \right)^{1/p}$$

Furthermore, for the 1-norm (i.e. $p = 1$), the score $\underline{S}$ obeys:

$$S_B = \frac{q_1}{p_1} S_1 + \frac{q_2}{p_2} S_2$$

Proof.

From the definition of the score:

$$\underline{S} = \int_0^{+\infty} dt\, w(t)\, \underline{s}(t) = \int_0^{+\infty} dt\, w(t)\, \underline{\mathscr{RN}}_p \left(\underline{\nabla}_{\underline{\underline{P}}^{\text{r.w.}}} \underline{\mathscr{R}}(t) \right)$$

Defining $\Delta\mathscr{R}_{ij}(t) \equiv \mathscr{R}_i(t) - \mathscr{R}_j(t)$, noting that $\Delta\mathscr{R}_{ij} = -\Delta\mathscr{R}_{ji}$, and using # for elements whose finite values drop out:

$$\nabla_{\underline{\underline{P}}} \mathscr{R}(t) = \begin{bmatrix} \# & 0 & p_1 \\ 0 & \# & p_2 \\ q_1 & q_2 & \# \end{bmatrix} \odot \begin{bmatrix} 0 & \# & \Delta\mathscr{R}_{1B}(t) \\ \# & 0 & \Delta\mathscr{R}_{2B}(t) \\ -\Delta\mathscr{R}_{1B}(t) & -\Delta\mathscr{R}_{2B}(t) & 0 \end{bmatrix}$$
$$= \begin{bmatrix} 0 & 0 & p_1\Delta\mathscr{R}_{1B}(t) \\ 0 & 0 & p_2\Delta\mathscr{R}_{2B}(t) \\ -q_1\Delta\mathscr{R}_{1B}(t) & -q_2\Delta\mathscr{R}_{2B}(t) & 0 \end{bmatrix}$$

Then:

$$\underline{\mathscr{RN}}_p \left(\underline{\nabla}_{\underline{\underline{P}}} \underline{\mathscr{R}}(t) \right) = \begin{bmatrix} p_1 \left|\Delta\mathscr{R}_{1B}(t)\right| \\ p_2 \left|\Delta\mathscr{R}_{2B}(t)\right| \\ \left((q_1 \left|\Delta\mathscr{R}_{1B}(t)\right|)^p + (q_2 \left|\Delta\mathscr{R}_{2B}(t)\right|)^p \right)^{1/p} \end{bmatrix} \tag{1.26}$$

which implies

$$\underline{s}(t) = \underline{\mathscr{RN}}_p \left(\underline{\nabla}_{\underline{\underline{P}}} \underline{\mathscr{R}}(t) \right) = \begin{bmatrix} s_1(t) \\ s_2(t) \\ s_B(t) \end{bmatrix} = \begin{bmatrix} s_1(t) \\ s_2(t) \\ \left(\left(\frac{q_1}{p_1} s_1(t) \right)^p + \left(\frac{q_2}{p_2} s_2(t) \right)^p \right)^{1/p} \end{bmatrix} \tag{1.27}$$

For the $p = 1$ norm, the linearity of integration then implies $S_B = \frac{q_1}{p_1} S_1 + \frac{q_2}{p_2} S_2$. □

In this simple model, the score on the boundary vertex v_B is a weighted sum of the scores on the cluster vertices $\{v_1, v_2\}$. The weights in the sum are precisely those ratios $\frac{q_n}{p_n}$ that determine how much more likely the diffusion process is to leave the boundary to a cluster than it is to leave a cluster to the boundary. This is, of course, large in the domain in interest.

Note that no particular properties of $\underline{\mathscr{R}}(t)$ were used in this computation! Defining our score with any vertex function would have produced an identical result, which here is a consequence of the limited freedom in a 3-vertex graph, the structure of $\underline{\underline{P}}^{\text{r.w.}}$, and its use to define the graph difference operator $\nabla_{\underline{\underline{P}}^{\text{r.w.}}}$. Once internal substructure is added to the clusters and bridge, however, the choice of $\underline{\mathscr{R}}$ becomes relevant.

1.5.2 Adding Substructure

It is common practice to coarse-grain large graphs by grouping together sets of vertices in the original graph into a single vertex in the new graph. There is a rich body of work applying graph diffusion-based methods to inform this coarse-graining in the hopes of preserving some underlying structure (Lafon and Lee 2006; Gfeller and De Los Rios 2007; Villegas et al. 2023). Here, we will do the opposite, and "decorate" an initial coarse-grained representation of a clustered graph to analyze the more general case.

We may add substructure to the graph in figure (1.1) as illustrated in figure (1.2) by replacing each vertex v_* with a set of vertices $V_* \subset V$, and each real non-negative edge weight $w_{ij} \equiv w(v_i, v_j)$ by a real matrix with non-negative entries $W(V_i, V_j) \equiv \underline{\underline{W}}^{ij} \in \mathbb{R}^{|V_i|} \times \mathbb{R}^{|V_j|}$ obeying $(\underline{\underline{W}}^{ij})^\dagger = \underline{\underline{W}}^{ji}$, corresponding to the block structure of the weight matrix:

$$\underline{\underline{W}} = \begin{bmatrix} \underline{\underline{W}}^{11} & 0 & \underline{\underline{W}}^{1B} \\ 0 & \underline{\underline{W}}^{22} & \underline{\underline{W}}^{2B} \\ \underline{\underline{W}}^{B1} & \underline{\underline{W}}^{B2} & \underline{\underline{W}}^{BB} \end{bmatrix} \tag{1.28}$$

For any matrix $\underline{\underline{A}}_{N\times M}$, define the row sums vector $\underline{A}_{\text{r.s.}}$ to be the N-element column vector with the sum of each row of $\underline{\underline{A}}$ as its elements, i.e. $\underline{A}_{\text{r.s.}} \equiv \underline{\underline{A}}\,\underline{\mathbb{1}}$. Then the full-graph degrees $\overset{\circ\!\!-\!\!\circ}{\underline{d}}{}^*$

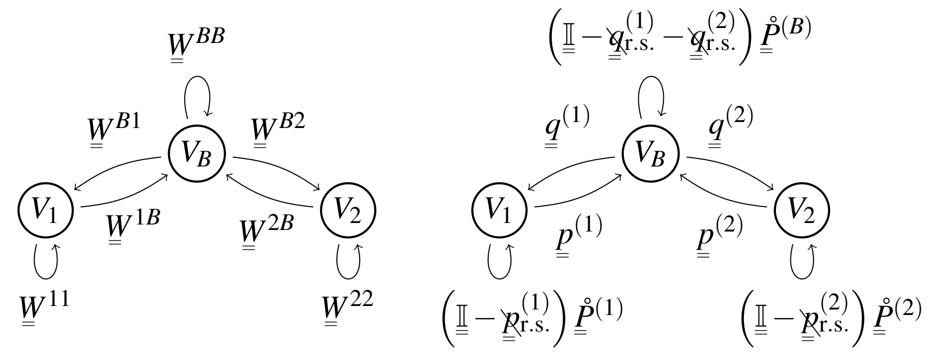

$\underline{\underline{W}}^{BB}$
$\underline{\underline{W}}^{B1}$
$\underline{\underline{W}}^{B2}$
V_B
V_1
V_2
$\underline{\underline{W}}^{1B}$
$\underline{\underline{W}}^{2B}$
$\underline{\underline{W}}^{11}$
$\underline{\underline{W}}^{22}$
$\left(\underline{\underline{\mathbb{I}}} - \underline{\underline{\diagdown\!\!\!q}}^{(1)}_{\text{r.s.}} - \underline{\underline{\diagdown\!\!\!q}}^{(2)}_{\text{r.s.}}\right) \underline{\underline{\mathring{P}}}^{(B)}$
$\underline{\underline{q}}^{(1)}$
$\underline{\underline{q}}^{(2)}$
$\underline{\underline{p}}^{(1)}$
$\underline{\underline{p}}^{(2)}$
$\left(\underline{\underline{\mathbb{I}}} - \underline{\underline{\diagdown\!\!\!p}}^{(1)}_{\text{r.s.}}\right) \underline{\underline{\mathring{P}}}^{(1)}$
$\left(\underline{\underline{\mathbb{I}}} - \underline{\underline{\diagdown\!\!\!p}}^{(2)}_{\text{r.s.}}\right) \underline{\underline{\mathring{P}}}^{(2)}$

Figure 1.2. A more realistic model with two clusters with vertex sets V_1, V_2, and a set of bridge vertices V_B connecting them. On the left is the (now directed) weighted graph, where the matrix-weights necessitate a directed representation to accommodate the matrix transpose associated with the pairing of the two edges (V_i, V_j) and (V_j, V_i). At the right is the corresponding random walk state transition diagram induced by the weighted graph, where the "states" V_* now correspond to an element in $\mathbb{R}^{|V_*|}$.

and sub-graph degrees $\underline{\mathring{d}}^*$ are

$$
\begin{aligned}
\underline{\mathring{d}}^{(1)} &= \underline{W}^{11}_{\text{r.s.}} & \qquad \underline{\overset{\circ\!-\!\circ}{d}}^{(1)} &= \underline{\mathring{d}}^{(1)} + \underline{W}^{1B}_{\text{r.s.}} \\
\underline{\mathring{d}}^{(2)} &= \underline{W}^{22}_{\text{r.s.}} & \qquad \underline{\overset{\circ\!-\!\circ}{d}}^{(2)} &= \underline{\mathring{d}}^{(2)} + \underline{W}^{2B}_{\text{r.s.}} \\
\underline{\mathring{d}}^{(B)} &= \underline{W}^{BB}_{\text{r.s.}} & \qquad \underline{\overset{\circ\!-\!\circ}{d}}^{(B)} &= \underline{W}^{B1}_{\text{r.s.}} + \underline{W}^{B2}_{\text{r.s.}} + \underline{\mathring{d}}^{(B)}
\end{aligned}
\tag{1.29}
$$

so that the full degree vector is $\underline{d} = \underline{\overset{\circ\!-\!\circ}{d}}^{(1)} \oplus \underline{\overset{\circ\!-\!\circ}{d}}^{(2)} \oplus \underline{\overset{\circ\!-\!\circ}{d}}^{(B)}$. The row-normalized transition matrix is then:

$$
\underline{\underline{P}} = \underline{\underline{\diagdown\!\!\!d}}^{-1}\underline{\underline{W}} = \begin{bmatrix}
\left(\underline{\underline{\diagdown\!\!\!\overset{\circ\!-\!\circ}{d}}}^{(1)}\right)^{-1} \underline{\underline{W}}^{11} & \underline{\underline{0}} & \left(\underline{\underline{\diagdown\!\!\!\overset{\circ\!-\!\circ}{d}}}^{(1)}\right)^{-1} \underline{\underline{W}}^{1B} \\
\underline{\underline{0}} & \left(\underline{\underline{\diagdown\!\!\!\overset{\circ\!-\!\circ}{d}}}^{(B)}\right)^{-1} \underline{\underline{W}}^{22} & \left(\underline{\underline{\diagdown\!\!\!\overset{\circ\!-\!\circ}{d}}}^{(B)}\right)^{-1} \underline{\underline{W}}^{2B} \\
\left(\underline{\underline{\diagdown\!\!\!\overset{\circ\!-\!\circ}{d}}}^{(B)}\right)^{-1} \underline{\underline{W}}^{B1} & \left(\underline{\underline{\diagdown\!\!\!\overset{\circ\!-\!\circ}{d}}}^{(B)}\right)^{-1} \underline{\underline{W}}^{B2} & \left(\underline{\underline{\diagdown\!\!\!\overset{\circ\!-\!\circ}{d}}}^{(B)}\right)^{-1} \underline{\underline{W}}^{BB}
\end{bmatrix}
\tag{1.30}
$$

We can put this into a more suggestive form by expanding the inverse of the full-graph degrees $\left(\overset{\circ\!-\!\circ}{\underline{\underline{\diagdown\!\!\!d}}}^{(*)}\right)^{-1}$ in terms of the subgraph degrees $\overset{\circ}{\underline{\underline{\diagdown\!\!\!d}}}^{(*)}$ by repeatedly applying the identity $\frac{1}{x+\delta} = \left(1 - \frac{\delta}{x+\delta}\right)\frac{1}{x}$. In particular, take $x = \overset{\circ}{\underline{\underline{\diagdown\!\!\!d}}}^{(*)}$, and $x+\delta = \overset{\circ\!-\!\circ}{\underline{\underline{\diagdown\!\!\!d}}}^{(*)}$. Since every diagonal element of both $\overset{\circ}{\underline{\underline{\diagdown\!\!\!d}}}^{(*)}$ and $\overset{\circ\!-\!\circ}{\underline{\underline{\diagdown\!\!\!d}}}^{(*)}$ is strictly greater than 0, there are no issues with division by zero. Then, define

$$
\begin{aligned}
\underline{\underline{p}}^{(1)} &\equiv \left(\overset{\circ\!-\!\circ}{\underline{\underline{\diagdown\!\!\!d}}}^{(1)}\right)^{-1}\underline{\underline{W}}^{1B} & \qquad \underline{\underline{p}}^{(2)} &\equiv \left(\overset{\circ\!-\!\circ}{\underline{\underline{\diagdown\!\!\!d}}}^{(2)}\right)^{-1}\underline{\underline{W}}^{2B} \\
\underline{\underline{q}}^{(1)} &\equiv \left(\overset{\circ\!-\!\circ}{\underline{\underline{\diagdown\!\!\!d}}}^{(B)}\right)^{-1}\underline{\underline{W}}^{B1} & \qquad \underline{\underline{q}}^{(2)} &\equiv \left(\overset{\circ\!-\!\circ}{\underline{\underline{\diagdown\!\!\!d}}}^{(B)}\right)^{-1}\underline{\underline{W}}^{B2} \\
\overset{\circ}{\underline{\underline{P}}}^{(1)} &\equiv \left(\overset{\circ}{\underline{\underline{\diagdown\!\!\!d}}}^{(1)}\right)^{-1}\underline{\underline{W}}^{11} & \qquad \overset{\circ}{\underline{\underline{P}}}^{(2)} &\equiv \left(\overset{\circ}{\underline{\underline{\diagdown\!\!\!d}}}^{(2)}\right)^{-1}\underline{\underline{W}}^{22}
\end{aligned}
\tag{1.31}
$$

and introduce the notation: for any matrix $\underline{\underline{A}} \in \mathbb{R}^{N\times M}$ with row sum vector $\underline{A}_{\text{r.s.}} \in \mathbb{R}^{N}$, the matrix $\underline{\underline{\diagdown\!\!\!A}}_{\text{r.s.}} \in \mathbb{R}^{N\times N}$ is the diagonal square matrix with the row sums of $\underline{\underline{A}}$ on its main diagonal, i.e. $[\underline{\underline{\diagdown\!\!\!A}}_{\text{r.s.}}]_{ij} = \delta_{ij}\left(\sum_{k=1}^{M}[\underline{\underline{A}}]_{ik}\right)$. With these definitions in hand, the block transition matrix can be written in an analogous form to the three-vertex case in equation (1.25),

$$
\underline{\underline{P}} = \begin{bmatrix}
(\underline{\underline{\mathbb{I}}} - \underline{\underline{\diagdown\!\!\!p}}_{\text{r.s.}}^{(1)})\overset{\circ}{\underline{\underline{P}}}^{(1)} & \underline{\underline{0}} & \underline{\underline{p}}^{(1)} \\
\underline{\underline{0}} & (\underline{\underline{\mathbb{I}}} - \underline{\underline{\diagdown\!\!\!p}}_{\text{r.s.}}^{(2)})\overset{\circ}{\underline{\underline{P}}}^{(2)} & \underline{\underline{p}}^{(2)} \\
\underline{\underline{q}}^{(1)} & \underline{\underline{q}}^{(2)} & (\underline{\underline{\mathbb{I}}} - \underline{\underline{\diagdown\!\!\!q}}_{\text{r.s.}}^{(1)} - \underline{\underline{\diagdown\!\!\!q}}_{\text{r.s.}}^{(2)})\overset{\circ}{\underline{\underline{P}}}^{(B)}
\end{bmatrix},
\tag{1.32}
$$

as shown in in the state-transition diagram in figure (1.2).

A few notes about equation (1.32):

1. $\underline{\underline{p}}^{(*)} \in \mathbb{R}^{|V_*|\times|V_B|}$ controls the outflow from V_* to the bridge vertices V_B. In the weak coupling limit, we expect $\sum_j p_{ij}^{(*)} = \left[\underline{\underline{\diagdown\!\!\!p}}_{\text{r.s.}}^{(*)}\right]_{ii} \ll 1$

2. $\underline{\underline{q}}^{(*)} \in \mathbb{R}^{|V_B|\times|V_*|}$ controls the inflow from the bridge vertices V_B to the cluster V_*. While $\sum_j q_{ij}^{(*)} = \left[\underline{\underline{\diagdown\!\!\!q}}_{\text{r.s.}}^{(*)}\right]_{ii} \in [0,1]$, the weak coupling limit does *not* guarantee that the row-sums of $\underline{\underline{q}}^{(*)}$ (i.e. the diagonal elements of $\underline{\underline{\diagdown\!\!\!q}}^{(*)}$) are small. Indeed, the bridge vertices are precisely

those that have fast outflow for diffusion processes starting on them, versus slow inflow for diffusion processes starting in the clusters.

3. $\underline{\underline{\mathring{P}}}^{(1)}$ and $\underline{\underline{\mathring{P}}}^{(2)}$ are the transition matrices for the induced subgraphs G_{V_1} and G_{V_2} of clusters 1 and 2. Since the outflow from the clusters to the bridge vertices is small in the weak coupling limit, we expect the internal diffusion dynamics in clusters 1 and 2 to be dominated by $\underline{\underline{\mathring{P}}}^{(1)}$, $\underline{\underline{\mathring{P}}}^{(2)}$, and the inflow from the bridge vertices V_B. Furthermore, since $\underline{\underline{\mathring{P}}}^{(1)}$ and $\underline{\underline{\mathring{P}}}^{(2)}$ represent the internal dynamics of clusters, we expect them to induce dynamics with rapid mixing times.

4. $\underline{\underline{\mathring{P}}}^{(B)}$ is the transition matrix for the induced subgraph G_{V_B} of the bridge vertices. Even in the weak coupling limit, we expect $\underline{\underline{\mathring{P}}}^{(B)}$ to be a poor predictor of the diffusion dynamics in V_B due to the strong outflow from these vertices to clusters A and B. Furthermore, the subgraph of bridge vertices may exhibit various pathological features, such as disconnectedness or extremely long mixing times.

1.5.3 N-cluster subgraph equilibrium coupling model

Consider a generalization of the 2-cluster model in section 1.5.1 with N_C clusters and a set of bridge vertices. If the coupling between each cluster and the bridge is only via their *subgraph equilibrium distributions* $\underline{\mathring{\pi}}^{(*)}$, this model can be exactly related to a *coarse grained* model consisting of an $(N_C + 1)$-vertex graph $G_{\text{c.g.}}$.

The weight matrix for this model is:

$$\underline{\underline{W}} = \begin{bmatrix} \underline{\underline{W}}^{11} & & 0 & a_1\,\mathring{\underline{\pi}}^{(1)}\left(\mathring{\underline{\pi}}^{(B)}\right)^{\dagger} \\ & \ddots & & \vdots \\ 0 & & \underline{\underline{W}}^{N_C N_C} & a_{N_C}\,\mathring{\underline{\pi}}^{(N_C)}\left(\mathring{\underline{\pi}}^{(B)}\right)^{\dagger} \\ a_1\,\mathring{\underline{\pi}}^{(B)}\left(\mathring{\underline{\pi}}^{(1)}\right)^{\dagger} & \cdots & a_{N_C}\,\mathring{\underline{\pi}}^{(B)}\left(\mathring{\underline{\pi}}^{(N_C)}\right)^{\dagger} & \underline{\underline{W}}^{BB} \end{bmatrix} \tag{1.33}$$

where $\underline{\underline{W}}^{nB} = a_n\,\mathring{\underline{\pi}}^{(n)}\left(\mathring{\underline{\pi}}^{(B)}\right)^{\dagger}$, with subgraph equilibrium distributions $\mathring{\underline{\pi}}^{(*)} \equiv \mathring{\underline{d}}^{(*)}/\mathring{D}^{(*)},\ \mathring{D}^{(*)} \equiv \underline{\mathbb{1}}^{\dagger}\mathring{\underline{d}}^{(*)}$. This results in a simple structure for the full graph degree vector and equilibrium distribution, with the components on each cluster and bridge subgraph proportional to the corresponding subgraph components:

$$\underline{d} = \left(\bigoplus_{n=1}^{N_C} \overset{\circ\!-\!\circ}{\underline{d}}^{(n)}\right) \oplus \overset{\circ\!-\!\circ}{\underline{d}}^{(B)} = \left(\bigoplus_{n=1}^{N_C} \frac{\mathring{D}_n + a_n}{\mathring{D}_n}\,\mathring{\underline{d}}^{(n)}\right) \oplus \frac{\mathring{D}_B + a_1 + \cdots + a_n}{\mathring{D}_B}\,\mathring{\underline{d}}^{(B)} \tag{1.34}$$

$$\underline{\pi} = \left(\bigoplus_{n=1}^{N_C} \frac{\mathring{D}_n + a_n}{D}\,\mathring{\underline{\pi}}^{(n)}\right) \oplus \frac{\sum_n a_n + \mathring{D}_B}{D}\,\mathring{\underline{\pi}}^{(B)}, \qquad D \equiv \sum_{n=1}^{N_C}\left(\mathring{D}_n + 2a_n\right) + \mathring{D}_B \tag{1.35}$$

as well as a simple structure to the corresponding transition and Laplacian matrices:

$$\underline{\underline{P}} = \underline{\underline{\bcancel{d}}}^{-1}\underline{\underline{W}} = \begin{bmatrix} (1-p_1)\mathring{\underline{\underline{P}}}^{(1)} & & 0 & p_1\,\underline{\mathbb{1}}^{(1)}\left(\mathring{\underline{\pi}}^{(B)}\right)^{\dagger} \\ & \ddots & & \vdots \\ 0 & & (1-p_{N_C})\mathring{\underline{\underline{P}}}^{(N_C)} & p_{N_C}\,\underline{\mathbb{1}}^{(N_C)}\left(\mathring{\underline{\pi}}^{(B)}\right)^{\dagger} \\ q_1\,\underline{\mathbb{1}}^{(B)}\left(\mathring{\underline{\pi}}^{(1)}\right)^{\dagger} & \cdots & q_{N_C}\,\underline{\mathbb{1}}^{(B)}\left(\mathring{\underline{\pi}}^{(N_C)}\right)^{\dagger} & (1-q_B)\mathring{\underline{\underline{P}}}^{(B)} \end{bmatrix} \tag{1.36}$$

$$\underline{\underline{L}} \equiv \underline{\underline{\mathbb{I}}} - \underline{\underline{P}} = \begin{bmatrix} (1-p_1)\mathring{\underline{\underline{L}}}^{(1)} + p_1\underline{\underline{\mathbb{I}}}^{(1)} & & 0 & -p_1\,\underline{\mathbb{1}}^{(1)}\left(\mathring{\underline{\pi}}^{(B)}\right)^{\dagger} \\ & \ddots & & \vdots \\ 0 & & (1-p_{N_C})\mathring{\underline{\underline{L}}}^{(N_C)} + p_{N_C}\underline{\underline{\mathbb{I}}}^{(N_C)} & -p_{N_C}\,\underline{\mathbb{1}}^{(N_C)}\left(\mathring{\underline{\pi}}^{(B)}\right)^{\dagger} \\ -q_1\,\underline{\mathbb{1}}^{(B)}\left(\mathring{\underline{\pi}}^{(1)}\right)^{\dagger} & \cdots & -q_{N_C}\,\underline{\mathbb{1}}^{(B)}\left(\mathring{\underline{\pi}}^{(N_C)}\right)^{\dagger} & (1-q_B)\mathring{\underline{\underline{L}}}^{(B)} + q_B\underline{\underline{\mathbb{I}}}^{(B)} \end{bmatrix} \tag{1.37}$$

where $q_B \equiv \sum_{n=1}^{N_C} q_n$, and the definitions of p_* and q_* are implied by the matrix equations.

Combining this the time-evolution equation of a diffusion process, $\partial_t \underline{v}^{\dagger}(t) = -\underline{v}^{\dagger}(t)\underline{\underline{L}}$, and a compatible partitioning of $\underline{v} = \left(\bigoplus_{n=1}^{N_C} \overset{\circ\!-\!\circ}{\underline{v}}^{(n)}\right) \oplus \overset{\circ\!-\!\circ}{\underline{v}}^{(B)}$, we have a system of N_C+1 differential equations governing the time evolution of $\underline{v}^{\dagger}(t)$:

$$\partial_t \overset{\circ\!-\!\circ}{\underline{v}}^{(n)\dagger} = -(1-p_n)\overset{\circ\!-\!\circ}{\underline{v}}^{(n)\dagger}\mathring{\underline{\underline{L}}}^{(n)} - p_n \overset{\circ\!-\!\circ}{\underline{v}}^{(n)\dagger} + q_n\left(\overset{\circ\!-\!\circ}{\underline{v}}^{(B)\dagger}\,\underline{\mathbb{1}}^{(B)}\right)\mathring{\underline{\pi}}^{(n)\dagger} \tag{1.38}$$

$$\partial_t \overset{\circ\!-\!\circ}{\underline{v}}^{(B)\dagger} = -(1-q_B)\overset{\circ\!-\!\circ}{\underline{v}}^{(B)\dagger}\mathring{\underline{\underline{L}}}^{(B)} - q_B \overset{\circ\!-\!\circ}{\underline{v}}^{(B)\dagger} + \left(\sum_{n=1}^{N_C} p_n\,\overset{\circ\!-\!\circ}{\underline{v}}^{(n)\dagger}\,\underline{\mathbb{1}}^{(n)}\right)\mathring{\underline{\pi}}^{(B)\dagger}. \tag{1.39}$$

With the only coupling between these equations happening between the subgraph equilibrium distributions, they can be solved with a *coarse-graining* ansatz:

$$\overset{\circ\!-\!\circ}{\underline{v}}^{(n)}(t) = \mu_n(t)\mathring{\underline{\pi}}^{(n)} + \underline{\gamma}^{(n)}(t), \qquad \underline{\gamma}^{(n)\dagger}(t)\,\underline{\mathbb{1}}^{(n)} \equiv 0, \tag{1.40}$$

where $\mu_n(t) = \overset{\circ\!-\!\circ}{\underline{v}}^{(n)}(t)^{\dagger}\,\underline{\mathbb{1}}^{(n)}$ represents the total probability mass in cluster n at time t (mirroring the use of μ as a symbol for mass in physics), and $\gamma^{(n)}(t)$ represents the deviation from the subgraph equilibrium distribution (mirroring the use of γ to represent the *skewness* of a distribution in statistics). By dotting these equations on the right with $\underline{\mathbb{1}}$ or $\underline{\underline{\mathbb{I}}} - \underline{\mathbb{1}}\underline{\pi}^{\dagger}$, we can separate the behavior of $\underline{\mu}(t) \in \mathbb{R}^{N_C+1}$ from the $\underline{\gamma}^{(*)}$ terms. Then $\underline{\mu}$ satisfies a coarse-grained time evolution equation:

$$\partial_t \underline{\mu}^\dagger(t) = -\underline{\mu}^\dagger(t)\underline{\underline{L}}^{\text{c.g.}}, \qquad \underline{\underline{L}}^{\text{c.g.}} = \begin{bmatrix} p_1 & & 0 & -p_1 \\ & \ddots & & \vdots \\ 0 & & p_{N_C} & -p_{N_C} \\ -q_1 & \cdots & -q_{N_C} & q_B \end{bmatrix}, \tag{1.41}$$

which has the formal solution $\underline{\mu}^\dagger(t) = \underline{\mu}^\dagger(0)e^{-\underline{\underline{L}}^{\text{c.g.}}t}$, and the $\underline{\gamma}^{(*)}(t)$ terms satisfy:

$$\partial_t \underline{\gamma}^{(n)^\dagger}(t) = -\underline{\gamma}^{(n)^\dagger}(t)\left(p_n\underline{\underline{\mathbb{I}}}^{(n)} + (1-p_n)\mathring{\underline{\underline{L}}}^{(n)}\right) \tag{1.42}$$

$$\partial_t \underline{\gamma}^{(B)^\dagger}(t) = -\underline{\gamma}^{(B)^\dagger}(t)\left(q_B\underline{\underline{\mathbb{I}}}^{(B)} + (1-q_B)\mathring{\underline{\underline{L}}}^{(B)}\right), \tag{1.43}$$

so their time evolution is governed solely by the corresponding subgraph Laplacians and the transition rates out of each cluster:

$$\underline{\gamma}^{(n)^\dagger}(t) = -\underline{\gamma}^{(n)^\dagger}(0)\, e^{-(1-p_n)\mathring{\underline{\underline{L}}}^{(n)}t}e^{-p_n t} \tag{1.44}$$

$$\underline{\gamma}^{(B)^\dagger}(t) = -\underline{\gamma}^{(B)^\dagger}(0)\, e^{-(1-q_B)\mathring{\underline{\underline{L}}}^{(B)}t}e^{-q_B t}. \tag{1.45}$$

Using the notation $\underline{e}^{(*,i)} \in \mathbb{R}^{|V_*|}$ for the i^{th} standard basis vector in partition $*$'s vector space, the return probabilities for a diffusion process starting on vertex i of cluster n are:

$$\overset{\circ\!-\!\circ}{R}{}_i^{(n)}(t) = \underbrace{\left[e^{-\underline{\underline{L}}^{\text{c.g.}}t}\right]_{nn}}_{\mu_n(t)}\mathring{\pi}_i^{(n)} + \underbrace{\left(\underline{e}^{(n,i)} - \mathring{\underline{\pi}}^{(n)}\right)^\dagger}_{\underline{\gamma}^{(n)^\dagger}(0)} \cdot e^{-(1-p_n)\mathring{\underline{\underline{L}}}^{(n)}t} \cdot \underline{e}^{(n,i)}e^{-p_n t} \tag{1.46}$$

$$= \left[e^{-\underline{\underline{L}}^{\text{c.g.}}t}\right]_{nn}\mathring{\pi}_i^{(n)} + \left(\left[e^{-(1-p_n)\mathring{\underline{\underline{L}}}^{(n)}t}\right]_{ii} - \mathring{\pi}_i^{(n)}\right)e^{-p_n t}, \tag{1.47}$$

and the return probabilities for starting on vertex i of the bridge vertices are:

$$\overset{\circ\!-\!\circ}{R}{}_i^{(B)}(t) = \left[e^{-\underline{\underline{L}}^{\text{c.g.}}t}\right]_{BB}\mathring{\pi}_i^{(B)} + \left(\left[e^{-(1-q_B)\mathring{\underline{\underline{L}}}^{(B)}t}\right]_{ii} - \mathring{\pi}_i^{(B)}\right)e^{-q_B t}, \tag{1.48}$$

The *rescaled* return probabilities are found by element-wise dividing these quantities by the global equilibrium distribution $\underline{\pi}$, which can be written in terms of $\underline{\mu}^{\pi} \equiv \lim_{t\to+\infty} \underline{\mu}(t)$ as $\underline{\pi} = \left(\bigoplus_{n=1}^{N_C} \mu_n^{\pi} \underline{\mathring{\pi}}^{(n)}\right) \oplus \mu_B^{\pi} \underline{\mathring{\pi}}^{(B)}$:

$$\overset{\circ\!-\!\circ}{\mathscr{R}}_i^{(n)}(t) \equiv \frac{R_i^{(n)}(t)}{\overset{\circ\!-\!\circ}{\pi}_i^{(n)}} = \frac{\left[e^{-\underline{\underline{L}}^{\text{c.g.}}t}\right]_{nn}}{\mu_n^{\pi}} + \frac{1}{\mu_n^{\pi}} \left(\frac{\left[e^{-(1-p_n)\underline{\underline{\mathring{L}}}^{(n)}t}\right]_{ii}}{\mathring{\pi}_i^{(n)}} - 1 \right) e^{-p_n t} \tag{1.49}$$

$$\overset{\circ\!-\!\circ}{\mathscr{R}}_i^{(B)}(t) \equiv \frac{R_i^{(B)}(t)}{\overset{\circ\!-\!\circ}{\pi}_i^{(B)}} = \frac{\left[e^{-\underline{\underline{L}}^{\text{c.g.}}t}\right]_{BB}}{\mu_B^{\pi}} + \frac{1}{\mu_B^{\pi}} \left(\frac{\left[e^{-(1-q_B)\underline{\underline{\mathring{L}}}^{(B)}t}\right]_{ii}}{\mathring{\pi}_i^{(B)}} - 1 \right) e^{-q_B t} \tag{1.50}$$

Defining the *coarse-grained* and *subgraph* rescaled return probabilities as

$$\mathscr{R}_n^{\text{c.g.}}(t) \equiv \frac{\left[e^{-\underline{\underline{L}}^{\text{c.g.}}(t)}\right]_{nn}}{\mu_n^{\pi}} \qquad \mathscr{R}_B^{\text{c.g.}}(t) \equiv \frac{\left[e^{-\underline{\underline{L}}^{\text{c.g.}}(t)}\right]_{BB}}{\mu_B^{\pi}} \tag{1.51}$$

$$\mathring{\mathscr{R}}_i^{(n)}(t) \equiv \frac{\left[e^{-\underline{\underline{\mathring{L}}}^{(n)}(t)}\right]_{ii}}{\mathring{\pi}_i^{(n)}} \qquad \mathring{\mathscr{R}}_i^{(B)}(t) \equiv \frac{\left[e^{-\underline{\underline{\mathring{L}}}^{(B)}(t)}\right]_{ii}}{\mathring{\pi}_i^{(B)}} \tag{1.52}$$

the full-graph rescaled return probabilities can be rewritten as

$$\overset{\circ\!-\!\circ}{\mathscr{R}}_i^{(n)}(t) = \mathscr{R}_n^{\text{c.g.}}(t) + \frac{1}{\mu_n^{\pi}} \left(\mathring{\mathscr{R}}_i^{(n)}((1-p_n)t) - 1 \right) e^{-p_n t} \tag{1.53}$$

$$\overset{\circ\!-\!\circ}{\mathscr{R}}_i^{(B)}(t) = \mathscr{R}_B^{\text{c.g.}}(t) + \frac{1}{\mu_B^{\pi}} \left(\mathring{\mathscr{R}}_i^{(B)}((1-q_B)t) - 1 \right) e^{-q_B t} \tag{1.54}$$

The rescaled return probabilities of subgraph equilibrium coupling model are thus determined by a simple combination of the coarse-grained reduced model with only $N_C + 1$ vertices and the subgraph behavior of each part of the graph partition.

We can now relate the instantaneous score $\overset{\circ\!-\!\circ}{\underline{s}}^{(*)}(t)$ to the subgraph and coarse grained scores $\underline{\mathring{s}}^{(*)}(t)$ and $s_*^{\text{c.g.}}(t)$ using the effective instantaneous score $\underline{s}^{\text{eff.}}$ defined analogously to $\underline{S}^{\text{eff.}}$

in equation (1.23):

$$s_i^{\text{eff.}} \equiv \lim_{T\to 0^+} \frac{1}{T} s_i(t)\big|_{p=1} = \sum_j \left|P_{ij}\left(\mathscr{R}_i(t) - \mathscr{R}_j(t)\right)\right|. \tag{1.55}$$

Using the relevant differences in rescaled full graph return probabilities expressed in terms of the subgraph and coarse-grained quantities

$$\begin{aligned}
\overset{\circ\!-\!\circ}{\mathscr{R}}{}_i^{(n)}(t) - \overset{\circ\!-\!\circ}{\mathscr{R}}{}_j^{(n)}(t) &= \frac{1}{\mu_n^\pi}\left(\mathring{\mathscr{R}}_i^{(n)}((1-p_n)t) - \mathring{\mathscr{R}}_j^{(n)}((1-p_n)t)\right)e^{-p_n t} \\
\overset{\circ\!-\!\circ}{\mathscr{R}}{}_i^{(n)}(t) - \overset{\circ\!-\!\circ}{\mathscr{R}}{}_i^{(B)}(t) &= \left(\mathscr{R}_n^{\text{c.g.}}(t) - \mathscr{R}_B^{\text{c.g.}}(t)\right) \\
&\quad + \left(\frac{\mathring{\mathscr{R}}_i^{(n)}((1-p_n)t) - 1}{\mu_n^\pi} e^{-p_n t} - \frac{\mathring{\mathscr{R}}_j^{(B)}((1-q_B)t) - 1}{\mu_B^\pi} e^{-q_B t}\right) \\
\overset{\circ\!-\!\circ}{\mathscr{R}}{}_i^{(B)}(t) - \overset{\circ\!-\!\circ}{\mathscr{R}}{}_j^{(B)}(t) &= \frac{1}{\mu_B^\pi}\left(\mathring{\mathscr{R}}_i^{(B)}((1-q_B)t) - \mathring{\mathscr{R}}_j^{(B)}((1-q_B)t)\right)e^{-q_B t}
\end{aligned}$$

and the expressions for the subgraph and coarse grained scores

$$\begin{aligned}
s_n^{\text{c.g.}}(t) &= p_n\left|\mathscr{R}_n^{\text{c.g.}}(t) - \mathscr{R}_B^{\text{c.g.}}(t)\right| \\
s_B^{\text{c.g.}}(t) &= \sum_{n=1}^{N_C} q_n\left|\mathscr{R}_n^{\text{c.g.}}(t) - \mathscr{R}_B^{\text{c.g.}}(t)\right| = \sum_{n=1}^{N_C} \frac{q_n}{p_n} s_n^{\text{c.g.}}(t) \\
\mathring{s}_i^{(n)}(t) &= \sum_{j=1}^{|V_n|}\left|\mathring{P}_{ij}^{(n)}\left(\mathring{\mathscr{R}}_i^{(n)}(t) - \mathring{\mathscr{R}}_j^{(n)}(t)\right)\right| \\
\mathring{s}_i^{(B)}(t) &= \sum_{j=1}^{|V_B|}\left|\mathring{P}_{ij}^{(B)}\left(\mathring{\mathscr{R}}_i^{(B)}(t) - \mathring{\mathscr{R}}_j^{(B)}(t)\right)\right|
\end{aligned}$$

some simple but tedious manipulations and the fact that $|x+y| \in \Big[\big||x|-|y|\big|, |x|+|y|\Big]_{\mathbb{R}} \in$

$\Big[|x|-|y|,|x|+|y|\Big]_{\mathbb{R}}$ yield:

$$\overset{\circ\!-\!\circ}{s}{}_i^{(n)}(t) = s_n^{\text{c.g.}}(t) + \frac{1-p_n}{\mu_n^\pi} e^{-p_n t} \mathring{s}_i^{(n)}((1-p_n)t) + \delta\overset{\circ\!-\!\circ}{s}{}_i^{(n)}(t) \tag{1.56}$$

$$\overset{\circ\!-\!\circ}{s}{}_i^{(B)}(t) = s_B^{\text{c.g.}}(t) + \frac{1-q_B}{\mu_B^\pi} e^{-q_B t} \mathring{s}_i^{(B)}((1-q_B)t) + \delta\overset{\circ\!-\!\circ}{s}{}_i^{(B)}(t) \tag{1.57}$$

where the spread of possible values are dictated by the $\delta\overset{\circ\!-\!\circ}{s}{}_i^{(*)}(t)$ terms:

$$\delta\overset{\circ\!-\!\circ}{s}{}_i^{(n)}(t) \in \left[-p_n \sum_{j=1}^{|V_B|} \mathring{\pi}_j^{(B)} \left|\Delta_{ij}^{(nB)}(t)\right|,\ +p_n \sum_{j=1}^{|V_B|} \mathring{\pi}_j^{(B)} \left|\Delta_{ij}^{(nB)}(t)\right|\right] \tag{1.58}$$

$$\delta\overset{\circ\!-\!\circ}{s}{}_i^{(B)}(t) \in \left[-\sum_{n=1}^{N_C} q_n \sum_{j=1}^{|V_n|} \mathring{\pi}_j^{(n)} \left|\Delta_{ji}^{(nB)}(t)\right|,\ +\sum_{n=1}^{N_C} q_n \sum_{j=1}^{|V_n|} \mathring{\pi}_j^{(n)} \left|\Delta_{ji}^{(nB)}(t)\right|\right] \tag{1.59}$$

$$\Delta_{ij}^{(nB)}(t) \equiv \frac{\mathring{\mathscr{R}}_i^{(n)}((1-p_n)t)-1}{\mu_n^\pi} e^{-p_n t} - \frac{\mathring{\mathscr{R}}_j^{(B)}((1-q_B)t)-1}{\mu_B^\pi} e^{-q_B t}. \tag{1.60}$$

These $\delta\overset{\circ\!-\!\circ}{s}{}^{(*)}$ terms capture the comparison between the internal subgraph dynamics of the N_C cluster partitions and the internal dynamics of the bridge points, so it should not be surprising that a simpler exact relation is not forthcoming in this context.

Let's pause and appreciate the results in equations (1.56) and (1.57): The instantaneous score for a coupled graph of N_C clusters and a set of bridge points can be expressed in terms of the scores of a coarse-grained model with only N_C+1 vertices, and the internal dynamics of the subgraphs corresponding to the clusters and the bridge, up to some smearing that reflects the differences of the internal dynamics of the cluster and the bridge subgraphs. Furthermore, since $s_B^{\text{c.g.}}(t) = \sum_n \frac{q_n}{p_n} s_n^{\text{c.g.}}(t)$, the coarse-grained model terms strongly push the instantaneous scores of the bridge vertices *above* the scores of the vertices in the clusters as long as the rate of transitioning *out* of the bridge and *into* cluster n, q_n, is large compared to the rate of transitioning *from* cluster n *into* the bridge, namely p_n.

1.6 Time-domain Arbitrary Weak-Coupling Limit

We will now examine the weak-coupling limit of a diffusion process on a graph with arbitrary bridge-cluster coupling structure. The diffusion process on such a graph with N_C clusters is governed by:

$$\partial_t \underline{v}^\dagger(t) = -\underline{v}^\dagger(t)\,\underline{\underline{L}} \qquad\qquad \underline{\underline{L}} = \underline{\underline{\mathbb{I}}} - \underline{\underline{P}} \tag{1.61}$$

$$\underline{\underline{P}} = \begin{bmatrix} (\underline{\underline{\mathbb{I}}} - \bcancel{\underline{\underline{P}}}_{\text{r.s.}}^{(1)})\underline{\underline{\mathring{P}}}^{(1)} & \underline{\underline{0}} & \underline{\underline{p}}^{(1)} \\ & \ddots & \vdots \\ \underline{\underline{0}} & (\underline{\underline{\mathbb{I}}} - \bcancel{\underline{\underline{P}}}_{\text{r.s.}}^{(N_C)})\underline{\underline{\mathring{P}}}^{(N_C)} & \underline{\underline{p}}^{(N_C)} \\ \underline{\underline{q}}^{(1)} \quad \cdots & \underline{\underline{q}}^{(N_C)} & (\underline{\underline{\mathbb{I}}} - \bcancel{\underline{\underline{q}}}_{\text{r.s.}}^{B})\underline{\underline{\mathring{P}}}^{(B)} \end{bmatrix} \tag{1.62}$$

$$\underline{\underline{L}} = \begin{bmatrix} \underline{\underline{\mathring{L}}}^{(1)} + \bcancel{\underline{\underline{P}}}_{\text{r.s.}}^{(1)}\underline{\underline{\mathring{P}}}^{(1)} & \underline{\underline{0}} & -\underline{\underline{p}}^{(1)} \\ & \ddots & \vdots \\ \underline{\underline{0}} & \underline{\underline{\mathring{L}}}^{(N_C)} + \bcancel{\underline{\underline{P}}}_{\text{r.s.}}^{(N_C)}\underline{\underline{\mathring{P}}}^{(N_C)} & -\underline{\underline{p}}^{(N_C)} \\ -\underline{\underline{q}}^{(1)} \quad \cdots & -\underline{\underline{q}}^{(N_C)} & \underline{\underline{\mathring{L}}}^{B} + \bcancel{\underline{\underline{q}}}_{\text{r.s.}}^{B}\underline{\underline{\mathring{P}}}^{(B)} \end{bmatrix} \tag{1.63}$$

Writing $\underline{v}^\dagger(t) = \left(\bigoplus_{n=1}^{N_C} \underline{v}^{(n)\dagger}(t)\right) \oplus \underline{v}^{(B)\dagger}(t)$, we find:

$$\partial_t \underline{v}^{(n)\dagger}(t) = -\underline{v}^{(n)\dagger}(t)\left(\underline{\underline{\mathring{L}}}^{(n)} + \bcancel{\underline{\underline{P}}}_{\text{r.s.}}^{(n)}\underline{\underline{\mathring{P}}}^{(n)}\right) + \underline{v}^{(B)\dagger}(t)\underline{\underline{q}}^{(n)} \tag{1.64}$$

$$\partial_t \underline{v}^{(B)^\dagger}(t) = \sum_{n=1}^{N_C} \underline{v}^{(n)^\dagger}(t)\underline{\underline{p}}^{(n)} - \underline{v}^{(B)^\dagger}(t)\left(\mathring{\underline{\underline{L}}}^{(B)} + \underline{\underline{q}}^{(B)}_{\text{r.s.}}\mathring{\underline{\underline{P}}}^{(B)}\right). \tag{1.65}$$

In the weak coupling limit of interest, the $\underline{\underline{p}}^{(*)}$ terms are small in magnitude but otherwise arbitrary, while the $\underline{\underline{q}}^{(*)}$ terms are unconstrained. To achieve this, first, replace all $\underline{\underline{p}}^{(*)}$ with $\eta\underline{\underline{p}}^{(*)}$, where $\eta \in (0,1]_{\mathbb{R}}$. Then we will perform our analysis by looking for the limiting behavior as $\eta \to 0^+$.

1.6.1 Timescale separation

Defining a new timescale through $\tau \equiv \eta t$ and dividing through by η allows us to separate terms more clearly:

$$\partial_\tau \underline{v}^{(n)^\dagger}(\tau) = -\underline{v}^{(n)^\dagger}(\tau)\left(\frac{1}{\eta}\mathring{\underline{\underline{L}}}^{(n)} + \underline{\underline{p}}^{(n)}_{\text{r.s.}}\mathring{\underline{\underline{P}}}^{(n)}\right) + \underline{v}^{(B)^\dagger}\frac{1}{\eta}\underline{\underline{q}}^{(n)} \tag{1.66}$$

$$\partial_\tau \underline{v}^{B^\dagger}(\tau) = \sum_{n=1}^{N_C} \underline{v}^{(n)^\dagger}(\tau)\underline{\underline{p}}^{(n)} - \underline{v}^{(B)^\dagger}(\tau)\left(\frac{1}{\eta}\mathring{\underline{\underline{L}}}^{(B)} + \frac{1}{\eta}\underline{\underline{q}}^{(B)}_{\text{r.s.}}\mathring{\underline{\underline{P}}}^{(B)}\right) \tag{1.67}$$

Rearranging to group most of the $\frac{1}{\eta}$ terms on the left with the time derivative, and using the notation for a left-acting derivative as $f(x)\overleftarrow{\partial_x} \equiv \partial_x f(x)$, we find:

$$\underline{v}^{(n)^\dagger}(\tau)\left[\overleftarrow{\partial_\tau} + \frac{1}{\eta}\mathring{\underline{\underline{L}}}^{(n)}\right] = \underline{v}^{(B)^\dagger}(\tau)\frac{1}{\eta}\underline{\underline{q}}^{(n)} - \underline{v}^{(n)^\dagger}\underline{\underline{p}}^{(n)}_{\text{r.s.}}\mathring{\underline{\underline{P}}}^{(n)} \tag{1.68}$$

$$\underline{v}^{B^\dagger}(\tau)\left[\overleftarrow{\partial_\tau} + \frac{1}{\eta}\left(\mathring{\underline{\underline{L}}}^{(B)} + \underline{\underline{q}}^{(B)}_{\text{r.s.}}\mathring{\underline{\underline{P}}}^{(B)}\right)\right] = \sum_{m=1}^{N_C} \underline{v}^{(m)^\dagger}(\tau)\underline{\underline{p}}^{(m)} \tag{1.69}$$

1.6.2 Decoupling Clusters from the Bridge

Now, for brevity, introduce the notation for a *lossy laplacian* as:

$$\widetilde{\underline{\underline{L}}}^{(B)} \equiv \mathring{\underline{\underline{L}}}^{(B)} + \underline{\underline{q}}^{(B)}_{\text{r.s.}}\mathring{\underline{\underline{P}}}^{(B)} = \underline{\underline{\mathbb{I}}} - (\underline{\underline{\mathbb{I}}} - \underline{\underline{q}}^{(B)}_{\text{r.s.}})\mathring{\underline{\underline{P}}}^{(B)} \tag{1.70}$$

where the structure of $\left[\underline{\underline{\gamma}}^{(B)}_{\text{r.s.}}\right]_{ii} \in [0,1)$ guarantees that $\underline{\underline{\widetilde{L}}}^{(B)}$ has strictly positive eigenvalues[10] (see the appendix for a discussion of the properties of these lossy Laplacians). With this, equation (1.69) can be formally solved by using the Green's function for the operator $\left[\overleftarrow{\partial}_\tau + \frac{1}{\eta}\underline{\underline{\widetilde{L}}}^{(B)}\right]$ satisfying the initial condition $\underline{v}^{(B)^\dagger}(0^+) = \underline{v}^{(B)^\dagger}_{\text{i.c.}}$:

$$\underline{v}^{(B)^\dagger}(\tau) = \underline{v}^{(B)^\dagger}_{\text{i.c.}} e^{-\frac{1}{\eta}\underline{\underline{\widetilde{L}}}^{(B)}\tau} + \int_0^\infty d\tau' \left(\sum_{m=1}^{N_C} \underline{v}^{(n)^\dagger}(\tau')\underline{\underline{p}}^{(m)}\right) e^{-\frac{1}{\eta}\underline{\underline{\widetilde{L}}}^{(B)}(\tau-\tau')}\Theta(\tau-\tau') \tag{1.71}$$

where $\Theta(*)$ is the standard Heaviside step function.

Because the matrix exponential decays rapidly as τ increases for $\tau > \tau'$ due to the factor of $\frac{1}{\eta}$ and the strictly positive eigenvalues of $\underline{\underline{\widetilde{L}}}^{(B)}$, this integral can be evaluated by Taylor expanding the term in parentheses about the point $\tau' = \tau$ and integrating order by order. Using the results of section B.1, we have:

$$\begin{aligned}
\underline{v}^{(B)^\dagger}(\tau) &= \overbrace{\left(\sum_{m=1}^{N_C} \underline{v}^{(m)^\dagger}(\tau)\underline{\underline{p}}^{(m)}\right) \frac{\eta\left(\underline{\underline{\widetilde{L}}}^{(B)}\right)^{-1}}{\underline{\underline{\mathbb{I}}} + \eta\overleftarrow{\partial}_\tau\left(\underline{\underline{\widetilde{L}}}^{(B)}\right)^{-1}}}^{\text{long-time behavior}} \\
&\quad + \underbrace{\left[\underline{v}^{(B)^\dagger}_{\text{i.c.}} - \left(\sum_{m=1}^{N_C} \underline{v}^{(m)^\dagger}(\tau)\underline{\underline{p}}^{(m)}\right) \frac{\eta\left(\underline{\underline{\widetilde{L}}}^{(B)}\right)^{-1}}{\underline{\underline{\mathbb{I}}} + \eta\overleftarrow{\partial}_\tau\left(\underline{\underline{\widetilde{L}}}^{(B)}\right)^{-1}}\Bigg|_{\tau=0}\right]}_{\equiv \underline{b}^\dagger_{\text{i.t.}} \text{ is the source of the bridge initial transient}} e^{-\frac{1}{\eta}\underline{\underline{\widetilde{L}}}^{(B)}\tau} \\
&= \left(\sum_{m=1}^{N_C} \underline{v}^{(m)^\dagger}(\tau)\underline{\underline{p}}^{(m)}\right) \frac{\eta\left(\underline{\underline{\widetilde{L}}}^{(B)}\right)^{-1}}{\underline{\underline{\mathbb{I}}} + \eta\overleftarrow{\partial}_\tau\left(\underline{\underline{\widetilde{L}}}^{(B)}\right)^{-1}} + \underline{b}^\dagger_{\text{i.t.}} e^{-\frac{1}{\eta}\underline{\underline{\widetilde{L}}}^{(B)}\tau}
\end{aligned} \tag{1.72}$$

where $\underline{\underline{\widetilde{L}}}^{(B)}$ is guaranteed to be invertible due to its strictly positive eigenvalues, and $\frac{\eta(\underline{\underline{\widetilde{L}}}^{(B)})^{-1}}{\underline{\underline{\mathbb{I}}}+\eta\,\overleftarrow{\partial}_\tau(\underline{\underline{\widetilde{L}}}^{(B)})^{-1}}$ is formal shorthand for the geometric series expansion of the denominator. Note there is no

[10]You might ask how we can be sure that $\underline{\underline{\gamma}}^{(B)}_{\text{r.s.}}$ has the correct structure to ensure that $\underline{\underline{\widetilde{L}}}^{(B)}$ has strictly positive eigenvalues, i.e. at least one non-zero entry on each of the connected components of the bridge vertex subset V_B? Because the bridge vertices are precisely those with rapid outflow to the cluster vertices, so each of the connected components of the bridge subgraph *must* be connected to at least one of the clusters, guaranteeing the condition we need for any reasonable identification of the bridge vertex subset V_B.

ordering ambiguity from the fraction notation here, as every part of the denominator individually commutes with the numerator of this expression.

Using equation (1.72) to eliminate $\underline{v}^{(B)^\dagger}$ from equation (1.68) yields:

$$\begin{aligned}
&\underline{v}^{(n)^\dagger}(\tau)\left[\overleftarrow{\partial}_\tau + \frac{1}{\eta}\mathring{\underline{\underline{L}}}^{(n)}\right] - \underline{b}^\dagger_{\text{i.t.}}\frac{1}{\eta}e^{-\frac{1}{\eta}\widetilde{\underline{\underline{L}}}^{(B)}\tau}\underline{\underline{q}}^{(n)} \\
&\qquad = \sum_{m=1}^{N_C}\underline{v}^{(m)^\dagger}(\tau)\underline{\underline{p}}^{(m)}\frac{(\widetilde{\underline{\underline{L}}}^{(B)})^{-1}}{\underline{\underline{\mathbb{I}}}+\eta\,\overleftarrow{\partial}_\tau(\widetilde{\underline{\underline{L}}}^{(B)})^{-1}}\underline{\underline{q}}^{(n)} - \underline{v}^{(n)^\dagger}(\tau)\cancel{\underline{\underline{p}}}^{(n)}_{\text{r.s.}}\mathring{\underline{\underline{P}}}^{(n)}
\end{aligned}\tag{1.73}$$

where all terms that directly contribute to rapid time-variation (due to various factors of $\frac{1}{\eta}$) are grouped on the left. Defining the left-acting matrix-differential *bridge-mediated distributor* for brevity:

$$\begin{aligned}
\overleftarrow{\underline{\underline{D}}}^{m\to n} &\equiv \underline{\underline{p}}^{(m)}\frac{(\widetilde{\underline{\underline{L}}}^{(B)})^{-1}}{\underline{\underline{\mathbb{I}}}+\eta\,\overleftarrow{\partial}_\tau(\widetilde{\underline{\underline{L}}}^{(B)})^{-1}}\underline{\underline{q}}^{(n)} \\
&= \underline{\underline{p}}^{(m)}(\widetilde{\underline{\underline{L}}}^{(B)})^{-1}\underline{\underline{q}}^{(n)} + \eta\left(-\overleftarrow{\partial}_\tau\underline{\underline{p}}^{(m)}(\widetilde{\underline{\underline{L}}}^{(B)})^{-2}\underline{\underline{q}}^{(n)}\right) + \eta^2\left(\overleftarrow{\partial}^2_\tau\underline{\underline{p}}^{(m)}(\widetilde{\underline{\underline{L}}}^{(B)})^{-3}\underline{\underline{q}}^{(n)}\right) + \cdots \\
&\equiv \underline{\underline{D}}_0^{m\to n} + \eta\overleftarrow{\underline{\underline{D}}}_1^{m\to n} + \eta^2\overleftarrow{\underline{\underline{D}}}_2^{m\to n} + \cdots, \qquad \overleftarrow{\underline{\underline{D}}}_r^{m\to n} \equiv (-\overleftarrow{\partial}_\tau)^r\underline{\underline{p}}^{(m)}\left(\widetilde{\underline{\underline{L}}}^{(B)}\right)^{-(r+1)}\underline{\underline{q}}^{(n)},
\end{aligned}\tag{1.74}$$

and including the initial condition for $\underline{v}^{(n)^\dagger}$ explicitly with a delta function, we can rewrite equation (1.73) as:

$$\begin{aligned}
&\underline{v}^{(n)^\dagger}(\tau)\left[\overleftarrow{\partial}_\tau + \frac{1}{\eta}\mathring{\underline{\underline{L}}}^{(n)}\right] - \underline{v}^{(n)^\dagger}_{\text{i.c.}}\delta(\tau) - \underline{b}^\dagger_{\text{i.t.}}\frac{1}{\eta}e^{-\frac{1}{\eta}\widetilde{\underline{\underline{L}}}^{(B)}\tau}\underline{\underline{q}}^{(n)} \\
&\qquad\qquad = \sum_{m=1}^{N_C}\underline{v}^{(m)^\dagger}(\tau)\overleftarrow{\underline{\underline{D}}}^{m\to n} - \underline{v}^{(n)^\dagger}(\tau)\cancel{\underline{\underline{p}}}^{(n)}_{\text{r.s.}}\mathring{\underline{\underline{P}}}^{(n)}.
\end{aligned}\tag{1.75}$$

This is an exact closed system of N_C equations which is the result of integrating out the degrees of freedom in the boundary vertices. The penalty we pay for this is hidden in $\underline{b}^\dagger_{\text{i.t.}}$ and $\overleftarrow{\underline{\underline{D}}}^{*\to *}$, namely arbitrarily high order derivatives $\overleftarrow{\partial}_\tau$ and powers of $\left(\widetilde{\underline{\underline{L}}}^{(B)}\right)^{-1}$. Fortunately, these high-order contributions are controlled by positive powers of our coupling scaling parameter η.

1.6.3 Separation of the Cluster Equilibrium Subspace

We are now notationally and conceptually prepared to show the main result of this development: in the weak-coupling limit, $\underline{v}^{\dagger}(\tau)$ should behave as

$$\lim_{\eta\to 0^+} \underline{v}^{\dagger}(\tau > 0) = \mu^{(1)}(\tau)\underline{\mathring{\pi}}^{(1)^{\dagger}} \oplus \cdots \oplus \mu^{(N_C)}(\tau)\underline{\mathring{\pi}}^{(N_C)^{\dagger}} \oplus \underline{v}^{(B)^{\dagger}}(\tau), \tag{1.76}$$

where the behavior on the boundary vertices, $\underline{v}^{(B)^{\dagger}}(\tau)$, is left unspecified for now. To derive this, recall the definition of the weak-coupling limit to be the $\eta \approx 0^+$ regime, where equation (1.75) takes the form

$$\begin{aligned} &\underline{v}^{(n)^{\dagger}}(\tau)\left[\overleftarrow{\partial}_{\tau} + \frac{1}{\eta}\underline{\underline{\mathring{L}}}^{(n)}\right] - \underline{v}^{(n)^{\dagger}}_{\text{i.c.}}\delta(\tau) - \underline{v}^{(B)^{\dagger}}_{\text{i.c.}}\frac{1}{\eta}e^{-\frac{1}{\eta}\underline{\underline{\widetilde{L}}}^{(B)}\tau}\underline{\underline{q}}^{(n)} \\ &\qquad = \sum_{m=1}^{N_C}\underline{v}^{(m)^{\dagger}}(\tau)\underline{\underline{q}}^{(m)}\left(\underline{\underline{\widetilde{L}}}^{(B)}\right)^{-1}\underline{\underline{p}}^{(n)} - \underline{v}^{(n)^{\dagger}}(\tau)\underline{\underline{\cancel{p}}}^{(n)}_{\text{r.s.}}\underline{\underline{\mathring{P}}}^{(n)}. \end{aligned} \tag{1.77}$$

This has the formal solution

$$\begin{aligned} \underline{v}^{(n)^{\dagger}}(\tau) = {}&\underline{v}^{(n)^{\dagger}}_{\text{i.c.}}e^{-\frac{1}{\eta}\underline{\underline{\mathring{L}}}^{(n)}\tau} + \int_0^{\tau}d\tau'\,\underline{v}^{(B)^{\dagger}}_{\text{i.c.}}\frac{1}{\eta}e^{-\frac{1}{\eta}\underline{\underline{\widetilde{L}}}^{(B)}\tau'}\underline{\underline{q}}^{(n)}e^{-\frac{1}{\eta}\underline{\underline{\mathring{L}}}^{(n)}(\tau-\tau')} \\ &+ \int_0^{\tau}d\tau'\,\underbrace{\left(\sum_{m=1}^{N_C}\underline{v}^{(m)^{\dagger}}(\tau')\underline{\underline{q}}^{(m)}\left(\underline{\underline{\widetilde{L}}}^{(B)}\right)^{-1}\underline{\underline{p}}^{(n)} - \underline{v}^{(n)^{\dagger}}(\tau')\underline{\underline{\cancel{p}}}^{(n)}_{\text{r.s.}}\underline{\underline{\mathring{P}}}^{(n)}\right)}_{\equiv \underline{f}^{(n)^{\dagger}}(\tau')}e^{-\frac{1}{\eta}\underline{\underline{\mathring{L}}}^{(n)}(\tau-\tau')}. \end{aligned} \tag{1.78}$$

In the region where $\eta \approx 0^+$ and any non-infinitesimal $\tau > 0$,[11] we can perform the integral involving $\underline{v}^{(B)^{\dagger}}_{\text{i.c.}}$, yielding:

$$\underline{v}^{(n)^{\dagger}}(\tau) \xrightarrow[\eta\approx 0^+]{\tau>0} \underline{v}^{(n)^{\dagger}}_{\text{i.c.}}\underline{\underline{\mathscr{P}}}^{(n)}_{\pi} + \underline{v}^{(b)^{\dagger}}_{\text{i.c.}}\left(\underline{\underline{\widetilde{L}}}^{(B)}\right)^{-1}\underline{\underline{q}}^{(n)}\underline{\underline{\mathscr{P}}}^{(n)}_{\pi} + \int_0^{\tau}d\tau'\,\underline{f}^{(n)^{\dagger}}(\tau')e^{-\frac{1}{\eta}\underline{\underline{\mathring{L}}}^{(n)}(\tau-\tau')}, \tag{1.79}$$

[11] Henceforth, in the $\eta \approx 0^+$ regime, stating $\tau > 0$ should be considered synonymous with the regime where matrix exponentials of the form $e^{-\frac{1}{\eta}(*)\tau}$ can be replaced with their infinite-τ limits.

where $\underset{=}{\mathscr{P}}_{\pi}^{(n)} \equiv \underline{\mathbb{1}}^{(n)}\underline{\mathring{\pi}}^{(n)}$. Differentiating annihilates the first two terms, and the Leibniz integral rule on the third term yields:

$$
\begin{aligned}
\partial_\tau \underline{v}^{(n)^\dagger}(\tau) &\xrightarrow[\eta\approx 0^+]{\tau>0} \underline{f}^{(n)^\dagger}(\tau) + \int_0^\tau d\tau'\, \underline{f}^{(n)^\dagger}(\tau') e^{-\frac{1}{\eta}\underset{=}{\mathring{L}}^{(n)}(\tau-\tau')} \left(-\frac{1}{\eta}\underset{=}{\mathring{L}}^{(n)}\right) \\
&= \underline{f}^{(n)^\dagger}(\tau) - \underline{f}^{(n)^\dagger}(\tau)\left(\underset{=}{\mathring{L}}^{(n)}\right)^{+}\underset{=}{\mathring{L}}^{(n)} \\
&= \underline{f}^{(n)^\dagger}(\tau)\underbrace{\left(\underset{=}{\mathbb{I}}^{(n)} - \left(\underset{=}{\mathring{L}}^{(n)}\right)^{+}\underset{=}{\mathring{L}}^{(n)}\right)}_{=\underset{=}{\mathscr{P}}_{\pi}^{(n)}} = \underline{f}^{(n)^\dagger}(\tau)\underset{=}{\mathscr{P}}_{\pi}^{(n)},
\end{aligned} \tag{1.80}
$$

where we have used the fact that $\underline{f}^{(n)^\dagger}(\tau)$ can be treated as a constant in the integral[12] because of the rapid exponential decay of $e^{-\frac{1}{\eta}\underset{=}{\mathring{L}}^{(n)}(\tau-\tau')}$, and $\underset{=}{\mathbb{I}}^{(n)} - \left(\underset{=}{\mathring{L}}^{(n)}\right)^{+}\underset{=}{\mathring{L}}^{(n)} = \underset{=}{\mathscr{P}}_{\pi}^{(n)}$ due to the properties of the the Moore-Penrose pseudoinverse and the fact that the subgraph of the n^{th} cluster is connected.

Having established that $\partial_\tau \underline{v}^{(n)^\dagger}(\tau) \xrightarrow[\eta\approx 0^+]{\tau>0} \left(\underline{f}^{(n)^\dagger}(\tau)\underline{\mathbb{1}}^{(n)}\right)\underline{\mathring{\pi}}^{(n)}$, we can integrate back from $\tau = +\infty$ to find:

$$
\begin{aligned}
\underline{v}^{(n)^\dagger}(\tau) &= \underline{v}^{(n)^\dagger}(+\infty) - \int_\tau^{+\infty} d\tau\, \partial_\tau \underline{v}^{(n)^\dagger}(\tau) \\
&\xrightarrow[\eta\approx 0^+]{\tau>0} \mu_\pi^{(n)}\underline{\mathring{\pi}}^{(n)} - \left(\int_\tau^{+\infty} d\tau\, \underline{f}^{(n)^\dagger}(\tau)\underline{\mathbb{1}}^{(n)}\right)\underline{\mathring{\pi}}^{(n)} \propto \underline{\mathring{\pi}}^{(n)}
\end{aligned} \tag{1.81}
$$

where the fact that $\underline{v}^{(n)^\dagger}(+\infty) = \underline{\overset{\infty}{\pi}}^{(n)} = \mu_\pi^{(n)}\underline{\mathring{\pi}}^{(n)} + O(\eta)$ has been used, $\mu_\pi^{(n)}$ is the equilibrium probability mass in cluster n, and the terms scaling with η have been thrown away. Combining this with the $\eta \to 0^+$ limit of equation (1.72) then gives the full weak-coupling limit expression for $\underline{v}^\dagger(\tau > 0)$ to leading order in η as:

$$
\underline{v}^\dagger(\tau) \xrightarrow[\eta\approx 0^+]{\tau>0} \left(\bigoplus_{n=1}^{N_C} \mu^{(n)}(\tau)\underline{\mathring{\pi}}^{(n)^\dagger}\right) \oplus \left(\eta \sum_{m=1}^{N_C} \mu^{(m)}(\tau)\underline{\mathring{\pi}}^{(m)}\underset{=}{p}^{(m)}\left(\underset{=}{\widetilde{L}}^{(B)}\right)^{-1}\right). \tag{1.82}
$$

[12] $\underline{f}^{(n)^\dagger}(\tau)$ is the result of a fixed (η-independent in the $\eta \approx 0^+$ regime) linear operator acting on $\underline{v}^\dagger(\tau)$, so it must be a sum of decaying exponentials in τ, with clear timescale separation inherited from the spectrum of $\underset{=}{L}\big|_{p^*\to\eta p^*}$. This justifies replacing $\underline{f}^{(n)^\dagger}(\tau')$ with $\underline{f}^{(n)^\dagger}(\tau)$ in the integral, leading to the stated result.

The time evolution of the cluster probability masses $\mu^{(n)}(\tau)$ can be recovered by inserting this expression into the basic time evolution equation for the diffusion process, $\partial_\tau \underline{v}^\dagger(\tau) = -\underline{v}^\dagger(\tau)\frac{1}{\eta}\underline{\underline{L}}$, yielding the coarse-grained effective time evolution equation:

$$\partial_\tau \underline{\mu}^\dagger(\tau) = -\underline{\mu}^\dagger(\tau)\underline{\underline{L}}_{\mu}^{\text{c.g.}}, \tag{1.83}$$

where $\underline{\mu}^\dagger(\tau) \in \left(\mathbb{R}^{N_C}\right)^\dagger$ is the vector of cluster probability masses with elements $\left[\underline{\mu}^\dagger(\tau)\right]_n = \mu^{(n)}(\tau)$, and $\underline{\underline{L}}_{\mu}^{\text{c.g.}} \in \mathbb{R}^{N_C \times N_C}$ is the effective Laplacian on this graph, with elements

$$\left[\underline{\underline{L}}_{\mu}^{\text{c.g.}}\right]_{nm} \equiv \underline{\mathring{\pi}}^{(n)^\dagger} \underline{\underline{p}}^{(m)} \left(\widetilde{\underline{\underline{L}}}^{(B)}\right)^{-1} \underline{q}_{\text{r.s.}}^{(m)} - \delta_{nm} \underline{\mathring{\pi}}^{(m)^\dagger} \underline{p}_{\text{r.s.}}^{(m)}. \tag{1.84}$$

Note that this coarse-grained Laplacian governing the cluster probability masses $\underline{\mu}^\dagger(\tau)$ does not have the special mostly-diagonal structure of the coarse-grained Laplacian in the N-cluster subgraph equilibrium coupling model of section 1.5.3, but rather is more general.

This chapter, in part, is being prepared for submission for publication of the material. The dissertation author was the primary investigator and author of this material. Professor Alexander Cloninger is the sole co-author of the material being prepared for submission for publication.

Chapter 2

Graph Clustering Numerics

2.1 Graph Transition and Laplacian Matrices

A connected undirected weighted graph $G=(V,E,w)$ has both a random-walk transition matrix $\underline{\underline{P}}^{\text{r.w.}} \equiv \underline{\underline{\diagdown\!\!\!\!d}}^{-1}\underline{\underline{W}}$ and a symmetric transition matrix $\underline{\underline{P}}^{\text{sym.}} \equiv \underline{\underline{\diagdown\!\!\!\!d}}^{-\frac{1}{2}}\underline{\underline{W}}\,\underline{\underline{\diagdown\!\!\!\!d}}^{-\frac{1}{2}}$. Since $\underline{\underline{P}}^{\text{sym.}}$ is a real symmetric matrix, it has an eigendecomposition in terms of real orthonormal eigenvectors $\underline{x}_n \in \mathbb{R}^{|V|}$ and real eigenvalues $\lambda_n^{(P)} \in \mathbb{R}$:

$$\underline{\underline{P}}^{\text{sym.}} = \sum_{i=1}^{|V|} \lambda_i^{(P)} \underline{x}_i \underline{x}_i^{\dagger} \qquad \underline{x}_i^{\dagger} \cdot \underline{x}_j = \delta_{ij} \tag{2.1}$$

The eigen-decomposition of the random-walk transition matrix then trivially follows from the fact that $\underline{\underline{P}}^{\text{sym.}}$ and $\underline{\underline{P}}^{\text{r.w.}} = \underline{\underline{\diagdown\!\!\!\!d}}^{-\frac{1}{2}}\underline{\underline{P}}^{\text{sym.}}\underline{\underline{\diagdown\!\!\!\!d}}^{+\frac{1}{2}}$ are similar matrices:

$$\underline{\underline{P}}^{\text{r.w.}} = \sum_{i=1}^{|V|} \lambda_i^{(P)} \underline{r}_i \underline{l}_i^{\dagger} \qquad \underline{r}_i \equiv \underline{\underline{\diagdown\!\!\!\!d}}^{-\frac{1}{2}}\underline{x}_i, \quad \underline{l}_i \equiv \underline{\underline{\diagdown\!\!\!\!d}}^{+\frac{1}{2}}\underline{x}_i, \quad \underline{l}_i^{\dagger} \cdot \underline{r}_j = \delta_{ij} \tag{2.2}$$

where $\left\{ \underline{r}_i \,\middle|\, \underline{\underline{P}}^{\text{r.w.}}\underline{r}_i = \lambda_i^{(P)}\underline{r}_i,\ i \in [1,|V|]_{\mathbb{Z}} \right\}$ and $\left\{ \underline{l}_i \,\middle|\, \underline{l}_i^{\dagger}\underline{\underline{P}}^{\text{r.w.}} = \lambda_i^{(P)}\underline{l}_i^{\dagger},\ i \in [1,|V|]_{\mathbb{Z}} \right\}$ are the right and left eigenvectors of $\underline{\underline{P}}^{\text{r.w.}}$, respectively.

The random walk Laplacian $\underline{\underline{L}}^{\text{r.w.}} \equiv \underline{\underline{\mathbb{I}}} - \underline{\underline{P}}^{\text{r.w.}}$ has the same eigenvectors as $\underline{\underline{P}}^{\text{r.w.}}$, with its eigenvalues shifted and reversed as:

$$\underline{\underline{L}}^{\text{r.w.}} = \sum_{i=1}^{|V|} \lambda_i^{(L)} \underline{r}_i \underline{l}_i^\dagger \qquad \lambda_i^{(L)} \equiv 1 - \lambda_i^{(P)}. \tag{2.3}$$

Since the eigenvalues of the transition matrices $\underline{\underline{P}}^{\text{r.w.}}$ and $\underline{\underline{P}}^{\text{sym.}}$ lie in the interval $[-1,1]_{\mathbb{R}}$ (easily proven with the Gershgorin Circle Theorem, see Bell 1965), the eigenvalues of $\underline{\underline{L}}$ lie in the interval $[0,2]_{\mathbb{R}}$. Henceforth, $\lambda_i \equiv \lambda_i^{(L)}$ will be assumed to refer to the eigenvalues of a *Laplacian*, not a transition matrix, unless otherwise specified.

It is sometimes convenient to define a *lazy random walk*[1] transition matrix by averaging $\underline{\underline{P}}$ with the identity matrix:

$$\underline{\underline{P}}_{\text{lazy}} \equiv \frac{1}{2}\left(\underline{\underline{\mathbb{I}}} + \underline{\underline{P}}\right) \tag{2.4}$$

Because the eigenvalues of lazy random walks are guaranteed to lie in the interval $[0,1]_{\mathbb{R}}$, they allow an exact correspondence between a discrete time process accessed by integer powers of $\underline{\underline{P}}_{\text{lazy}}$, and a continuous time process defined by

$$\underline{\underline{P}}^t_{\text{lazy}} = e^{-t\underline{\underline{L}}_{\text{lazy}}} \qquad \underline{\underline{L}}_{\text{lazy}} \equiv -\ln\left(\underline{\underline{P}}_{\text{lazy}}\right) \qquad t \in [0,+\infty)_{\mathbb{R}} \tag{2.5}$$

where the eigenvalues of $\underline{\underline{L}}_{\text{lazy}}$ are in the interval $[0,+\infty)_{\mathbb{R}}$.[2]

This correspondence between $\underline{\underline{P}}^t_{\text{lazy}}$ and $e^{-t\underline{\underline{L}}_{\text{lazy}}}$ is convenient for two reasons. First, this discrete/continuous time correspondence allows simple validation of the behavior of numerical routines based on eigendecompositions of $\underline{\underline{L}}$ used to define continuous time processes. Second, certain computations[3] are more efficiently done by performing a small number of matrix-vector multiplications to compute the action of $\underline{\underline{P}}^t_{\text{lazy}}$ on a vector rather than resorting to a (partial or full) eigendecomposition to define a continuous-time process. For these reasons, the numerical

[1] See Oliveira and Peres n.d. for discussion of the behavior of lazy random walks

[2] Strictly speaking, if $\underline{\underline{P}}_{\text{lazy}}$ has a zero eigenvalue, then $\underline{\underline{L}}_{\text{lazy}}$ will have an infinite eigenvalue. This situation can be avoided by adding a self-loop with an arbitrarily small positive weight on each vertex to enforce $\lambda_i^{(P_{\text{lazy}})} \in (0,1]_{\mathbb{R}}$.

[3] In particular, computations at short times when $e^{-t\underline{\underline{L}}}$ has significant contributions from many eigenvectors are computationally expensive in an eigendecomposition formulation.

results presented here generally use *lazy* random walk transition and Laplacian matrices.

2.2 Computing the Score

The general version of the score,

$$\underline{S} = \int_0^{+\infty} dt\, w(t)\, \underline{\mathscr{RN}}_p \left(\nabla_{\underline{\underline{K}}(T)} \underline{\mathscr{R}}(t) \right) \qquad \underline{\underline{K}}(T) \equiv e^{-T \underline{\underline{L}}^{\text{r.w.}}} \tag{2.6}$$

contains multiple tunable elements: $w(t) : \mathbb{R}_+ \to \mathbb{R}_+$, the time weighting function used to focus the score's attention on specific time regimes; p in the matrix row norm $\underline{\mathscr{RN}}_p(*)$, used to control which p-norm we use to define the scalar magnitude of the difference between $\mathscr{R}_i(t)$ and nearby values of the $\underline{\mathscr{R}}(t)$; and the diffusion time $T \in [0, +\infty)$, used to define how far on the graph we compare values in the graph difference operator $\nabla_{\underline{\underline{K}}(T)}$.

For the sake of simplicity, computational efficiency, and to avoid over-tuning our algorithm to the specific datasets under consideration, we make the following *fixed* choices for all numerical results that follow:

- $w(t) = \Theta(t - t_{\text{min}})$, so all timescales are weighted equally and minimal tuning is done to the data at hand. In practice, a truncated eigendecomposition of $\underline{\underline{L}}$ is used with only the slowest-evolving modes (i.e. those with the smallest eigenvalues $\lambda_i^{(L)}$) retained. The error this truncation induces decreases exponentially above $t_{\text{threshold}} \equiv$
- $p = 1$, which is in some sense the *natural* choice in this context since $\underline{\mathscr{RN}}_{p=1} \left(\underline{\underline{K}}(T) \right) = \underline{\mathbb{1}}$; and
- Replace $\underline{\underline{K}}(T)$ with $\underline{\underline{P}}^{\text{r.w.}}$, which is equivalent to using the quantity $\lim_{T \to 0^+} \frac{1}{T} \underline{S}$ as our score.

These choices mean that our effective score function in this chapter is:

$$S_i^{\text{eff.}} \equiv \int_0^{+\infty} dt\ \Theta(t - t_{\text{min}})\ \left[\underline{\mathscr{R}\mathscr{N}}_{p=1}\Big(\underline{\nabla}_{\underline{\underline{P}}^{\text{r.w.}}} \underline{\mathscr{R}}(t)\Big)\right]_i = \int_{t_{\text{min}}}^{+\infty} dt\ \sum_{j=1}^{|V|} P_{ij}^{\text{r.w.}} \left|\mathscr{R}_i(t) - \mathscr{R}_j(t)\right| \quad (2.7)$$

where we have pulled $P_{ij}^{\text{r.w.}}$ out of the absolute value sign since its elements are non-negative. The use of $\underline{\underline{P}}^{\text{r.w.}}$ instead of $\underline{\underline{K}}(T)$ in particular allows computations to take advantage of the sparsity structure of the weight matrix, since if $\underline{\underline{W}}$ is sparse, $\underline{\underline{P}}^{\text{r.w.}}$ is as well, but $\underline{\underline{K}}(T)$ is not in general. This is important for large problems, as often $\underline{\underline{P}}^{\text{r.w.}}$ will be extremely sparse, and $\underline{\mathscr{R}}(t)$ can be efficiently approximated from its the top few eigenvectors which are efficiently computable (see Lehoucq, Sorensen, and Yang 1998).

To compute the effective score $\underline{S}^{\text{eff.}}$, we need only form the matrix $\underline{\underline{P}}^{\text{r.w.}}$, be able to efficiently approximate $\underline{\mathscr{R}}(t)$ for arbitrary $t \in [0, +\infty)_{\mathbb{R}}$, and evaluate the integral in equation (2.7). The rescaled return probability $\underline{\mathscr{R}}(t)$ is formed from the element-wise division of the the return probability $R_i(t) = \left[e^{-t\underline{\underline{L}}^{\text{r.w.}}}\right]_{ii}$ by the ergodic distribution $\underline{\pi} \equiv \underline{d}/D,\ D \equiv \underline{d}^{\dagger}\underline{\mathbb{1}}$. The ergodic distribution is trivial to compute, so we focus our attention on computing $R_i(t)$.

The power series definition of the matrix exponential implies that

$$e^{-t\underline{\underline{L}}^{\text{r.w.}}} = e^{-t\underline{\underline{\bcancel{d}}}^{-\frac{1}{2}}\underline{\underline{L}}^{\text{sym.}}\underline{\underline{\bcancel{d}}}^{+\frac{1}{2}}} = \underline{\underline{\bcancel{d}}}^{-\frac{1}{2}} e^{-t\underline{\underline{L}}^{\text{sym.}}} \underline{\underline{\bcancel{d}}}^{+\frac{1}{2}}. \tag{2.8}$$

Since we are only interested in the diagonal elements of $e^{-t\underline{\underline{L}}^{\text{r.w.}}}$, the powers of the diagonal degree matrix drop out completely:

$$\left[e^{-t\underline{\underline{L}}^{\text{r.w.}}}\right]_{ii} = d_i^{-\frac{1}{2}} \left[e^{-t\underline{\underline{L}}^{\text{sym.}}}\right]_{ii} d_i^{+\frac{1}{2}} = \left[e^{-t\underline{\underline{L}}^{\text{sym.}}}\right]_{ii}, \tag{2.9}$$

and using the eigendecomposition of $\underline{\underline{L}}^{\text{sym.}} = \sum_{k=1}^{|V|} \lambda_k^{(L)} \underline{x}_k \underline{x}_k^{\dagger}$ yields:

$$\left[e^{-t\underline{\underline{L}}^{\text{r.w.}}}\right]_{ii} = \sum_{k=1}^{|V|} e^{-\lambda_k^{(L)} t} \left([\underline{x}_k]_i\right)^2 \equiv \sum_{k=1}^{|V|} e^{-\lambda_k^{(L)} t} \left[\underline{x}_k \odot \underline{x}_k\right]_i \tag{2.10}$$

where $\odot$ is the hadamard product (i.e. the element-wise product). The rescaled return probability $\underline{\mathscr{R}}(t)$ is then:

$$\underline{\mathscr{R}}(t) = \sum_{k=1}^{|V|} e^{-\lambda_k^{(L)} t} \underbrace{\underline{x}_k \odot \underline{x}_k \oslash \underline{\pi}}_{=\underline{\xi}_k} = \sum_{k=1}^{|V|} e^{-\lambda_k^{(L)} t} \underline{\xi}_k \tag{2.11}$$

where we have defined $\underline{\xi}_k$ as a notational convenience. Comparing the definition of $\underline{\xi}_k$ to those of the right eigenvectors of $\underline{\underline{L}}^{\text{r.w.}}$ (see equations 2.2 and 2.3) reveals that

$$\underline{\xi}_k = (\underline{d}^\dagger \mathbb{1}) \, (\underline{r}_k)^{\odot 2} \quad \Longrightarrow \quad \underline{\mathscr{R}}(t) = (\underline{d}^\dagger \mathbb{1}) \sum_{k=1}^{|V|} e^{-\lambda_k^{(L)} t} (\underline{r}_k)^{\odot 2} \tag{2.12}$$

so the rescaled return probability $\underline{\mathscr{R}}(t)$ is a time-varying weighted sum of the element-wise squares of the right eigenvectors of the random walk Laplacian $\underline{\underline{L}}^{\text{r.w.}}$.

The connection between the $\underline{\xi}$ vectors and the eigenvectors of $\underline{\underline{P}}^{\text{sym.}}$ and $\underline{\underline{L}}^{\text{sym.}}$ allows cheap evaluation of $\underline{\mathscr{R}}(t)$ for any $t \in [0, +\infty)_{\mathbb{R}}$ once the eigenvalues and eigenvectors are in hand:

$$\underline{\mathscr{R}}(t) = \sum_{k=1}^{|V|} e^{-\lambda_k^{(L)} t} \underline{\xi}_k = \sum_{k=1}^{|V|} \left(\lambda_k^{(P)}\right)^t \underline{\xi}_k. \tag{2.13}$$

In practice, for computations on large graphs, only the eigenvector/eigenvalue pairs corresponding to the long-time behavior (i.e. those pairs with Laplacian eigenvalues below a cutoff $\lambda_{\text{threshold}}^{(L)}$) are computed. Then, if one hopes to approximate the true value of $\underline{S}^{\text{eff.}}$, $t_{\min}$ should be set above a cutoff $t_{\min} > c/\lambda_{\text{threshold}}^{(L)}$, where the truncation error decreases exponentially as the constant $c \in (0, +\infty)_{\mathbb{R}}$ is increased.[4]

The last part of the computation of the score is the evaluation of the time integral. For this, a variety of numerical integration algorithms can be used. However, because of the widely disparate timescales present in the integrand due to the dynamics of diffusion on the graph, a vectorized numerical integration algorithm which adaptively subdivides the integration interval should be preferred. Here, QUADPACK (Piessens et al. 2012) is used through its inclusion in

[4]This can be trivially confirmed by examining the exact error term from the omitted eigenvector/eigenvalue pairs in equation (**??**) to bound the error on $\underline{\mathscr{R}}(t)$, then using the triangle inequality to connect this to the error in $\underline{S}^{\text{eff.}}$.

SciPy (Virtanen et al. 2020).

2.3 Clustering Strategies with the Score in Hand

With $\underset{\sim}{S}^{\text{eff.}}$ computed on a graph G, we now wish to exploit this separation of *boundary* vertices, where the value of $\underset{\sim}{S}^{\text{eff.}}$ is expected to be large, from *volume* vertices, where the value of $\underset{\sim}{S}^{\text{eff.}}$ is expected to be small. In this work this partition will be applied as a front-end to two clustering algorithms: spectral clustering using a k-means backend (see Liu and Han 2018 for an exposition), and active learning using label diffusion (Zhu 2005) to disseminate the true labels of queried vertices to the rest of the graph.

In both cases, we will use a simple strategy to make use of $\underset{\sim}{S}^{\text{eff.}}$: Define the set of *boundary* vertices, V_{bndry}, to be the $|V_{\text{bndry}}| \in (0,|V|)_{\mathbb{Z}}$ vertices in G with the largest values of $\underset{\sim}{S}^{\text{eff.}}$, and define the set of *volume* vertices as $V_{\text{vol.}} \equiv V \backslash V_{\text{bndry}}$. Then form the corresponding induced subgraphs G_{bndry} and $G_{\text{vol.}}$ from this partition of the vertices of G. With better separation between the clusters in $G_{\text{vol.}}$ than in the original graph G, we may then perform clustering on $G_{\text{vol.}}$ in whatever way we see fit so long as that method will be able to take advantage of the better separation between clusters in $G_{\text{vol.}}$ as compared to G.

The question of how to handle label assignments for the vertices in V_{bndry} then remains, and is context-specific. For some tasks, these may be vertices you *do not wish* to assign labels to at all—e.g. for graphs derived from some underlying data set, it may be that vertices that join two clusters correspond to data points that do not closely align with the features of the archetypal examples of any cluster. For other data sets, complete labeling may be important, and we may then use a technique like label diffusion on the full graph G to propagate the computed labels of $V_{\text{vol.}}$ to the remaining vertices in V_{bndry}, even if our initial labeling was performed with spectral clustering with a k-means backend.

The centrality of label diffusion in these discussions and the results that follow is due to the fact that it has the same root source of 'distance' on the graph that the boundary-identifying score $\underline{S}$ has, namely the action of the Laplacian on vertex functions of the graph. It is thus a natural way of propagating labels across the graph that will be improved by the boundary vertex pruning the score $\underline{S}$ enables.

2.3.1 Spectral Clustering with *k*-Means: a Brief Definition

Spectral Clustering with *k*-Means as a backend is a well-known algorithm that, given a weighted undirected graph $G = (V, E, w)$ and a number of requested clusters $N_C^{\text{req.}}$, computes a label assignment according to the following algorithm:

Algorithm 1. A brief overview of spectral clustering with a *k*-means backend

```
Require: $\underline{\underline{W}} \in \mathbb{R}^{|V| \times |V|}$ is a symmetric weight matrix with non-negative elements
    $\underline{\underline{L}}^{\text{r.w.}} \leftarrow \underline{\underline{\mathbb{I}}} - \underline{\underline{\bcancel{d}}}^{-1} \underline{\underline{W}}$
    $\underline{\underline{V}} \leftarrow \text{right\_eigvects}\left(\underline{\underline{L}}^{\text{r.w.}}\right)$        ▷ columns of $\underline{\underline{V}}$ are the right eigenvectors of $\underline{\underline{L}}^{\text{r.w.}}$
    $\underline{\underline{X}}^{\text{spect.}} \leftarrow \underline{\underline{V}}_{:,\, 1:N_C^{\text{req.}}}$        ▷ columns of $\underline{\underline{X}}^{\text{spect.}} \in \mathbb{R}^{|V| \times N_C^{\text{req.}}}$ are the $N_C^{\text{req.}}$ eigenvectors with the smallest eigenvalues
    $\underline{Y} \leftarrow \text{k\_means}\left(\underline{\underline{X}}^{\text{spect.}},\, N_C^{\text{req.}}\right)$        ▷ $\underline{Y} \in \mathbb{Z}^{|V|}$ are the integer labels
```

The right eigenvectors of $\underline{\underline{L}}^{\text{r.w.}}$ are preferred for the spectral embedding $\underline{\underline{X}}^{\text{spect.}}$ in this work because, in the case of N_C disconnected components in G, the elements of the N_C eigenvectors of $\underline{\underline{L}}^{\text{r.w.}}$ with the lowest eigenvalues will be element-wise constant on each connected component, making the *k*-means clustering on the spectral embedding trivial (Liu and Han 2018). In particular, because we are performing spectral clustering on the volume subgraph $G_{\text{vol.}}$, which we expect to be closer to the disconnected case even for a strongly-connected full graph G after removal of the boundary vertices, we want to use a spectral embedding that takes advantage of this property.

2.3.2 Active Clustering using Label Diffusion

The notion of diffusion on a graph discussed so far involves a process which preserves the sum of the elements of the vector undergoing the diffusion process:

$$\underline{v}^\dagger(t) \equiv \underline{v}^\dagger(0)e^{-t\underline{\underline{L}}^{\text{r.w.}}} \implies \underline{v}^\dagger(t)\underline{\mathbb{1}} = \underline{v}^\dagger(0)e^{-t\underline{\underline{L}}^{\text{r.w.}}}\underline{\mathbb{1}} = \underline{v}^\dagger(0)\underline{\mathbb{1}}, \tag{2.14}$$

which is easily proven by using the power series definition of the matrix exponential and the fact that $\underline{\underline{L}}^{\text{r.w.}}\underline{\mathbb{1}} = \underline{0}$. In particular, the infinite time limit is the ergodic distribution:

$$\underline{v}^\dagger(t) \xrightarrow[t\to+\infty]{} c\underline{\pi} \propto \underline{d},\, c \in \mathbb{R} \tag{2.15}$$

which is undesirable if one wishes to use a diffusion process to propagate labels across a graph as the final result will incorporate the degrees of the vertices.

Instead, label diffusion acts $\underline{\underline{L}}$ or $\underline{\underline{P}}$ to the *right* on a column vector:

$$\underline{h}(t) = e^{-t\underline{\underline{L}}^{\text{r.w.}}}\underline{h}(0) \tag{2.16}$$

which has a constant vector as its infinite time limit:

$$\underline{h}(t) \xrightarrow[t\to+\infty]{} c\underline{\mathbb{1}},\, c \in \mathbb{R}, \tag{2.17}$$

which is more suitable to propagating labels across a graph, as a reasonable requirement of any label propagation scheme is that vertices the algorithm is certain have the same label should have the same value at the end of label propagation process. In general, an D_L-element vector is used as a label for each vertex, whence h becomes a matrix $\underline{\underline{h}} \in \mathbb{R}^{|V|\times D_L}$, with each row being the vector label assigned to the corresponding vertex. While there are many more complex variations (such as Wang, Tu, and Tsotsos 2013), a simplified algorithm for label diffusion is presented in algorithm (2), where the last step assigning an integer label to each vertex is performed under the assumption that the ground-truth vertex labels are encoded as one-hot (i.e. standard basis) vectors.

Algorithm 2. A brief overview of label diffusion

```
Require: $\underline{\underline{W}} \in \mathbb{R}_+^{|V|\times|V|}$    ▷ graph weight matrix
Require: $\underline{\underline{\mathscr{P}}}_{\text{known}}^2 = \underline{\underline{\mathscr{P}}}_{\text{known}},\ \underline{\underline{\mathscr{P}}}_{\text{unknown}}^2 = \underline{\underline{\mathscr{P}}}_{\text{unknown}}$    ▷ diagonal projectors
Require: $\underline{\underline{\mathscr{P}}}_{\text{known}} + \underline{\underline{\mathscr{P}}}_{\text{unknown}} = \underline{\underline{\mathbb{I}}}$    ▷ vertices split into known- and unknown-label subsets
Require: $\underline{\underline{h}}_{\text{known}} \in \mathbb{R}^{|V|\times D_L}$ s.t. $\underline{\underline{\mathscr{P}}}_{\text{known}}\underline{\underline{h}}_{\text{known}} = \underline{\underline{h}}_{\text{known}}$    ▷ the known label matrix
  $\underline{\underline{h}} = \underline{\underline{0}}_{|V|\times D_L}$    ▷ label matrix, each row is a $D_L$-element label vector
  $\underline{\underline{P}}^{\text{r.w.}} \leftarrow \underline{\underline{d}}^{-1}\underline{\underline{W}}$
  repeat
      $\underline{\underline{h}} \leftarrow \underline{\underline{\mathscr{P}}}_{\text{unknown}}\underline{\underline{P}}^{\text{r.w.}}\underline{\underline{h}} + \underline{\underline{h}}_{\text{known}}$
  until $\underline{\underline{h}}$ converges
  $y_i = \text{argmax}_j(h_{ij})$    ▷ $y_i$ is the integer label assigned to $v_i$
```

For active learning where the algorithm can request the true labels for a small number of data points/vertices of its choosing (see Settles 2011 for details), it is natural for us to prefer a method of point selection that is intrinsically compatible with the features of the graph that are being enhanced by boundary vertex removal. Algorithm (3) uses the label diffusion process outlined in algorithm (2) to find vertices that take a long *time* under label diffusion to be labeled by the already-queried points. This is appealing, as the boundary vertex set removal used to define the volume subgraph will create very weakly connected clusters. Then algorithm (3) is intuitively likely to query at least one label from each cluster before it begins querying multiple labels from any cluster, modulo common considerations of widely disparate cluster sizes and inter-cluster connection strengths.

It is notable that algorithm (3) decides all of the data points it wishes to query labels for at once, which corresponds to the *batch-mode* active learning as discussed in Settles 2011. Knowing all the labels we wish to query in advance is often preferable in practice, as there may be significant overhead or time delays to acquiring one label at a time.

Algorithm 3. Selecting distant points on a graph for active learning as determined by the label diffusion dynamics

```
Require: $\underline{\underline{W}} \in \mathbb{R}_{+}^{|V|\times|V|}$    ▷ the weight matrix of a graph
Require: $N_Q \in [1, |V_{\text{vol.}}|)$    ▷ number of vertices to choose whose labels will be queried
  $\underline{\underline{P}}^{\text{r.w.}} \leftarrow \underline{\underline{\cancel{d}}}^{-1}\underline{\underline{W}}$
  $Q \leftarrow \{q_{\text{random}}\}$    ▷ seed the set of vertex indices to query with $q_{\text{random}} \in [1, |V|]_{\mathbb{Z}}$
  while $|Q| < N_Q$ do
      $\underline{h} \leftarrow \underline{0}_{|V|\times 1}$
      $\underline{\underline{\cancel{\mathscr{P}}}}_{\text{unknown}} \leftarrow \underline{\underline{\mathbb{I}}}$
      for all $q \in Q$ do
          $h_q \leftarrow 1$    ▷ initialize with ones where you've already decided to query
          $\left[\underline{\underline{\cancel{\mathscr{P}}}}_{\text{unknown}}\right]_{qq} \leftarrow 0$    ▷ construct unknown label projector for label diffusion
      end for
      $\underline{h}_{\text{known}} \leftarrow \underline{h}$
      repeat
          $\underline{h} \leftarrow \underline{\underline{\cancel{\mathscr{P}}}}_{\text{unknown}}\underline{\underline{P}}^{\text{r.w.}}\underline{h} + \underline{h}_{\text{known}}$
      until partial diffusion on the graph is achieved
      $Q \leftarrow Q \cup \{\text{argmin}_i(h_i)\}$
  end while
```

2.4 Graph Construction from Point Clouds

While the boundary identifying score $\underline{S}$, computed from the rescaled return probability $\underline{\mathscr{R}}(t)$, is defined solely by the graph G, much real-world data is in the form of point clouds $\left\{ \underline{x}^{(n)} \middle| \underline{x}^{(n)} \in \mathbb{R}^D,\ n \in [1, N_p]_{\mathbb{Z}} \right\}$. In this work we use a Gaussian kernel with an adaptive bandwidth set by the k^{th} nearest neighbor distance to define G from a point cloud, as shown in algorithm 4.

Algorithm 4. Constructing an undirected graph weight matrix from point cloud data

```
$X \leftarrow \left\{ \underline{x}^{(n)} \middle| \underline{x}^{(n)} \in \mathbb{R}^D,\ n \in [1, N_p]_{\mathbb{Z}} \right\}$    ▷ the point cloud
$d_{ij} \equiv \left| \underline{x}^{(i)} - \underline{x}^{(j)} \right|$    ▷ pairwise Euclidean distance matrix
$\underline{\sigma}^{(k)} = \left[ \text{rowsort}(\underline{\underline{d}}) \right]_{:,k}$    ▷ sort the rows of $\underline{\underline{d}}$ and take the $k^{\text{th}}$ smallest distance from each row
$W_{ij} \leftarrow \exp\left( -\frac{d_{ij}}{\sigma_i^{(k)} \sigma_j^{(k)}} \right)$
```

The adaptive bandwidth construction has a number of appealing qualities, first and foremost being that the weight matrix it produces $\underline{\underline{W}}$ is invariant not only to global spatial dilation of the point cloud $\underline{x} \to c\underline{x}\ \forall \underline{x} \in X$, but for sufficiently well-separated clusters is also approximately invariant to *per-cluster* spatial dilation. There is no reason to generically expect that a single fixed bandwidth for the entire point cloud is appropriate for well-separated portions of the data which may be generated by completely different mechanisms. We argue—solely qualitatively—that it is *changes in density* that matter, not the value of the density itself in nearly constant-density regions of the point cloud.

2.5 Benchmarking Clustering

At its core, the goal of clustering is to take a set of data, $X = \{ x_n \,|\, n \in [1, N_x]_{\mathbb{Z}} \}$[5], and produce some number N_C of subsets of those data points $\left\{ C_i \middle| C_i \subset X,\ i \in [1, N_C]_{\mathbb{Z}} \right\}$, called

[5]The elements of D are often called *data points*. We will use this terminology generically, even for elements of the data set that are not *points* in the intuitive geometric meaning of the word.

clusters, such that points within each subset are similar to each other in some meaningful way. Depending on the nature of the data and any downstream tasks that will use the clustering result, N_C may be specified as part of the input to the algorithm, or it may be dynamically determined; the clusters may be disjoint (i.e. $C_i \cap C_j = \emptyset \; \forall i, j \,|\, i \neq j$), or some data points may belong to multiple clusters; the union of these clusters may be equal to the full data set (i.e. $\bigcup_{i=1}^{N_C} C_i = D$), or there may be elements from the full data set that are not included in any cluster.

Quantifying the quality of a clustering on a particular data set is a challenging task in its own right, at the most basic level due to the fact that the notion of a good *clustering* is itself not universal, but depends on the specific notion of what constitutes a good *cluster*. The notion of a good *cluster* can either be generic—e.g. an isotropically distributed set of points in Euclidean space all close to their mean[6]—or specific to one data set—i.e. a gold standard, *ground-truth clustering* is provided to which all predicted clusterings should be compared.

For the results presented here we will focus solely on data for which a ground-truth clustering is available, and where the ground-truth labeling is such that every data point belongs to exactly *one* ground-truth cluster, called its *ground-truth label*. In such cases, a perfect predicted clustering[7] is simple to identify. However, there is still no unique way of deciding which of two *imperfect* predicted clusterings is “better” than the other.

We will rely on three quantitative notions when evaluating a predicted clustering: the error rate, the V-measure, and the P_4-metric. Their formulation is simplest in terms of the so-called *confusion matrix*.

2.5.1 The Confusion Matrix

For a data set X, assume we have both a ground-truth clustering with $N_C^{\text{g.t.}}$ distinct labels and a predicted clustering with $N_C^{\text{pred.}}$ distinct labels. Then the confusion matrix $\underline{\underline{M}} \in \mathbb{Z}_+^{N_C^{\text{g.t.}} \times N_C^{\text{pred.}}}$

[6]This is the notion of a ”good cluster“ that the k-means clustering algorithm relies upon

[7]In this context, a perfect predicted clustering means that the predicted label of every point matches that point's ground-truth label, modulo an inconsequential one-to-one matching between the name assigned to the predicted labels and ground-truth labels.

is a matrix where M_{ij} is the number of data points with ground-truth label i and predicted label j. For example, consider the confusion matrix

$$\underline{\underline{M}} = \begin{bmatrix} 6 & 0 & 0 \\ 0 & 0 & 9 \\ 0 & 8 & 1 \end{bmatrix}. \tag{2.18}$$

The first row of $\underline{\underline{M}}$ tells us that every data point with the first ground-truth label has the first predicted label (corresponding to the first column); the second row of $\underline{\underline{M}}$ tells us that every data point with the second ground-truth label has the third predicted label (column 3); the third row of $\underline{\underline{M}}$ tells us that the data points with the third ground-truth label are split, with 8 of them given the second predicted label, and one of them given the third predicted label.

Several features are worth noting:

- The sum of all the entries of $\underline{\underline{M}}$ is the number of data points, $|X|$.
- The row sums of $\underline{\underline{M}}$ are the numbers of data points in each of the ground-truth clusters.
- The column sums of $\underline{\underline{M}}$ are the numbers of data points in each of the predicted clusters.
- A reasonable approach to finding a good matching between the ground-truth label names and the predicted label names is to find the permutation of the columns that maximizes the sum of the diagonal entries.

Some clustering algorithms inherently produce predicted cluster names with no relation to the ground-truth cluster names, such as spectral k-means clustering. In these cases, if $N_C^{\text{g.t.}} = N_C^{\text{pred.}}$, this approach of permuting the columns to maximize the diagonal sum of the confusion matrix is a necessary first step before some clustering metrics which assume a one-to-one mapping between the ground-truth and predicted cluster names can be used. For square confusion matrices where this column permutation to maximize the sum of the diagonal entries has already been done, we will use the notation $\underline{\underline{M}}^{\text{matched}}$. This is called the *assignment problem*, and is well-studied (Burkard and Cela 1999).

2.5.2 The Error Rate

This is conceptually the simplest, but also potentially the most misleading clustering metric. Assuming a one-to-one correspondence between the names used for the labels in the ground-truth clustering and predicted clustering already exists, and defining $N_C \equiv N_C^{\text{g.t.}} = N_C^{\text{pred.}}$, the error rate is:

$$ER \equiv \frac{\text{number of incorrect labels}}{\text{number of data points}} = \frac{\sum_{i=1}^{N_C} M_{ii}^{\text{matched}}}{\sum_{i=1}^{N_C} \sum_{j=1}^{N_C} M_{ij}^{\text{matched}}} \tag{2.19}$$

While simple to compute, grossly imbalanced class sizes will result in extremely low error rates for what could reasonably be considered “bad” predicted clusterings. For example, if D consists of 10^6 data points in two classes, with 10 data points in one class, and $10^6 - 10$ points in the other, a predicted clustering which assigned every data point to the label of the larger ground-truth cluster would have an error rate of:

$$ER = \frac{10}{10^6} = 10^{-5} = 0.001\%. \tag{2.20}$$

But in applications, identifying elements of the rarer class is frequently the most important and relevant task, leaving the error rate grossly unsuited as a general clustering metric. It also requires making a label name correspondence in advance.

However, in cases where the ground-truth cluster sizes are not too dissimilar and a suitable one-to-one matching between the ground-truth and predicted cluster names is available, it can provide a simple and intuitive measure of the quality of a predicted clustering.

2.5.3 The V-measure

The V-measure relies on information-theoretic notions to remove the dependence on a label name matching, and combines two relevant notions of the quality of a predicted clustering—homogeneity and completeness—to produce a single real number in the interval $[0, 1]_{\mathbb{R}}$, where a

V-measure of 1 indicates a perfect matching (Rosenberg and Hirschberg 2007).

The Homogeneity

The *homogeneity* $h \in [0,1]_{\mathbb{R}}$ of a predicted clustering answers the question "how much information does knowing the predicted label of a data point tell me about the ground-truth label of that data point?" It is 1 when every data point in a predicted cluster shares the same ground-truth label for every predicted cluster. On the other hand, the homogeneity is low when the predicted clusters contain mixtures of ground-truth labels. It is defined as

$$h = \begin{cases} 1 - \frac{H(C_{\text{g.t.}} | C_{\text{pred.}})}{H(C_{\text{g.t.}})} & \text{if } H(C_{\text{g.t.}}, C_{\text{pred.}}) \neq 0 \\ 1 & \text{if } H(C_{\text{g.t.}}, C_{\text{pred.}}) = 0 \end{cases} \tag{2.21}$$

where the two entropies H can be more easily expressed by defining the normalized confusion matrix probability distribution $\underline{\underline{p}}$ and its marginals as:

$$p_{ij} \equiv \frac{1}{|D|} M_{ij} \qquad p_i^{\text{g.t.}} \equiv \sum_{j=1}^{N_C^{\text{pred.}}} p_{ij} \qquad p_j^{\text{pred.}} \equiv \sum_{i=1}^{N_C^{\text{g.t.}}} p_{ij} \tag{2.22}$$

where p_{ij} is the fraction of data points which are in the i^{th} ground-truth cluster and the j^{th} predicted cluster. Then:

$$H(C_{\text{g.t.}}) \equiv -\sum_{i=1}^{N_C^{\text{g.t.}}} p_i^{\text{g.t.}} \ln p_i^{\text{g.t.}} \tag{2.23}$$

$$H(C_{\text{g.t.}} | C_{\text{pred.}}) \equiv -\sum_{i=1}^{N_C^{\text{g.t.}}} \sum_{j=1}^{N_C^{\text{pred.}}} p_{ij} \ln \left(\frac{p_{ij}}{p_j^{\text{pred.}}} \right) \tag{2.24}$$

The Completeness

The *completeness* $c \in [0,1]_{\mathbb{R}}$ of a predicted clustering answers the question "how much information does knowing the ground-truth label of a data point tell me about the predicted label of that data point?" It is 1 when every data point in a ground-truth cluster shares the same predicted label for every ground-truth cluster. On the other hand, the completeness is low when the ground-truth clusters contain mixtures of predicted labels. It is defined as

$$c \equiv \begin{cases} 1 - \frac{H(C_{\text{pred.}} \middle| C_{\text{g.t.}})}{H(C_{\text{pred.}})} & \text{if } H(C_{\text{g.t.}}, C_{\text{pred.}}) \neq 0 \\ 1 & \text{if } H(C_{\text{g.t.}}, C_{\text{pred.}}) = 0 \end{cases} \tag{2.25}$$

where the two relevant entropies are:

$$H(C_{\text{pred.}}) \equiv -\sum_{j=1}^{N_C^{\text{pred.}}} p_j^{\text{pred.}} \ln p_j^{\text{pred.}} \tag{2.26}$$

$$H(C_{\text{pred.}} \middle| C_{\text{g.t.}}) \equiv -\sum_{i=1}^{N_C^{\text{g.t.}}} \sum_{j=1}^{N_C^{\text{pred.}}} p_{ij} \ln\left(\frac{p_{ij}}{p_i^{\text{g.t.}}}\right) \tag{2.27}$$

The V-measure

The V-measure is then defined as the harmonic mean of the homogeneity and the completeness:

$$v_{\text{measure}} \equiv \frac{2}{\frac{1}{h} + \frac{1}{c}} = \frac{2hc}{h+c} \tag{2.28}$$

The use of the harmonic mean has a few nice consequences:

- $v_{\text{measure}} \in [0,1]_{\mathbb{R}}$
- If either the homogeneity or the completeness is zero, then the V-measure is also zero.
- The V-measure satisfies $v_{\text{measure}} \leq 2\min(h,c)$, so if either the homogeneity or the completeness is small, the V-measure must be, as well.
- $v_{\text{measure}} = 1$ requires that $h = 1$ *and* $c = 1$, implying a perfect matching between the predicted and ground-truth clusterings.

2.5.4 The P$_4$-metric

The P$_4$-metric is a recently introduced binary classification metric (Sitarz 2023). It evaluates the performance of a binary classifier, so it is only suited to provide a single number to quantify the quality of a clustering algorithm when both the ground-truth and predicted clusterings contain two clusters each. However, it can still be used to derive a per-cluster notion of how good a predicted clustering is when we have a matched confusion matrix $\underline{\underline{M}}^{\text{matched}}$ and its associated normalized confusion matrix probability distribution $\underline{\underline{p}} \in \mathbb{R}^{N_C \times N_C}$. The P$_4$-metric for cluster $n \in [1, N_C]_{\mathbb{Z}}$ is the harmonic mean of four conditional probabilities for a data point $d \in D$ drawn uniformly from the entire data set:

- $P(d \in C_n^{\text{g.t.}} \,|\, d \in C_n^{\text{pred.}}) = \frac{p_{nn}}{p_n^{\text{pred.}}}$,
- $P(d \in C_n^{\text{pred.}} \,|\, d \in C_n^{\text{g.t.}}) = \frac{p_{nn}}{p_n^{\text{g.t.}}}$,
- $P(d \notin C_n^{\text{g.t.}} \,|\, d \notin C_n^{\text{pred.}}) = \frac{1 - p_n^{\text{pred.}} - p_n^{\text{g.t.}} + p_{nn}}{1 - p_n^{\text{pred.}}}$, and
- $P(d \notin C_n^{\text{pred.}} \,|\, d \notin C_n^{\text{g.t.}}) = \frac{1 - p_n^{\text{pred.}} - p_n^{\text{g.t.}} + p_{nn}}{1 - p_n^{\text{g.t.}}}$.

Then an N_C class clustering will have N_C P$_4$-metrics, one for each cluster.

2.6 Synthetic Benchmarks

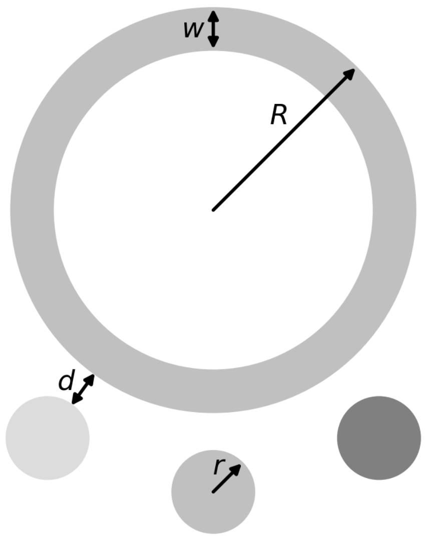
2.6.1 Annulus with Disk anomalies

This class of synthetic distributions consists of a point cloud in two dimensions with $N_C = 4$ disjoint regions where points are drawn, as shown in figure (2.1). The number of points drawn in each disjoint region of the distribution is fixed in order to avoid situations where random sampling may lead to uninformative variations in the number of points in the smaller regions of the distribution.

For all of the results on this "annulus with disk anomalies" point clouds that follow, certain parameters defining the point clouds are fixed and do not change:

- The annulus has $R = 1$, $w = 0.05$, and is filled with 950 points drawn at random from a uniform distribution in the interior of the annulus.
- The small disks have $r = 0.05$, and counterclockwise from the leftmost disk are filled with 12, 24, 48 points drawn from a uniform distribution in the interior of the disk. These correspond to the disks having point densities that are approximately half of, equal to, and double the point densities in the annulus.
- The graphs are constructed with an adaptive bandwidth with $k = 5$ held fixed. The effective score $\underline{S}^{\text{eff.}}$ was computed with $t_{\min} = 0$, keeping the 10% of the eigenvector/value pairs with the slowest time evolution (i.e. the 10% of eigenpairs with the smallest values of $\lambda^{(L)}$).

The euclidean distance separating the region of support for the disk anomalies and the annulus is uniform for all three disk anomalies for each realization of this point cloud, and is called d. This distributional gap d is a parameter that we will vary in what follows to control how difficult this data set is to cluster.



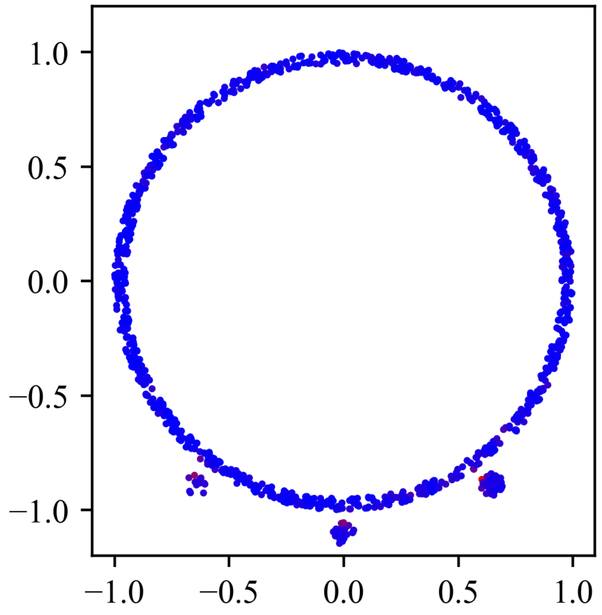


Figure 2.1. The annulus with disk anomalies distribution (left), and an example point cloud drawn from this distribution color coded by score $\underline{S}^{\text{eff.}}$ (right). The use of this class of test distributions was inspired by X. Cheng and Mishne 2020

Annulus Spectral k-means Results

The results of figures (2.2) and (2.3) show an obvious improvement over standard spectral clustering with no boundary point pruning, providing good clustering performance down to approximately *two thirds* of the distribution gap where standard spectral clustering breaks down. We will first focus on figure (2.2)(a,c), which show the clustering performance on the volume subgraph after removing the identified boundary vertices.

Figure (2.2)(a) shows the clustering performance as a function of both the distribution gap d and the fraction of the data points that are pruned to produce the volume subgraph. The standard spectral clustering result lies on the x-axis of this plot, and a clear improvement in the V-measure is observed at low distribution gap values of $d \lesssim 0.1$, as is evident in figure (2.2)(c). Furthermore, improvement in the V-measure is stable for a wide range of amounts of pruning, showing that our approach is stable and is not dependent on careful fine-tuning of the degree of pruning.

If we look at a constant amount of pruning of approximately 2% or more in figure figure (2.2)(a), we see a *decrease* in performance relative to the no-pruning case in the approximate distribution gap range $d \approx 0.15$, with higher V-measures both below and above these values of the distribution gap. We attribute this behavior to the following: at large distribution gaps $d > 0.15$, the disk anomalies and the annulus are nearly or completely disconnected, and the score $\underset{\sim}{S}^{\text{eff.}}$ is primarily identifying widely-distributed points in the point cloud due to random density fluctuations. In this regime, the clustering is robust against a large number of points pruned. As the distribution gap enters the $d \approx 0.15$ regime, however, the score identifies points on the annulus near the disk anomalies. With a sufficient number of points pruned in this regime, the annulus is cut into multiple segments, severely degrading clustering performance. When $d < 0.15$, the vertices with the largest values of the score are a mixture between points on the annulus due to random density fluctuations, and points near the anomalies, restoring robustness to large amounts of point pruning and preventing the segmentation of the annulus.

Figure (2.2)(b,d) show the clustering performance when we take the labels computed by spectral clustering on the volume subgraph and use label diffusion to produce labels on the full graph. The behavior is largely similar to the behavior on the volume subgraph shown in (2.2)(a,c), showing that this method respects our intuition that there is an obvious label that should be provided to every data point on this data set.

The per-cluster performance shown in figure (2.3) on the full graph (i.e. using label diffusion to propagate labels from the volume subgraph to the full graph when pruning boundary data points) validates that the per-cluster performance is quite good down to distribution gaps approximately two thirds of those where standard spectral clustering begins to break down. The half-density disk in figure (2.3)(d) is the most difficult to to cluster, and it is the breakdown in clustering of this half density disk (which contains only 12 points) which dictates the initial drop in performance in the annulus P_4-metric as the distribution gap falls below 0.10.

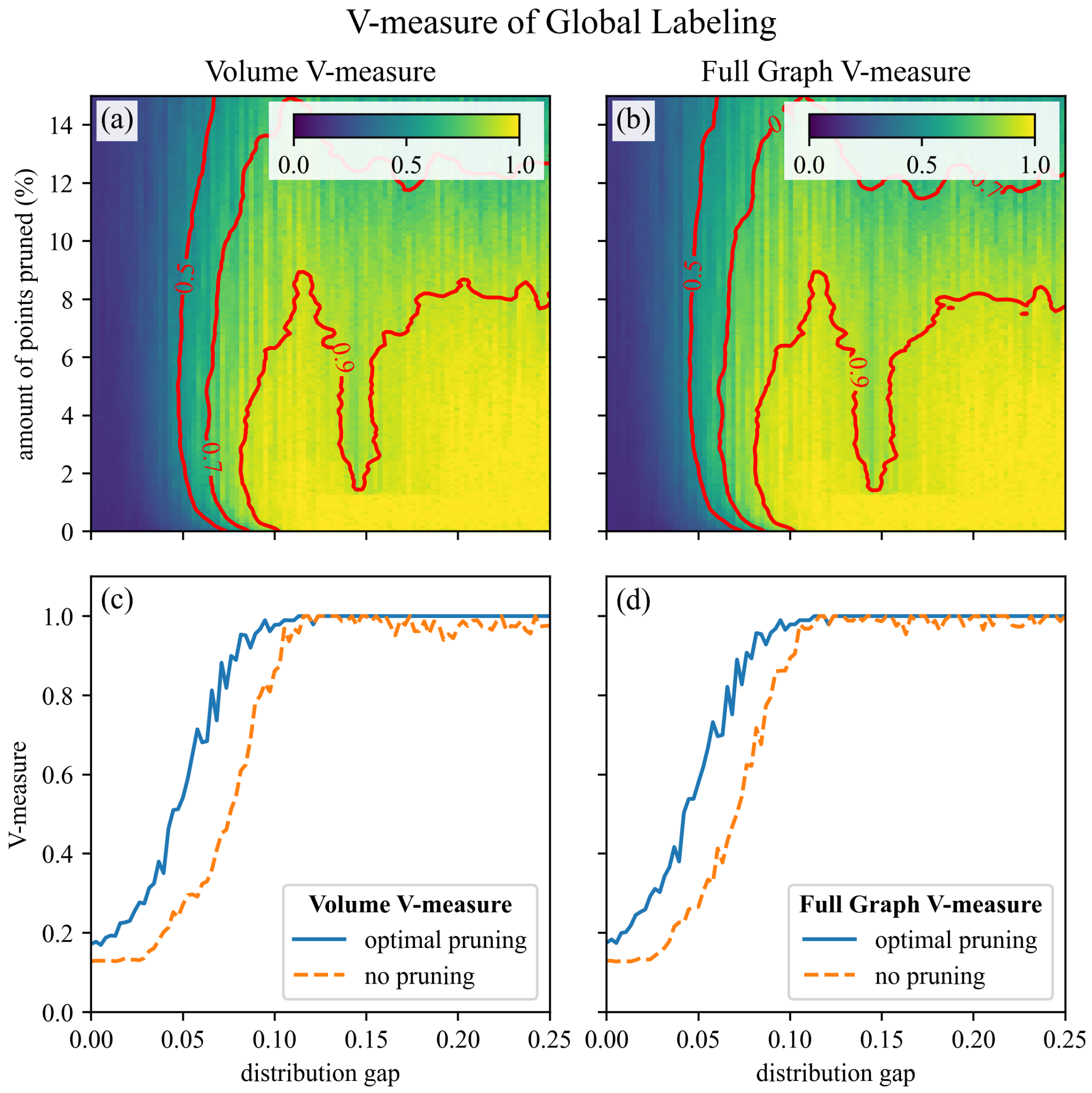


Figure 2.2. The global performance of classical spectral clustering (*no pruning*) versus optimally pruned and backfilled spectral clustering (*optimal pruning*), as measured by the V-measure. The constant-V-measure contour lines in (a) and (b) are computed on a gaussian-filter-smoothed version of the underlying image in order to produce comprehensible contours, with a standard deviation of approximately 1 image pixel. The *optimal pruning* lines in (c) and (d) show the maximum value of the V-measure over all possible amounts of point pruned per distribution gap value, while the *no pruning* lines correspond to directly applying spectral clustering with a k-means backend on the full graph without identifying or removing any boundary points first. Note that the volume subgraph used in (a) and (c) is variable, i.e. it contains fewer vertices as the amount of points pruned increases.

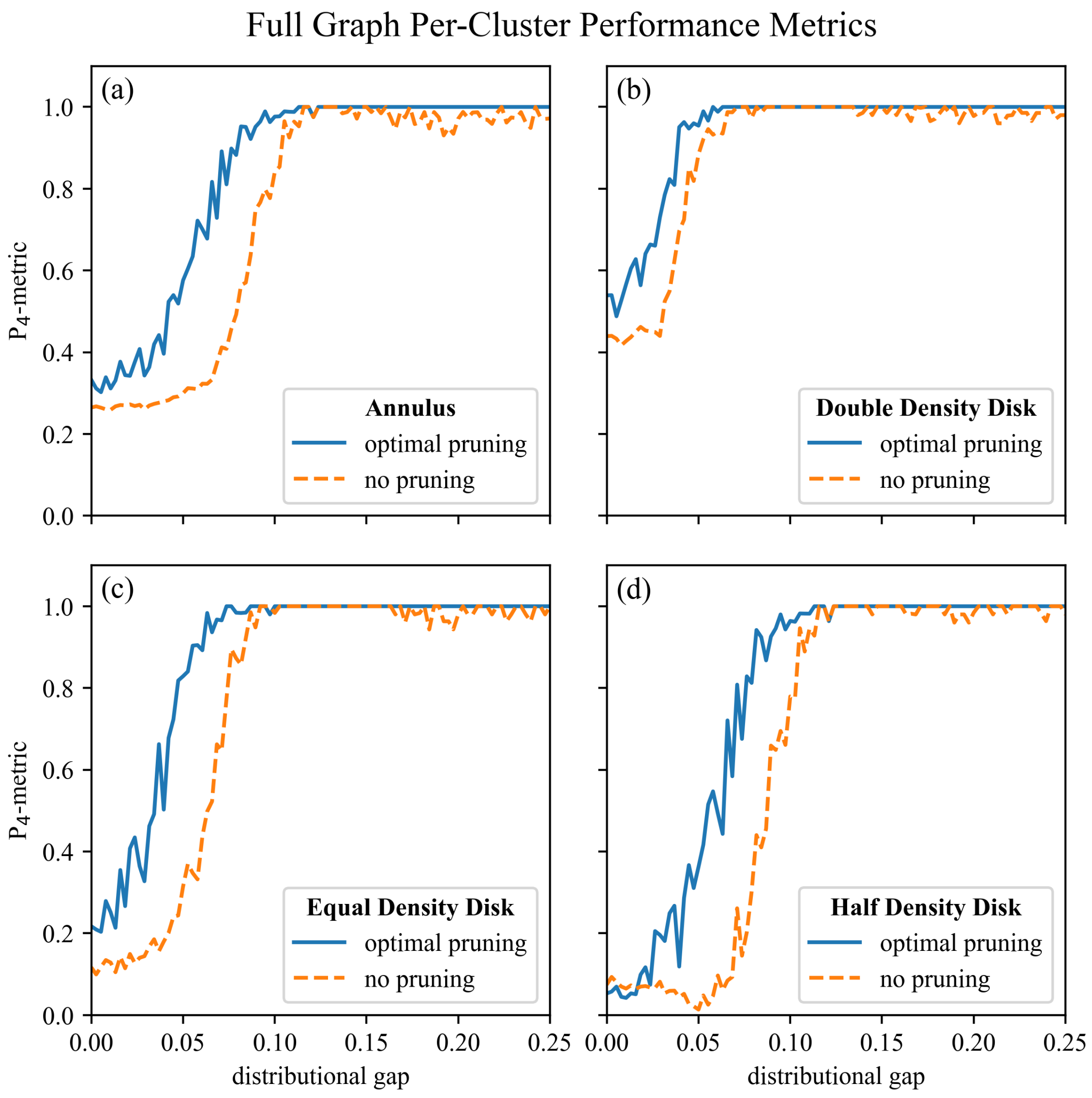


Figure 2.3. The per-cluster performance of standard spectral clustering with a k-means backend (*no pruning*) versus optimally pruned and back-filled spectral clustering (*optimal pruning*). Note that the amount of pruning as a function of the distributional gap used to define the "optimal pruning" plots is the same for all four graphs, and is determined by the amount of pruning that maximizes the global V-measure on the volume subgraph, as in figure 2.2(a) and (c).

Annulus Active Learning Results

Figure (2.4) shows the global clustering behavior for active learning on the annulus dataset, where $N_Q = 4$ groundtruth labels have been queried in all cases. Without pruning of boundary vertices, the active learning algorithm shows markedly better V-measure performance at low-distribution gaps than standard spectral clustering with a k-means backend shown in figure (2.2). The optimal-pruning performance for the active learning results shows a similarly notable improvement.

Where active clustering performs dramatically better than standard spectral clustering with a k-means backend is in the large-amount-of-pruning limit, and in general in insensitivity to the specific amount of pruning. The dramatic dips in performance by standard spectral clustering for distributional gaps of $d \approx 0.15$ that begin with as low as $\sim 1.5\%$ of points pruned simply do not appear. Recall that the failure mode of standard spectral clustering using a k-means backend at large amounts of pruning is the annulus becoming severed by the boundary vertex pruning, leading to the k-means backend preferring to target the large segmented pieces of the annulus for clustering over the smaller disk anomalies.

However, the segmentation of the annulus does not matter to the active learning algorithm so long as the segments of the annulus produced by pruning are still more tightly coupled to the rest of the annulus than to the disk anomalies under label diffusion (algorithm (2)). This shows the strong benefit of choosing an active learning point selection strategy that is intrinsically compatible with our boundary identification, as algorithm (3) is.

The per-cluster active learning performance shown in figure (2.5) tells a similar story to that of the standard spectral clustering per-cluster performance in figure (2.3).

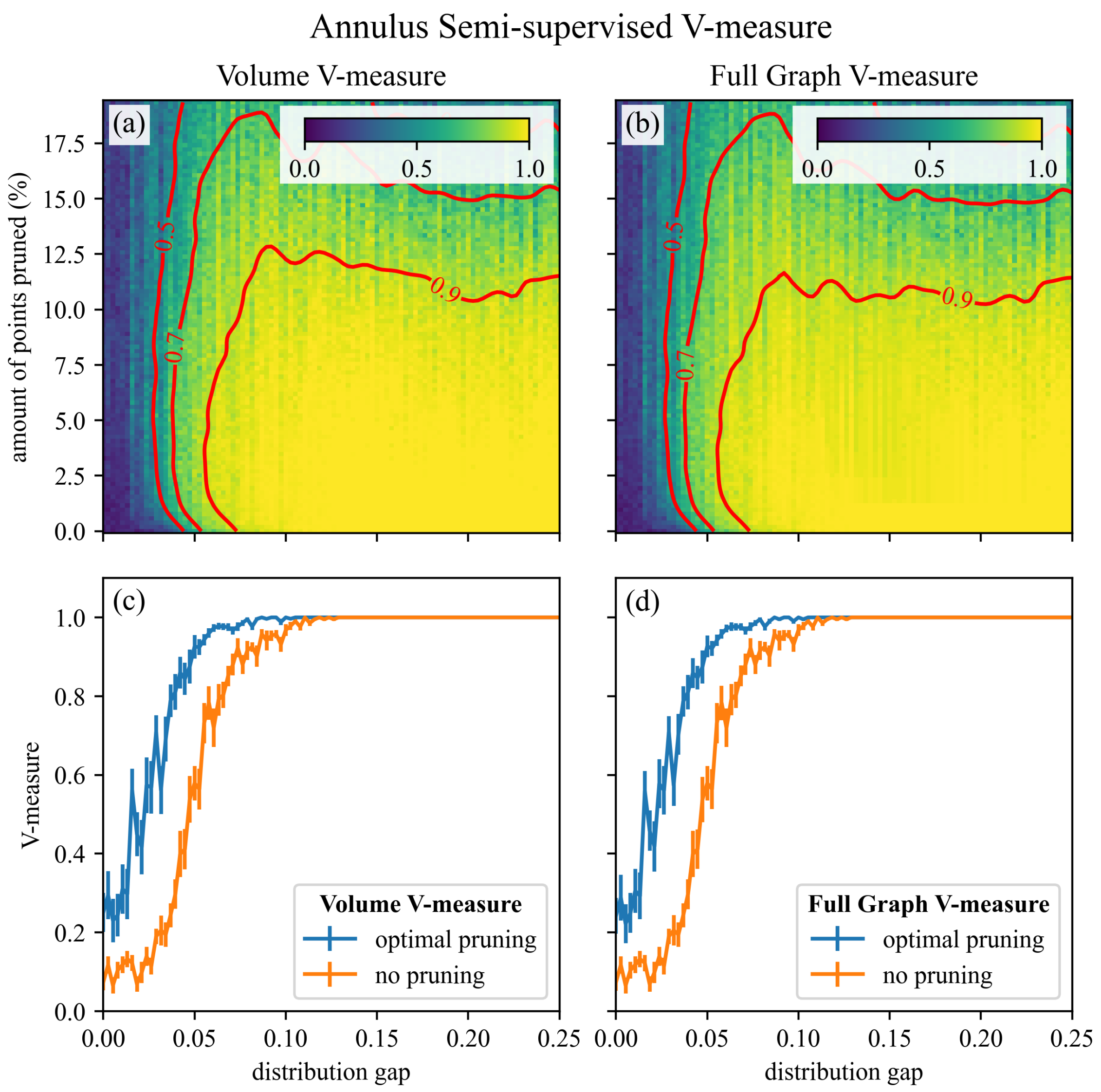


Figure 2.4. The performance of classical spectral clustering (*no pruning*) versus optimally pruned and backfilled spectral clustering (*optimal pruning*), as measured by the V-measure. The constant-V-measure contour lines in (a) and (b) are computed on a gaussian-filter-smoothed version of the underlying image in order to produce comprehensible contours, with a standard deviation of approximately one image pixel. The *optimal pruning* lines in (c) and (d) show the maximum value of the V-measure over all possible amounts of point pruned per distribution gap value, while the *no pruning* lines correspond to directly applying spectral clustering with a k-means backend on the full graph without identifying or removing any boundary points first. The error bars in (c) and (d) are the standard errors of the estimate of the mean from a bootstrap analysis of 20 independent simulations per pixel in (a) and (c). Note that the volume subgraph used in (a) and (c) is variable, i.e. it contains fewer vertices as the amount of points pruned increases.

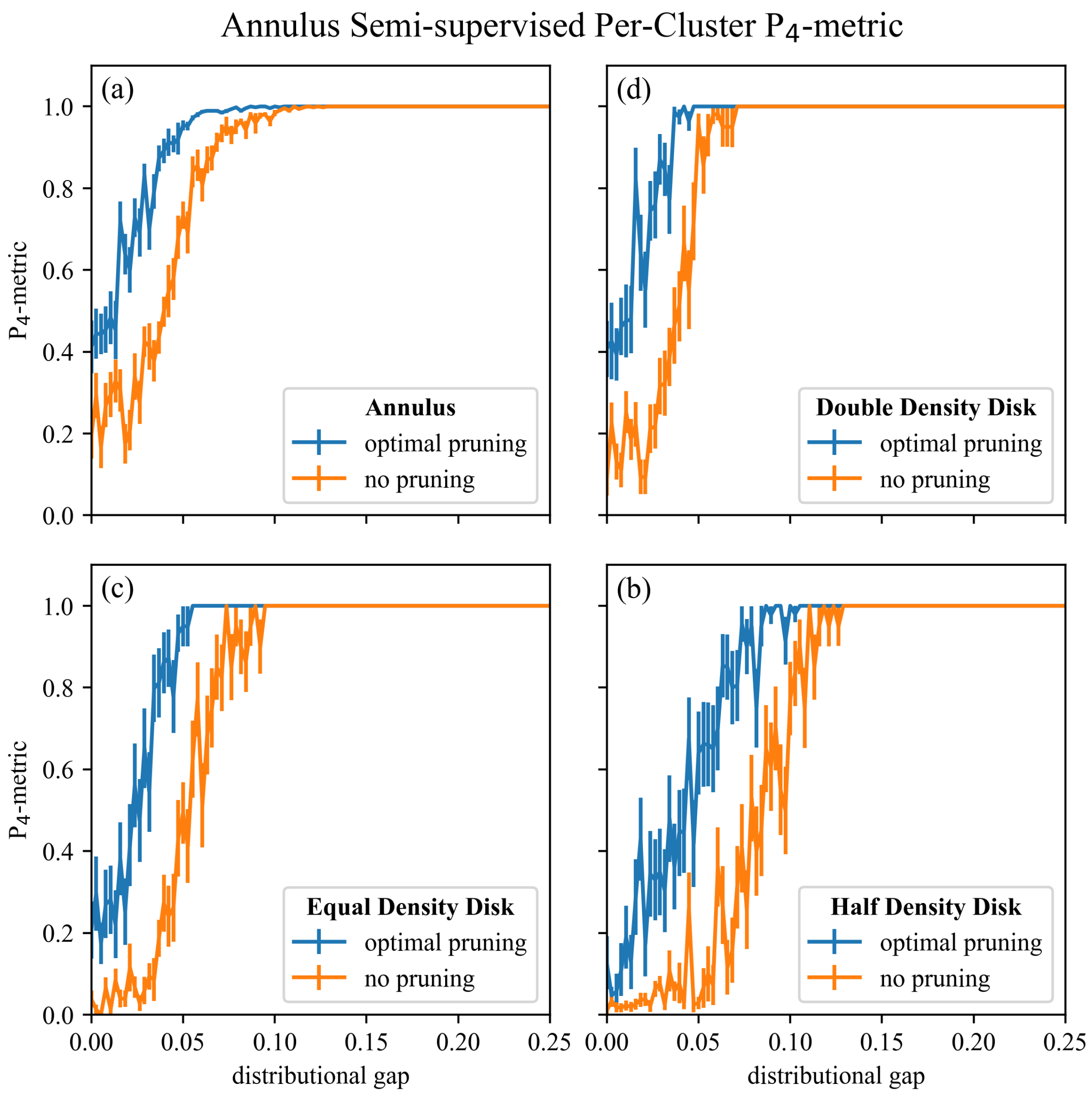


Figure 2.5. The per-cluster performance of classical spectral clustering (*no pruning*) versus optimally pruned and backfilled spectral clustering (*optimal pruning*). The error bars are the standard errors of the estimate of the mean from a bootstrap analysis.

2.7 Salinas-A Hyperspectral Imagery

The corrected[8] Salinas-A Scene Hyperspectral image (HSI) data set (Green et al. 1998) is an 86×83 pixel hyperspectral image with the spectrum of each pixel having 204 bands, resulting a point cloud of 7138 points in 204 dimensions. There are 6 ground-truth labels, as well as 1790 unlabeled points in the ground-truth, which were omitted in this analysis, leaving a 5348 point labeled data set. The data was clustered using the active learning label diffusion algorithm. Note that since the smallest ground-truth cluster (corresponding to the 'Brocoli_green_weeds_1' label) constitutes over 7.3% of the data points, and many of the error rates under consideration are well below this, the error rate is meaningful and will be reported in addition to the V-measure.

For the large-N_Q (large number of ground truth labels queried) experiments of figures (2.6-2.9), approximately half of the data points were used by deleting every other row of the image to reduce computation time to gather sufficient statistics. The graph was constructed with an adaptive bandwidth scheme with $k = 3$, and the score $\underline{S}^{\text{eff.}}$ was computed with $t_{\min} = 10$ and a truncated eigendecomposition keeping the 10% of eigenpairs with the smallest values of $\lambda^{(L)}$.

The variable volume subgraph performance shown in figure (2.6) shows the error rate dropping by a factor of two to four (depending on the number of labels queried) when comparing the no-pruning case to the to the $\gtrsim 250$ points pruned cases, with approximately stable performance from approximately 250 to 400 points pruned. This is a clear indication that the first boundary points we are removing are precisely those that are difficult to cluster, and clustering on the variable volume subgraph is easier than on the full graph. However, the full graph error rate *slightly increases* as the number of vertices pruned increases. The V-measure plots of figure (2.8) tell a similar story.

Figure 2.7 suggests an answer to this puzzling behavior. It shows the label error rate on the *fixed volume subgraph* of the same $2677 - 400 = 2277$ vertices from the maximally-pruned case. The fixed volume subgraph plot of this figure makes clear that the pruning is *not* improving

[8]*Corrected* refers to the deletion of certain water absorption bands from the data.

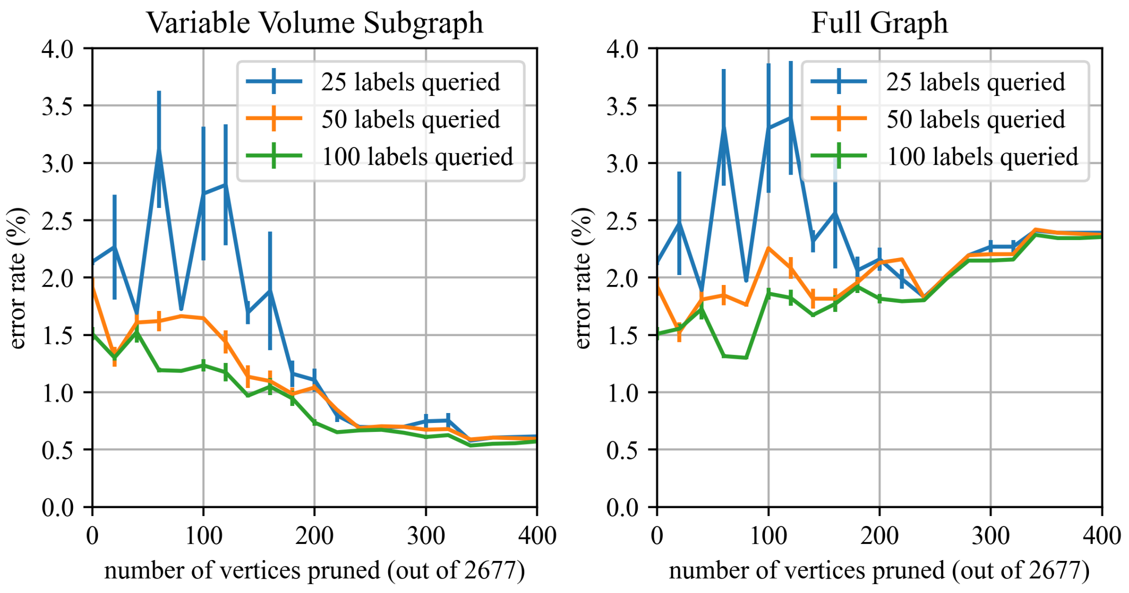


Figure 2.6. Active learning labeling in the large-N_Q limit on a downsampling of the Salinas-A scene as measured by the error rate on the variable (i.e. scaling with the number of vertices pruned) volume subgraphs and the full graph. The error bars show the standard error of a bootstrap estimate of the mean of repeated labeling experiments.

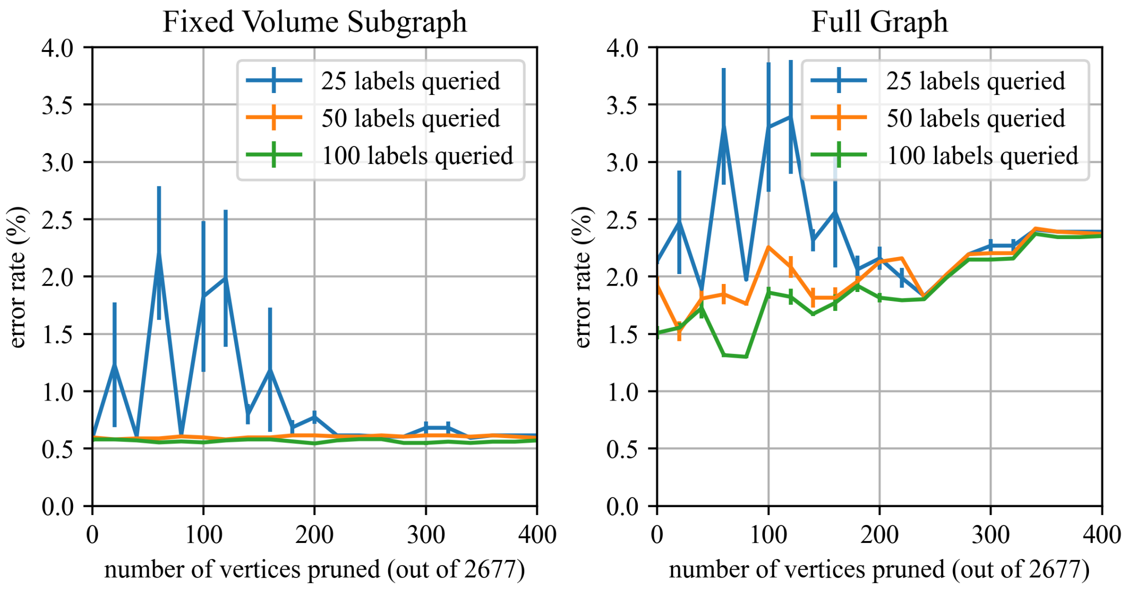


Figure 2.7. The error rate of the same experiments used to generate figure (2.6), with the volume subgraph error rate evaluated on the *fixed* volume subgraph from the maximally pruned case (i.e. the $2677 - 400 = 2277$ vertex) volume subgraph.

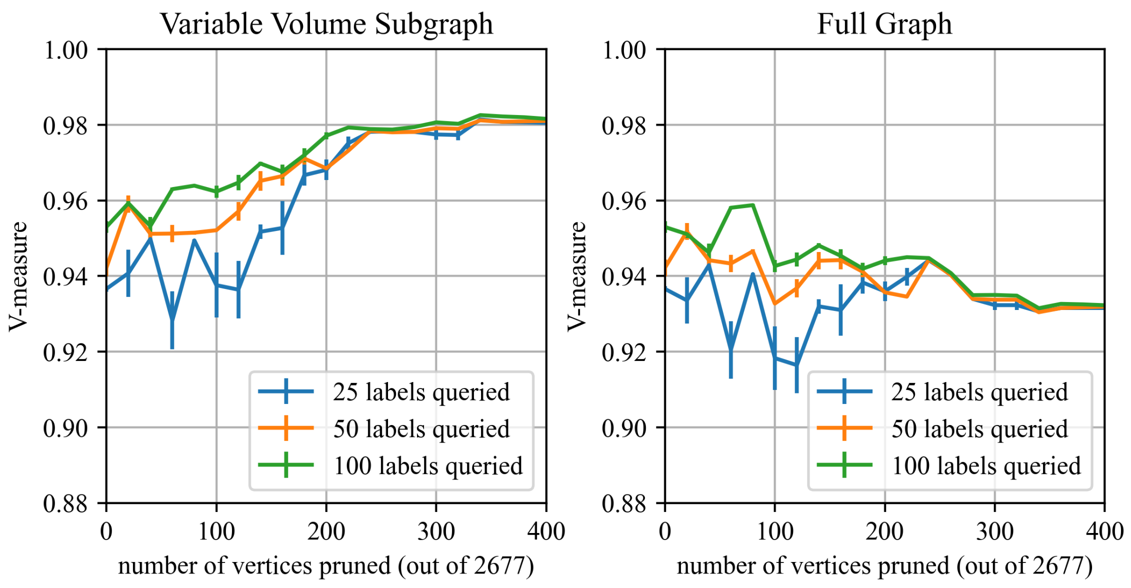


Figure 2.8. Active learning labeling in the large-N_Q limit on a downsampling of the Salinas-A scene as measured by the V-measure on the variable (i.e. scaling with the number of vertices pruned) volume subgraphs and the full graph. The error bars show the standard error of a bootstrap estimate of the mean of repeated labeling experiments.

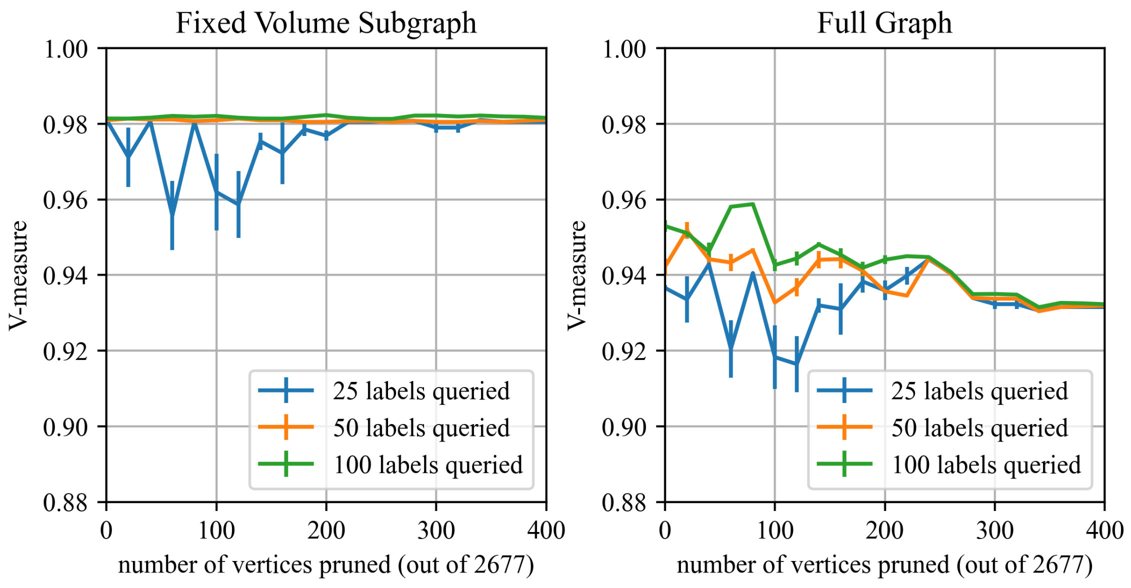


Figure 2.9. The V-measure of the same experiments used to generate figure (2.8), with the volume subgraph V-measure evaluated on the *fixed* volume subgraph from the maximally pruned case (i.e. the $2677 - 400 = 2277$ vertex) volume subgraph.

the labeling error rate on the 2277 volume vertices of this fixed subgraph for 50 or 100 labels queried, with complex behavior observed for 25 labels queried until more than 200 vertices have been pruned. Again, the V-measure plots of figure (2.9) corroborate this.

We hypothesize that these seemingly conflicting performance trends as a function of pruning can be reconciled by recognizing that the ground truth labels do not actually respect the boundaries between clusters as defined by the diffusion process on our graph. Rather, along the diffusion boundaries, the ground-truth labels have a complex structure. If we query a large fraction of ground-truth labels on the full graph with no pruning, we can directly discover this complex structure. However, removing the boundary points before querying for ground-truth labels impedes discovery of this structure, thus leading to a drop in performance as a function of increased pruning as measured by the full-graph error rate and V-measure.

This naturally suggests that this boundary-pruning active-learning algorithm should perform best at *low query numbers*. For these low-N_Q experiments, the full Salinas-A scene data set was used with all 5348 data points. The graph was constructed with an adaptive bandwidth scheme with $k = 5$, and the score $\underset{\sim}{S}^{\text{eff.}}$ was computed with $t_{\min} = 10$ and a truncated eigendecomposition keeping the 10% of eigenpairs with the smallest values of $\lambda^{(L)}$.

The performance story is shockingly good as the number of labels queried drops to approximately twice the number of distinct ground-truth labels, as shown by the error rate and V-measure in figures (2.10-2.13). In this low-N_Q regime, both the variable *and* the fixed volume subgraphs show clearly improved error rates and V-measures above approximately 850 vertices pruned, the we see approximately 1% error rates in the vicinity of 1000 points pruned across 10, 12, and 14 labels queried. Even with only 12 labels queried, *which is only twice the number of distinct labels in the data set being analyzed*, error rates of approximately 1-4% are seen across a wide range of 850-1500 vertices pruned. Querying 14 labels further improves the independence of the performance from any need to precisely tune the degree of pruning in this regime. Depending on the points of comparison, this is a reduction in the error rate by *up to an order of magnitude*.

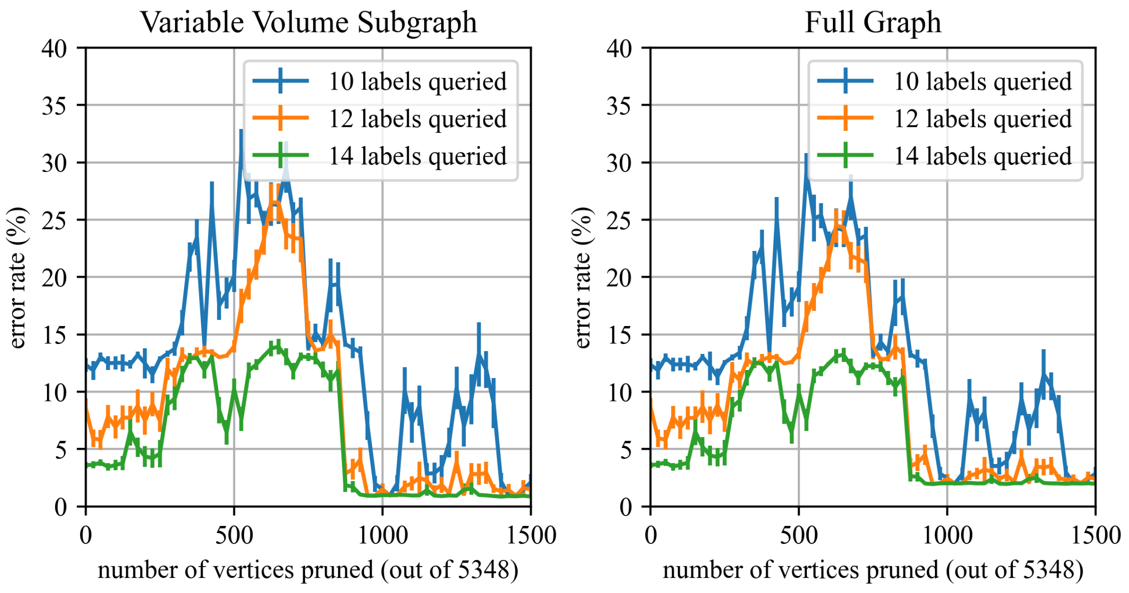


Figure 2.10. Active learning labeling in the small-N_Q limit on the full Salinas-A scene as measured by the error rate on the variable (i.e. scaling with the number of vertices pruned) volume subgraphs and the full graph. The error bars show the standard error of a bootstrap estimate of the mean of repeated labeling experiments.

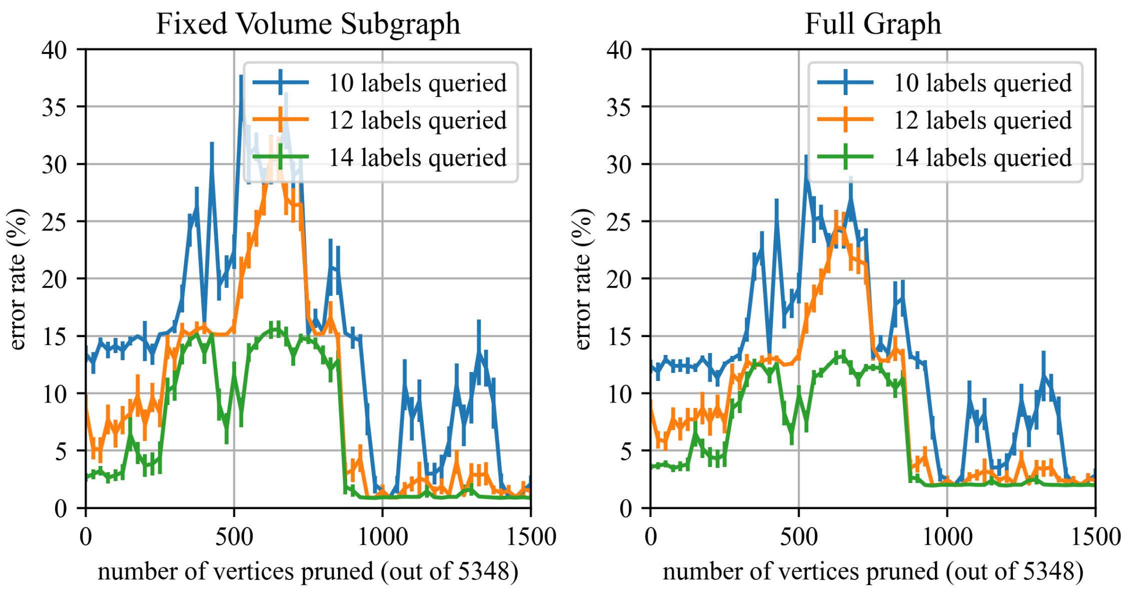


Figure 2.11. The error rate of the same experiments used to generate figure (2.10), with the volume subgraph error rate evaluated on the *fixed* volume subgraph from the maximally pruned case (i.e. the $5348 - 1500 = 3848$ vertex) volume subgraph.

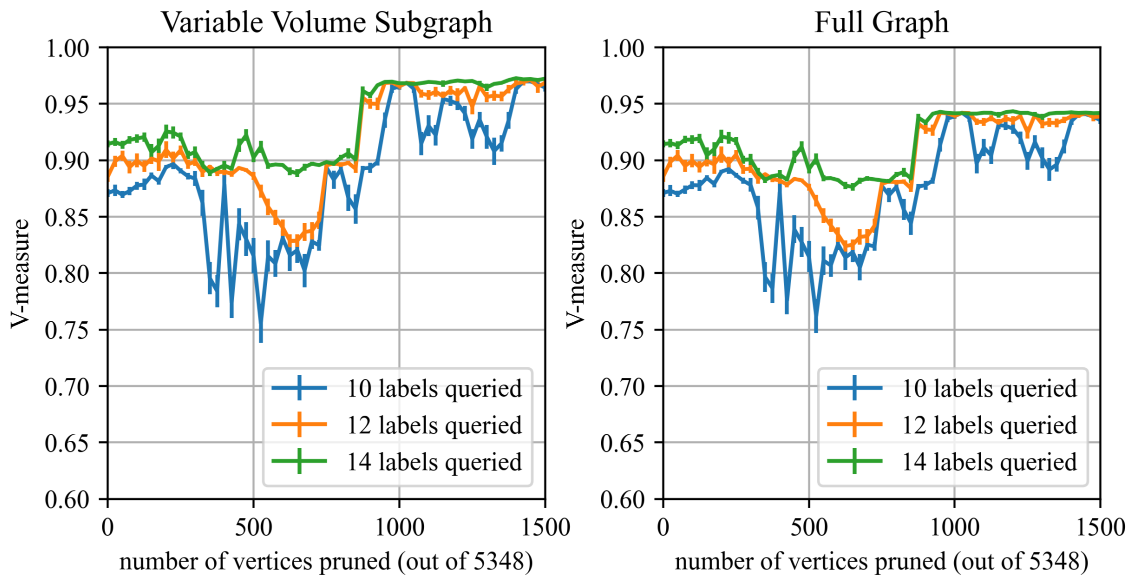


Figure 2.12. Active learning labeling in the small-N_Q limit on the full Salinas-A scene as measured by the V-measure on the variable (i.e. scaling with the number of vertices pruned) volume subgraphs and the full graph. The error bars show the standard error of a bootstrap estimate of the mean of repeated labeling experiments.

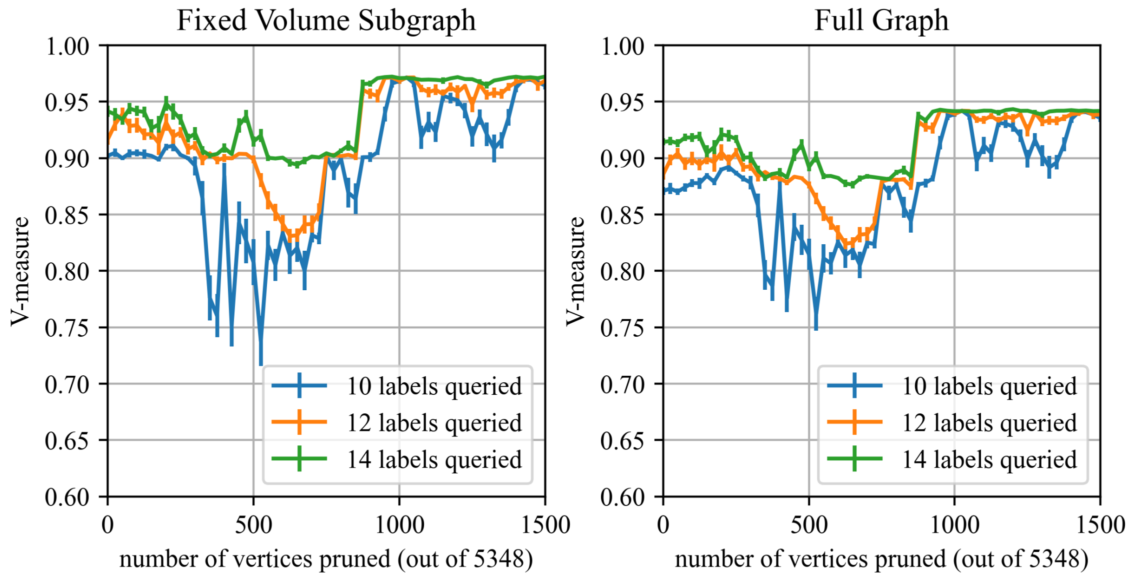


Figure 2.13. The V-measure of the same experiments used to generate figure (2.12), with the volume subgraph V-measure evaluated on the *fixed* volume subgraph from the maximally pruned case (i.e. the $5348 - 1500 = 3848$ vertex) volume subgraph.

The performance on the full graph shows a higher error rate and lower V-measure than on the Volume subgraphs in the 850-1500 vertices pruned regime, but despite this still shows a $\sim$1.5- to 5-fold improvement in the error rate from the 0 pruning case, depending on the number of labels queried.

All of these Salinas-A results were obtained on the raw hyperspectral image data, with no pre-processing with any other method other than excluding the '0' label (corresponding to no cluster in the original dataset). This method compares favorably to methods relying on the point cloud directly, such as A. Cloninger and Mhaskar 2021, where the data was pre-processed with PCA and a spatial-density estimator was used to remove points from low-density regions prior to label querying to achieve an approximately 4% error rate at 10 labels queried, only reaching an approximately 2% error rate when $\gtrsim 200$ labels were queried.

This chapter, in part, is being prepared for submission for publication of the material. The dissertation author was the primary investigator and author of this material. Professor Alexander Cloninger is the sole co-author of the material being prepared for submission for publication.

Chapter 3

Tensor Network Theory

3.1 Introduction

Tensor networks can be used to provide a low-dimensional parameterization of an extremely high dimensional vector space having a tensor product structure (Orús 2019). Prototypical tensor network examples include Tensor Train Networks/Matrix Product States (TTN/MPS) and Projected Entangled Pair States (PEPS) (Orús 2014), and Multiscale Entanglement Renormalization Ansatz (MERA) (Evenbly and Vidal 2011). For a review of the basic concepts of tensor networks, which we will not rehash here, see Biamonte and Bergholm 2017.

3.2 PEPS Patch Networks

In this work two dimensional PEPS network will form the basis of a generative image model. While PEPS networks provide a memory-efficient way to *store* vectors in the total Hilbert space, contractions of such networks are notoriously difficult, in general exhibiting exponential complexity in the size of the network (Pang et al. 2020). Indeed, it is precisely this difficulty which explains the popularity of tree tensor networks (S. Cheng et al. 2019) and matrix product states (Stoudenmire and Schwab 2016) on image data, despite their inability to respect the two-dimensional nature of the correlations in a natural fashion.

To address this, we propose slicing a PEPS network into patches as shown in figure (3.1) and using these patches to perform local computations using only information in spatially local portions of the image and network. For $N \times M$ pixel greyscale images, denote by $\underline{\underline{x}} \in \mathbb{R}^{N\times M}$ the $N \times M$ element matrix encoding the brightness of each pixel (where $x_{ij} \in [0,1]$, with 0 corresponding to black and 1 corresponding to white). Next, define vector spaces corresponding to the pixels, patches, and images as:

Pixel $\mathscr{H}_{nm}$ is the local Hilbert space associated with the pixel at row n and column m, with dimension $d_{nm} \equiv \dim(\mathscr{H}_{nm})$

Patch $\mathscr{H}_{nm}^{(p)} \equiv \bigotimes_{n'=n-p}^{n+p} \bigotimes_{m'=m-p}^{m+p} \mathscr{H}_{nm}$ is the *patch* Hilbert space consisting of the tensor product of all local Hilbert spaces within $p \in \mathbb{Z}_{++}$ of (n,m) in ∞-norm. It is understood that if the indices n', m' in these tensor products runs out of the valid range $n \in [1,N]_{\mathbb{Z}}$, $m \in [1,M]_{\mathbb{Z}}$, that those terms are omitted.

Image $\mathscr{H}_{\text{tot}} \equiv \bigotimes_{n=1}^{N} \bigotimes_{m=1}^{M} \mathscr{H}_{nm}$ is the hilbert space associated with the full-image embedding of dimension $D = \prod_{n=1}^{N} \prod_{m=1}^{M} d_{nm}$.

We can then define feature maps for pixels, patches, and whole images, where we lift the description of an $N \times M$ pixel image to a high-dimensional vector space defined by a tensor product structure:

Pixel $\phi : \mathbb{R} \to \mathscr{H}_{nm}$ is a pixel feature map. $\underline{\phi}(x) = [\cos(\pi x/2), \sin(\pi x/2)]^{\dagger}$ is one example of such a map, which will be used throughout the rest of this work.

Patch $\underline{\Phi}_{nm}^{(p)}(\underline{\underline{x}}) \equiv \bigotimes_{n'=n-p}^{n+p} \bigotimes_{m'=m-p}^{m+p} \underline{\phi}(x_{nm})$ is the feature map of the patch centered at pixel (n,m) of ∞-norm radius p. Note that $\underline{\Phi}_{nm}^{(p)}(\underline{\underline{x}}) \in \mathscr{H}_{nm}^{(p)}$.

Image $\underline{\Phi}(\underline{\underline{x}}) = \bigotimes_{n=1}^{N} \bigotimes_{m=1}^{M} \underline{\phi}(x_{nm})$ is the feature map of the entire image. Clearly, $\underline{\Phi}(\underline{\underline{x}}) \in \mathscr{H}_{\text{tot}}$

The model we propose for image generation is to define a set of positive semi-definite (PSD) matrices $\left\{ \underline{\underline{\rho}}_{nm}^{(p)} \,\middle|\, \underline{\underline{\rho}}_{nm}^{(p)} \in \mathscr{H}_{nm}^{(p)} \otimes \mathscr{H}_{nm}^{(p)\dagger} \ \forall n \in [1,N]_{\mathbb{Z}},\ m \in [1,M]_{\mathbb{Z}} \right\}$. These PSD matrices are then used for image generation as detailed by algorithm (5).

Algorithm 5. Image generation algorithm

```
  $\forall\, n \in \{1,...,N\}$, $m \in \{1,...,M\}$, define a PSD matrix $\underline{\underline{\rho}}^{(p)}_{nm} \in \mathscr{H}^{(p)}_{nm} \otimes \mathscr{H}^{(p)\dagger}_{nm}$ from the training data.

  Initialize a random image $\underline{\underline{x}} \in [0,1]^{N\times M}_{\mathbb{R}}$

  $\underline{\phi}^{(nm)} \equiv \underline{\phi}(x_{nm})$                    ▷ per-pixel feature map

Require: $\eta \in \mathbb{R}_+$                    ▷ image generation step size
  repeat

      for n=1 to N do
          for m=1 to M do
              $\Delta x_{nm} \leftarrow \eta \left(\frac{\partial \underline{\phi}^{nm}}{\partial x_{nm}}\right) \left(\partial_{\underline{\phi}^{nm}} \ln\left(\left(\underline{\Phi}^{(p)}_{nm}(\underline{\underline{x}})\right)^{\dagger} \cdot \underline{\underline{\rho}}^{(p)}_{nm} \cdot \underline{\Phi}^{(p)}_{nm}(\underline{\underline{x}})\right)\right)$
          end for
      end for

      for n=1 to N do
          for m=1 to M do
              $x_{nm} \leftarrow x_{nm} + \Delta x_{nm}$
          end for
      end for

  until convergence of $\underline{\underline{x}}$
```

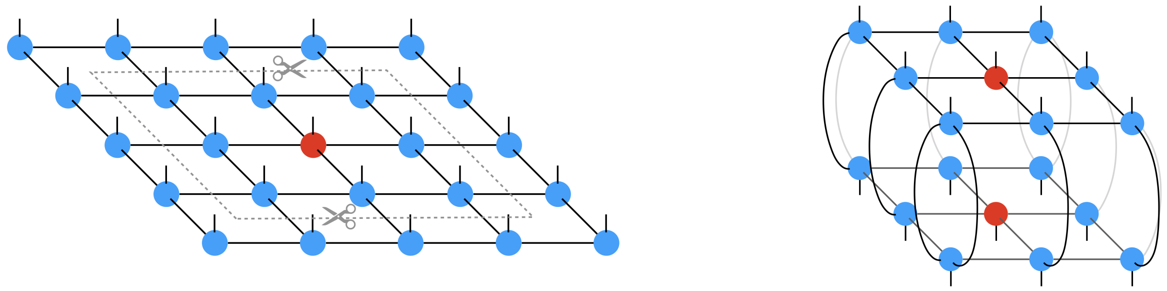


Figure 3.1. At left, a full PEPS network is shown, with the tensor corresponding to pixel (n,m) highlighted in red. The dangling legs here are the *physical* legs which correspond to the Hilbert spaces $\mathscr{H}_{nm}$. For patch radius $p=1$, we can cut out the 3×3 sub-network shown. At right, the (unnormalized) patch density matrix $\underline{\underline{\rho}}_{nm}^{(p)} \in \mathscr{H}_{nm}^{(p)} \otimes \mathscr{H}_{nm}^{(p)\dagger}$ is shown, where the cut internal legs along the edge of the 3×3 patch are joined to a second copy of the 3×3 sub-network, which represents its adjoint.

3.3 Training PEPS Patch Networks

The choice of how to define $\underline{\underline{\rho}}_{nm}^{(p)}$ from the training data

$$X_{\text{train}} \equiv \left\{ \underline{\underline{x}}_{\text{train}}^{(n)} \,\middle|\, \underline{\underline{x}}_{\text{train}}^{(n)} \in [0,1]_{\mathbb{R}}^{N\times M},\; n\in[1,N_{\text{train}}] \right\} \tag{3.1}$$

and which pixel feature map $\underline{\phi}(x)$ is used to define $\vec{\Phi}_{nm}^{(p)}(\underline{\underline{x}})$ give different variations of this image generation algorithm. The simplest construction is to take a linear combination of outer products of the patch feature map vectors for each training image:

$$\underline{\underline{\rho}}_{nm}^{(p),\text{naive}} = \sum_{n=1}^{N_{\text{train}}} \underline{\Phi}_{nm}^{(p)}\left(\underline{\underline{x}}_{\text{train}}^{(n)}\right)\left(\underline{\Phi}_{nm}^{(p)}\left(\underline{\underline{x}}_{\text{train}}^{(n)}\right)\right)^{\dagger} \tag{3.2}$$

where the overall normalization of this sum is irrelevant because of the logarithm in the image generation algorithm (5). This naive construction is tantamount to memorizing the training set, however, and so is undesirable from a memory (and memory requirements, for large data sets) perspective.

Instead of this naive construction, we will define our patch density matrices by defining a single PEPS network, and using overlapping patches of it to define our patch density matrices, as shown in figure (3.1). This allows parameter sharing between the different patch density matrices, dramatically reducing the size of the model, especially for larger patch radii p.

Our image generation algorithm (5) implies that $\underline{\Phi}_{nm}^{(p)}(\underline{\underline{x}})^{\dagger} \cdot \underline{\underline{\rho}}_{nm}^{(p)} \cdot \underline{\Phi}_{nm}^{(p)}(\underline{\underline{x}}) \in [0, +\infty)_{\mathbb{R}}$ should be *large* if the image patch centered at pixel (n,m) in $\underline{\underline{x}}$ is similar to the training data, and *small* otherwise. We propose algorithm (6) to achieve this. It features a loss function with two notable features. The first is that the matrix elements $\underline{\Phi}^{\dagger}\underline{\rho}\underline{\Phi}$ only appear as their logs, which allows the overall normalization of the patch density matrices $\underline{\rho}$ to be ignored in derivatives of this quantity, as in the image generation algorithm (5). The second feature is that the loss functions $\mathscr{L}$ try to push all matrix elements of images in the training set to a particular value, $\ln(C)$, where $C \in (0, +\infty)_{\mathbb{R}}$ is a tunable training parameter. This ensures that features common to many training examples do not wash out rarer features in the training set and end up producing a trivial model.

3.3.1 Network Cuts and the *R*-Term Tensor Decomposition

The arbitrariness of the network cuts used to define the patch density matrices $\underline{\underline{\rho}}_{nm}^{(p)}$ seems like a fatal flaw of this proposal. Each leg of every tensor is used in two distinct ways: it is contracted with a leg of a neighboring tensor, but it is also contracted with the corresponding leg of that same tensor adjoint when that leg is cut to define a $(2p+1) \times (2p+1)$ PEPS patch and used to define the density matrix.

This arbitrariness can be lifted by enforcing specific behavior when any of a tensor's internal bond legs are contracted with its adjoint. We achieve this by writing each tensor as an R-term decomposition. WOLOG we consider the case of two order-two tensors (i.e. matrices)[1],

[1]See appendix C.1 for details on this.

Algorithm 6. Model Training Algorithm with Explicitly Represented Tensors

```
Initialize a 2D $N \times M$ PEPS tensor network, with tensors $\left\{ \mathbf{T}^{(nm)} \middle| n \in [1,N]_{\mathbb{Z}},\ m \in [1,M]_{\mathbb{Z}} \right\}$.
Require: $C \in (0,+\infty)_{\mathbb{R}}$                                    ▷ target value for matrix elements
$\mathscr{L}^{(k)}_{nm} \equiv \left( \ln \left( \left( \underline{\Phi}^{(p)}_{nm}(\underline{\underline{x}}^{(k)}_{\text{train}}) \right)^{\dagger} \cdot \underline{\underline{\rho}}^{(p)}_{nm}(\mathbf{T}^{(nm)}) \cdot \underline{\Phi}^{(p)}_{nm}(\underline{\underline{x}}^{(k)}_{\text{train}}) \right) - \ln(C) \right)^2$   ▷ per-training image loss
$\mathscr{L}^{\text{tot.}}_{nm} \equiv \frac{1}{N^{\text{train}}} \sum_{k=1}^{N_{\text{train}}} \mathscr{L}^{(k)}_{nm}$                    ▷ average loss over the training set
Require: $\eta \in \mathbb{R}_{++}$                                          ▷ tensor gradient descent step size

repeat
    for n=1 to N do
        for m=1 to M do
            $\Delta\mathbf{T}^{(nm)} \leftarrow -\eta \, \partial_{\mathbf{T}^{(nm)}} \mathscr{L}^{\text{tot.}}_{nm}$
        end for
    end for

    for n=1 to N do
        for m=1 to M do
            $\mathbf{T}^{(nm)} \leftarrow \mathbf{T}^{(nm)} + \Delta\mathbf{T}^{(nm)}$
        end for
    end for

until convergence of $\{\mathbf{T}^{(nm)}\}$
```

where the outer product construction is used in place of the tensor product notation for clarity:

$$\mathbf{A} = \sum_{r_A=1}^{R_A} \alpha_{r_A} \hat{u}^{(r_A)} \hat{a}_{\text{bond}}^{(r_A)\,\dagger} \tag{3.3}$$

$$\mathbf{B} = \sum_{r_B=1}^{R} \beta_{r_B} \hat{b}_{\text{bond}}^{(r_B)} \hat{v}^{(r_B)\,\dagger} \tag{3.4}$$

where $\hat{a}_{\text{bond}}^{(r_A)}$, $\hat{b}^{(r_B)} \in \mathscr{H}_{\text{bond}}$. The original $N \times M$ PEPS network that $\mathbf{A}$ and $\mathbf{B}$ are tensors of contracts to the same quantity so long as the value of $\mathbf{A} \cdot \mathbf{B}$ is preserved. This contraction of $\mathbf{A}$ with $\mathbf{B}$ is insensitive to the specific values of the unit bond vectors $\{\hat{a}_{\text{bond}}^{(*)}\}$ and $\{\hat{b}_{\text{bond}}^{(*)}\}$ so long as their dot products are preserved:

$$\hat{a}^{(r_A)\,\dagger} \cdot \hat{b}^{(r_B)} = G_{r_A r_B}^{AB} \tag{3.5}$$

which is the manifestation of the so-called *gauge invariance* present in tensor networks (Evenbly 2022). This invariance to the specific values of the bond vectors in the R-term decomposition can then be used to require

$$\hat{a}^{(r_A)\,\dagger} \cdot \hat{a}^{(r'_A)} = G_{r_A r'_A}^{A} = \delta_{r_A r'_A} \qquad \hat{b}^{(r_B)\,\dagger} \cdot \hat{b}^{(r'_B)} = G_{r_B r'_B}^{B} = \delta_{r_B r'_B} \tag{3.6}$$

while keeping $\hat{a}^{(r_A)\,\dagger} \cdot \hat{b}^{(r_B)} = G_{r_A r_B}^{AB}$ fixed.

With this condition on the bond vectors applied, the contractions on the internal legs of every tensor in the network satisfy:

$$\mathbf{A}\mathbf{A}^{\dagger} = \sum_{r_A=1}^{R_A} |\alpha_{r_A}|^2 \hat{u}^{(r_A)} \hat{u}^{(r_A)\,\dagger} \tag{3.7}$$

$$\mathbf{B}\mathbf{B}^{\dagger} = \sum_{r_B=1}^{R_B} |\beta_{r_B}|^2 \hat{v}^{(r_B)} \hat{v}^{(r_B)\,\dagger} \tag{3.8}$$

$$\mathbf{A}\mathbf{B} = \sum_{r_A=1}^{R_A} \sum_{r_B=1}^{R_B} \alpha_{r_A} \beta_{r_B} \left(\hat{a}_{\text{bond}}^{(r_A)\,\dagger} \cdot \hat{b}_{\text{bond}}^{(r_B)} \right) \hat{u}^{(r_A)} \hat{v}^{(r_B)\,\dagger}. \tag{3.9}$$

It should be emphasized that this choice of gauge is largely motivated by computational complexity. With this choice in place, contracting the outer ring of blue tensors in figure (3.1) used to calculate the patch density matrix $\underline{\underline{\rho}}_{nm}^{(p)}$ prevents a quadratic increase in the number of terms in the R-term representation of those contracted edge tensors.

3.3.2 Contracting Patch Density Matrices

The contraction order for computing $\underline{\Phi}_{nm}^{(p)}(\underline{\underline{x}})^{\dagger} \underline{\underline{\rho}}_{nm}^{(p)} \underline{\Phi}_{nm}^{(p)}(\underline{\underline{x}})$ is shown in figure (3.2). The computational complexity of this contraction is well controlled until the last step due to the properties of the patch network. Following the computation step by step:

1. Because the pixel feature maps $\underline{\phi}$ have only one term in an R-term decomposition, as they are simple vectors, contracting the pixel feature maps (the small magenta diamonds) into the outer ring of tensors in the patch (the blue circles) does not increase the number of terms in their R-term decompositions, but rather only rescales the constant of each term.

2. Collapsing the two outer rings of tensors requires us to contract along the bonds that were cut from the full network and rejoined to form the density matrix. These bonds are precisely those discussed in section 3.3.1, and since the gram matrix of the vectors on the bond in the R-term decomposition is orthogonal, this process does not increase R, but rather only rescales the constants in the R-term decomposition.

3. The four corner tensors in the ring of eight tensors are order two, i.e. they are matrices. They can be directly multiplied onto the vectors in the R-term decompositions of the order four tensors in the ring without increasing R.

4. The final contraction of the four order-four tensors into a single order eight tensor unavoidably results in a tensor with R^4 terms in its R-term decomposition in general.

The complexity of the final step, which leads to a quartic increase in the number of terms required to represent the intermediate tensor, leads us to consider a stochastic sampling scheme

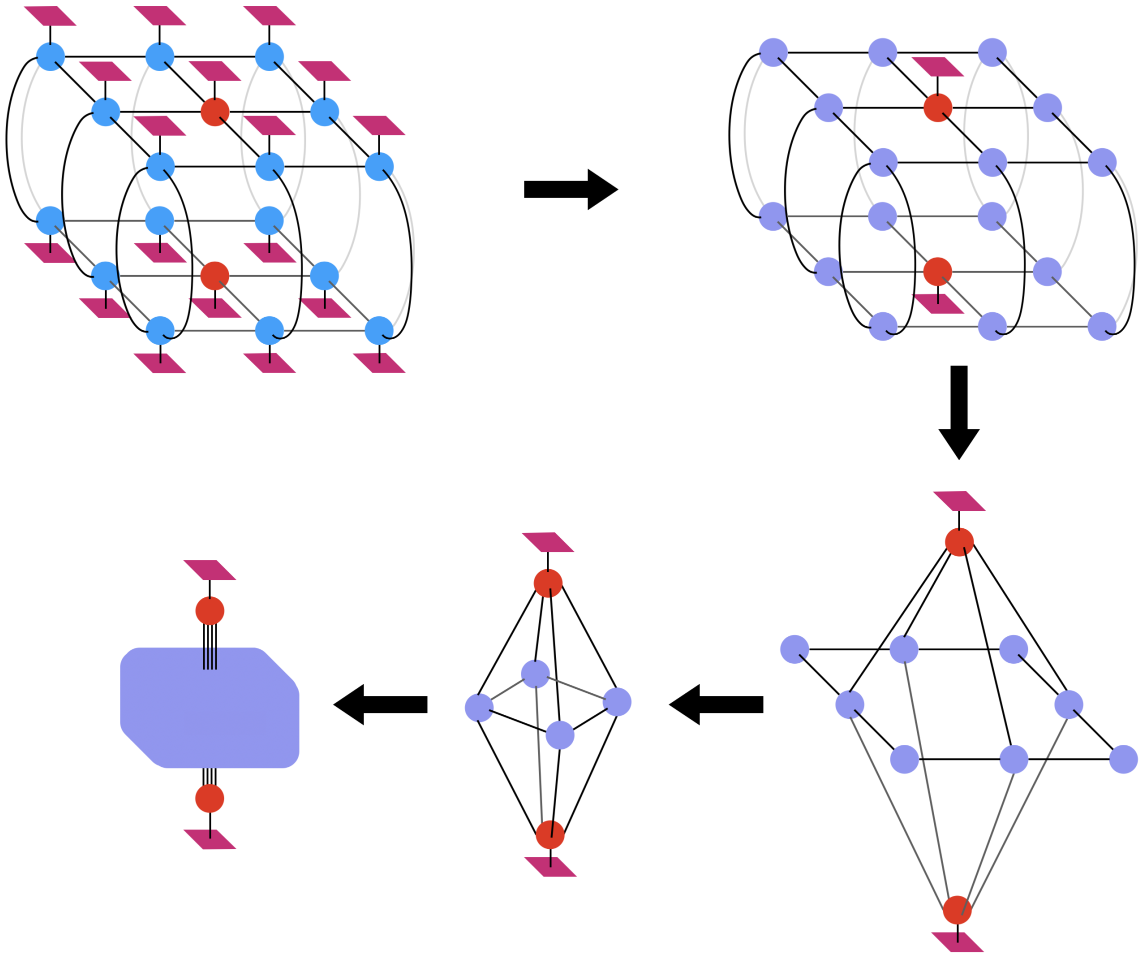


Figure 3.2. Evaluating the scalar $\left(\underline{\Phi}^{(p)}_{nm}(\underline{x})\right)^{\dagger}\underline{\rho}^{(p)}_{nm}\,\underline{\Phi}^{(p)}_{nm}(\underline{x})$ is performed by contracting the tensor network shown, starting from the top left and proceeding clockwise. The red circles represent $\mathbf{T}^{(nm)}$, the tensors at the center of the patch, the blue circles are the surrounding tensors, and the small magenta diamonds are the feature maps $\underline{\phi}(x_{ij})$ of individual pixels in the image. Note that, except for the last contraction step, nothing here increases the number of terms in the R-term representation of the intermediate tensors.

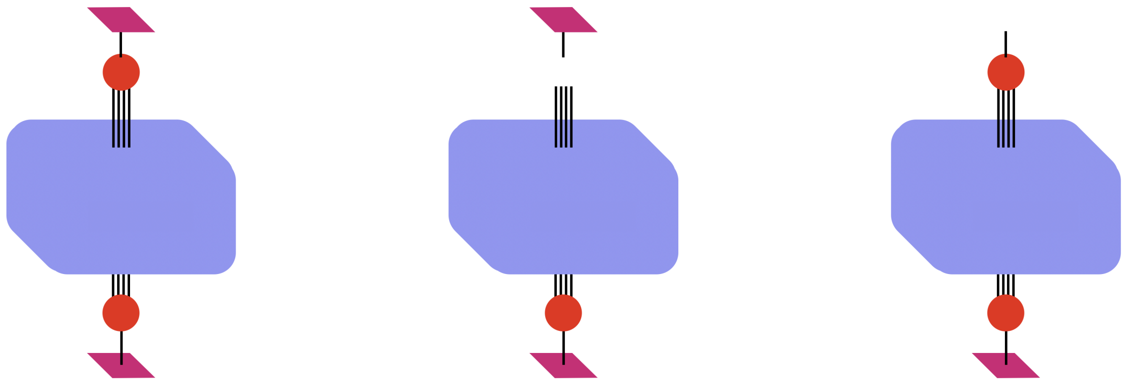


Figure 3.3. From the tensor network representation of the patch density matrix element $g_{nm}^{(p)} \equiv \left(\underline{\Phi}_{nm}^{(p)}(\underline{\underline{x}})\right)^{\dagger} \underline{\rho}_{nm}^{(p)} \underline{\Phi}_{nm}^{(p)}(\underline{\underline{x}})$ (left), we can derive (up to an inessential multiplicative factor of 2) the tensor network representations of the derivative needed for the model training algorithm (6), $\partial_{\mathbf{T}^{(nm)}} g_{nm}^{(p)}$ (center tensor network), as well as the derivative needed for the image generation algorithm (5), $\partial_{\underline{\phi}^{nm}} g_{nm}^{(p)}$ (right tensor network).

which is discussed in section 3.5.

3.4 Gradient Optimization of Tensor Decompositions

If we wish to make use of the convenient properties of the R-term decomposition in these algorithms, our gradient-based optimization of the model tensors must be in the parameter space of the R-term decomposition, not in the explicit representation of the tensor where every element is stored. Assume an order N tensor has an R-term representation

$$\mathbf{T} = \sum_{r=1}^{R} \alpha_r \hat{v}_r^{(1)} \otimes \cdots \otimes \hat{v}_r^{(N)}, \tag{3.10}$$

where $\hat{v}_r^{(n)}$ is the unit vector on leg n from term r in the expansion, and not just one element of that vector. Storing each parameter of this representation of $\mathbf{T}$ involves storing $R\left(1+\sum_{n=1}^{N} d_n\right)$ real numbers. We refer to the the list of these numbers, stored as a $R\left(1+\sum_{n=1}^{N} d_n\right)$ element vector, as $\underline{\mathrm{serial}}(\mathbf{T})$, and will also use the shorthand $\underline{p}$ for these parameters. With this notation, the tensor can also be written as a function of the serialized parameter vector $\mathbf{T}(\underline{p})$. Note that, since R-term decompositions are highly non-unique, "$\underline{\text{serial}}(\cdot)$" cannot be a function of the tensor T itself at all. Nevertheless, the notation is useful and will be assumed to refer to a particular R-term decomposition of the argument of "$\underline{\text{serial}}(\cdot)$", which is typically clear from context. $\mathbf{T}(\underline{p})$, on the other hand, is unambiguous, as each list of parameters maps to a single tensor with knowledge of the dimensions of each leg and the order of serialization.

In the model training algorithm (6) the tensors $\mathbf{T}^{(nm)}$ must be updated to approximate another tensor $\mathbf{T}_{\text{target}} = \mathbf{T} + \Delta$ while, for complexity reasons, keeping the number of terms in the decomposition of $\mathbf{T}$ fixed at R. We can do this by updating the parameters of the R-term decomposition of $\mathbf{T}$ to approximate $\mathbf{T} + \Delta$ in l_2-norm. Computing the gradient of $\left|\mathbf{T}_{\text{target}} - \mathbf{T}(\underline{p} + \delta\underline{p})\right|_2$ with respect to $\delta\underline{p}$ is conceptually and computationally simple. The problem comes at the update step, where we wish to update the parameter list using a classic gradient descent correction:

$$\underline{p}' = \underline{p} - \eta_p \underbrace{\partial_{\underline{p}} \left|\mathbf{T}_{\text{target}} - \mathbf{T}(\underline{p} + \delta\underline{p})\right|_2^2}_{\equiv \underline{g}} = \underline{p} - \eta_p \underline{g}. \tag{3.11}$$

Generically, as $\eta_p \to 0^+$, this results in a first order correction to $\mathbf{T}' = \mathbf{T} + \eta_p \,\delta\mathbf{T} + O(\eta_p^2)$. However, for finite parameter space step sizes η_p, $\mathbf{T}'$ is an $(N+1)^{\text{th}}$ order polynomial in η_p, and for large η_p grows in norm as $\eta_p^{(N+1)}$. What value of η_p is large enough to transition from the safe, linear correction regime to the poorly conditioned regime is specific to $\mathbf{T}$, the ambient-space correction Δ, and thus is dynamic over the course of training.

This presents a *massive* challenge, as this produces a poorly conditioned optimization problem in the parameter space of the R-term decompositions of the tensors since most of the tensors being optimized are order five in a $2D$ PEPS network. We resolve this problem by augmenting the parameter-space gradient descent optimization with information from the ambient space $\mathscr{H} \ni \mathbf{T}$. Our approach is to assume a pre-computed ambient-space correction $\Delta \in \mathscr{H}$ to $\mathbf{T}$. Then do a numerical line-search to find the parameter-space step size η_p in the direction indicated by the gradient which minimizes the error $E(\eta_p)$ between the target tensor $\mathbf{T}_{\text{target}}$ and the corrected tensor $\mathbf{T}(\underline{p} - \eta_p \underline{g})$. This process is detailed in algorithm (7).

Algorithm 7. Ambient-space-informed tensor parameter optimization step

```
    $\Delta \in \mathscr{H}$, $\underline{p} \in \mathbb{R}^{(1+d_1+\cdots+d_N)R}$        $\triangleright$ $\Delta$ is the ambient space step computed elsewhere
**Require:** $|\Delta|_2 > 0$
    $\mathbf{T}(\cdot) : \mathbb{R}^{(1+d_1+\cdots+d_N)R} \to \mathscr{H}$
    $\mathbf{T}_{\text{target}} \equiv \mathbf{T}(\underline{p}) + \Delta$
    $\underline{g} \leftarrow \partial_{\delta \underline{p}} \left|\mathbf{T}_{\text{target}} - \mathbf{T}(\underline{p} + \delta \underline{p})\right|_2^2 \Big|_{\delta \underline{p}=0}$
    $E(\eta_p) \equiv \left|\mathbf{T}_{\text{target}} - \mathbf{T}(\underline{p} - \eta_p \underline{g})\right|_2$
    $\eta_p^{\min} \leftarrow \text{argmin}(E(\eta_p) \,|\, \eta_p \geq 0)$
    $\underline{p} \leftarrow \underline{p} - \eta_p^{\min} \underline{g}$
```

The argmin optimization over the error $E(\eta_p)$ can be efficiently performed by a variety of line search optimizers. $E(\eta_p)$ can be written as a function of three inner products:

$$E(\eta_p) = \sqrt{\mathbf{T}_{\text{target}}^\dagger \cdot \mathbf{T}_{\text{target}} - 2\left(\mathbf{T}_{\text{target}}^\dagger \cdot \mathbf{T}(\underline{p} - \eta_p \underline{g})\right) + \mathbf{T}^\dagger(\underline{p} - \eta_p \underline{g}) \cdot \mathbf{T}(\underline{p} - \eta_p \underline{g})}. \tag{3.12}$$

While the $\mathbf{T}_{\text{target}}^\dagger \cdot \mathbf{T}_{\text{target}}$ term is high computational complexity since generally the update Δ that is being approximated here has a large number of terms in its R-term representation, this term need only be computed once at the beginning of the optimization process. While $\mathbf{T}_{\text{target}}^\dagger \cdot \mathbf{T}(\underline{p} - \eta_p \underline{g})$ and $\mathbf{T}^\dagger(\underline{p} - \eta_p \underline{g}) \cdot \mathbf{T}(\underline{p} - \eta_p \underline{g})$ must be re-computed at every evaluation of $E(\eta_p)$ during the argmin optimization, they are comparatively cheap due to the low number of terms in the R-term representation of $\mathbf{T}(\underline{p} - \eta \underline{g})$.

3.5 Unbiased Sampling of MPS/MPO Networks

The quartic contraction complexity of the last contraction in used to construct the patch quantities in figure 3.2 encourages exploration of statistical, unbiased estimates of the diamond of four tensors, and whether this can be done efficiently using the R-term decomposition[2] of the relevant tensors. To investigate this, use the tensor origami procedure outlined in appendix C.1 to transform an arbitrary-order tensor **A** into a vector $\underline{A} = \sum_{\vec{r}} \alpha_{\vec{r}} \hat{v}_{\vec{r}}$. Then define a random variable $\widetilde{\underline{A}}$ as follows:

$$\widetilde{\underline{A}} = \sum_{\vec{r}} \widetilde{\alpha}_{\vec{r}} \hat{v}_{\vec{r}} \tag{3.13}$$

We wish to choose the properties of $\widetilde{\alpha}_{\vec{r}}$ such that $\mathbb{E}(\widetilde{\underline{A}}) = \underline{A}$. Exploiting the linearity of expected values:

$$\mathbb{E}(\widetilde{\underline{A}} - \underline{A}) = \mathbb{E}\left(\sum_{\vec{r}} (\widetilde{\alpha}_{\vec{r}} - \alpha_{\vec{r}}) \hat{v}_{\vec{r}}\right) = \sum_{\vec{r}} \mathbb{E}(\widetilde{\alpha}_{\vec{r}} - \alpha_{\vec{r}}) \hat{v}_{\vec{r}} = 0 \tag{3.14}$$

A sufficient (though not necessary) condition to satisfy this equation is

$$\mathbb{E}(\widetilde{\alpha}_{\vec{r}}) = \alpha_{\vec{r}} \tag{3.15}$$

3.6 Controlling the Sampling Variance

3.6.1 Fixing the Mean

When approximating a vector $|B\rangle = \sum_{i=1}^{N_B} b_i |i\rangle$ as $|\tilde{B}\rangle = \sum_{i=1}^{N_B} Y_i |i\rangle$, where Y_i are random variables and $\langle i|j\rangle = G_{ij}$, the linearity of the expectation guarantees that $\mathbb{E}(|\tilde{B}\rangle) = |B\rangle$ so long as $\mathbb{E}(Y_i) = b_i$. The variance will depend on more detailed properties of the random variables $\{Y_i\}$,

[2]The R-term decomposition, also called the CANDECOMP/PARFAC decomposition, and other tensor decompositions are reviewed in Grasedyck, Kressner, and Tobler 2013.

however, as follows:

$$\begin{aligned}
\mathbb{E}\left(\left||B\rangle - |\tilde{B}\rangle\right|_2^2\right) &= \mathbb{E}\left(\langle B|B\rangle + \langle\tilde{B}|\tilde{B}\rangle - \langle\tilde{B}|B\rangle - \langle B|\tilde{B}\rangle\right) \\
&= \mathbb{E}\left(\langle\tilde{B}|\tilde{B}\rangle\right) - \langle B|B\rangle \\
&= \sum_{i,j}\left[\mathbb{E}\left(Y_i^*Y_j\right) - b_i^*b_j\right]G_{ij} \\
&= \sum_{i,j}\left[\sigma_{ij} + \mathbb{E}(Y_i^*)\mathbb{E}(Y_j) - b_i^*b_j\right]G_{ij} \\
\mathbb{E}\left(\left||B\rangle - |\tilde{B}\rangle\right|_2^2\right) &= \sum_{i,j}\sigma_{ij}G_{ij}
\end{aligned}$$

where $\sigma_{ij} \equiv \mathbb{E}\left(Y_i^*Y_j\right) - \mathbb{E}(Y_i^*)\mathbb{E}(Y_j)$ is the covariance matrix of the Y_i random variables, and it is assumed above that $\mathbb{E}(Y_i) = b_i$.

If we define $\{Y_i | i \in \{1, 2, \ldots N_B\}\}$ as

$$Y_i \equiv c_i \sum_{m=1}^{M_B} \delta_{i,Q_m^{(\alpha)}} \tag{3.16}$$

where $c_i \in \mathbb{C}$, and $\{Q_m^{(\alpha)} | m \in \{1, 2, \ldots, M\}\}$ is a set of M i.i.d. integer-valued random variables taking on values $n \in \{1, 2, \ldots, N\}$ with probabilities

$$p_n(\alpha) \equiv \frac{|b_n|^\alpha}{\sum_{k=1}^N |b_k|^\alpha} \tag{3.17}$$

where $\alpha \in [0, \infty)$ is a parameter allowing us to affect the sampling procedure by rescaling the relative probability for sampling a given term in $|B\rangle$ out of the N terms present.

Choosing the value of c_i to enforce $\mathbb{E}(Y_i) = b_i$ as follows:

$$\mathbb{E}(Y_i) = c_i \sum_{m=1}^{M} \mathbb{E}\left(\delta_{i,Q_m^{(\alpha)}}\right) = c_i \sum_{m=1}^{M} p_i(\alpha) = c_i M p_i(\alpha) = b_i \tag{3.18}$$

$$c_i = \frac{b_i}{M p_i(\alpha)} = \frac{b_i}{M}\frac{\sum_{j=1}^N |b_j|^\alpha}{|b_i|^\alpha} \tag{3.19}$$

3.6.2 Computing the Variance

To get a handle on the variance, we first compute $\mathbb{E}(Y_i^* Y_j)$:

$$\mathbb{E}\left(Y_i^* Y_j\right) = \mathbb{E}\left[\left(c_i \sum_{m=1}^{M} \delta_{i,Q_m^{(\alpha)}}\right)^* \left(c_j \sum_{m'=1}^{M} \delta_{j,Q_{m'}^{(\alpha)}}\right)\right] \tag{3.20}$$

$$= c_i^* c_j \left[\sum_{\substack{m,m'=1 \\ m\neq m'}}^{M} \mathbb{E}\left(\delta_{i,Q_m^{(\alpha)}} \delta_{j,Q_{m'}^{(\alpha)}}\right) + \sum_{m=1}^{M} \delta_{i,Q_m^{(\alpha)}} \delta_{j,Q_m^{(\alpha)}}\right] \tag{3.21}$$

For $i = j$, this reduces to (suppressing the α dependence on c and p for brevity):

$$\mathbb{E}\left(Y_i^* Y_j\right)|_{i=j} = |c_i|^2 \left[\sum_{\substack{m,m'=1 \\ m\neq m'}}^{M} p_i p_j + \sum_{m=1}^{M} p_i\right] = |c_i|^2 \left[(M^2 - M) p_i^2 + M p_i\right] \tag{3.22}$$

and for $i \neq j$:

$$\mathbb{E}\left(Y_i^* Y_j\right)|_{i\neq j} = c_i^* c_j \left[\sum_{\substack{m,m'=1 \\ m\neq m'}}^{M} p_i p_j + 0\right] = c_i^* c_j \left(M^2 - M\right) p_i p_j \tag{3.23}$$

More compactly, all cases are covered by:

$$\mathbb{E}(Y_i^* Y_j) = c_i^* c_j \left[(M^2 - M) p_i p_j + M p_i \delta_{ij}\right] \tag{3.24}$$

The covariance σ_{ij} is then (liberally using the fact that $b_i = c_i M p_i(\alpha)$ throughout):

$$\sigma_{ij} = \mathbb{E}(Y_i^* Y_j) - \mathbb{E}(Y_i^*)\mathbb{E}(Y_j) \tag{3.25}$$

$$= c_i^* c_j (M^2 - M) p_i p_j + c_i^* c_j M p_i \delta_{ij} - (c_i^* M p_i)(c_j M p_j) \tag{3.26}$$

$$= c_i^* c_i M p_i \delta_{ij} - c_i^* c_j M p_i p_j \tag{3.27}$$

$$\sigma_{ij} = \frac{1}{M p_i(\alpha)} |b_i|^2 \delta_{ij} - \frac{1}{M} b_i b_j \tag{3.28}$$

which gives us a variance of our estimate of $|B\rangle$ of:

$$\mathbb{E}\left(\left| |B\rangle - |\tilde{B}\rangle \right|_2^2 \right) = \sum_{i,j} \sigma_{ij} G_{ij} \tag{3.29}$$

$$= \frac{1}{M} \sum_{ij} G_{ij} \left[\frac{1}{p_i(\alpha)} |b_i|^2 \delta_{ij} - b_i^* b_j \right] \tag{3.30}$$

$$= \frac{1}{M} \sum_i G_{ii} \frac{|b_i|^2}{p_i(\alpha)} - \frac{1}{M} \langle B|B\rangle \tag{3.31}$$

WOLOG we may assume that the diagonal entries of $\underline{\underline{G}}$ are all 1 by a rescaling of the associated vectors in the sum expression for $|B\rangle$, yielding the following explicit form for the variance as a function of α:

$$\mathbb{E}\left(\left| |B\rangle - |\tilde{B}\rangle \right|_2^2 \right) = \frac{1}{M} \left[\left(\sum_i |b_i|^{2-\alpha} \right) \left(\sum_j |b_j|^{\alpha} \right) - \left(\sum_{i,j} b_i^* G_{ij} b_j \right) \right] \tag{3.32}$$

Since a basic requirement of our sampling procedure is that the variance should not diverge due to infinitesimal non-zero components of $|B\rangle$, it is clear that we must avoid any negative powers of $|b_{\dots}|$ in (3.32). Thus, $\alpha \in [0, 2]$.

3.6.3 Minimizing the Variance

Since the variance of our estimate of $|B\rangle$ depends on α as shown in (3.32), we can optimize over α to minimize the variance. Noting that the α dependence is contained entirely in the first term, and assuming that $G_{ii} = 1$ WOLOG, we find:

$$\partial_\alpha \mathbb{E}\left(\left||B\rangle - |\tilde{B}\rangle\right|_2^2\right) = \frac{1}{M}\sum_i |b_i|^2 \left(\partial_\alpha \frac{1}{p_i(\alpha)}\right) \tag{3.33}$$

$$= \frac{1}{M}\sum_i |b_i|^2 \left[\frac{\sum_k |b_k|^\alpha \ln|b_k|}{|b_i|^\alpha} - \frac{\ln|b_i| \sum_k |b_k|^\alpha}{|b_i|^\alpha}\right] \tag{3.34}$$

$$= \frac{1}{M}\sum_{k,i} \left[|b_k|^\alpha |b_i|^{2-\alpha} \ln|b_k| - |b_k|^\alpha |b_i|^{2-\alpha} \ln|b_i|\right] \tag{3.35}$$

$$= 0 \implies \alpha = 1 \tag{3.36}$$

The second derivative test verifies that $\alpha = 1$ is a local minimum:

$$\partial_\alpha^2 \mathbb{E}\left(\left||B\rangle - |\tilde{B}\rangle\right|_2^2\right) = \frac{1}{M}\sum_i |b_i|^2 \left(\partial_\alpha^2 \frac{1}{p_i(\alpha)}\right) \tag{3.37}$$

$$= \frac{1}{M}\sum_{i,k} |b_i|^{2-\alpha} |b_k|^\alpha \left(\ln|b_i| - \ln|b_k|\right)^2 \tag{3.38}$$

which is clearly positive for $\alpha \in [0,2]$, implying that $\alpha = 1$ is the minimum in this interval.

Our variance for the sampling procedure corresponding to $\alpha = 1$ is then:

$$\mathbb{E}\left(\left||B\rangle - |\tilde{B}\rangle\right|_2^2\right)\Bigg|_{\substack{G_{kk}=1\\ \alpha=1}} = \frac{1}{M}\left[\left(\sum_i |b_i|\right)^2 - \left(\sum_{i,j} b_i^* G_{ij} b_j\right)\right] \tag{3.39}$$

where the only assumption made is that the diagonal elements of $\underline{\underline{G}}$ are 1, i.e. the vectors $|i\rangle$ in the expression $|B\rangle = \sum_{i=1}^N b_i |i\rangle$ are normalized.

The relative error can then be defined as:

$$\sqrt{\frac{\mathbb{E}\left(\left||B\rangle - |\tilde{B}\rangle\right|^2\right)}{\langle B|B\rangle}} = \frac{1}{\sqrt{M}}\sqrt{\frac{\left(\sum_i |b_i|\right)^2}{\sum_{i,j} b_i^* G_{ij} b_j} - 1} \tag{3.40}$$

In the case where the constituent vectors of $|B\rangle$ are orthonormal, i.e. $G_{ij} = \delta ij$, this simplifies to a function of the ratio of the L_1 and L_2 norms:

$$\left.\sqrt{\frac{\mathbb{E}\left(\left||B\rangle - |\tilde{B}\rangle\right|^2\right)}{\langle B|B\rangle}}\right|_{\substack{G_{ij}=\delta_{ij}\\ \alpha=1}} = \frac{1}{\sqrt{M}}\sqrt{\left(\frac{|\underline{b}|_1}{|\underline{b}|_2}\right)^2 - 1} \tag{3.41}$$

where $|\underline{b}|_p$ is the p-norm of the column vector of the list of coefficients of the expansion of $|B\rangle$ in the $\{|i\rangle\}$ frame.

Chapter 4

Tensor Network Numerics

To validate the general approach, we turn to image inpainting (Elharrouss et al. 2020). Using the naive density matrices constructed directly from the training set as in equation (3.2), reproduced here:

$$\underline{\underline{\rho}}_{nm}^{(p),\text{naive}} = \sum_{n=1}^{N_{\text{train}}} \underline{\underline{\Phi}}_{nm}^{(p)} \left(\underline{\underline{x}}_{\text{train}}^{(n)} \right) \left(\underline{\underline{\Phi}}_{nm}^{(p)} \left(\underline{\underline{x}}_{\text{train}}^{(n)} \right) \right)^{\dagger}. \tag{4.1}$$

Image inpainting takes various forms, and here we use the following variety:

1. Draw a seed image $\underline{\underline{x}}^{\text{seed}}$ from the validation set of one of the classes from the MNIST[1] data set.

2. Create an initial image $\underline{\underline{x}}^{(0)}$ for the inpainting algorithm by copying a fraction $p \in (0,1)_{\mathbb{R}}$ of the pixels of the seed image, and initializing the remaining pixels in the initial image to a value of $\frac{1}{2}$. [2]

3. Use the image generation algorithm (5) on the initial image $\underline{\underline{x}}^{(0)}$ while *holding the value of the pixels drawn from* $\underline{\underline{x}}^{\text{seed}}$ *fixed* using the *training* set from the same MNIST class to compute $\underline{\underline{\rho}}_{nm}^{(p),\text{ naive}}$.

[1]The MNIST data set is a common test bed for machine learning algorithms consisting of 28×28 pixel greyscale images of handwritten digits with corresponding labels. See LeCun et al. 1998 for details.

[2]Pixels in greyscale images are assumed to have values in $[0,1]_{\mathbb{R}}$ throughout this chapter, where 0 corresponds to a black pixel, and 1 to a white pixel.

Step 3 involves evaluating following matrix element and its derivative w.r.t. x_{nm}:

$$\left(\underline{\Phi}_{nm}^{(p)}\left(\underline{\underline{x}}\right)\right)^{\dagger} \underline{\underline{\rho}}_{nm}^{(p),\text{ naive}}\, \underline{\Phi}_{nm}^{(p)}\left(\underline{\underline{x}}\right) \tag{4.2}$$

This is prohibitive to perform at every gradient step if $\underline{\underline{\rho}}_{nm}^{(p),\text{ naive}}$ is constructed from the *entire* training set. Instead, this work uses a stochastic gradient optimization where at each update, the gradient is computed from an approximate patch density matrix

$$\widetilde{\underline{\underline{\rho}}}_{nm}^{(p),\text{ naive}} \equiv \sum_{\underline{\underline{x}}\in X_{\text{batch}}} \underline{\Phi}_{nm}^{(p)}\left(\underline{\underline{x}}\right)\left(\underline{\Phi}_{nm}^{(p)}\left(\underline{\underline{x}}\right)\right)^{\dagger}, \quad X_{\text{batch}} \subset X_{\text{train}} \tag{4.3}$$

Using $p = 2$ (i.e. 5×5 patches) on the 28×28 pixel images from the '6' MNIST class, image inpainting was performed at mask percentages of 20%, 40%, 60%, and 80%, with euclidean reconstruction errors and the pairwise distances between the images in the validation set used for seed images reported in figure (4.1). The reconstruction errors are significantly below the mean pairwise distances in the validation set, suggesting that the inpainting is performing well.

With a validity check cleared, we turn to examining the behavior of patch PEPS networks under training. Figure 4.2 compares the Shannon entropy of the normalized list of constants in each tensor's R-term decomposition to the standard deviation of the pixels in the '6' class of the MNIST training set. It is consistent with but not persuasive of the notion that the information from the training set is directly informing information content of the tensors as training goes on.

4.1 Future Directions

The clear next step in this work is to do an exhaustive evaluation of the generative abilities of a trained patch PEPS network, and comparing the performance of this compressive model to the performance of the naive model. Because the lowest complexity contractions are only available for $p = 1$, i.e. 3×3 pixel patches, it is likely that significant computational problems

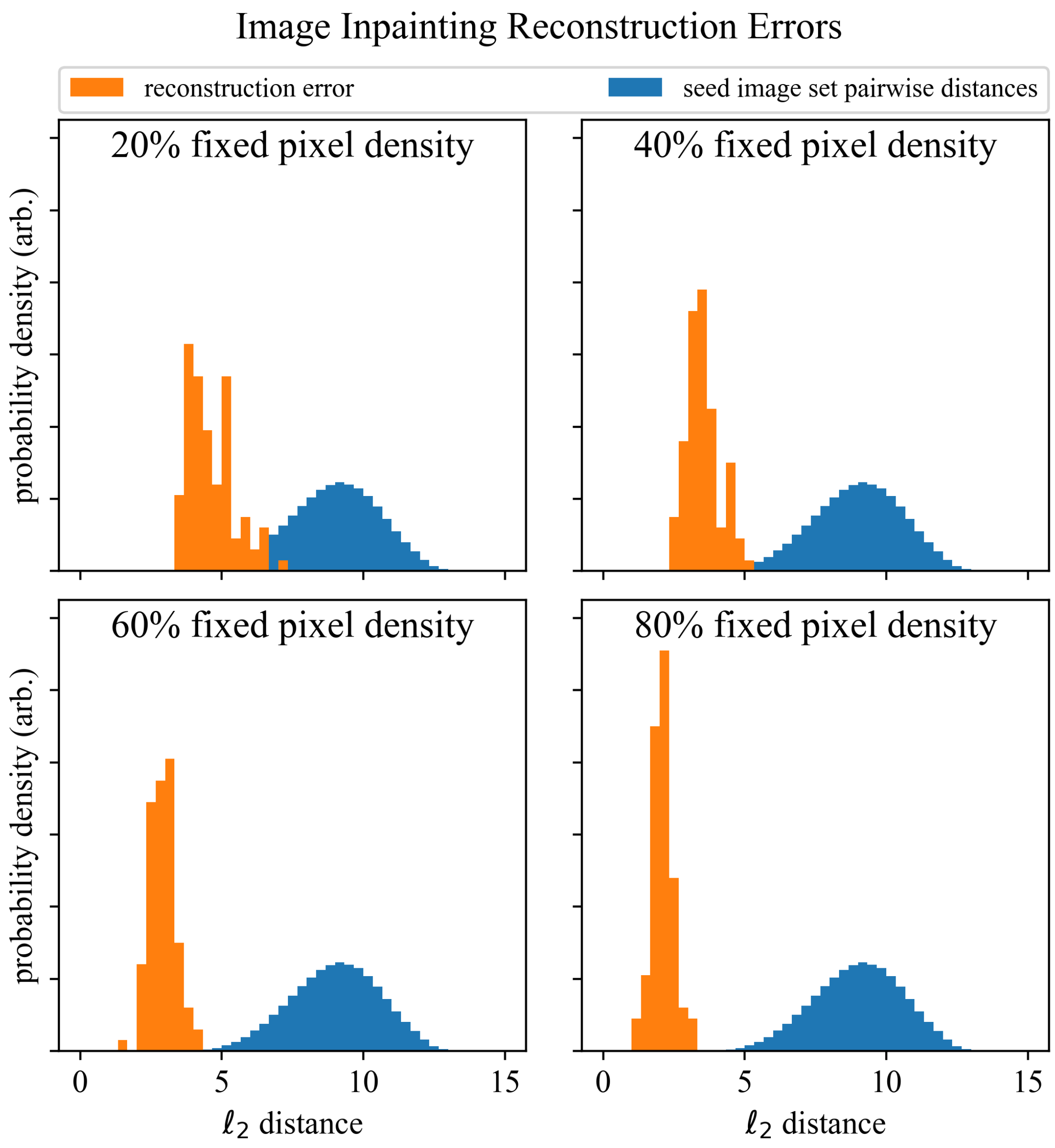


Figure 4.1. The reconstruction errors of the inpainted images relative to their seed images are shown for four different fractions of fixed pixels from the seed image. To produce these histograms a set of 100 images were inpainted at each fixed pixel density. To provide a reference scale for the reconstruction errors, the distribution of pairwise distances in the seed image set (i.e. the validation set of the MNIST ‘6’ class) is shown on every plot.

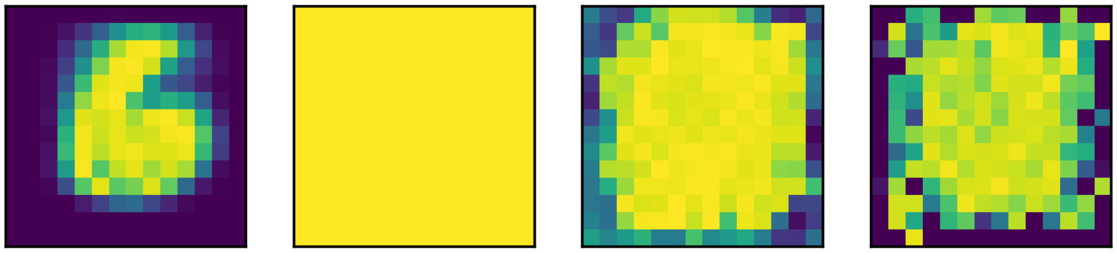


Figure 4.2. A comparison of the standard deviation of the pixel values in the '6' class in the MNIST training set (left), and the evolution over a small number of optimization steps of the entropy of the constants in the R-term decomposition of the PEPS tensor, (right three images).

may arise as the image size scales up and where larger patches that allow for incorporating longer-distance correlations directly may be beneficial. Characterizing the properties of the *full* tesor network trained in this patch fashion is also a potentially interesting direction of work. Since image-local correlations were forced to be stored and resolved locally during the training process, is there some sense in which this is reflected in the full PEPS network? Is it is more amenable to approximate contraction if it arises in this fashion?

Appendix A

Notation

A.1 Sets

$\mathbb{Z}$	$\{\cdots,-1,0,1,\cdots\}$
$\mathbb{Z}_+$	$\{0,1,2,\cdots\}$
$\mathbb{Z}_{++}$	$\{1,2,\cdots\}$
$[n,m]_\mathbb{Z}$	$\{q\in\mathbb{Z}\|n\le q\le m\}$ for $n,m\in\mathbb{Z}$
$(n,m)_\mathbb{Z}$	$\{q\in\mathbb{Z}\|n\le q< m\}$ for $n,m\in\mathbb{Z}$
$\mathbb{R}$	The real numbers
$\mathbb{R}_+$	$\{x\in\mathbb{R}\|x\ge 0\}$
$\mathbb{R}_{++}$	$\{x\in\mathbb{R}\|x> 0\}$
$[a,b]_\mathbb{R}$	$\{x\in\mathbb{R}\|a\le x\le b\}$ for $a,b\in\mathbb{R}$, $a\le b$
$(a,b)_\mathbb{R}$	$\{x\in\mathbb{R}\|a< x< b\}$ for $a,b\in\mathbb{R}$, $a\le b$
$\mathbb{C}$	The complex numbers

A.2 Arrays

An order N array is a finite grid of scalars, indexed by N integers $(i_1, i_2, ..., i_N)$. The nth index takes on values $i_n \in [1, \dim(n)]_{\mathbb{Z}}$, where $\dim(n) \in \mathbb{Z}_{++}$ is the *dimension* of the nth index. When representing an array as a whole rather than a particular indexed element of an array, we will frequently (but not always, especially for high-order tensors) denote the order of the array by the number of underlines beneath its symbol, i.e. $\underline{\underline{W}}$ is an order 2 array (a matrix), $\underline{x}$ is an order 1 array (a column vector by convention), and $\underline{\underline{W}}\,\underline{x}$ represents matrix multiplication.

The transpose of a column vector $\underline{x}$ is a row vector, denoted by $\underline{x}^\dagger$. The transpose of a matrix $\underline{\underline{M}}$ is also a matrix, and is similarly denoted by $\underline{\underline{M}}^\dagger$.

Square matrices can be constructed from vectors by filling the main diagonal with the entries of the vector, and setting all off-diagonal entries to 0. This is denoted by $\underline{\underline{\backslash v}}$ s.t. $\backslash v_{ij} = \delta_{ij} v_i$, where the backwards slash is meant to be evocative of the main diagonal of a matrix.

Certain special matrices and vectors receive their own symbols. The (square) identity matrix is denoted by $\underline{\underline{\mathbb{I}}}$. Arrays with every element set to 1 are denoted by $\mathbb{1}$, with the appropriate number of underlines denoting the order of the array, i.e. the ones column vector $\underline{\mathbb{1}}$, the ones row vector $\underline{\mathbb{1}}^\dagger$, the ones matrix $\underline{\underline{\mathbb{1}}}$, etc. The ith one-hot (or standard basis) vector with all elements zero except the ith element, which is one, is denoted by $\underline{e}^{(i)}$.

Element-wise binary operations between arrays are common and are defined analogously for all order arrays of identical dimensions. Using vectors to illustrate these operations:

multiplication	$\underline{x} \odot \underline{y}$	$\Longrightarrow \quad \left[\underline{x} \odot \underline{y}\right]_i = x_i y_i$
division	$\underline{x} \oslash \underline{y}$	$\Longrightarrow \quad \left[\underline{x} \oslash \underline{y}\right]_i = x_i / y_i$
exponentiation	$q^{\odot \underline{x}}$	$\Longrightarrow \quad [q^{\odot \underline{x}}]_i = q^{x_i}$
	$\underline{x}^{\odot q}$	$\Longrightarrow \quad [\underline{x}^{\odot q}]_i = (x_i)^q$
	$\underline{x}^{\odot \underline{y}}$	$\Longrightarrow \quad \left[\underline{x}^{\odot \underline{y}}\right]_i = (x_i)^{y_i}$

A.3 Graphs

A.3.1 General Scalar Weighted Graphs

A weighted graph is a tuple $G = (V, E, w)$ consisting of the vertex set V, the edge set $E = V \times V$, and a weight function $w : E \to \mathbb{R}_+$, where $\mathbb{R}_+ \equiv \{x \in \mathbb{R} | x \geq 0\}$. By convention we include *all* tuples of vertices in the edge set, and encode a missing edge by setting its weight to 0.

By choosing an ordering for the vertices we may refer to them as $v_i \in V$, where $i \in \{1, 2, ..., |V|\}$. Once this ordering is chosen, the structure of the graph's connections and weights can be represented by a single matrix with non-negative entries $W_{ij} = w(v_i, v_j)$. Furthermore, functions on the vertex set $f : V \to \mathbb{R}$ can be written as a vector $\underline{f} \in \mathbb{R}^{|V|}$, where the i^{th} element $f_i = f(v_i)$.

A.3.2 Partitioned Weighted Graphs

We will frequently consider graphs with some ground truth partitioning of its vertex set. The *full graph* is $G = (V, E, w)$. Its vertices are assumed to have a natural partitioning $V = V_1 \sqcup \cdots \sqcup V_{N_P}$, and an ordering $(v_1, \cdots, v_{|V|})$ that respects this partitioning so they may be indexed with a natural number $i \in \{1, \cdots, |V|\}$. This partitioning induces a block structure of the weight matrix:

$$\underline{\underline{W}}^{G} = \begin{bmatrix} \underline{\underline{W}}^{(1,1)} & \cdots & \underline{\underline{W}}^{(1,N_P)} \\ \vdots & \ddots & \vdots \\ \underline{\underline{W}}^{(N_P,1)} & \cdots & \underline{\underline{W}}^{(N_P,N_P)} \end{bmatrix} \tag{A.1}$$

This natural partitioning can be used to define N_P *induced subgraphs*

$$\left\{ G_n = (V_n, E_n, w_n) \,\middle|\, n \in \{1, \cdots, N_P\},\ E_n = V_n \times V_n,\ w_n = w|_{E_n} \right\} \tag{A.2}$$

where $w|_{E_n}$ is the *restriction* of the weight function $w : E \to \mathbb{R}_+$ to the domain E_n of edges between vertices in V_n. The weight matrix of each induced subgraph from this partition of V is then one of the diagonal blocks of $\underline{\underline{W}}^{G}$:

$$\underline{\underline{W}}^{G_n} = \underline{\underline{W}}^{(n,n)} \tag{A.3}$$

This vertex partitioning can be reflected in a vertex function $f_G : V \to \mathbb{R}$ by partitioning its values as $\underline{f} = \underline{f}^{(1)} \oplus \cdots \oplus \underline{f}^{(N_P)}$, where $\underline{f}^{(n)} \in \mathbb{R}^{|V_n|}$ and $\forall n \in [1, N_P], j \in [1, |V_n|]$, $\exists v \in V_n$ s.t. $f_j^{(n)} = f(v)$.

Appendix B

Graph Clustering

B.1 Green's Function Solution to Linear First-Order Time-Independent Inhomogeneous ODEs

Consider the following first order differential equation for $\underline{u}^{\dagger}(x)$:

$$\underline{u}^{\dagger}(x)\left[\overleftarrow{\partial}_x + \frac{1}{\eta}\underline{\underline{M}}\right] = \underline{f}^{\dagger}(x) + \underline{u}_0^{\dagger}\delta(x), \tag{B.1}$$

where $\underline{\underline{M}} \in \mathbb{R}^{N\times N}$ is a diagonalizable square matrix with non-negative eigenvalues, $\eta \in (0,1]$, $\underline{f}^{\dagger}(x)$ is a known function—the so-called *forcing function*, $\underline{u}^{\dagger}(x)$ is subject to the initial condition $\underline{u}^{\dagger}(x<0) = \underline{0}^{\dagger}$, and $\underline{u}_0^{\dagger}$ is constant vector enconding the initial value of $\underline{u}^{\dagger}(x=0^{+}) = \underline{u}_0^{\dagger}$. Note that we are not assuming $\underline{\underline{M}}$ is positive semi-definite, as we will need the asymmetric case.

This can be solved for $\underline{u}^{\dagger}(x \geq 0)$ with the Green's function of the linear operator. Noting that for any x-independent vector $\underline{c}$,

$$\underline{c}^{\dagger}e^{-\frac{1}{\eta}\underline{\underline{M}}x}\Theta(x)\left[\overleftarrow{\partial}_x + \frac{1}{\eta}\underline{\underline{M}}\right] = \underline{c}^{\dagger}\delta(x) \tag{B.2}$$

and taking advantage of the linearity of the operator $\overleftarrow{\partial}_x + \frac{1}{\eta}\underline{\underline{M}}$, the formal solution to equation

(B.1) is a convolution of the forcing term with the Green's function:

$$\underline{u}^\dagger(x) = \underline{u}_0^\dagger e^{-\frac{1}{\eta}\underline{\underline{M}}x} + \underbrace{\int_0^\infty dx' \underline{f}^\dagger(x') e^{-\frac{1}{\eta}\underline{\underline{M}}(x-x')}\Theta(x-x')}_{\equiv \mathscr{I}}. \tag{B.3}$$

We will henceforth refer to the integral in equation (B.3) as $\mathscr{I}$.

Further analysis of this integral is facilitated by examining the eigendecomposition of $\underline{\underline{M}}$ and splitting $\underline{f}^\dagger(x)$ into two parts accordingly. Write $\underline{\underline{M}}$ as:

$$\underline{\underline{M}} = \sum_{n=1}^{N} \lambda_n \underline{w}_n^R \underline{w}_n^{L\dagger} \tag{B.4}$$

where requiring $\underline{w}_n^{L\dagger} \underline{w}_m^R = \delta_{nm}$ implies that $\underline{\underline{M}}\underline{x}_n^R = \lambda_n \underline{x}_n^R$ and $\underline{x}_n^{L\dagger}\underline{\underline{M}} = \lambda_n \underline{x}_n^{L\dagger}$. Eigenvalues may be repeated—or not—without affecting this construction. This implies the matrix exponential in the Green's function may be written as:

$$e^{-\frac{1}{\eta}\underline{\underline{M}}x} = \sum_{n=1}^{N} e^{-\frac{1}{\eta}\lambda_n x} \underline{w}_n^R \underline{w}_n^{L\dagger} = \sum_{\substack{n\in\{1,\cdots,N\}\\ \lambda_n=0}} \underline{w}_n^R \underline{w}_n^{L\dagger} + \sum_{\substack{n\in\{1,\cdots,N\}\\ \lambda_n>0}} e^{-\frac{1}{\eta}\lambda_n x} \underline{w}_n^R \underline{w}_n^{L\dagger}. \tag{B.5}$$

The fact that $\underline{w}_n^{L\dagger} \underline{w}_m^R = \delta_{nm}$ implies that the first term in equation (B.5) is a projector:

$$\underline{\underline{P}}_0 \equiv \sum_{\substack{n\in\{1,\cdots,N\}\\ \lambda_n=0}} \underline{w}_n^R \underline{w}_n^{L\dagger} \implies \underline{\underline{P}}_0^2 = \underline{\underline{P}}_0 \tag{B.6}$$

Using the identities $\underline{\underline{\mathbb{I}}} = \underline{\underline{P}}_0 + \left(\underline{\underline{\mathbb{I}}} - \underline{\underline{P}}_0\right)$ and $\underline{\underline{P}}_0\left(\underline{\underline{\mathbb{I}}} - \underline{\underline{P}}_0\right) = \underline{\underline{0}}$, and noting that the second sum in equation (B.5) is annihilated by $\underline{\underline{P}}_0$:

$$\underline{\underline{P}}_0 \left(\sum_{\substack{n\in\{1,\cdots,N\}\\ \lambda_n>0}} e^{-\frac{1}{\eta}\lambda_n x} \underline{w}_n^R \underline{w}_n^{L\dagger} \right) = \underline{\underline{0}}, \tag{B.7}$$

we can split the integrand of equation (B.3) into two parts according to the nature of the action of the matrix exponential:

$$\begin{aligned}\underline{f}^\dagger(x')e^{-\frac{1}{\eta}\underline{\underline{M}}(x-x')} &= \underline{f}^\dagger(x')\left(\underline{\underline{P}}_0 + \left(\underline{\underline{\mathbb{I}}} - \underline{\underline{P}}_0\right)\right)e^{-\frac{1}{\eta}\underline{\underline{M}}(x-x')} \\ &= \underbrace{\underline{f}^\dagger(x')\underline{\underline{P}}_0}_{\equiv \underline{f}^\dagger_{\lambda=0}(x')} + \underbrace{\underline{f}^\dagger(x')\left(\underline{\underline{\mathbb{I}}} - \underline{\underline{P}}_0\right)}_{\equiv \underline{f}^\dagger_{\lambda>0}(x')} e^{-\frac{1}{\eta}\underline{\underline{M}}(x-x')} \\ &= \underline{f}^\dagger_{\lambda=0}(x') + \underline{f}_{\lambda>0}(x')^\dagger e^{-\frac{1}{\eta}\underline{\underline{M}}(x-x')}\end{aligned} \tag{B.8}$$

where we are guaranteed that $\underline{f}^\dagger_{\lambda>0}(x')e^{-\frac{1}{\eta}\underline{\underline{M}}(x-x')} \xrightarrow[x\to\infty]{} \underline{0}^\dagger$. Thus:

$$\begin{aligned}\mathscr{I} &= \int_{0^-}^{\infty} dx' \underline{f}^\dagger(x')e^{-\frac{1}{\eta}\underline{\underline{M}}(x-x')}\Theta(x-x') \\ &= \underbrace{\int_0^x dx' \underline{f}^\dagger_{\lambda=0}(x')}_{\equiv \mathscr{I}_{\lambda=0}} + \underbrace{\int_0^\infty dx' \underline{f}^\dagger_{\lambda>0}(x')e^{-\frac{1}{\eta}\underline{\underline{M}}(x-x')}\Theta(x-x')}_{\equiv \mathscr{I}_{\lambda>0}}.\end{aligned} \tag{B.9}$$

Our attention will henceforth be focused on the more interesting of these two terms, $\mathscr{I}_{\lambda>0}$.

Analytic forcing function

For small η and suitable restrictions on the growth rate of the magnitude of $\underline{f}^\dagger_{\lambda>0}(x')$, the value of $\mathscr{I}_{\lambda>0}(x)$ will be dominated by the value of the integrand where $x' \approx x$. Under the assumption that $\underline{f}^\dagger_{\lambda>0}(x')$ is analytic in x', we can Taylor expand it about $x' = x$ to capture this behavior:

$$\underline{f}^\dagger_{\lambda>0}(x') = \sum_{r=0}^{\infty} \frac{1}{r!}\left(\partial_x{}^r \underline{f}^\dagger_{\lambda>0}(x)\right)(x'-x)^r. \tag{B.10}$$

Inserting this into the integral for $\mathscr{I}_{\lambda>0}$, using the fact that $\underline{f}^\dagger_{\lambda>0} = \underline{f}^\dagger_{\lambda>0}\left(\underline{\underline{\mathbb{I}}} - \underline{\underline{P}}_0\right)$, and interchanging the sum and integral yields:

$$\mathscr{I}_{\lambda>0}(x) = \sum_{r=0}^{\infty} \frac{1}{r!}\left(\partial_x{}^r \underline{f}^\dagger_{\lambda>0}(x)\right)\int_{0^-}^{\infty} dx'(x'-x)^r\left(\underline{\underline{\mathbb{I}}} - \underline{\underline{P}}_0\right)e^{-\frac{1}{\eta}\underline{\underline{M}}(x-x')}\Theta(x-x'). \tag{B.11}$$

The remaining integral can be evaluated exactly by using the eigen-decomposition of the argument of the integral:

$$\int_{0^-}^{\infty} dx'(x'-x)^r(\underline{\underline{\mathbb{I}}}-\underline{\underline{P}}_0)e^{\frac{1}{\eta}\underline{\underline{M}}(x-x')}\Theta(x-x') \tag{B.12}$$

$$= \int_{0^-}^{x} dx'(x'-x)^r \sum_{\substack{n\in\{1,\cdots,N\}\\ \lambda_n>0}} e^{-\frac{1}{\eta}\lambda_n(x-x')}\underline{w}_n^R\underline{w}_n^{L\dagger} \tag{B.13}$$

$$= \sum_{\substack{n\in\{1,\cdots,N\}\\ \lambda_n>0}} \underline{w}_n^R\underline{w}_n^{L\dagger}\int_{0^-}^{x} dx'(x'-x)^r e^{-\frac{1}{\eta}\lambda_n(x-x')} \tag{B.14}$$

$$= (-1)^r(r!)\eta^{r+1}\sum_{\substack{n\in\{1,\cdots,N\}\\ \lambda_n>0}} \underline{w}_n^R\underline{w}_n^{L\dagger}\left(\frac{1}{\lambda_n}\right)^{r+1}\left[1-\left(\sum_{k=0}^{r}\frac{1}{k!}\left(\frac{\lambda_n}{\eta}x\right)^k\right)e^{-\frac{\lambda_n}{\eta}x}\right]. \tag{B.15}$$

Defining the rth order truncation of the power series of the exponential function as:

$$\exp_{\lceil r\rceil}(z) \equiv e^{z}_{\lceil r\rceil} \equiv \sum_{k=0}^{r}\frac{z^k}{k!}, \tag{B.16}$$

we can write this cleanly as:

$$\int_{0^-}^{\infty} dx'(x'-x)^r(\underline{\underline{\mathbb{I}}}-\underline{\underline{P}}_0)e^{\frac{1}{\eta}\underline{\underline{M}}(x-x')}\Theta(x-x')$$
$$= (-1)^r(r!)\eta^{r+1}\left(\underline{\underline{M}}^+\right)^{r+1}\left[\underline{\underline{\mathbb{I}}}-e_{\lceil r\rceil}^{+\frac{1}{\eta}\underline{\underline{M}}x}e^{-\frac{1}{\eta}\underline{\underline{M}}x}\right] \tag{B.17}$$

where $\underline{\underline{M}}^+$ is the Moore-Penrose inverse, which eliminates the need to include an explicit factor of $\left(\underline{\underline{\mathbb{I}}}-\underline{\underline{P}}_0\right)$. This notation makes it clear that for small x/η, the term in square brackets goes to $\underline{\underline{0}}$ due to the two exponentials multiplying to approximately $\underline{\underline{\mathbb{I}}}$ since the order-r approximation to the exponential is good in this region, while for large x/η, the term in square brackets tends to $\underline{\underline{\mathbb{I}}}$ since $e_{\lceil r\rceil}^{+\frac{1}{\eta}\underline{\underline{M}}x}e^{-\frac{1}{\eta}\underline{\underline{M}}x} \sim \left(\frac{1}{\eta}\underline{\underline{M}}x\right)^r e^{-\frac{1}{\eta}\underline{\underline{M}}x} \xrightarrow[x/\eta\to\infty]{} \underline{\underline{0}}$.

This allows us to eliminate the integral from equation (B.11), yielding:

$$\mathscr{I}_{\lambda>0}(x) = \sum_{r=0}^{\infty} \frac{1}{r!} \left({\partial_x}^r \underline{f}^{\dagger}_{\lambda>0}(x) \right) \int_{0^-}^{\infty} dx' (x'-x)^r \left(\underline{\underline{\mathbb{I}}} - \underline{\underline{P}}_0 \right) e^{-\frac{1}{\eta}\underline{\underline{M}}(x-x')} \Theta(x-x') \tag{B.18}$$

$$= \underline{f}^{\dagger}_{\lambda>0}(x)\, \eta \underline{\underline{M}}^{+} \sum_{r=0}^{\infty} \left(-\eta \overleftarrow{\partial}_x \underline{\underline{M}}^{+} \right)^r \left[\underline{\underline{\mathbb{I}}} - e_{\lceil r \rceil}^{+\frac{1}{\eta}\underline{\underline{M}}x} e^{-\frac{1}{\eta}\underline{\underline{M}}x} \right] \tag{B.19}$$

$$= \underline{f}^{\dagger}_{\lambda>0}(x) \left[\underbrace{\frac{\eta \underline{\underline{M}}^{+}}{\underline{\underline{\mathbb{I}}} + \eta \overleftarrow{\partial}_x \underline{\underline{M}}^{+}}}_{\text{large } x/\eta \text{ limit}} - \underbrace{\eta \underline{\underline{M}}^{+} \sum_{r=0}^{\infty} \left(-\eta \overleftarrow{\partial}_x \underline{\underline{M}}^{+} \right)^r e_{\lceil r \rceil}^{+\frac{1}{\eta}\underline{\underline{M}}x} e^{-\frac{1}{\eta}\underline{\underline{M}}x}}_{\text{small } x/\eta \text{ correction}} \right]. \tag{B.20}$$

The "small x/η correction" term can be dramatically simplified by grouping terms by powers of η:

$$\sum_{r=0}^{\infty} \left(-\eta \overleftarrow{\partial}_x \underline{\underline{M}}^{+} \right)^r e_{\lceil r \rceil}^{+\frac{1}{\eta}\underline{\underline{M}}x} \tag{B.21}$$

$$= \sum_{r=0}^{\infty} \left(-\eta \overleftarrow{\partial}_x \underline{\underline{M}}^{+} \right)^r \left(\sum_{k=0}^{r} \frac{1}{k!} \left(\frac{1}{\eta} \underline{\underline{M}} x \right)^k \right) \tag{B.22}$$

$$= \underbrace{\sum_{r=0}^{\infty} \frac{1}{r!} \overleftarrow{\partial}_x^{\,r} (-x)^r}_{k=r} + \left(-\eta \overleftarrow{\partial}_x \underline{\underline{M}}^{+} \right) \underbrace{\sum_{r=1}^{\infty} \frac{1}{(r-1)!} \overleftarrow{\partial}_x^{\,r-1} (-x)^{r-1}}_{k=r-1} + \cdots \tag{B.23}$$

$$= \underbrace{\sum_{m=0}^{\infty} \left(-\eta \overleftarrow{\partial}_x \underline{\underline{M}}^{+} \right)^m}_{= \left[\underline{\underline{\mathbb{I}}} + \eta \overleftarrow{\partial}_x \underline{\underline{M}}^{+} \right]^{-1}} \underbrace{\sum_{r=0}^{\infty} \frac{1}{r!} \overleftarrow{\partial}_x^{\,r} (-x)^r}_{\equiv \overleftarrow{\mathscr{T}}_0} \tag{B.24}$$

$$= \frac{\underline{\underline{\mathbb{I}}}}{\underline{\underline{\mathbb{I}}} + \eta \overleftarrow{\partial}_x \underline{\underline{M}}^{+}} \overleftarrow{\mathscr{T}}_0 \tag{B.25}$$

where $\overleftarrow{\mathscr{T}}_0$ applied to an arbitrary analytic function of $g(x)$ on the right gives the value of that function at $x = 0$, assuming an adequate radius of convergence for the Taylor series of $g(x)$ centered at x:

$$g(x) \overleftarrow{\mathscr{T}}_0 = \sum_{m=0}^{\infty} \frac{1}{r!} \left(\partial_x^m g(x) \right) (-x)^m = g(0). \tag{B.26}$$

So the “small x/η correction” term applied to $\underline{f}^{\dagger}_{\lambda>0}$ gives:

$$\begin{aligned}
\underline{f}^{\dagger}_{\lambda>0}(x)\,\eta\underline{\underline{M}}^{+}\sum_{r=0}^{\infty}\left(-\eta\overleftarrow{\partial}_x\underline{\underline{M}}^{+}\right)^{r} e_{\lceil r\rceil}^{+\frac{1}{\eta}\underline{\underline{M}}x}\, e^{-\frac{1}{\eta}\underline{\underline{M}}x}& \\
&= \underline{f}^{\dagger}_{\lambda>0}(x)\frac{\eta\underline{\underline{M}}^{+}}{\underline{\underline{\mathbb{I}}}+\eta\overleftarrow{\partial}_x\underline{\underline{M}}^{+}}\overleftarrow{\mathscr{T}}_0\, e^{-\frac{1}{\eta}\underline{\underline{M}}x} \\
&= \left.\left(\underline{f}^{\dagger}_{\lambda>0}(x)\frac{\eta\underline{\underline{M}}^{+}}{\underline{\underline{\mathbb{I}}}+\eta\overleftarrow{\partial}_x\underline{\underline{M}}^{+}}\right)\right|_{x=0} e^{-\frac{1}{\eta}\underline{\underline{M}}x}.
\end{aligned} \tag{B.27}$$

With this result in hand, the value of $\mathscr{I}_{\lambda>0}$ can be written cleanly as:

$$\mathscr{I}_{\lambda>0} = \underline{f}^{\dagger}_{\lambda>0}(x)\frac{\eta\underline{\underline{M}}^{+}}{\underline{\underline{\mathbb{I}}}+\eta\overleftarrow{\partial}_x\underline{\underline{M}}^{+}} - \left.\left(\underline{f}^{\dagger}_{\lambda>0}(x)\frac{\eta\underline{\underline{M}}^{+}}{\underline{\underline{\mathbb{I}}}+\eta\overleftarrow{\partial}_x\underline{\underline{M}}^{+}}\right)\right|_{x=0} e^{-\frac{1}{\eta}\underline{\underline{M}}x} \tag{B.28}$$

and the overall solution for $\underline{u}^{\dagger}(x)$ is:

$$\begin{aligned}
\underline{u}^{\dagger}(x) = {}& \underbrace{\underline{u}^{\text{i.c.}\,\dagger}_{\lambda=0} + \int_0^x dx'\,\underline{f}^{\dagger}_{\lambda=0}(x') + \underline{f}^{\dagger}_{\lambda>0}(x)\frac{\eta\underline{\underline{M}}^{+}}{\underline{\underline{\mathbb{I}}}+\eta\overleftarrow{\partial}_x\underline{\underline{M}}^{+}}}_{\text{long-time behavior}} \\
&+ \underbrace{\left[\underline{u}^{\text{i.c.}\,\dagger}_{\lambda>0} - \left.\left(\underline{f}^{\dagger}_{\lambda>0}(x)\frac{\eta\underline{\underline{M}}^{+}}{\underline{\underline{\mathbb{I}}}+\eta\overleftarrow{\partial}_x\underline{\underline{M}}^{+}}\right)\right|_{x=0}\right] e^{-\frac{1}{\eta}\underline{\underline{M}}x}}_{\text{initial transient}}
\end{aligned} \tag{B.29}$$

B.2 Graph Laplacians

For any undirected weighted graph $G=(V,E,w)$, a random-walk-normalized transition matrix can be formed from a symmetric weight matrix $\underline{\underline{W}}\in\mathbb{R}^{|V|\times|V|}$, $\underline{\underline{W}}^{\dagger}=\underline{\underline{W}}$, $W_{ij}\geq 0$ with degrees $d_i=\sum_{j=1}^{|V|}W_{ij}>0$ as:

$$\underline{\underline{P}}_{\text{r.w.}} \equiv: \bcancel{\underline{\underline{d}}}^{-1}\underline{\underline{W}}. \tag{B.30}$$

$\underline{\underline{P}}_{\text{r.w.}}$ is guaranteed to be diagonalizble with real eigenvalues as it is similar to the real symmetric transition matrix $\underline{\underline{P}}_{\text{sym}}\equiv\bcancel{\underline{\underline{d}}}^{\frac{1}{2}}\underline{\underline{P}}_{\text{r.w.}}\bcancel{\underline{\underline{d}}}^{-\frac{1}{2}}$. As similar matrices, the eigenvalues of $\underline{\underline{P}}_{\text{r.w.}}$ and $\underline{\underline{P}}_{\text{sym}}$ are identical and lie in the interval $[-1,+1]\in\mathbb{R}$ due to a simple application of the Gershgorin circle theorem and exploiting the fact that $[\underline{\underline{P}}_{\text{r.w.}}]_{ij} \geq 0$ and $\sum_{j=1}^{|V|} [\underline{\underline{P}}_{\text{r.w.}}]_{ij} = 1$:

$$\lambda_n^{P_{\text{r.w.}}} \in \bigcup_{i=1}^{|V|} [P_{ii} - (1 - P_{ii}), P_{ii} + (1 - P_{ii})] = \bigcup_{i=1}^{|V|} [2P_{ii} - 1, 1] \in [-1, 1] \tag{B.31}$$

Furthermore, if $P_{ii} > 0 \; \forall \; i \in \{1, \cdots, |V|\}$ and the undirected graph G is connected, there is a single non-degenerate eigenvalue satisfying $|\lambda| = 1$ corresponding to the equilibrium left- and right-eigenvectors:

$$\underline{\underline{P}}_{\text{r.w.}} \underline{\mathbb{1}} = \underline{\mathbb{1}} \qquad \underline{\pi}^\dagger \underline{\underline{P}}_{\text{r.w.}} = \underline{\pi}^\dagger \qquad \pi_i \equiv \frac{d_i}{\sum_{j=1}^{|V|} d_j} \qquad \lim_{q \to +\infty} \left(\underline{\underline{P}}_{\text{r.w.}}\right)^q = \underline{\mathbb{1}}\, \underline{\pi}^\dagger \tag{B.32}$$

The Laplacian matrix can be defined from the transition matrix as:

$$\underline{\underline{L}}_{\text{r.w.}} \equiv \underline{\underline{\mathbb{I}}} - \underline{\underline{P}}_{\text{r.w.}} = \underline{\underline{\mathbb{I}}} - \underline{\underline{d}}^{-1} \underline{\underline{W}}. \tag{B.33}$$

By construction, $\underline{\underline{L}}_{\text{r.w.}}$ has identical left- and right-eigenvectors as $\underline{\underline{P}}_{\text{r.w.}}$ with shifted eigenvalues:

$$\left.\begin{array}{ll} \underline{\underline{P}}_{\text{r.w.}} \underline{x}_n^R = \lambda_n^P \underline{x}_n^R & \underline{\underline{L}}_{\text{r.w.}} \underline{x}_n^R = (1 - \lambda_n^P) \underline{x}_n^R \\ \left(\underline{x}_n^L\right)^\dagger \underline{\underline{P}}_{\text{r.w.}} = \left(\underline{x}_n^L\right)^\dagger \lambda_n^P & \left(\underline{x}_n^L\right)^\dagger \underline{\underline{L}}_{\text{r.w.}} = \left(\underline{x}_n^L\right)^\dagger (1 - \lambda_n^P) \end{array}\right. \implies \quad \lambda_n^L = 1 - \lambda_n^P \tag{B.34}$$

Since $\underline{\underline{P}}_{\text{r.w.}}$ has a complete basis of left- and right-eigenvectors, so does $\underline{\underline{L}}_{\text{r.w.}}$. The diagonalizability of $\underline{\underline{L}}_{\text{r.w.}}$ is also guaranteed by a similarity relation to the so-called *symmetric Laplacian*:

$$\underline{\underline{L}}_{\text{sym}} \equiv \underline{\underline{d}}^{+\frac{1}{2}} \underline{\underline{L}}_{\text{r.w.}} \underline{\underline{d}}^{-\frac{1}{2}} = \underline{\underline{\mathbb{I}}} - \underline{\underline{d}}^{-\frac{1}{2}} \underline{\underline{W}}\, \underline{\underline{d}}^{-\frac{1}{2}}. \tag{B.35}$$

A well-known theorem of spectral graph theory guarantees that the multiplicity of 0 as an eigenvalue of both $\underline{\underline{L}}_{\text{r.w.}}$ and $\underline{\underline{L}}_{\text{sym}}$ is equal to the number of connected components in the undirected graph G with weight matrix $\underline{\underline{W}}$ (Chung 1997).

Lossy Laplacians

Define a random-walk-normalized lossy Laplacian as:

$$\widetilde{\underline{\underline{L}}}_{\text{r.w.}} \equiv \underline{\underline{L}}_{\text{r.w.}} + \underline{\underline{\rho}}\,\underline{\underline{P}}_{\text{r.w.}} = (\underline{\underline{\mathbb{I}}} - \underline{\underline{P}}_{\text{r.w.}}) + \underline{\underline{\rho}}\,\underline{\underline{P}}_{\text{r.w.}} = \underline{\underline{\mathbb{I}}} - (\underline{\underline{\mathbb{I}}} - \underline{\underline{\rho}})\underline{\underline{P}}_{\text{r.w.}} = \underline{\underline{\mathbb{I}}} - (\underline{\underline{\mathbb{I}}} - \underline{\underline{\rho}})\underline{\underline{d}}^{-1}\underline{\underline{W}} \tag{B.36}$$

where $\underline{\underline{\rho}} = \operatorname{diag}\left(\underline{\rho}\right)$, and $\rho_i \in [0,1)$. In the case where $\underline{\rho} = \underline{0}$, this reduces to the ordinary Laplacian. However, when $\underline{\rho} \neq \underline{0}$, the properties of $\widetilde{\underline{\underline{L}}}_{\text{r.w.}}$ diverge from those of $\underline{\underline{L}}_{\text{r.w.}}$.

Lemma 2. *Any random-walk-normalized lossy Laplacian $\widetilde{\underline{\underline{L}}}_{\text{r.w.}}$ is similar to a real symmetric matrix, and is thus diagonalizable with real eigenvalues. Furthermore, its eigenvalues are non-negative.*

Proof.

A similarity transform with $\underline{\underline{S}} = (\underline{\underline{\mathbb{I}}} - \underline{\underline{\rho}})\underline{\underline{d}}^{+\frac{1}{2}}$ reveals:

$$\widetilde{\underline{\underline{L}}}_{\text{sym}} \equiv \underbrace{(\underline{\underline{\mathbb{I}}} - \underline{\underline{\rho}})^{-\frac{1}{2}}\underline{\underline{d}}^{+\frac{1}{2}}}_{\underline{\underline{S}}}\,\widetilde{\underline{\underline{L}}}_{\text{r.w.}}\,\underbrace{\underline{\underline{d}}^{-\frac{1}{2}}(\underline{\underline{\mathbb{I}}} - \underline{\underline{\rho}})^{+\frac{1}{2}}}_{\underline{\underline{S}}^{-1}} = (\underline{\underline{\mathbb{I}}} - \underline{\underline{\rho}})^{+\frac{1}{2}}\underline{\underline{L}}_{\text{sym}}(\underline{\underline{\mathbb{I}}} - \underline{\underline{\rho}})^{+\frac{1}{2}} + \underline{\underline{\rho}}. \tag{B.37}$$

Since $\widetilde{\underline{\underline{L}}}_{\text{sym}}$ is the sum of two real matrices, each manifestly symmetric, $\widetilde{\underline{\underline{L}}}_{\text{sym}}$ is itself a real symmetric matrix. Thus $\widetilde{\underline{\underline{L}}}_{\text{sym}}$ and (by similarity) $\widetilde{\underline{\underline{L}}}$ are both diagonalizable with real eigenvalues. Moreover, the eigenvalues of $\widetilde{\underline{\underline{L}}}$ are guaranteed to lie in the interval $[0,2]$ by the Gershgorin circle theorem:

$$\lambda_n\left(\widetilde{\underline{\underline{L}}}\right) \in \bigcup_{i=1}^{|V|} [\rho_i,\ 1-(1-\rho_i)(2P_{ii}-1)] \in [0,2] \tag{B.38}$$

$\square$

This Gershgorin circle theorem result guarantees that $\lambda_n > \min\left(\underline{\rho}\right)\ \forall\, n \in \{1,\cdots,|V|\}$. If $\min\left(\underline{\rho}\right) > 0$, this ensures the eigenvalues of $\widetilde{\underline{\underline{L}}}_{\text{r.w.}}$ are strictly positive, which implies that $\lim_{x\to+\infty} e^{-\widetilde{\underline{\underline{L}}}_{\text{r.w.}}x} = \underline{\underline{0}}$, among other useful corollaries. We will now establish generalizations of this result showing that, given suitable conditions on $\underline{\rho}$, all eigenvalues of $\underline{\underline{\widetilde{L}}}_{\text{r.w.}}$ are bounded away from 0 even if $\underline{\rho}$ has some zero entries.

Theorem 2. *Any connected undirected weighted graph $G = (V, E, w)$ with weight matrix $\underline{\underline{W}} \in \mathbb{R}^{|V|\times|V|}$, $W_{ij} \geq 0$, degree vector $\underline{d} \in \mathbb{R}^{|V|}$, $d_i = \sum_{j=1}^{|V|} W_{ij} > 0$, and loss vector $\underline{\rho} \in \mathbb{R}^{|V|}$, $\rho_i \in [0,1)$ has an associated random-walk-normalized lossy Laplacian matrix $\underline{\underline{\widetilde{L}}}_{\text{r.w.}} \equiv \underline{\underline{\mathbb{I}}} - (\underline{\underline{\mathbb{I}}} - \underline{\underline{\rho}})\underline{\underline{P}}_{\text{r.w.}}$ (where $\underline{\underline{P}}_{\text{r.w.}} \equiv \underline{\underline{d}}^{-1}\underline{\underline{W}}$) which is diagonalizable, has non-negative eigenvalues, and whose eigenvalues are strictly greater than* 0 *as long as $\underline{\rho} \neq \underline{0}$.*

Proof.

Lemma (2) guarantees the diagonalizability and non-negative eigenvalues of $\underline{\underline{\widetilde{L}}}_{\text{r.w.}}$. To prove all eigenvalues of $\underline{\underline{\widetilde{L}}}_{\text{r.w.}}$ are strictly positive, examine $\underline{\underline{\widetilde{L}}}_{\text{sym}}$ since it shares identical eigenvalues. The Rayleigh quotient is:

$$R(\underline{\underline{\widetilde{L}}}_{\text{sym}}, \underline{v} \neq \underline{0}) \equiv \frac{\underline{v}^\dagger \underline{\underline{\widetilde{L}}}_{\text{sym}} \underline{v}}{\underline{v}^\dagger \underline{v}} = \frac{1}{\underline{v}^\dagger \underline{v}} \left(\underline{v}^\dagger (\underline{\underline{\mathbb{I}}} - \underline{\underline{\rho}})^{+\frac{1}{2}} \underline{\underline{L}}_{\text{sym}} (\underline{\underline{\mathbb{I}}} - \underline{\underline{\rho}})^{+\frac{1}{2}} \underline{v} + \underline{v}^\dagger \underline{\underline{\rho}}\, \underline{v} \right) \tag{B.39}$$

It is well known that the set of eigenvalues of a self-adjoint matrix is contained in the range of R, so proving $R > 0$ implies $\lambda_n > 0\ \forall\, n$. Since both $(\underline{\underline{\mathbb{I}}} - \underline{\underline{\rho}})^{+\frac{1}{2}} \underline{\underline{L}}_{\text{sym}} (\underline{\underline{\mathbb{I}}} - \underline{\underline{\rho}})^{+\frac{1}{2}}$ and $\underline{\underline{\rho}}$ are positive semi-definite, we must only show that they cannot simultaneously be zero.

Because $\underline{\underline{L}}_{\text{sym}}$ is the symmetric Laplacian of a connected graph, it has a single non-degenerate zero eigenvalue. A simple computation shows that the corresponding eigenvector must be proportional to $\underline{d}^{\odot+\frac{1}{2}}$:

$$\underline{\underline{L}}_{\text{sym}} \underline{d}^{\odot+\frac{1}{2}} = \underbrace{\underline{\underline{d}}^{+\frac{1}{2}} \underline{\underline{L}}_{\text{r.w.}} \underline{\underline{d}}^{-\frac{1}{2}}}_{=\underline{\underline{L}}_{\text{sym}}} \underbrace{\underline{\underline{d}}^{+\frac{1}{2}} \underline{\mathbb{1}}}_{=\underline{d}^{\odot+\frac{1}{2}}} = \underline{\underline{d}}^{+\frac{1}{2}} \underline{\underline{L}}_{\text{r.w.}} \underline{\mathbb{1}} = \underline{0} \tag{B.40}$$

where the last equality follows from the fact that the row sums of $\underline{\underline{L}}$ are zero by construction of the random-walk-normalized Laplacian. Since all other eigenvalues of $\underline{\underline{L}}_{\text{sym}}$ are strictly greater than 0, we know that

$$\underline{v}^{\dagger}(\underline{\underline{\mathbb{I}}}-\underline{\underline{\rho}})^{+\frac{1}{2}}\underline{\underline{L}}_{\text{sym}}(\underline{\underline{\mathbb{I}}}-\underline{\underline{\rho}})^{+\frac{1}{2}}\underline{v}=0 \implies \underline{v} \propto (\underline{\underline{\mathbb{I}}}-\underline{\underline{\rho}})^{-\frac{1}{2}}\underline{d}^{\odot+\frac{1}{2}}. \tag{B.41}$$

But since we require $d_n > 0\,\forall\, n \in \{1,\cdots,|V|\}$, this gives a non-zero contribution from $\underline{v}^{\dagger}\underline{\underline{\rho}}\,\underline{v}$ as long as $\underline{\rho} \neq \underline{0}$ since every element of $(\underline{\underline{\mathbb{I}}}-\underline{\underline{\rho}})^{-\frac{1}{2}}\underline{d}^{\odot+\frac{1}{2}}$ is strictly positive:

$$(\underline{d})^{\odot+\frac{1}{2}}(\underline{\underline{\mathbb{I}}}-\underline{\underline{\rho}})^{-\frac{1}{2}}\underline{\underline{\rho}}(\underline{\underline{\mathbb{I}}}-\underline{\underline{\rho}})^{-\frac{1}{2}}\underline{d}^{\odot+\frac{1}{2}} = \sum_{n=1}^{|V|}\frac{\rho_n}{1-\rho_n}d_n > 0 \tag{B.42}$$

thus since the Rayleigh quotient is the sum of two non-negative terms, and since at least one of these two terms is always positive, $R(\underline{\underline{\widetilde{L}}}_{\text{sym}},\underline{v}\neq\underline{0})>0 \implies \lambda_n > 0\,\forall\, n\in\{1,\cdots,|V|\}$. □

This can be generalized to the case of a graph made up of multiple connected components.

Theorem 3. *Any undirected weighted graph $G=(V,E,w)$ with weight matrix $\underline{\underline{W}} \in \mathbb{R}^{|V|\times|V|}$, $W_{ij}\geq 0$, degree vector $\underline{d}\in\mathbb{R}^{|V|}$, $d_i=\sum_{j=1}^{|V|}W_{ij}>0$, and loss vector $\underline{\rho}\in\mathbb{R}^{|V|}$, $\rho_i\in[0,1)$ has an associated random-walk-normalized lossy Laplacian matrix $\underline{\underline{\widetilde{L}}}_{\text{r.w.}} \equiv \underline{\underline{\mathbb{I}}}-(\underline{\underline{\mathbb{I}}}-\underline{\underline{\rho}})\underline{\underline{P}}_{\text{r.w.}}$ (where $\underline{\underline{P}}_{\text{r.w.}}\equiv\underline{\underline{d}}^{-1}\underline{\underline{W}}$) satisfying:*

- *$\underline{\underline{\widetilde{L}}}_{\text{r.w.}}$ is diagonalizable with real, non-negative eigenvalues; and*
- *as long as $\underline{\rho}$ has at least one non-zero element on each of the connected components of G, then the eigenvalues of $\underline{\underline{\widetilde{L}}}_{\text{r.w.}}$ are strictly positive.*

Proof.

We can always choose an ordering of the standard basis vectors so that the weight matrix has a block diagonal structure with M blocks, each block corresponding to the weight matrix of one of the M connected components of G. Writing $\underline{\underline{W}}$, $\underline{d}$, and $\underline{\rho}$ in this basis:

$$\underline{\underline{W}} = \begin{bmatrix} \underline{\underline{W}}^{(1)} & & & 0 \\ & \underline{\underline{W}}^{(2)} & & \\ & & \ddots & \\ 0 & & & \underline{\underline{W}}^{(M)} \end{bmatrix} \qquad \underline{d} = \begin{bmatrix} \underline{d}^{(1)} \\ \underline{d}^{(2)} \\ \vdots \\ \underline{d}^{(M)} \end{bmatrix} \qquad \underline{\rho} = \begin{bmatrix} \underline{\rho}^{(1)} \\ \underline{\rho}^{(2)} \\ \vdots \\ \underline{\rho}^{(M)} \end{bmatrix} \tag{B.43}$$

Forming the Lossy Laplacian $\underline{\underline{\widetilde{L}}} \equiv \underline{\underline{\mathbb{I}}} - (\underline{\underline{\mathbb{I}}} - \underline{\underline{\bcancel{\rho}}})\underline{\underline{\bcancel{d}}}^{-1}\underline{\underline{W}}$ preserves this block structure:

$$\underline{\underline{\widetilde{L}}} = \begin{bmatrix} \underline{\underline{\widetilde{L}}}^{(1)} & & & 0 \\ & \underline{\underline{\widetilde{L}}}^{(2)} & & \\ & & \ddots & \\ 0 & & & \underline{\underline{\widetilde{L}}}^{(M)} \end{bmatrix} \qquad \underline{\underline{\widetilde{L}}}^{(m)} \equiv \underline{\underline{\mathbb{I}}}^{(m)} - (\underline{\underline{\mathbb{I}}}^{(m)} - \underline{\underline{\bcancel{\rho}}}^{(m)}) \left(\underline{\underline{\bcancel{d}}}^{(m)}\right)^{-1} \underline{\underline{W}}^{(m)} \tag{B.44}$$

The set of eigenvalues of a block diagonal matrix is the union of the set of eigenvalues of each diagonal block. But each block of $\underline{\underline{\widetilde{L}}}$ is the lossy laplacian of a connected component of G, so theorem (2) guarantees that as long as $\underline{\rho}^{(m)} \neq \underline{0}\ \forall\, m \in \{1, \cdots, M\}$, then the eigenvalues of every block are strictly positive, and thus the eigenvalues of $\underline{\underline{\widetilde{L}}}$ are strictly positive, as well. $\square$

Appendix C

Tensor Networks

C.1 Tensor Origami

When $\mathscr{H} \ni \mathbf{A}$ is composed of a tensor product of more than one Hilbert space, we can unfold or fold $\mathbf{A}$ into a vector or various different matrices as we see fit. To see this, let $U = \{\mathscr{H}_1, \mathscr{H}_2, \ldots \mathscr{H}_N\}$ be the set of all the component Hilbert spaces that together form $\mathscr{H} \equiv \mathscr{H}_U = \bigotimes_{h \in U} h$. Now define a bipartition $U = X \sqcup Y$. This defines a natural split of $\mathscr{H}_U$ into two parts such that $\mathscr{H}_U = \mathscr{H}_X \otimes \mathscr{H}_Y$, where $\mathscr{H}_X$ and $\mathscr{H}_Y$ are defined analogously to $\mathscr{H}_U$ from the sets X, Y. This suggests array representations of $\mathbf{A}$ as both a vector and a matrix, as:

$$\underline{A} = \sum_{\vec{r}} \alpha_{\vec{r}} \hat{v}_{\vec{r}} \quad \longleftrightarrow \quad \underline{\underline{A}} = \sum_{\vec{r}_x} \sum_{\vec{r}_y} \alpha_{(\vec{r}_x, \vec{r}_y)} \hat{x}_{\vec{r}_x} \hat{y}^{\dagger}_{\vec{r}_y} \tag{C.1}$$

where $\vec{r}_x$ and $\vec{r}_y$ are vectorized indices of the subset of the components of $\vec{r}$ corresponding to $\mathscr{H}_X$ and $\mathscr{H}_Y$, $\hat{x}_{\vec{r}_x} \in \mathscr{H}_X$, and $\hat{y}_{\vec{r}_y} \in \mathscr{H}_Y$. The single and double underlines on $\underline{A}$ and $\underline{\underline{A}}$ visually differentiate between the vector and matrix forms of $\mathbf{A}$. The same process can be used to join legs on tensors of arbitrary order, or split a leg whenever the corresponding vector space has a tensor product structure that may be exploited.

Bibliography

[1] Enrique Amigó et al. “A comparison of extrinsic clustering evaluation metrics based on formal constraints”. In: *Information retrieval* 12 (2009), pp. 461–486.

[2] Kenneth I Appel and Wolfgang Haken. *Every planar map is four colorable*. Vol. 98. American Mathematical Soc., 1989.

[3] Howard E Bell. “Gershgorin’s theorem and the zeros of polynomials”. In: *The American Mathematical Monthly* 72.3 (1965), pp. 292–295.

[4] Jacob Biamonte and Ville Bergholm. “Tensor networks in a nutshell”. In: *arXiv preprint arXiv:1708.00006* (2017).

[5] Rainer E Burkard and Eranda Cela. “Linear assignment problems and extensions”. In: *Handbook of combinatorial optimization: Supplement volume A*. Springer, 1999, pp. 75–149.

[6] Song Cheng et al. “Tree tensor networks for generative modeling”. In: *Physical Review B* 99.15 (2019), p. 155131.

[7] Xiuyuan Cheng and Gal Mishne. “Spectral Embedding Norm: Looking Deep into the Spectrum of the Graph Laplacian”. In: *SIAM Journal on Imaging Sciences* 13.2 (2020), pp. 1015–1048. DOI: 10.1137/18M1283160. eprint: https://doi.org/10.1137/18M1283160. URL: https://doi.org/10.1137/18M1283160.

[8] Fan RK Chung. *Spectral graph theory*. Vol. 92. American Mathematical Soc., 1997.

[9] A. Cloninger and H.N. Mhaskar. “Cautious active clustering”. In: *Applied and Computational Harmonic Analysis* 54 (2021), pp. 44–74. ISSN: 1063-5203. DOI: https://doi.org/10.1016/j.acha.2021.02.002. URL: https://www.sciencedirect.com/science/article/pii/S1063520321000154.

[10] Alexander Cloninger and Wojciech Czaja. “Eigenvector localization on data-dependent graphs”. In: *2015 International Conference on Sampling Theory and Applications*. 2015, pp. 608–612. DOI: 10.1109/SAMPTA.2015.7148963.

[11] Omar Elharrouss et al. “Image inpainting: A review”. In: *Neural Processing Letters* 51 (2020), pp. 2007–2028.

[12] Glen Evenbly. “A practical guide to the numerical implementation of tensor networks i: Contractions, decompositions, and gauge freedom”. In: *Frontiers in Applied Mathematics and Statistics* 8 (2022), p. 806549.

[13] Glen Evenbly and Guifré Vidal. “Tensor network states and geometry”. In: *Journal of Statistical Physics* 145 (2011), pp. 891–918.

[14] Piotr Formanowicz and Krzysztof Tanaś. “A survey of graph coloring - its types, methods and applications”. In: *Foundations of Computing and Decision Sciences* 37.3 (3912), pp. 223–238. DOI: doi:10.2478/v10209-011-0012-y. URL: https://doi.org/10.2478/v10209-011-0012-y.

[15] David Gfeller and Paolo De Los Rios. “Spectral Coarse Graining of Complex Networks”. In: *Phys. Rev. Lett.* 99 (3 July 2007), p. 038701. DOI: 10.1103/PhysRevLett.99.038701. URL: https://link.aps.org/doi/10.1103/PhysRevLett.99.038701.

[16] Lars Grasedyck, Daniel Kressner, and Christine Tobler. “A literature survey of low-rank tensor approximation techniques”. In: *GAMM-Mitteilungen* 36.1 (2013), pp. 53–78.

[17] Robert O Green et al. “Imaging Spectroscopy and the Airborne Visible/Infrared Imaging Spectrometer (AVIRIS)”. In: *Remote Sensing of Environment* 65.3 (1998), pp. 227–248. ISSN: 0034-4257. DOI: https://doi.org/10.1016/S0034-4257(98)00064-9. URL: https://www.sciencedirect.com/science/article/pii/S0034425798000649.

[18] S. Lafon and A.B. Lee. “Diffusion maps and coarse-graining: a unified framework for dimensionality reduction, graph partitioning, and data set parameterization”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 28.9 (Sept. 2006), pp. 1393–1403. ISSN: 1939-3539. DOI: 10.1109/TPAMI.2006.184.

[19] Yann LeCun et al. “Gradient-based learning applied to document recognition”. In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324.

[20] Richard B Lehoucq, Danny C Sorensen, and Chao Yang. *ARPACK users’ guide: solution of large-scale eigenvalue problems with implicitly restarted Arnoldi methods*. SIAM, 1998.

[21] Jialu Liu and Jiawei Han. “Spectral clustering”. In: *Data clustering*. Chapman and Hall/CRC, 2018, pp. 177–200.

[22] Boaz Nadler and Meirav Galun. “Fundamental Limitations of Spectral Clustering”. In: *Advances in Neural Information Processing Systems*. Ed. by B. Schölkopf, J. Platt, and T. Hoffman. Vol. 19. MIT Press, 2006. URL: https://proceedings.neurips.cc/paper_files/paper/2006/file/bdb6920adcd0457aa17b53b22963dad9-Paper.pdf.

[23] Roberto I. Oliveira and Yuval Peres. “Random walks on graphs: new bounds on hitting, meeting, coalescing and returning”. In: *2019 Proceedings of the Meeting on Analytic Algorithmics and Combinatorics (ANALCO)*, pp. 119–126. DOI: 10.1137/1.9781611975505.13. eprint: https://epubs.siam.org/doi/pdf/10.1137/1.9781611975505.13. URL: https://epubs.siam.org/doi/abs/10.1137/1.9781611975505.13.

[24] Román Orús. “A practical introduction to tensor networks: Matrix product states and projected entangled pair states”. In: *Annals of physics* 349 (2014), pp. 117–158.

[25] Román Orús. “Tensor networks for complex quantum systems”. In: *Nature Reviews Physics* 1.9 (2019), pp. 538–550.

[26] Yuchen Pang et al. “Efficient 2D Tensor Network Simulation of Quantum Systems”. In: *SC20: International Conference for High Performance Computing, Networking, Storage and Analysis*. Nov. 2020, pp. 1–14. DOI: 10.1109/SC41405.2020.00018.

[27] David J Pearce. “An improved algorithm for finding the strongly connected components of a directed graph”. In: *Victoria University, Wellington, NZ, Tech. Rep* (2005).

[28] Robert Piessens et al. *Quadpack: a subroutine package for automatic integration*. Vol. 1. Springer Science & Business Media, 2012.

[29] Andrew Rosenberg and Julia Hirschberg. “V-measure: A conditional entropy-based external cluster evaluation measure”. In: *Proceedings of the 2007 joint conference on empirical methods in natural language processing and computational natural language learning (EMNLP-CoNLL)*. 2007, pp. 410–420.

[30] Naoki Saito. “How Can We Naturally Order and Organize Graph Laplacian Eigenvectors?” In: *2018 IEEE Statistical Signal Processing Workshop (SSP)*. June 2018, pp. 483–487. DOI: 10.1109/SSP.2018.8450808.

[31] Geoffrey Schiebinger, Martin J. Wainwright, and Bin Yu. “The geometry of kernelized spectral clustering”. In: *The Annals of Statistics* 43.2 (2015), pp. 819–846. DOI: 10.1214/14-AOS1283. URL: https://doi.org/10.1214/14-AOS1283.

[32] Burr Settles. “From theories to queries: Active learning in practice”. In: *Active learning and experimental design workshop in conjunction with AISTATS 2010*. JMLR Workshop and Conference Proceedings. 2011, pp. 1–18.

[33] Jianbo Shi and J. Malik. “Normalized cuts and image segmentation”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 22.8 (2000), pp. 888–905. DOI: 10.1109/34.868688.

[34] Mikołaj Sitarz. *Extending F1 Metric, Probabilistic Approach. Advances in Artificial Intelligence and Machine Learning. 2023; 3 (2): 61*. 2023.

[35] Edwin Stoudenmire and David J Schwab. “Supervised learning with tensor networks”. In: *Advances in neural information processing systems* 29 (2016).

[36] R.Endre Tarjan. “A note on finding the bridges of a graph”. In: *Information Processing Letters* 2.6 (1974), pp. 160–161. ISSN: 0020-0190. DOI: https://doi.org/10.1016/0020-0190(74)90003-9. URL: https://www.sciencedirect.com/science/article/pii/0020019074900039.

[37] Tom Vieijra, Laurens Vanderstraeten, and Frank Verstraete. “Generative modeling with projected entangled-pair states”. In: *arXiv preprint arXiv:2202.08177* (2022).

[38] Pablo Villegas et al. “Laplacian renormalization group for heterogeneous networks”. In: *Nature Physics* 19.3 (2023), pp. 445–450.

[39] Pauli Virtanen et al. “SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python”. In: *Nature Methods* 17 (2020), pp. 261–272. DOI: 10.1038/s41592-019-0686-2.

[40] Bo Wang, Zhuowen Tu, and John K. Tsotsos. “Dynamic Label Propagation for Semi-supervised Multi-class Multi-label Classification”. In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. Dec. 2013.

[41] Xiaojin Zhu. *Semi-supervised learning with graphs*. Carnegie Mellon University, 2005.