January 2016

Physics based supervised and unsupervised learning of graph structure

Ayan Sinha
Purdue University

Graduate School Form
30 Updated 12/26/2015

PURDUE UNIVERSITY
GRADUATE SCHOOL
Thesis/Dissertation Acceptance

This is to certify that the thesis/dissertation prepared

By Ayan Tuhinendu Sinha

Entitled

PHYSICS BASED SUPERVISED AND UNSUPERVISED LEARNING OF GRAPH STRUCTURE

For the degree of Doctor of Philosophy

Is approved by the final examining committee:

Karthik Ramani
Chair

David Gleich

Niklas Elmqvist

Jitesh Panchal

To the best of my knowledge and as understood by the student in the Thesis/Dissertation Agreement, Publication Delay, and Certification Disclaimer (Graduate School Form 32), this thesis/dissertation adheres to the provisions of Purdue University's "Policy of Integrity in Research" and the use of copyright material.

Approved by Major Professor(s): Karthik Ramani

Approved by: Jay P. Gore 7/16/2016

Head of the Departmental Graduate Program Date

PHYSICS BASED SUPERVISED AND UNSUPERVISED LEARNING OF GRAPH STRUCTURE

A Dissertation

Submitted to the Faculty

of

Purdue University

by

Ayan T. Sinha

In Partial Fulfillment of the

Requirements for the Degree

of

Doctor of Philosophy

August 2016

Purdue University

West Lafayette, Indiana

Dedicated to my family and friends

ACKNOWLEDGMENTS

I have had the privilege of learning and working with outstanding educators through my journey as a graduate student at Purdue University. I want to thank everyone who has made helped me realize my research ideas. First of all, I thank my advisor Dr. Karthik Ramani for his support and guidance beyond the realms of academia, and taking an active interest towards a complete education. He pushed me to perform to the best of my abilities and showed me the big picture. His drive and ambition to produce high impact research helped me realize the qualities of an exemplary researcher. I gained a lot from his insights into academia, and his talks have inspired me to enter academia. I thank Dr. David Gleich for being an exemplary mentor and a good friend. He helped me gain clarity in thought while doing research, and refined my ability to communicate my research ideas. I would like to to thank Dr. Jitesh Panchal for his mentorship and support from the time I was doing my masters at Georgia Tech to now. I thank my committee member Dr.Niklas Elmqvist for providing intellectual guidance based on his expertise. I acknowledge Dr. Mark Newman of University of Michigan for insightful discussions which inspired some ideas for my research. I am especially grateful to Mrs. Michelle Sarault for handling all of my conference paperwork and Mrs. Cathy Elwell for handling my academic paperwork. Furthermore, I thank all friends and students in the C Design Lab for their support in my initiatives. I owe the greatest thanks to my brother for his continuous support and encouragement during the course of my studies. He has been a mentor in rough times and the best role model. I also am grateful to my mother and father for being the support pillars throughout my education, and giving me the freedom to pursue my dreams. Furthermore, I acknowledge the funding sources, which made this PhD possible.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

Sinha, Ayan T. Ph.D., Purdue University, August 2016. Physics Based Supervised and Unsupervised Learning of Graph Structure. Major Professor: Karthik Ramani, School of Mechanical Engineering.

Graphs are central tools to aid our understanding of biological, physical, and social systems. Graphs also play a key role in representing and understanding the visual world around us, 3D-shapes and 2D-images alike. In this dissertation, I propose the use of physical or natural phenomenon to understand graph structure. I investigate four phenomenon or laws in nature: (1) Brownian motion, (2) Gauss's law, (3) feedback loops, and (3) neural synapses, to discover patterns in graphs.

Random walks is the mathematical formalization of Brownian motion on graphs. I discuss the connection between the Laplace-Beltrami operator governing the diffusion equation and the transition matrix of a random walk. This connection is used to construct multiscale shape signatures with several desirable properties, and applied to shape matching, segmentation and search.

Next, I establish a connection between the Laplacian of a general graph and the flux of a potential field over a boundary. This insight is exactly an extension of Gauss law for charge distributions applied to discrete graphs. I design a novel method for data clustering using this law in conjunction with laws of conservation and spectral graph theory. My method to identify community boundaries is significantly more robust to noise than traditional detection of strongly knit modules.

Feedback is pervasive in natural as well as artificial systems. I design a new method to separate feedback effects in recommender systems such as Netflix from intrinsic preferences of users. This method is based on a simple deconvolution of an iterative feedback loop common in physical systems that admit feedback. I am able to identify items rec-

ommended to a user by the recommender system by simply inputting the user-item ratings matrix to my algorithm.

Finally, I study artificial neural networks inspired by neural synapses in the human brain. I develop a technique to create a geometric image from a point cloud, so that convolutional neural networks used pervasively for image understanding can be used for 3D shape understanding. I construct deep neural models which use the parametric knowledge of point clouds from a large database to parameterize unknown and incomplete point clouds. This bridges the gap between traditional geometric processing and data-driven supervised image processing techniques. Furthermore, I demonstrate how pooled activations of neurons in the networks can be used for real-time understanding of point cloud data. I infer the hand skeleton structure from depth images as a prototypical application.

Overall, I show how interesting structures in networks, shapes and images can be inferred using physical laws of nature.

1. INTRODUCTION

I aim to answer the most pressing problems of Big data by combining physical laws and machine learning. Today, there are two pressing issues yet unresolved in Big-data i.e.

1. How to deal with the large amounts of noise in the data set?
2. How to deal with feedback within the processes that generate the data?

I have developed new techniques to address both of these issues that deliver faster and more reliable algorithms to uncover the underlying patterns in the data. The secret sauce in these techniques is inspired by physical laws and processes like diffusion, Gauss law, and biological neural networks. To test the robustness, I have implemented these techniques to a wide variety of domains by representing data points and their pairwise relationships as a graph structure without rebuilding them for each domain. I focus on three types of graph structures: Networks as a graph, shapes or meshes as a graph and finally an image as a graph. In networks a node represents a physical, social or biological object of interest, whereas in shapes, a node is point on the mesh surface. In images the nodes represent pixels and the edges represent adjacent pixel similarity. Often times, one representation may be more valuable than another in a particular task. I introduce these three types of graphs in the context of physical laws used to understand these graph structures, however, note that these representation are interchangeable as they all are graphs.

1.1 Shapes as Graphs

Geometric meshes are simply a graph structure detailing the connection between vertices embedding in a 3-dimensional manifold. I use random walks and convolutional neural networks (CNNs) to understand the structure of geometry meshes. Using random walks, I develop multiscale signatures for shape analysis. I discuss the creation of geometry images

from a mesh so that CNNs can be used to infer the class of shapes. I motivate understanding shapes using random walks and the creation of geometry images for analysis using CNNs here, and the technical details are discussed in the next section.

1.1.1 Random walks for understanding shapes

Traditional methods for shape analysis resort to a single level approach, where the mesh structure is used to formulate global metrics. However the paradigm is shifting to multi-scale methods which help understand a shape over multiple levels and discover salient features on the mesh at global as well as local scales. Current multiscale methods include the heat kernel [1] and multiscale biharmonic kernel [2] which operate over the mesh domain with a tuning parameter, and have been applied to shape segmentation, correspondence and retrieval [3–5]. These kernels differ in weighting of eigenvalues and sets up one motivating question for this dissertation, '*Is there a generalized principle guiding the weighting of eigenvalues and hence the construction of multiscale kernels, while being intuitive in the choice of kernel or scale?*' In this thesis, I aim to answer this challenging question using the Markov chain framework of random walks. The choice of this framework is motivated by the fact that Markov chains emerge as a generalization of the heat equation and the intimate relationship between the discrete Laplace-Beltrami operator and the rate matrix of a random walk. For a general discussion on random walks, the reader is asked to refer to the excellent survey by Lovász [6]. As meshes are point-sampled and subsequently triangulated, the random walk framework naturally fits into general mesh analysis. It has already been used in a wide array of applications including mesh segmentation, cutting, denoising etc. [7–9]. In this thesis, I use random walks to construct multiscale kernels with two intuitive parameter choices, which generalize the current approaches under a single framework. All formulated kernels are proven to be convergent and positive definite. Hence they inherit all the '*nice*' properties of popular kernel embeddings (GPS, spectral [10], etc.) and kernel distance metrics (biharmonic, commute-time, etc.) which I briefly review.

The Heat Kernel Signature (HKS), related to diffusion kernels [11, 12], was proposed as a point signature. The idea is to extract information about the object using time t as a scaling parameter. However, choosing a suitable t is not an easy task and most approaches rely on heuristic tuning with no straightforward interpretation [13]. My approach to this problem comes from the mixing rate of a Markov chain which is intimately connected to the rate of change of the heat kernel, and hence, the diffusion process. As the kernel approaches uniform distribution, I contend its rate of convergence to equilibrium is equally (if not more) informative. This is one motivation of this dissertation and key separating factor from other approaches, i.e., to extract information about shapes using the rate of diffusion in a random walk setting. My work is also inspired by the multiscale biharmonic kernel and biharmonic distance [14]. The biharmonic distance derived from the biharmonic operator (square of Laplacian) is a parameter free distance metric on meshes with inverse square weighting of the eigenvalues. Rustamov's multiscale biharmonic paper advances a novel way to construct general multiscale kernels by minimizing the Laplacian energy subject to the sparsity inducing lasso constraint. On the surface, it is unclear how the inverse exponent weighting of eigenvalues achieves the scaling. As demonstrated in [15], the answer is interlinked with wavelet transforms on graphs and point clouds. Similar observations have been made in the machine learning community in the context of iterated ranking on manifolds using the Green's function [16].

As I shall shortly show, my approach provides an intuitive explanation while generalizing the biharmonic kernel, GPS embedding [17], biharmonic distance and commute time distance [18, 19] on meshed surfaces. To do this, I introduce novel multiscale kernels using the random walk framework and derive corresponding embeddings and pairwise distances. The fractional moments of the rate of continuous time random walk (equivalently diffusion rate) are used to discover higher order kernels (or similarities) between pair of points. Specifically, the multiscale kernels $\tau^{\alpha n}$ are constructed by integrating the moments of the probability transition rates of a random walk and governed by two parameters n and α:

$$\tau^{\alpha n} = \int_{t=0}^{\infty} t^n [\Delta_{\alpha M} e^{-t\Delta_{\alpha M}}] dt \tag{1.1}$$

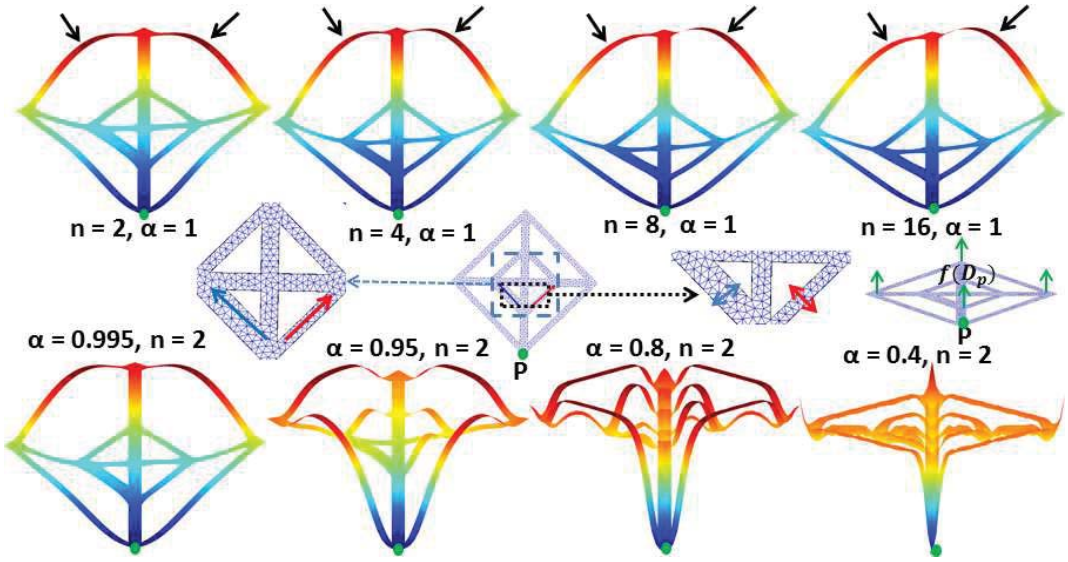


Figure 1.1. *Center: Asymmetric mesh with unequal widths (left strip (blue arrow) > right strip (red arrow)) along with exploded views. (Right) Distances are measured from point P and viewed as a height function $f(D_p)$ above mesh domain (green arrows). Color changes from blue to red as distance increases. Top Row: Varying t^n with $n = 2, 4, 8, 16$, respectively, scales underlying asymmetry in path length distribution with n acting as 'frequency filter'. High n separates distance from P to top left and top right strips displayed using black arrows. Bottom Row: Varying α with $\alpha = 0.995, 0.95, 0.8, 0.4$, respectively, makes asymmetry disappear with 'scale shift'. At small α all points are at approximately the same distance, apart from points in immediate vicinity.*

where t is time, M is the transition matrix and $\Delta_{\alpha M}$ is the lazy rate matrix. I discuss ties of M and $\Delta_{\alpha M}$ with the Laplace-Beltrami operator Δ later on. Figure 1.1 displays the multiscale distances from the green point P for the asymmetric mesh displayed in the center of the figure. The bottom left strip (blue arrow) is thicker than the bottom right strip (red arrow), and hence, there are more diffusion pathways from point P to nodes on the top left side of the mesh relative to the right. This underlying asymmetry in connecting pathways from P emerge as I increase the moment parameter n from 2 to 16, i.e., distances to nodes on the top right strip relative to the top left strip gradually increase (marked using arrows in top figure), made visible using color plot (red implies large distance) and height

field (distance along vertical direction with mesh as base). Decreasing α has the opposite effect of progressively suppressing the asymmetry (see bottom row). For $\alpha = 0.4$, the entire mesh is approximately at the same distance from P, apart from the points in the immediate vicinity. These observations can be explained in terms of Equation (1.1):

- First, replacing $\Delta_{\alpha M}$ with Δ, one immediately establishes that the $\Delta e^{-t\Delta}$ term in the integrand is the differential of the heat operator $e^{-t\Delta}$, hence equal to the negative rate of heat diffusion.

- Second, the n^{th} moment of diffusion rate with respect to time, t^n, scales the rate of diffusion, i.e., say $n = 2$ weights the diffusion rate between two points at large times more heavily than say $n = 1$ over the integral. In effect, diffusions occurring over long time periods are penalized for high values of n and are equally weighted for $n = 0$, hence the moment extracts scale (see Figure 1.1). From a signal processing viewpoint, the role of n is to filter frequencies similar to parameter t in the heat kernel, i.e., higher frequencies are progressively suppressed by increasing n [13].

- As $\Delta_{\alpha M}$ can be written as $I - \alpha M$, each element of the operator $e^{-t\Delta_{\alpha M}}$ can be expanded using the Maclaurin series of $e^{-t(I-\alpha M)}$ as:

$$e^{-t\Delta_{\alpha M}}(u, v) = e^{-t} \sum_{k=0}^{\infty} \alpha^k M^k(u, v) \frac{(t)^k}{k!} \tag{1.2}$$

  $M^k(u, v)$ can be interpreted as the sum of all random walks of length k joining points u and v [20]. In the series, paths of length k get weight α^k and hence shorter paths (say k_1) will be weighted more than longer ones (say k_2) for $0 < \alpha < 1$ ($\alpha^{k_2} < \alpha^{k_1}$) [21]. From a signal processing viewpoint, the role of α is to shift the scale of analysis as it is implicitly added to eigenvalues (see Figure 1.1).

- α^k and t^k are multiplicative parameters in Equation (1.2) and aid each other, i.e., decreasing α suppresses longer path lengths and large t discovers longer paths connecting two points. This is intuitive as time and distance are coupled notions. Allowing dual parameters α and n to control scale provides greater flexibility and reveals different multiscale information.

- Finally, integrating over the entire time interval eliminates the choice of the appropriate time interval and replaces it with the range of moment parameter which I perceive to be more intuitive. It also makes the kernel robust to noise and small topology changes.

The formulated kernels are isometry, scale, and tessellation invariant, can be made globally or locally shape aware, and are insensitive to partial objects and noise based on the moment and influence parameters. Additionally, the corresponding kernel distances and embeddings are convergent and efficiently computable. I introduce dual GMS signatures based on the kernels and discuss the applicability of the multiscale distance and embedding. Collectively, I present a unified view of popular embeddings and distance metrics while recovering intuitive probabilistic interpretations on discrete surface meshes.

1.1.2 Creating geometry images for deep understanding of shapes

A good generic 3D shape representation is vital to the progression of 3D object recognition. One strategy is to use a probability distribution on a 3D voxel grid [22, 23]. Other approaches quantify some measure of local or global variation of surface coordinates relative to a fixed frame of reference [24]. These methods are successful under rigid transformations (rotations, translations and reflections), but will naturally fail to recognize isometric deformation of a shape (say the deformation of a standing person to a sitting person). In other words representations bases on point coordinates, volumetric or surface-based, are *extrinsic* to the shape. I contend that a good shape representation should be invariant to isometry. In mathematical terms this means that the representation should preserve geodesic distances computed over shape surfaces. This contention is substantiated by the popularity of the *intrinsic* shape signatures for 3D deformable shape analysis in the geometry community [25].

The ground-breaking accuracy obtained by convolutional neural networks (CNNs) for image classification [26] marked the advent of deep learning methods for various vision tasks such as video recognition, human and hand pose tracking using 3D sensors, image segmentation and retrieval [27–29]. Consequently, researchers have tried to adapt the

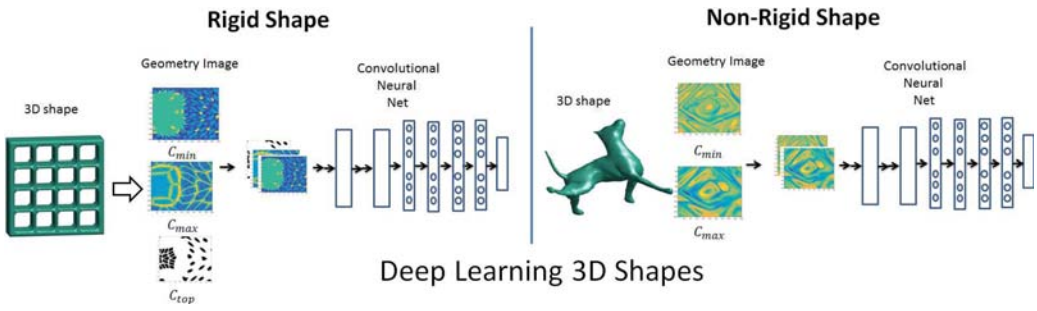


Figure 1.2. *Learning 3D shape surfaces using geometry images: My approach to learn intrinsic property of shapes using geometry images is applicable to rigid (left) as well as non-rigid objects undergoing isometric transformations (right). The geometry image encodes intrinsic properties of shape surfaces such as principal curvatures (C_{min}, C_{max}). Topology of a non-zero genus surface is accounted for by using a topological mask (C_{top}) as in the bookshelf example.*

CNN architecture for 3D deformable as well as rigid shape analysis. The lack of a unified shape representation has led researchers pursuing deformable and rigid shape analysis down different routes. Rigid shape analysis in deep learning architectures proceed using a voxelized input or panoramic representation of a shape using different viewpoints [22–24], which are inherently *extrinsic* representations. In contrast, CNN-based deformable shape analysis methods use geodesic convolutional filters as patches or model spectral-CNN's using the eigen decomposition of the Lalplace-Beltrami operator [30–32], which are *intrinsic* representations. In summary, the vision community has focussed on extrinsic representation of 3D shapes suitable for learning rigid shapes, whereas the geometry community has focussed on adapting CNN's to shift variant spaces Riemannian (or non-Euclidean) manifolds. A method to unify these two complementary approaches has remained elusive.

Instead of adapting the CNN architecture to support convolution on surfaces, I adopt the counter intuitive approach of molding the 3D shape surface to fit a regular grid structure as required by CNNs 1.2. I do this by introducing a method to transform a general mesh model into a flat and regular image, which I term 'geometry image', following [33]. In the

process, I also attempt to bridge the gap between traditional geometric processing and image processing techniques in this dissertation. The standard approach to create a geometry image is to first cut the surface into disk-like charts, then parameterize them piecewise in the plane followed by stitching them together into a texture atlas [34]. This approach fails to preserve the connectivity between different surfaces, vital for holistic shape analysis. Another approach is to spherically parameterize a genus-zero surface followed by sampling on a regular polyhedral domain namely the octahedron, tetrahedron and cube [35]. The sampled domain is then projected onto a regular planar domain by cutting along the polyhedral edges. However, spherical parametrization induces stretch distortion, leading to loss of intrinsic shape information. It is also not applicable to higher genus surfaces severely limiting its usage for general 3D shape analysis which come is various forms and topology.

My main contribution is a procedure for authalic spherical parametrization applicable to general 3D shapes. The critical limitation of spherical parametrization to genus zero surfaces is removed by filling topological holes using α-shapes along with heuristic hole-filling methods. I maintain a topological mask so as to be able to infer the holes in the original shape and recreate the mesh model. The accuracy of my authalic spherical parametrization stems from (1) an area restoring diffeomorphic flow modeled using discrete differential geometry, and (2) an iterative vertex displacement procedure from the original mesh to the spherical mesh using barycentric coordinates. I use this parametrization in conjunction with spherical area sampling and functional interpolation techniques to output a geometry image of a desired size. The geometry image simplifies complex 3D tasks such as noise removal or mesh morphing in the derived regular 2D domain. I focus on using these geometry images in a CNN architecture to classify and retrieve shapes. I demonstrate that in the presence of sufficient training data, the two principal curvatures suffice to accurately learn an intrinsic shape representation without resorting to complex multiscale shape signatures. In summary my contributions are:

1. Accurate authalic parametrization of genus-zero surface models using area restoring diffeomorphic flow and barycentric mapping.

2. Extension of authalic parametrization to higher genus surface models via a topological mask. I also demonstrate a heuristic procedure to rectify incorrect mesh models not following the Euler characteristic.

3. An approach to intrinsically learn 3D surfaces using a geometry image which encodes features invariant to isometry.

To summarize, surfaces are the most intuitive parametrization of 3D shapes. Learning surfaces using convolutional neural networks (CNNs) is a challenging task due to the shift variance of Riemannian manifolds. Current paradigms to tackle this challenge are to either adapt the convolutional filters to operate on surfaces, learn spectral descriptors defined by the Laplace-Beltrami operator, or to drop surfaces altogether in lieu of more favourable voxelized inputs. I adopt an antagonistic approach of converting the 3D shape into a *'geometry image'*, so that standard CNNs can directly be used to learn 3D shapes. This conversion is made possible by a low-distortion authalic (area conserving) parametrization of a genus-zero surface onto a spherical domain. This spherically parameterized shape is then sampled onto an octahedron and is subsequently *cut* to convert the original 3D shape into a flat and regular geometry image. In order to maintain generality of mu approach to learn shapes as geometry images, I extend spherical parametrization to surfaces with non genus-zero surfaces by introducing a topological mask. I show the efficacy of my approach to intrinsically learn 3D shapes using geometry images on several datasets. Furthermore, I validate that my representation is able to retain all pertinent information of a 3D shape.

1.2 Networks as Graphs

Networks help uncover the underlying mechanisms governing complex systems by collectively understanding dyadic interactions between physical, biological or social objects represented as nodes [21, 36]. In networks, I focus on three things: (1) Identify communities or subsets of associated nodes in a network which is a crucial step towards discovering the functional organization of the network [37–39], (2) Investigate whether it is possible to identify items affected by feedback loops in recommender systems and (3) Interpreting

popular kernels used in network science as operators on a random walk, or more generally a diffusion process on a graph. I motivate discovering communities using Gauss's law, deconvolving feedback loops to discover intrinsic preferences of users, and the use of diffusion on a graph for data mining and network science problems here. The technical details are discussed in the next section.

1.2.1 Gauss's Law for identifying community boundaries

Overlap characterized as the possibility of nodes having multiple membership [40, 41], and hierarchy representing the organization of communities over multiple scales [42, 43], are two prominent features of communities in real world networks. Although there is no unique definition, a common abstraction is that communities in networks are groups of tightly connected nodes (dense intra-connections) and weakly connected to each other (sparse inter-connections) [44, 45]. Community detection approaches formalize this abstraction and determine the best division of the network into communities by optimizing a local [41, 46, 47] or global [48–50] objective function. Local measures neglect mesoscopic topological interactions valuable for community detection, and consequently global measures have emerged as the leading choice [38], the most popular being modularity optimization [49]. However, the hierarchical organization of communities contradicts the existence of a best partition, but instead necessitates comparing the quality of partitions obtained at different organizational levels [51]. This obscures a precise formalism of both local and global objective functions. Local definitions intrinsically cannot compare partitions obtained at different hierarchical levels without introducing ad-hoc stability measures [46], whereas adaptive global definitions [52, 53] using a resolution parameter suffer from the resolution limit [54]. Further, overlap between communities breaks the fundamental assumption of communities being sparely inter-connected, as pervasive overlap leads to strong inter-community connections [41]. The presence of both these features in many real-world networks warrants a precise definition of overlapping communities consistent with communities organized at multiple scales. Although there are abundant methods for

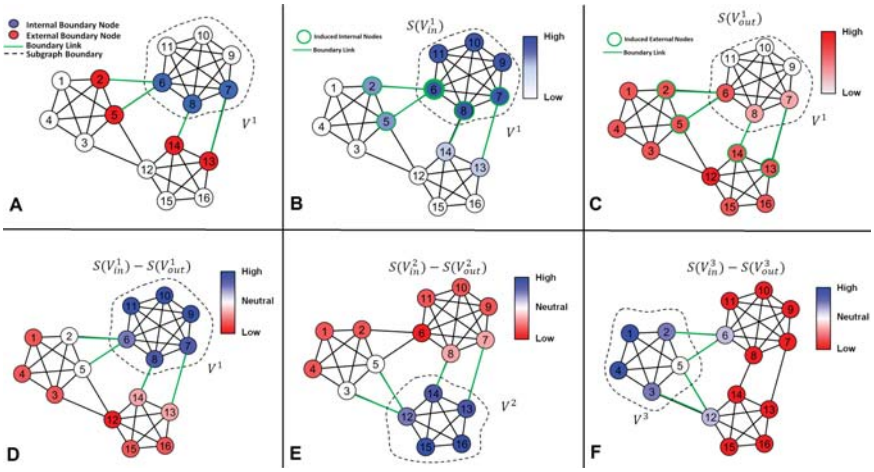


Figure 1.3. *Identifying module boundaries improves detection. a: The curvature (color) identifies the boundaries (high curvature) between the peaks that represent the values of the scalar field (height). b: (Top) A curvature-like metric of a network shows the same effect wherein the curvature of inter-community links is higher (darker shades of red) relative to intra-community links. (Bottom Left) The curvature of links in the network are displayed in the link curvature matrix. The curvature increases as color changes from blue to yellow. (Bottom Right) The histogram of curvature values shows a gap between the curvature of intra-module and inter-module links allowing easy detection. c: The same methodology is applicable to general graphs such as social networks, meshes and images and identifies module boundaries even in the presence of large noise. The identified modules are mapped to the corresponding noiseless graphs in the top-right corner for visual verification.*

community detection, there exists no quantitative definition of overlapping communities which supports hierarchical community organization [38].

Here, I seek to use information about the boundary between subgraphs to identify the communities themselves. This boundary information arises from the curvature of each link (Figure 1.3 a, b), and the information from the boundaries reliably indicates the module structure even in the presence of substantial noise (Figure 1.3 c). I will describe how a network analog of Gauss's law resolves the dichotomy between overlap and hierarchy because the my definition uncovers overlapping communities and that hierarchical communities are

nothing but completely overlapping communities at a coarser scale of analysis. I show that this network analog of Gauss's law translates into an efficiently optimizable quality function to reveal hierarchical and overlapping structure in networks. The results on several real world networks with known and unknown ground truth community assignment spanning diverse disciplines, conclusively validates the merit of my contribution relative to current approaches. Further, I show that expectations of diffusion rate generalize several network science tools like PageRank [55], harmonic kernels [56–58] and deconvolution matrices [59]. This framework advances a general methodology to deduce the similarity between nodes in addition to identifying community structure, and serves as a critical tool for network science analysis.

1.2.2 Deconvolving feedback loops in recommender systems

Recommender systems have been helpful to users for making decisions in diverse domains such as movies, wines, food, news among others [60, 61]. However, it is well known that the interface of these systems affect the users' opinion, and hence, their ratings of items [62, 63]. Thus, broadly speaking, a user's rating of an item is a either his or her intrinsic preference or the influence of the recommender system on the user [64]. As these ratings implicitly affect recommendations to other users through feedback, it is critical to quantify the role of feedback in content personalization [65].

Thus the primary motivating question for this work is: Given only a user-item rating matrix, is it possible to infer whether any preference values are influenced by a recommender system? Secondary questions include: Which preference values are influenced and to what extent by the recommender system? Furthermore, how do I recover the intrinsic preference value of an item to a user?

I develop an algorithm to answer these questions using the singular value decomposition (SVD) of the observed ratings matrix. The genesis of this algorithm follows by viewing the observed ratings at any point of time as union of true ratings and recommendations:

$$\boldsymbol{R}_{\text{obs}} = \boldsymbol{R}_{\text{true}} + \boldsymbol{R}_{\text{recom}} \tag{1.3}$$

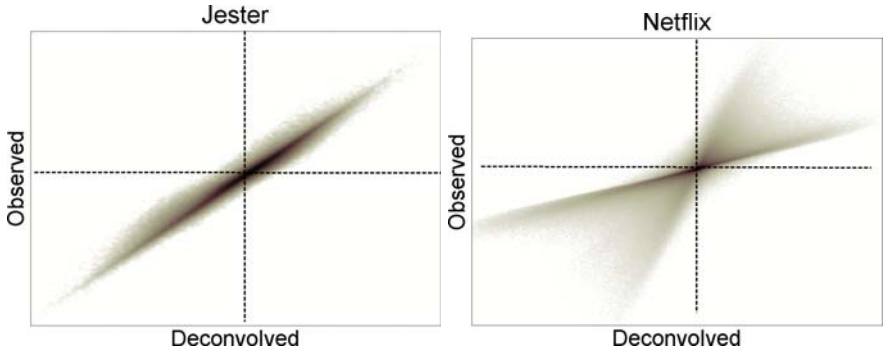


Figure 1.4. *A scatter plot of deconvolved and true ratings on the Jester joke dataset (Top) that had no feedback loops and on the Netflix dataset (Bottom) where their Cinematch algorithm was running. The Netflix data shows dispersive effects indicative of a recommender system whereas the Jester data is highly correlated indicating no feedback system.*

where $\boldsymbol{R}_{\text{obs}}$ is the observed rating matrix at a given instant of time, $\boldsymbol{R}_{\text{true}}$ is the rating matrix due to users' true preferences of items (along with any external influences such as ads, friends, and so on) and $\boldsymbol{R}_{\text{recom}}$ is the rating matrix which indicates the recommender system's contribution to the observed ratings. My more formal goal is to recover $\boldsymbol{R}_{\text{true}}$ from $\boldsymbol{R}_{\text{obs}}$. But this is impossible without strong modeling assumptions; any rating is just as likely to be a true rating as due to the system.

Thus, I make four strong, but plausible assumptions about a recommender system. With these assumptions, I am able to mathematically relate $\boldsymbol{R}_{\text{true}}$ and $\boldsymbol{R}_{\text{obs}}$. This enables me to find the centered rating matrix $\boldsymbol{R}_{\text{true}}$ (up to scaling). An example of the types of insights this enables is shown in Figure 1.4. This figure shows two scatter plots of the estimated *true* ratings (y-axis) compared with the *observed* ratings (x-axis) for two datasets. If there is no recommender system, then these should be perfectly correlated. If there is a system with feedback loops, I should see a dispersive plot. In the first plot (Jester) I see the results for a real-world system without any recommender system or feedback loops; the second plot (Netflix) shows the results on the Netflix ratings matrix, which did have a recommender

system impacting the data. In the experiments section, I further validate that this dispersive pattern indeed characterizes recommender system effects through extensive experiments.

Based on these insights, I develop a heuristic, but accurate, metric to quantitatively infer the influence of a recommender system (or any set of feedback effects) on a ratings matrix. The results of this metric are shown in the results section. A score near 0 indicates no recommender system and a large score indicates strong recommender system effects. Additionally, I propose a metric for evaluating the influence of a recommender system on each user-item rating pair. Aggregating these scores over all users helps identify putative highly recommended items. The fraction of non-zero values provide insight into the quality of recommendations, and these final metrics argue that Netflix had a better recommender than MovieLens, for example.

1.2.3 Graph diffusion for understanding kernels in network science

The starting point of any data mining application is the notion of similarity between two nodes in a graph. Although the similarity metrics proposed in literature abound, there lacks a systematic understanding of these metrics. Similarity metrics fall under the category of either local metrics such as cosine, Jaccard etc. or global metrics such as PageRank, diffusion, Katz etc. I observed that a common feature underlying global metrics is that the corresponding similarity matrices, or kernels, possess the same or similar eigen-spectrum. Specifically, all kernels grounded on random walk principles have the same set of eigenvectors and only differ in the eigenvalue weighting. Similarly, path-length based kernels have the eigenvectors of the adjacency matrix as a common orthonormal basis and differ in the functional form of eigenvalues. Global kernels are crucial in determining the centrality of nodes and ranking in graphs. I investigate the principles governing eigenvalue weighting in kernels, and propose a general mechanism to construct kernels for determining centrality and ranking.

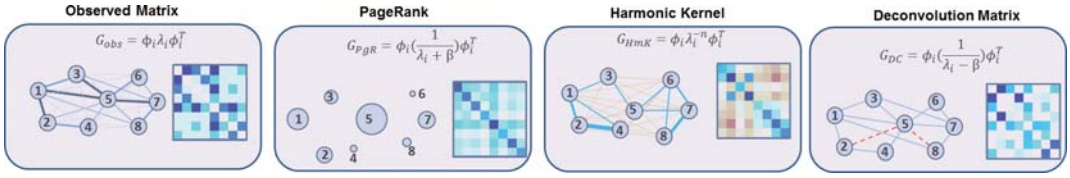


Figure 1.5. *Kernels constructed as expectations of the rate of diffusion kernel controlled by 2 parameters: frequency shift, β and frequency filter, n. The 4 matrices differ in β and n weighting of eigenvalues λ and have the same eigenvector basis ϕ. Left: Original matrix with edge width and color proportion to weight. The dashed edges represent noise. The corresponding color coded adjacency matrix is to the right of the network with darker shades of blue indicating greater edge weight. Left-Center: PageRank with node sizes proportional to its PageRank and color coded PageRank matrix to the right. Right-Center: Harmonic kernel with graph edges colored blue for positive affinity and red for negative affinity at $n = 3$. The corresponding color coded affinity matrix is to the right with darker shades of blue or red indicating greater positive or negative affinity respectively. Right: Deconvoluted network at $\beta = 2$ with inferred indirect relationships represented as dashed red lines. The color coded deconvolution matrix is to the right of the network with higher similarity colored in darker shades of blue.*

Specifically, I investigate kernels, τ of the form:

$$\tau = \int_{t=0}^{\infty} f(t)\dot{p}_t dt \tag{1.4}$$

where $f(t)$ is a function convoluting with $\dot{p}_t$, the rate of diffusion. Using diffusion, p_t, instead of $\dot{p}_t$, does not work in the integral due to the null space effect, specifically the presence of an eigenvalue 0 in the spectral expansion confounds the analysis. My key insight is that $\dot{p}_t$ removes the null space, and hence Laplace transform is directly applicable to evaluate the integral and output general kernels. Different functions, $f(t)$, result in different kernels with its own set of interpretation. Although, my formulation is applicable for general diffusion equation, I choose a particular form of the diffusion operator, also known as the Laplacian, which describes random walks. The contributions are:

- Spectral analysis of continuous time random walks and its relationship with the rate of diffusion.

- General mechanism to construct novel kernels by convoluting a function with the rate of diffusion and the properties of kernels constructed by convoluting polynomial functions.
- Extension of PageRank as an arithmetic-geometric sum of random walks, extension of harmonic kernels to a general real exponent and generalization of network deconvolution approach.
- Validation of my kernels as link predictors in graphs.

1.3 Images as Graphs

In images, pixels play the role of nodes and the pixel gradients play the role of edges. I use the image representation for 2 physical laws based applications: (1) Image segmentation using continuous random walks, or diffusion, and (2) Neural network based real time tracking of hand pose. I introduce the motivation for these 2 applications next.

1.3.1 Diffusion based image segmentation

Understanding the interplay between local scale and global effect is a challenging problem in image segmentation. My remedy is to use a learning approach to discern all pairwise affinities using sparse local connections. Traditional unsupervised image segmentation methods resort to grouping individual pixels based on similarities/dissimilarities in image features (e.g. brightness, color, texture) over local patches [66]. The three most common methods under this framework include Shi and Malik's normalized cut criterion [67], Comaniciu and Meer's mean-shift algorithm [68] and Felzenszwalb and Huttenlocher's (Felz-Hutt) graph-based segmentation [69]. Local image information is suitable for identifying feature rich regions; however, coherent perceptual grouping necessitates the use of global cues between image patches. Two parallel approaches exist for propagating local image information to global scales:

1. Explicitly or implicitly increase the connection radius between pixels in the affinity matrix

2. Use learning techniques on a sparse affinity matrix to construct a better similarity matrix or kernel.

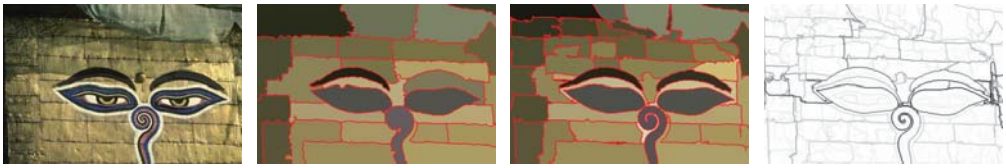


Figure 1.6. *Left to Right : Original image, single-scale segmentation, multi-scale segmentation and weighted boundary image obtained using variants of the generalized Green's function.*

Explicit increase of connection radius is computationally prohibitive, and hence, multi-scale frameworks for normalized cut [70] and 'over-segmentations' from Felz-Hutt, mean-shift algorithms [71, 72] focus on implicitly increasing the connection radius. The second emerging paradigm for systematic combination of local and global cues is to learn all pairwise affinities using locally connected graphs [73, 74]. The central notion to these approaches is propagating (diffusing) affinity on a sparse graph and is intimately tied to heat diffusion (equivalently continuous time random walks) [75, 76]. These approaches have led to state of the art results in retrieval and ranking [77]. More recently, higher order node affinities have been used for segmenting an image [74, 78]. Full-affinity cut [78] utilizes full pairwise affinity matrices for image segmentation and Self-diffusion [74] performs a diffusion process on the affinity matrix for a certain critical time-stamp. Other dense matrix approaches include, the use of commute time matrix for image segmentation [79] and use of biharmonic kernel for shape segmentation [80]. The isoperimetric criterion [81] calculates all affinities from a *'ground'* node by solving a set of linear equations and recursively partitions the obtained 2-way segmentation. However, a common disconnect underlying all these approaches is that the derived kernel is used as a *pre-processor* in conjunction with a

standard metric or method to induce partitions, i.e., Full- affinity cut and Self-diffusion use the normalized cut criterion, commute time in [79] uses spectral partitioning, biharmonic kernel in [80] relies on mean-shift and isoperimetric graph partition [81] uses the ratio cut criterion. Hence, there lacks a formal criterion which directly induces partitions using the structure of the kernel itself. I propose a novel global criterion, the expected diffusion modulus, which induces partitions based on every nodes' tendency to diffuse heat, subject to flow conservation. The diffusion modulus kernel implicitly learns all pairwise affinities using the n^{th} moment of diffusion rate. The moment parameter n infers resolution and naturally outputs the commonly used commute-time and biharmonic kernels as special cases. The special structure of the kernel enables fast Louvain-like optimization and automatically determines the number of segments for a given resolution n. Enforcing long range connections using an over-segmented image leads to fast high-quality segmentations on the BSDS and MSRC databases. I prove that the diffusion modulus is intricately related to the ubiquitous normalized cut criterion for image segmentation and modularity metric for community detection.

1.3.2 Deep neural networks for hand tracking

Robust hand tracking is central to human-computer interaction interfaces and augmented reality applications. Although, there exists robust and accurate methods for full body tracking, hand tracking is far more challenging [28, 82–89]. This is due to several reasons: (i) the hand pose exists in a high dimensional space because each finger and the palm is associated with several degrees of freedom, (ii) the fingers exhibit self similarity, are flexible and often occlude each other, (iii) noise in acquired data coupled with fast finger articulations confounds continuous hand tracking. Multi camera setups or GPU acceleration eases some of these challenges, but limits deployment to the general public.

I present a robust method for hand tracking with a single depth camera which achieves real time performance without a GPU. Specifically, I propose a novel matrix completion method to estimate the joint angle parameters on a per frame basis. My method is flexi-

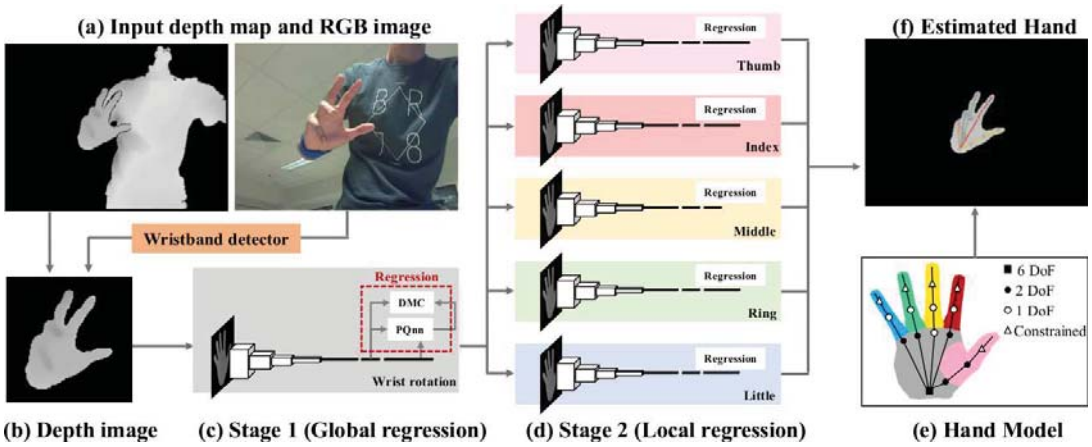


Figure 1.7. *An overview of the proposed approach. In a real-setting, I extract region of interest using depth map and RGB-based wrist band detector (a)-(b). The obtained depth image is fed into a ConvNet which outputs an activation feature. These activation features synchronize with other features in a population database using my deep matrix completion method and the global pose parameters are estimated(c). Based on this global pose initialization, I estimate the rest of the local joint parameters in the same recursive manner (d). The final hand pose is displayed on a multimedia screen (e).*

ble to operate with or without temporal information. This alleviates the need for explicit pose initialization if the method loses track or the hand disappears from the camera's view frustum. Furthermore, my pre-compiled database supports a large viewpoint coverage and my hierarchical pose estimation from global to local parameters is robust to severe finger occlusions.

At the core of my approach lies a convolutional neural net (ConvNet) architecture to discriminatively reduce the dimensionality of the depth map. ConvNet's have achieved ground-breaking performance in image classification and video recognition. A naive strategy to replace the classification layer in a deep neural net with a regression layer leads to errors, as the objective function often gets stuck in a local minima. Previous approaches has have shown that this error decreases by incorporating a prior [90] or a intermediate heat map features [28] into the ConvNet architecture. Different from these approaches, I

train several ConvNets to output a discriminative low dimensional activation feature in the penultimate fully connected layer. This activation vector represents either the global hand orientation or the local articulations of the five fingers, given a depth map. My main insight is that a pool of (spatially or temporally) nearby activation features to an activation feature can better represent the hand pose. For generating a population of activation features from which such a pool is extracted, I render realistic depth maps covering a large range of hand articulations and feed them into a deep ConvNet. The ConvNets automatically learn the scope of training (local or global), the finger type (thumb, ring, index, middle or little), and prevalent occlusions by simply inputting the discretized class of the pose parameter values, and do not require any additional information. I then store the activation features from the ConvNets for each depth map in the training data to create a population database of activation features. I demonstrate these activation features can be re-purposed on generic databases in my experiments. Additionally, the low dimensionality of the activation feature, coupled with product quantization enables efficient retrieval of approximate nearest neighbors from the population at runtime.

A pose estimation matrix is imputed with the *deep* activation vectors of the nearest neighbor, their corresponding joint angles and the activation vector of the input depth map. This is similar in spirit of the collaborating filtering approach proposed in [91]. However, neither do I use low fidelity BRIEF descriptors for nearest neighbor retrieval, nor do I use inefficient iterations to factorize and complete the matrix. Instead, I estimate the unknown values in the incomplete matrix (i.e., pose parameters of input depth map) by assuming a low-rank matrix structure with missing entries. I also add some temporal neighbors from previous frames in the pose estimation matrix which act as a regularizer and reduce jitter of the estimated pose.

Following the success of cascaded approaches to hand pose estimation [85,89], I hierarchically regress the hand pose from global to local joint angle parameters. The articulation complexity of the palm is lower than of the fingers, and hence, robust estimation of the global orientation is an easier task relative to that of the fingers. The ConvNet finetuned to the conditioned search space outputs more discriminative activation features for finger

articulations. This in turn leads to better accuracy for finger parameter estimation. I demonstrate that the ConvNet architecture significantly outperforms PCA [85] and random forests (RF) [89] for global pose initialization. My overall pipeline runs as 32 frame per second (FPS) on a standard computer. My main contributions are summarized as follows:

1. Initialization of the pose matrix using a low dimensional and discriminative representation of the global orientation or finger articulations as an activation feature using deep ConvNets, which aids efficient retrieval of nearest neighbors from a large population of pre-computed activation features using product quantization.

2. A deep matrix completion (DMC) method for estimating joint angle parameters using the initialized pose matrix.

3. A hierarchical pipeline for hand pose estimation that combines the global pose orientation and finger articulations in a principled way while maintaining real-time frame rates on a standard computer.

Overall, my techniques provide superior results in pattern recognition objectives such as image segmentation in computer vision, accurately retrieve similar shapes to a query from a large database of geometric mesh models, prevent false positives in recommender systems and identify the hierarchical and overlapping organization of networks.

The organization of the thesis is as follows: I first introduce the theoretical results for the three different graph types and reference prior literature as required in Chapter 2. In Chapter 3, I discuss the applications of the introduced theoretical concepts in Chapter 2. Finally, I summarize and discuss future work in Chapter 4.

2. THEORY

In this chapter I discuss theoretical details for the four physical laws and how they apply to understanding graph structure. Each subsection discusses the learning analogy to physical laws: (1) Brownian motion or random walks, (2) Gauss's law, (3) Feedback and (4) Deep Neural networks for understanding graph structure. Relevant background is introduced wherever applicable.

2.1 Random Walks

A Markov chain is a mathematical system that undergoes state transitions following the Markov property, i.e., future states depend only upon the present state and not on the past, or that the system is memoryless. Random walks can be viewed as special cases of finite time-reversible Markov chains [92]. A random walk starting from a node on a graph evolves with time, where time is either discrete (set of positive integers) or continuous (set of positive real numbers). Here, I briefly discuss the formalism of discrete and continuous time random walks, the inter-relationship between the two and relevant spectral decompositions which are used in subsequent sections. For a general discussion on random walks, the reader is asked to refer the excellent survey by Lovász [93]. In this section, I first detail the random walk operators for different graph based representations. Next, I discuss constructing kernels using random walks first in the context of networks and then on shapes. Finally, I describe how random walks can be used for image segmentation.

2.1.1 Random walk operators

First, I detail basics useful for analyzing discrete and continuous random walks. An unweighted, undirected network G of N nodes is completely characterized by $N \times N$ adjacency matrix A, a symmetric binary matrix of zeros and ones with $'1'$ indicating an edge

between two nodes and $'0'$ otherwise. Let $D = diag(d_i)$ be a diagonal matrix of the node degrees in the graph. The probability transition matrix M for the discrete walk is given by $M = D^{-1}A$, indicating the probability of leaving a node is evenly split among all its adjacent neighbors. As M is asymmetric in general, it is useful to convert it into a symmetric normalized matrix N for spectral analysis by considering $N = D^{1/2}MD^{-1/2}$. The study of continuous time walks is simplified by introducing the random walk Laplacian Δ_M and its normalized counterpart $\mathcal{L}$. Mathematically, $\Delta_M = I - M$ and $\mathcal{L} = I - N$ where I is an identity matrix of size N. Let the spectral decomposition of the normalized Laplacian be given by $\mathcal{L} = \Phi\Lambda\Phi^T$ where Λ is a diagonal matrix with the ordered eigenvalues, i.e., $0 = \lambda_1 \leq \lambda_2 ... \leq \lambda_N \leq 2$, Φ is the matrix with corresponding eigenvectors as columns. Then the eigen decomposition of the normalized transition matrix is $\Phi\Lambda'\Phi^T$ with same set of eigenvectors and eigenvalues related as $\Lambda' = I - \Lambda$ (see [94]). I first discuss discrete and continuous time random walks which are directly applicable to networks, and then describe the random walk operators specific to meshes and images, respectively.

Discrete time random walk: Discrete random walks take place over discrete time steps starting from $t = 0$ over the positive integer domain. The rule of walk on the graph can be expressed as

$$P_t = M^T P_{t-1} \tag{2.1}$$

Hence iterating,

$$P_t = (M^T)^t P_0 \tag{2.2}$$

where P_t is the probability distribution (summing to 1) over the nodes of the graph starting from the initial distribution P_0 and depends on the transpose of the state transition matrix M^T. The stationary distribution or the probability of being at a vertex after the walk has reached equilibrium (after long time or at $t = \infty$) is independent of the initial distribution. Formally $\pi = M^T\pi$ or $\pi = \mathbb{1}D/vol$ where $\mathbb{1}$ is a vector with all coordinates 1 and $vol = \sum_u d_u$ called the volume of the graph. The distribution at intermediate times, i.e., M^t can be determined using the spectrum of the normalized Laplacian or can be calculated using

elementary matrix multiplication. The spectral decomposition of the transition matrix is $M = D^{-1/2}\Phi\Lambda'\Phi^T D^{1/2}$ and $M^t = D^{-1/2}\Phi\Lambda'^t\Phi^T D^{1/2}$. Hence,

$$p_t(i,j) = \sum_{k\geq 1} \lambda_k'^t \Phi_k(i)\Phi_k(j)\sqrt{\frac{d_j}{d_i}} = \pi_j + \sum_{k\geq 2} \lambda_k'^t \Phi_k(i)\Phi_k(j)\sqrt{\frac{d_j}{d_i}} \tag{2.3}$$

where $p_t(i,j)$ is the probability of starting at i and reaching j in t steps and is the $(i,j)^{th}$ entry of the matrix M^t and tends to π_j in the long time limit as $\mid \lambda_k' \mid \leq 1$. I have used the orthogonality of eigenvectors $\Phi_i^T\Phi_j = \delta(i,j)$). Note that P_t is a $N \times 1$ vector whereas p_t is a $N \times N$ matrix. The limitation of discrete time steps is overcome in the continuous time limit and is the topic of my next discussion.

Continuous time random walk: The difference between the discrete and continuous setting of the random walk is the waiting time between hops of a random walker, i.e., constant 1 for the discrete walk (as $t = 0, 1, 2 \ldots$) and an exponential distribution for continuous time [95]. Recall, the exponential distribution characterizes the waiting time in a Poisson process and hence the number of jumps completed by a random walker at time t is a Poisson distribution. The rule of walk is expressed by the Kolmogorov differential equation:

$$\frac{dp_t}{dt} = -\Delta_M p_t \tag{2.4}$$

I solve the Kolmogorov equation to determine the probability distribution over all pairs of state spaces at time t and solution is the exponential matrix. I recover the probability distribution at time t by using the formula

$$P_t = [e^{-t\Delta_M}]^T P_0 \tag{2.5}$$

Where P_0 is the initial probability distribution vector and P_t is the vector at time t. Again using the orthogonality of the eigenvectors, the relation $\Delta_M = D^{-1/2}\Phi\Lambda\Phi^T D^{1/2}$ and noticing that the normalized Laplacian matrix is diagonalizable, the matrix exponential is calculated using the eigenvectors as $e^{-t\Delta_M} = D^{-1/2}\Phi e^{-t\Lambda}\Phi^T D^{1/2}$. And the spectral decomposition of the exponential matrix at time t is given as

$$p_t(i,j) = \pi_j + \sum_{k\geq 2} e^{-t\lambda_k}\Phi_k(i)\Phi_k(j)\sqrt{\frac{d_j}{d_i}} \tag{2.6}$$

As in the discrete setting, the $(i,j)^{th}$ entry $p_t(i,j)$ evaluates the probability of transition from i to j in time t and the probability tends to π_j in the long time limit as $\mid \lambda_k \mid > 0$. Note that probability conservation holds for both discrete and continuous time random walks, i.e., $\sum_j p_t(i,j) = 1 \; \forall i \in V$ or that the probability of a random walker to jump from state i to any other state is 1.

The diffusion equation is central to many graph theoretic applications ranging from consensus in multi-agent systems [96] to synchronization in community structure [97]. Simply put the diffusion equation models flow from high density regions to low density regions based on the density gradient and is succinctly captured by the equation:

$$\frac{d\Psi}{dt} = -\Delta\Psi. \tag{2.7}$$

Here Ψ is the quantity that flows (gas, heat, etc.) and Δ is the Laplace operator which governs the rate of flow. On discrete graphs, the Laplace operator takes different forms based on the context of application. It is apparent that if I take $\Delta = \mathbb{L}$, and $\Psi = P_t$, the equation describing the system is the same as Equation (2.4). The solution to the above differential equation also gives rise to the heat kernel [98], i.e., the heat kernel H_t is continuous time random walk matrix, p_t. From here on, I refer to the continuous time random walk matrix interchangeably as the heat kernel, H_t and represent the discrete time analog in terms of the transition matrix M.

Each element of H_t can be represented by expanding the exponential $e^{-\mathbb{L}t}$ using the Maclaurin series of $e^{-t(I-M)}$ as:

$$H_t(u,v) = e^{-\mathbb{L}t}(u,v) = e^{-t}\sum_{k=0}^{\infty} M^k(u,v)\frac{t^k}{k!}. \tag{2.8}$$

$M^k(u,v)$ is interpreted as the sum of all transfer probabilities of length k joining nodes u and v. t is a real positive number in the range $[0,\infty)$. As the heat kernel, H_t is an exponential sum of random walks, it follows that the derived node-to-node similarities are in some sense a measure of regular equivalence [99]. Regularly equivalent nodes serve the same role in a network, which in itself is a reasonable notion of a community (such as functionally related protein complexes or social circles). The random walk Laplacian governing the heat kernel can be viewed as a normalized counterpart of the standard Laplacian.

Random walk mesh Laplacian: The random walk Laplacian discussed before does not naturally extend to meshes. In order to coherently analyze mesh model, I discuss the formulation of the Laplacian for mesh surfaces. Specifically, I explain the construction of probability transition matrix M for discrete random walks, and the construction of rate matrix Q for continuous time random walks on a triangulated surface mesh. The triangulation (discrete state space) assumption is consistent with general analysis on point-sampled meshes.

The rate matrix Q is negative of the discrete Laplace-Beltrami operator Δ. This stems from the fact that Brownian motion and heat flow both satisfy the diffusion equation, and that Brownian motion is a generalization of continuous time random walks to continuous state spaces. This relationship can be directly observed by comparing the heat diffusion equation $\frac{\delta H_t}{\delta t} = -\Delta H_t$ (H_t is the heat kernel) and Kolmogorov's backward equation $\frac{\delta P_t}{\delta t} = QP_t$ (P_t is the Markov kernel) for continuous time Markov chains [100]. The Brownian interpretation of heat kernel was briefly explained in [1].

It is well known that the Laplace-Beltrami operator Δ is a generalization of the Laplacian from flat spaces to manifolds [101]. The operator can be constructed using the common cotangent discretization [101]. The weights w of the Laplace-Beltrami operator are given by $w(i,j) = \begin{cases} \sum_j \frac{cota_{i,j}+cotb_{i,j}}{2A_i} & \text{if} \quad j = i \\ -\frac{cota_{i,j}+cotb_{i,j}}{2A_i} & \text{if} \quad j \subseteq N_1(i) \\ -\frac{cota_{i,j}}{2A_i} & \text{if} \quad (i,j) \in E_B \\ 0 & \text{otherwise} \end{cases}$

where $\mathcal{N}_1(i)$ is the set of 1-ring neighbours of vertex i, $a_{i,j}, b_{i,j}$ are the two angles supporting the edge connecting vertices i and j, A_i is the associated surface patch or finite volume (usually barycentric or Voronoi) and E_B indicates a boundary edge. The $b_{i,j}$ term drops out for boundary points imposing the von Neumann boundary conditions. I assume that the triangulation is regular, i.e., $a_{i,j} + b_{i,j} \leq \pi$ so that all weights are positive. Another commonly used operator, the conformal Laplacian L_c [102] is related to the Laplace-Beltrami operator by $\Delta = A^{-1}L_c$, where A is a diagonal matrix of the Voronoi,

barycentric or mixed area (A_u for point u). As $Q = -\Delta$, the weights q of the rate matrix are given by

$$q(i,j) = \begin{cases} \sum_j \frac{-(cota_{i,j}+cotb_{i,j})}{2A_i} & \text{if} \quad i = j \\ \frac{cota_{i,j}+cotb_{i,j}}{2A_i} & \text{if} \quad j \subseteq N_1(i) \\ \frac{cota_{i,j}}{2A_i} & \text{if} \quad (i,j) \in E_B \\ 0 & \text{otherwise} \end{cases}$$

Verify that $\sum_v q_{ij} = 0 \; \forall j \in V$ where V is the set of all nodes, which means that the sum of all transition rates to or from a node is conserved, i.e., flow conservation. For discrete random walks, the probability transition matrix M satisfies:

- The single-step transition probability of jumping from any node u to an adjacent node v ($(p_{ij})_{j \in N_1(i)}$) is positive, i.e., $0 \leq p_{ij}$
- The probability of a random walker to jump from state u to any state is 1, i.e., $\sum_j p_{ij} = 1 \; \forall j \in V$ (conservation property).

I use the uniformization technique of Markov chains to build M satisfying the above properties. Uniformization is a technique to simulate continuous time chains using a discrete chain analog (see Lawler [100] for a detailed exposition). Suppose υ_i represents the absolute diagonal values of the rate matrix Q and $\Upsilon = \max_i \upsilon_i$, then the uniformized chain is given by

$$p(i,j) = \begin{cases} \frac{q(i,j)}{\Upsilon} & \text{if} \quad i \neq j \\ 1 - \frac{\upsilon_i}{\Upsilon} & \text{if} \quad i = j \end{cases}$$

Using the above transformation, the elements of the transition matrix M are

$$p(i,j) = \begin{cases} 1 - \sum_j \frac{cota_{i,j}+cotb_{i,j}}{2\Upsilon A_i} & \text{if} \quad j = i \\ \frac{cota_{i,j}+cotb_{i,j}}{2\Upsilon A_i} & \text{if} \quad j \subseteq N_1(i) \\ \frac{cota_{i,j}}{2\Upsilon A_i} & \text{if} \quad (i,j) \in E_B \\ 0 & \text{otherwise} \end{cases}$$

It is easy to verify that regular triangulation and scaling by Υ ensure satisfaction of Property 1 and 2 respectively. Note Property 1 does not hold for general meshes. The diagonal elements of the transition matrix are zero only if u coincides with the index of the maximum element Υ. The non-zero diagonal values ξ_u indicate presence of self-loops wherein a random walker remains

in the same state with positive probability and with probability $(1 - \xi_u)$ hops to a 1-ring neighbour (also called the lazy random walk). For notational consistency, I scale the values of the Laplace-Beltrami operator Δ by $1/\Upsilon$ and represent it by Δ_M. Note the relations $Q = -\Upsilon\Delta_M$ and $\Delta_{\alpha M} = I - \alpha M$, where $\Delta_{\alpha M}$ is the lazy rate operator and α is a tuning parameter.

Diffusion operator for images: Images are denoted using a weighted graph, by the triple $\mathcal{G} = (V, E, \mathcal{A})$ where V is the set of pixels/regions, $E \subseteq V \times V$ is the set of local edge affinities and $\mathcal{A}$ is the weighted adjacency matrix with weights $w(u, v)$ if $(u, v) \in E$. The transition matrix M that governs a random walk is related to $\mathcal{A}$ as $M = D^{-1}\mathcal{A}$ and corresponding random walk Laplacian Δ is given by $\Delta = I - M$. As Δ is unsymmetric I consider the normalized graph Laplacian $\mathcal{L}$ related to $\mathcal{A}$ and Δ as $\mathcal{L} = I - D^{-1/2}\mathcal{A}D^{-1/2} = D^{1/2}\Delta D^{-1/2}$. Its elements are:

$$\mathcal{L}(u, v) = \begin{cases} 1 & \text{if } u = v \\ -\frac{w(u,v)}{\sqrt{d_u d_v}} & \text{if } i \neq j \text{ and } (u, v) \in E \end{cases} \tag{2.9}$$

2.1.2 Multiscale kernels using random walks on networks

Different graph theoretic metrics are a result of different weighting scheme on the eigenvalues, i.e., the representation is of the general form $\tau = \Phi f(\Lambda)\Phi^T$ up to a scaling factor where $f(\Lambda)$ is an operator on the eigenvalues. This motivates me to investigate if there a generalized principle guiding the weighting of eigenvalues and hence the construction of the similarity matrices τ. In this discussion I establish that this general principle is guided by suitable expectations of the generalized rate of continuous time random walks, or equivalently and more intuitively the rate of diffusion matrix. The generalized rate of diffusion matrix $\tilde{B}$ is given as $(I - \alpha M)e^{(\alpha M - I)t}$. The expectations of $\tilde{B}$ over time are naturally given by:

$$\tau^{\alpha n} = \int_{t=0}^{\infty} t^n [\Delta_{\alpha M} e^{-t\Delta_{\alpha M}}] dt \tag{2.10}$$

where $\tau^{\alpha n}$ is the output similarity matrix governed by two parameters n and α and $\Delta_{\alpha M}$ is the generalized Laplacian. The generalized Laplacian $\Delta_{\alpha M}$ is an extension of the standard

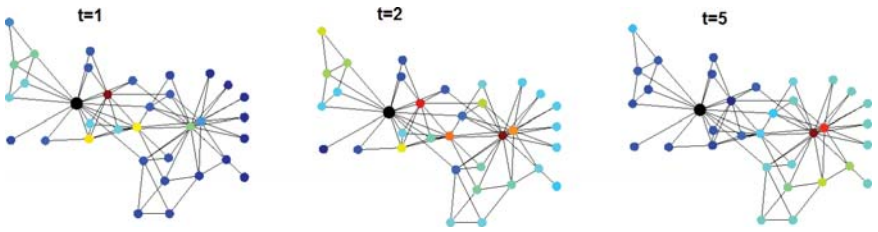


Figure 2.1. *Effect of time, t, on rate of diffusion for 3 values of t on the karate graph. Similarity decreases as color changes from blue to red spectrum.*

Laplacian with an additional parameter α and mathematically represented as $\Delta_{\alpha M} = I - \alpha M$. It immediately follows that, the generalized rate of diffusion matrix is the unscaled rate of diffusion matrix for $\alpha = 1$.

I discuss the implications of α and the moment parameter n in network graphs and provide additional insights for discrete geometric manifolds.

- First, $\Delta_{\alpha M} e^{-t\Delta_{\alpha M}}$ term in the integrand is the negative differential of the generalized diffusion equation $\frac{dp_t}{dt} = -\Delta_{\alpha M} p_t$. t^n scales the rate of diffusion between all pairs of nodes, i.e., large n weighs the diffusion rate at large times more heavily relative to small n. In effect, diffusions occurring over long time windows are weighed more for high values of n and are equally weighted for $n = 0$, hence the moment parameter n extracts scale. If I consider the rate between a node pair to be a signal over time, the role of n is to filter frequencies, i.e., higher frequencies are progressively suppressed by increasing n [13].

- Each element of $e^{-t\Delta_{\alpha M}}$ can be expanded using the Maclaurin series of $e^{-t(I-\alpha M)}$ as:

$$e^{-t\Delta_{\alpha M}}(i,j) = e^{-t} \sum_{k=0}^{\infty} \alpha^k M^k(i,j) \frac{(t)^k}{k!} \tag{2.11}$$

  $M^k(i,j)$ can be interpreted as the sum of all random walks of length k joining points i and j. For $0 \leq \alpha < 1$, $e^{-t\Delta_{\alpha M}}$ is probabilistically interpreted as a lazy random walk where with α probability a random walkers performs a normal random walk and with $(1 - \alpha)$ probability the walker leaves the current node and teleports to a

random node, without regard to graph structure. In Equation (2.12), paths of length k get weight α^k and hence shorter paths (say k_1) are weighed more than longer ones (say k_2) ($\alpha^{k_2} < \alpha^{k_1}$). For $-1 \leq \alpha < 0$, Equation (2.12) can no longer be interpreted probabilistically but instead, interpreted as a weighted signalling process on a graph. This is seen by expanding $e^{-t(I-\alpha M)}$ for $\alpha < 0$ as:

$$\begin{aligned} e^{-t\Delta_{-\alpha M}}(i,j) &= \sum_{k=0}^{\infty}(I+\alpha M)^k(i,j)\frac{(-t)^k}{k!} \\ &= \sum_{k=0}^{\infty} w_{sig}(I+\alpha M)^k(u,v) \end{aligned} \tag{2.12}$$

where w_{sig} are the weights of the signals transferred between nodes. $(I+\alpha M)^k$ is the random walk counterpart of the signalling graph in [103] with additional parameter α controlling the influence of signals based on path length k. From a signal processing viewpoint, the role of $\alpha(-1 \leq \alpha < 1)$ is to shift the scale of analysis as it implicitly shifts the eigenvalues in the spectral decomposition of $e^{-t\Delta_{\alpha M}}$. To see this, write $I - \alpha M$ as $\alpha(\beta I + \Delta_m)$ where $\beta = \frac{1-\alpha}{\alpha}$. The eigenvalues of $(\beta I + \Delta_m)$ are $(\beta I + \Lambda)$, and hence, $e^{-t\Delta_{\alpha M}}$ is spectrally decomposed as $D^{-1/2}\Phi e^{-\alpha t(\beta I+\Lambda)}\Phi^T D^{1/2}$. Expanding,

$$e^{-t\Delta_{\alpha M}}(i,j) = \sum_{k\geq} e^{-\alpha t(\beta+\lambda_k)}\Phi_k(i)\Phi_k(j)\sqrt{\frac{d_j}{d_i}} \tag{2.13}$$

- Allowing dual parameters α and n to control scale provides greater flexibility and reveals different multiscale information (see [58]) about node connectivity. Integrating over the entire time interval makes the resulting matrix robust to noise and small topology changes.

Figure 2.1 displays the effect of t and Figure 2.2 shows the effect of α on the negative rate of diffusion from the black node in the karate graph, where blue shades indicate similarity and red shades indicate dissimilarity. I see that nodes close to the black node progressively become similar as time increases suggesting that the black node reaches a local equilibrium with these nodes quicker than the rest of the nodes, and hence, becomes aware of its true neighbourhood. The effect of decreasing α is complimentary, in the sense

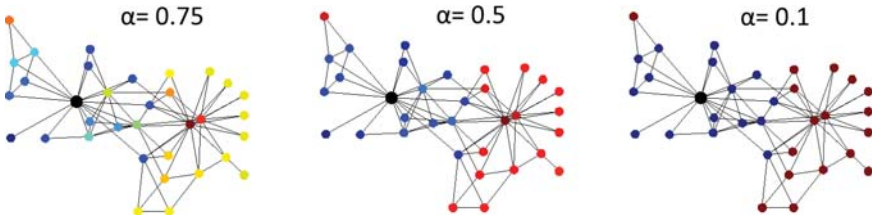


Figure 2.2. *Effect of teleportation parameter, α, on rate of diffusion for 3 values of α on the karate graph. Similarity decreases as color changes from blue to red spectrum.*

that the black node becomes progressively unaware of nodes further away from its neighbourhood as α decreases. Hence, allowing dual parameters α and t to control diffusion provides greater flexibility and reveals multiscale information about node connectivity.

The contribution here is the generalization of popular graph theoretic matrices. These matrices are derived as special cases of the spectral solution of Equation (2.10). I first prove the general solution to the equation in the next subsection. In subsequent subsections I prove the interrelationship between the general solution and different network science tools, as well as geometric distances.

Expectations of rate of diffusion matrix:I begin this section with the proof for continuous time random walks

Theorem 2.1.1 *The solution of*

$$\tau^{\alpha n} = \int_{t=0}^{\infty} t^n [\Delta_{\alpha M} e^{-t\Delta_{\alpha M}}] dt \tag{2.14}$$

exists for all real $n > 0$ and $-1 \leq \alpha \leq 1$ and is given by:

$$\tau^{\alpha n}(i,j) = \begin{cases} \Gamma(n+1) \sum\limits_{k\geq 2} \frac{1}{\lambda_k{}^n} \Phi_k(i)\Phi_k(j)\sqrt{\frac{d_j}{d_i}} & \textit{if} \quad \alpha = 1 \\ \frac{\Gamma(n+1)}{\alpha^n} \sum\limits_{k\geq 1} \frac{1}{(\beta+\lambda_k)^n} \Phi_k(i)\Phi_k(j)\sqrt{\frac{d_j}{d_i}} & \textit{if} \quad -1 \leq \alpha < 1 \end{cases}$$

Proof The term $\Delta_{\alpha M} e^{-t\Delta_{\alpha M}} dt$ can be written as $[e^{-t\Delta_{\alpha M}} - e^{-(t+dt)\Delta_{\alpha M}}]$ using Taylor series expansion. Using Equation (2.6), the eigenvalue λ_1 and corresponding eigenvector

drop out of the summand for $\alpha = 1$. Using the orthogonality of eigenvectors and evaluating the integral, τ^n simplifies to,

$$\tau^n(i,j) = \Gamma(n+1)\sum_{k\geq 2}\frac{1}{{\lambda_k}^n}\Phi_k(i)\Phi_k(j)\sqrt{\frac{d_j}{d_i}} \tag{2.15}$$

where Γ is the gamma function. Verify using Equation (2.13) that all eigenvalues of $e^{-t\Delta_{\alpha M}}$, hence all eigenvalues of $\Delta_{\alpha M}$ have magnitude greater than 0 for $-1 \leq \alpha < 1$. The matrix $\Delta_{\alpha M}$ is invertible for all values of α in the range $[-1, 1)$ and the solution for Equation (2.14) exists. Using the orthogonality of eigenvectors and evaluating the integral, $\tau^{\alpha n}$ is given by:

$$\tau^{\alpha n}(i,j) = \frac{\Gamma(n+1)}{\alpha^n}\sum_{k\geq 1}\frac{1}{(\beta+\lambda_k)^n}\Phi_k(i)\Phi_k(j)\sqrt{\frac{d_j}{d_i}} \tag{2.16}$$

Note the starting index k is different for the two equations above, based on the value of α. The eigenvectors are in decreasing smoothness order as I arrange the eigenvalues in increasing order, and hence, the order n scales the smoothness while filtering out small '*frequencies*'. Following the same reasoning, the smaller the α value in the range $[-1, 1)$, the larger the magnitude of β and hence α performs a '*spectra or scale shift*' [57]. ■

Figure 2.3 demonstrates this phenomenon for three functions: $f(t) = 1$, $f(t) = t^2$, $f(t) = sin(\pi t)$ convoluted with h_t for two pairs of nodes, $[34, 33]$ displayed in red and $[34, 1]$ displayed in blue for the karate graph. I observe that the convoluting function in effect changes the behavior of the affinity between nodes with different functions leading to different interpretations of the resulting affinity.

An important property demonstrated in the above theorem is that the rate of diffusion matrix $\Delta_M e^{-t\Delta_M}$, is integrable for all positive values of n over the time domain, whereas, the diffusion matrix $e^{-t\Delta_M}$ is not, due to the presence of eigenvalue $\lambda = 0$. Indeed, matrix surgical operations like pseudo inverse overcome this limitation of the diffusion matrix whereas, my results follow from standard algebraic operations on the rate of diffusion matrix. I next prove it for the discrete random walks.

The formulation equivalent to Equation (2.14) for discrete walks is:

$$\hat{\tau}^{\alpha n} = \sum_{t=1}^{\infty}(x)_n[(\alpha M)^{t-1} - (\alpha M)^t] \tag{2.17}$$

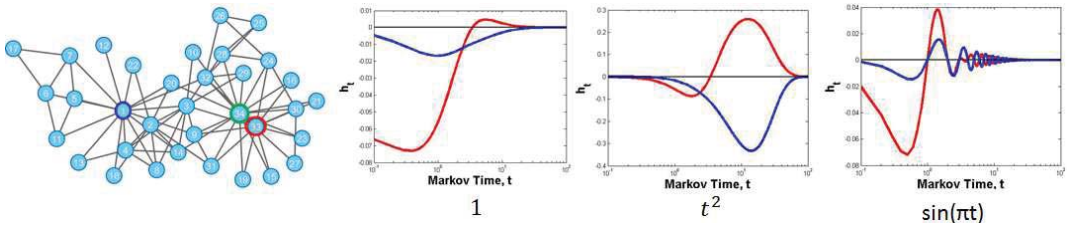


Figure 2.3. *Effect of the convolution function $f(t)$ on rate of diffusion between 2 node pairs:* $[34, 33]$ *in red and* $[34, 1]$ *in blue. Three functions considered are* $1, t^2, sin(\pi t)$.

where $\hat{\tau}^{\alpha n}$ denotes the operator defined over the function $(x)_n$, also called the Pochhammer symbol or rising factorial and represented as $(x)_n = x(x+1)(x+2)...(x+n-1)$. As M^t denotes the transition matrix after time t, $M^{t-1} - M^t$ denotes the change in transition probability after one time step and equivalent to rate of diffusion in the continuous time limit. More rigorously, the term in parenthesis in Equation (2.17) can be written as $[e^{-t\Delta_{\alpha M}} - e^{-(t+dt)\Delta_{\alpha M}}]$, where I have used the fact that $(\alpha M)^t$ or $(I - \Delta_{\alpha M})^t$ converges to $e^{-t\Delta_{\alpha M}}$ for continuous time walks (see [12] for derivation). Taylor series expansion of the derived $I - e^{-\Delta_{\alpha M} dt}$ term in $e^{-t\Delta_{\alpha M}}(I - e^{-\Delta_{\alpha M} dt})$ results in the rate of diffusion. An alternative interpretation of the uv^{th} term in $M^{t-1} - M^t$ is that it denotes the probability that a random walker is at node v in the t^{th} time-step starting from node u at time $t = 0$ (see page 159 [21] for case $(x)_1$). Parameters n in $(x)_n$ and α play a similar role of exaggerating/suppressing pathways (or '*faraway*' nodes), as in the continuous time case. Recall in Equation (2.6), eigenvalue 1 and the corresponding eigenvector appear as π_v in both M^t as well as M^{t-1}. Substituting the spectral representation of M in the Equation (2.17) for $\alpha = 1$, removes the eigenvalue 1 (of M) and the corresponding eigenvector from the summand, enabling me to use $I - M$ in an invertible setting (removal of eigenvalue 1 from M implies removal of eigenvalue 0 from $I - M$, hence killing its null space). Using the

sum of infinite series $\sum_{k=1}^{\infty} k(k+1)...(k+n-1)[x^{k-1}-x^k] = n!/(1-x)^n$ and spectral decomposition of $(I-M)^{-n}$ or Δ_M^{-n}, the operator is represented as

$$\hat{\tau}^n(i,j) = n! \sum_{k\geq 2} \frac{1}{{\lambda_k}^n} \Phi_k(i)\Phi_k(j)\sqrt{\frac{d_j}{d_i}} \tag{2.18}$$

Following a similar derivation as in the continuous time case, the representation for $0 < \alpha < 1$ is

$$\hat{\tau}^{\alpha n}(u,v) = \frac{n!}{\alpha^n} \sum_{k\geq 1} \frac{1}{(\beta+\lambda_k)^n} \Phi_k(i)\Phi_k(j)\sqrt{\frac{d_j}{d_i}} \tag{2.19}$$

Thus, the use of Pochhammer symbol in lieu of t^n results in the same resultant kernel as in the continuous case. I now proceed to showing the relationship of my formulated similarity matrix $\tau^{\alpha n}$ and several existing graph theoretic matrices commonly used in literature.

Theoretical Properties: The formulated kernels are robust to noise, provably converge on finite graphs, are positive definite and can be efficiently approximated.

Scale invariance, robustness and positive definite: For scale invariance, the kernel weighting function on the eigenvectors must satisfy $K(\gamma/\omega^2) = \omega^{2-2/p}K(\gamma)$. It can be verified that the weighting function $K(\beta+\lambda) = (\beta+\lambda)^{-1/p}$ satisfies this property and hence the kernels are scale invariant. The heat kernel is robust to noise as it is a weighted sum over all connecting pathways between nodes,and hence, is frequently used in biological applications wherein the networks are inherently noisy. Inductively, the rate of diffusion is also robust to noise. $\tau^{\alpha n}$ is derived by integrating over the entire time interval and further suppresses noise and small topological changes in the graph structure. $D^{1/2}\tau^{\alpha n}D^{-1/2}$ simplifies to $\frac{1}{(\beta+\lambda_k)^n}\Phi_k(i)\Phi_k(j)$ and is positive definite. Pre and post multiply with positive diagonal matrices $(diag(\pi))^{-1/2}$, $(diag(\pi))^{1/2}$ respectively, results in the kernel $\tau^{\alpha n}$ being positive definite.

Convergence and complexity: The convergence of $\tau^{\alpha n}$ over finite graphs where n is any positive real number can be verified using the property that the integral $\int_0^\infty e^{-t}t^{z-1}dt$ (Gamma function) is absolutely convergent for all z with $Re(z) > 0$ after doing a variable change $t' = t/\lambda_i$. As each term in the spectral representation converges and the representation is finite, it follows that the kernel converges. $\tau^{\alpha n}$ can be computed either

using the complete eigen spectrum $\mathcal{O}(|V|^2)$ for sparse graphs or efficiently approximated using the first k eigenvectors $\mathcal{O}(|V|^{3/2})$) due to the decreasing smoothness property of the eigen spectrum.

Generalization of network science kernels: In this subsection I discuss how the rate of rate of diffusion matrix generalizes several network science tools. Geometric manifolds are taken up in the next section. I discuss the relationship of $\tau^{\alpha n}$ to PageRank, harmonic and deconvolution kernels.

PageRank: PageRank is the original algorithm used by Google to rank search results on the world wide web [55]. Substituting $n = 1$ in Equation (2.18) gives:

$$\tau^{\alpha}(i,j) = \frac{1}{\alpha^n} \sum_{k \geq 1} \frac{1}{\beta + \lambda_k} \Phi_k(i) \Phi_k(j) \sqrt{\frac{d_j}{d_i}} \tag{2.20}$$

Which is precisely the representation of the PageRank kernel for general α. The original PageRank algorithm uses $\alpha = 0.85$ to rank search results on the web. The PageRank vector is retrieved my multiplying τ^{α} by a vector of ones $\mathbb{1}$, i.e.,

$$R = \tau^{\alpha} \mathbb{1} \tag{2.21}$$

where R is a $N \times 1$ vector of PageRanks. Recently, the heat kernel was proposed as the pagerank of a graph. The recurrence relationships for PageRank and heat kernel pagerank are given as

$$\begin{aligned} pr_{\alpha,f} &= \alpha f + (1-\alpha) pr_{\alpha,f} M \\ \frac{pr_{t,f}}{dt} &= -pr_{t,f}(I - M) \end{aligned} \tag{2.22}$$

Where $pr_{\alpha,f}$ is the PageRank, $pr_{t,f}$ is the heat kernel pagerank and f is the preference vector. The PageRank is a geometric sum whereas the heat kernel pagerank is an exponential of random walks. This is verified by their equivalent representations in terms of the transition matrix, M as:

$$\begin{aligned} pr_{\alpha,f} &= \alpha \sum_{k=0}^{\infty} (1-\alpha)^k f M^K \\ \frac{pr_{t,f}}{\delta t} &= e^{-t} \sum_{k=0}^{\infty} \frac{t^k}{k!} f M^k \end{aligned} \tag{2.23}$$

A geometric series decays very slowly whereas an exponential series decays too quickly. The kernel $\tau^{\alpha n}$ finds a middle ground for pagerank values as an arithmetic-geometric sum over three parameters α, n, f:

$$pr_{\alpha,n,f} = \alpha \sum_{k=0}^{\infty} (k+1)(k+2)...(k+n-1)(1-\alpha)^k f M^K \tag{2.24}$$

where I use the relationship

$$\sum_{k=0}^{\infty} (k+1)(k+2)...(k+n-1)(1-\alpha)^k f M^K = \frac{(n-1)!}{(I-\alpha M)^n} \tag{2.25}$$

Using the spectral decomposition of M in Equation (2.4), it can be verified that $(I-\alpha M)^{-n}$ takes the same form as $\tau^{\alpha n}$ upto a constant multiplicative factor. Whence my formulation shows the transition from geometric maps to exponential maps of the random walk.

Fundamental matrix and harmonic kernel: In the Markov chain theory, the fundamental matrix $\mathcal{Z}^0$ for continuous time random walk is defined as

$$\mathcal{Z}(i,j) = \int_0^{\infty} (p_t(i,j) - \pi_j) dt \tag{2.26}$$

This matrix is very useful for a wide array of calculations like expected number of visits to a state, expected return and access times [92]. It can be verified that $\mathcal{Z}$ reduces to the same spectral form as $\tau^{\alpha n}$ for $\alpha = 1$ and $n = 1$ (see page 107 [92]). A kernel is a positive semi definite similarity matrix and is a starting point of several algorithms in the machine learning literature. Recently, harmonic kernels $\mathcal{H}$ have received much attention in shape matching and ranking algorithms and are given by:

$$\mathcal{H}^n(i,j) = \sum_{k \geq 2} \frac{1}{\lambda_k^n} \Phi_k(i) \Phi_k(j) \tag{2.27}$$

for integer n. Pre and post multiplying Equation (2.15) by scaling matrices $D^{1/2}$ and $D^{-1/2}$ respectively outputs the harmonic kernels for any positive real n up to a constant. The associated distance metrics and embeddings also serve a wide range of applications and directly derivable using $\mathcal{H}$. The reader is asked to refer to the next subsection for a detailed discussion demonstrating the interrelationship of different distance metrics like commute time, biharmonic distance etc via my formulation.

Deconvolution kernel: Deconvolution is a recently proposed approach for removing indirect effects from correlated information [59]. The governing equation for network deconvolution is:

$$G_{obs} = \Omega + \Omega^2 + \Omega^3 + \cdots \tag{2.28}$$

where G_{obs} is the observed matrix and Ω is a matrix of direct dependencies, referred to as the deconvolution kernel. The solution to the deconvolution kernel Ω is given by:

$$\Omega(i,j) = \sum_{k \geq 1} \frac{\hat{\lambda}_k}{1+\hat{\lambda}_k} \hat{\Phi}_k(i) \hat{\Phi}_k(j) \tag{2.29}$$

where $\hat{\lambda}_k$ and $\hat{\Phi}_k$ are the k^{th} eigenvalue and eigenvector of the observed matrix G_{obs} with both direct and indirect effects. Now consider $\hat{\tau} = -D^{1/2}\tau^{\alpha n}D^{-1/2}$ for $\alpha = -1$ and $n = 1$. Substituting $\beta = \frac{1-\alpha}{\alpha} = -2$ I get

$$\begin{aligned}
\hat{\tau}(i,j) &= \sum_{k \geq 1} \frac{1}{(-2+\lambda_k)} \Phi_k(i)\Phi_k(j) \\
&= \sum_{k \geq 1} \frac{1}{(-2+1-\lambda'_k)} \Phi_k(i)\Phi_k(j) + I - I \\
&= \sum_{k \geq 1} \Big(\frac{-1}{(1+\lambda'_k)} + 1 \Big) \Phi_k(i)\Phi_k(j) - I \\
&= \sum_{k \geq 1} \frac{-1+1+\lambda'_k}{(1+\lambda'_k)} \Phi_k(i)\Phi_k(j) - I \\
&= \sum_{k \geq 1} \frac{\lambda'_k}{(1+\lambda'_k)} \Phi_k(i)\Phi_k(j) - I \\
&= \hat{\Omega} - I
\end{aligned} \tag{2.30}$$

It is clear that $\hat{\Omega}$ is structurally equivalent to Ω by considering the eigenvalues λ'_k associated with the normalized transition matrix $N_M = D^{-1/2}MD^{1/2} = D^{-1/2}AD^{-1/2}$ instead of G_{obs}. Inferring the direct from indirect effects between node pairs is not affected by the identity matrix $-I$, and hence, Ω is the deconvolution kernel for N_M. Note, [59] linearly scales the matrix so as to ensure the maximum magnitude of eigenvalues is less than 1 whereas I scale the observed matrix by suitable exponents of D so as to ensure the same criterion. Thus, $\hat{\tau}$ is the deconvolution kernel for N_M and varying α in the range $[-1, 0)$

outputs a generalized convolution kernel controlling the suppression of indirect effects, and immediately applicable to network deconvolution. The strong assumption that the powers of Ω all contribute equally to the observed matrix in the original formulation, is effectively relaxed to:

$$G_{obs} = \alpha\Omega + \alpha^2\Omega^2 + \alpha^3\Omega^3 + \cdots \tag{2.31}$$

where α is the relaxation parameter in my formulation.

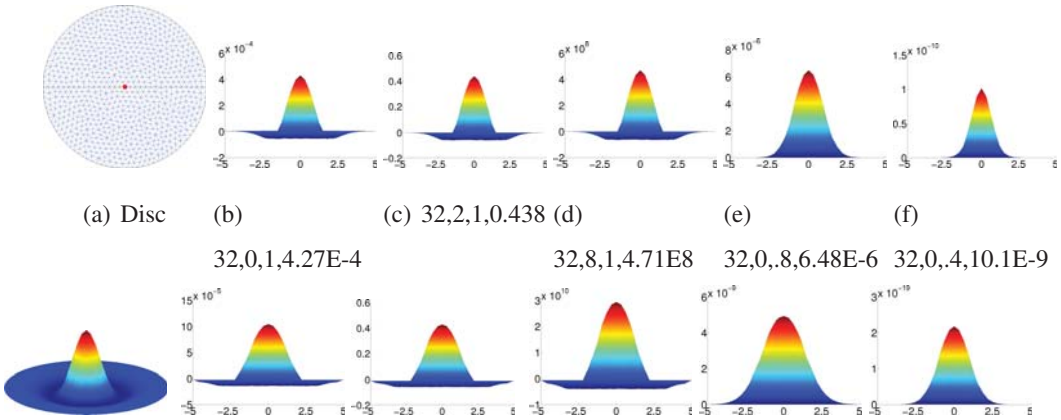
The results derived using the rate of diffusion matrix are also derivable using the discrete counterpart of the rate of diffusion matrix (see [58]). Further, t^n in Equation (2.10) can be replaced by a general functions $f(t)$ which may be suitable for specific applications. Overall, my key observation is that interpreting the parameters α and n as frequency shift and frequency filter respectively, generalizes several existing similarity matrices and can potentially guide creation of new similarity matrices suitable for the particular network to be analyzed.

2.1.3 Multiscale kernels using random walks on geometric manifolds

As discussed previously, the formulation of the multiscale kernels using the rate of diffusion matrix is based on two parameters- a moment parameter n and an influence parameter α, given by the following equation:

$$\tau^{\alpha n} = \int_{t=0}^{\infty} t^n [\Delta_{\alpha M} e^{-t\Delta_{\alpha M}}] dt \tag{2.32}$$

I focus my attention to interpreting the same on geometric manifolds. Although t^n can be replaced by general functions of t, I restrict my attention to powers in t (n is a positive fraction), and term $\tau^{\alpha n}$ as the moment time operator. The implication of this functional form was discussed at the beginning of this section for general graphs. I now turn my attention to geometric manifolds. Loosely speaking, n and α produce similar effect of exaggerating/suppressing '*faraway*' nodes. However, they do so in very different ways, which I illustrate by plotting the values of $t^n[\Delta_{\alpha M} e^{-t\Delta_{\alpha M}}]$ for different values of $[t, n, \alpha]$, from the center of a disc in Figure 2.4. Plot 2.4g shows the rate of diffusion values to all points from



(a) Disc (b) 32,0,1,4.27E-4 (c) 32,2,1,0.438 (d) 32,8,1,4.71E8 (e) 32,0,.8,6.48E-6 (f) 32,0,.4,10.1E-9

(g) 32,0,1,4.27E-4 (h) 64,0,1,1.04E-4 (i) 64,2,1,0.427 (j) 64,8,1,2.93E10 (k) 64,0,.8,4.93E-9 (l) 64,0,.4,2.1E-19

Figure 2.4. *(a) Disc of radius 5. (g) 3D plot of rate of diffusion from center (red point) of disc viewed as height function and color coded (rate decreases as color changes from red to blue). (b)-(f) and (h)-(l): 2D plot of function* $t^n[\Delta_{\alpha M} e^{-t\Delta_{\alpha M}}]$ *for different values of* t, n, α*, viewed as height field (function value) along vertical axis and distance from center along horizontal axis. The quadruple indicates* $[t, n, \alpha, \varsigma]$ *values where* ς *is the maximum value of function indicative of weightage in the integral.*

the center as a color plot (red implies higher values) and height field from the base of the disc. Note, the flow conservation property holds $\forall t > 0$. Formally, $\sum_j \Delta_M e^{-t\Delta_M}(i,j) = 0$ $\forall v \in V$. This can be verified by writing $\Delta_M e^{-t\Delta_M} dt$ as $(e^{-t\Delta_M} - e^{-(t+dt)\Delta_M})$. As all rows of $e^{-t\Delta_M}, e^{-(t+dt)\Delta_M}$ sum to 1 (Property 2), flow conservation follows. It does not hold in general for $\alpha \neq 1$. Plots 2.4b, 2.4h display the rate of diffusion at times $t = [32, 64]$ with almost equal maximum values $\approx [4 \cdot 10^{-4}, 10^{-4}]$. Weighting the rate of diffusion by t^n produces similarly shaped scaled plots for the same time, as t^n is a multiplicative factor to the term in parenthesis. It can be visually verified by comparing Plot 2.4b to Plots 2.4c, 2.4d and Plot 2.4h to Plots 2.4i, 2.4j. However, t^8 weighting produces two orders of magnitude difference in the maximum value $\approx [4.7 \cdot 10^8, 2.9 \cdot 10^{10}]$ at the two times. This validates the claim that parameter n penalizes the rate of diffusion at long time scales more than short ones in the integral, and large values of n provide global information about the

shape. The influence parameter α affects the diffusion rate on the disc in two ways. Firstly, it suppresses the diffusion to nodes further away from the center, consequently suppressing the rate (notice the progressive squeezing in Plots 2.4b to 2.4e to 2.4d and Plots 2.4h to 2.4k to 2.4l as α decreases from 1 to 0.8 to 0.4). Secondly, α decreases the magnitude of rate of diffusion, affecting the magnitude at long time scales more than short ones. The maximum value drops from $\approx 6 \cdot 10^{-6}$ (Plot 2.4e) to $\approx 10^{-10}$ (Plot 2.4f) for $t = 32$, but drops from $\approx 5 \cdot 10^{-9}$ (Plot 2.4k) to $\approx 2 \cdot 10^{-19}$ (Plot 2.4l) at larger time $t = 64$. This can be understood by considering the equivalent formulation for discrete random walks, Equation (2.17), and that the term in parenthesis can be written as $\alpha^t[M^{t-1}/\alpha - (M)^t]$, with α appearing as a multiplicative factor with time t as exponent. Hence, the notion of '*faraway*' nodes for parameters n, α can be stated as:

- For the moment parameter n, '*faraway*' is related to the rate of diffusion before converging to the equilibrium distribution, and hence, (loosely) tied to time scales.
- For the influence parameter α, '*faraway*' is related to the separation between the source and destination nodes, and hence, (loosely) tied to distance scales.

I do not treat time and distance as independents (both parameters influence time and distance) but attach the notion of time to n and distance to α, as it makes intuitive sense and for ease of comprehension. I demonstrate subsequently that each reveal useful multiscale information about the triangulated mesh. To sum it up, the basic intuition in defining these operators is that the '*rate of diffusion is informative for understanding shapes*'. This point is illustrated further in Figure 2.5 where I compare the diffusion kernel to the rate of diffusion kernel for $\alpha = 1$. The key separating feature is that the smallest eigenvalue and its corresponding (uninformative) constant eigenvector does not appear in the rate kernel. This is true as $\Delta_M e^{-t\Delta_M} dt$ can be written as $(e^{-t\Delta_M} - e^{-(t+dt)\Delta_M})$ and using Equation (2.6). This is beneficial as it suppresses the fastest decaying eigenvalue (λ_1) and most global eigenvector (Φ_1). Hence the rate kernel is '*more local*' relative to the heat kernel and does not converge to the uniform distribution, as the constant eigenvector which dominates at long time is removed. The isocontours of diffusion rate are more intuitive, as they follow

the shape locally and the derived signatures are discriminative at all scales, unlike the HKS. These claims are validated in Figure 2.5. The Rate of Heat Kernel Signature (RHKS) is defined similar to HKS on $\Delta_M e^{-t\Delta_M}$ instead of $e^{-t\Delta_M}$. Plots 2.5c, 2.5g indicate robust matching using RHKS to green anchor point and plots 2.5d, 2.5h indicate discriminative signatures at all scales unlike HKS. The isolines and color plots of the rate of diffusion from the anchor point are more localized at small times (plot 2.5a vs. 2.5e); also the isolines follow similar pattern for time $t = 2$. The color plots, plot 2.5b vs. 2.5f, are starkly different and can be explained by considering $1/\lambda_2$ value which is ≈ 2. $1/\lambda_2$ is the mixing time of diffusion and plot 2.5b clearly indicates that the rate of diffusion is small on the entire horse model, hence diffusion has nearly converged to equilibrium. Thus, I validate my claim that the rate of diffusion is equally informative and more discriminative, compared to the diffusion kernel. Having defined the operator, I establish desired symmetric and positive-definite kernels, and hence, develop the distance metric and the corresponding embedding.

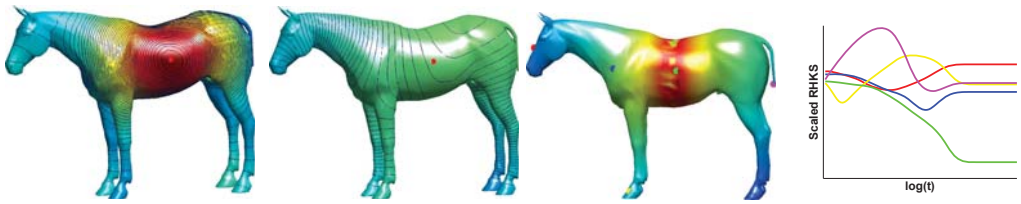
Kernel computation: I previously computed the multiscale operator using the eigen spectrum of the transition matrix and I now show that the result is intimately connected to the discrete Green's function, and hence the commute time and biharmonic kernel. Further, I use the kernel distance to evaluate pairwise distances and characterize the associated embedding. Recall that the spectral representation of $\tau^{\alpha n}$ simplifies to,

$$\tau^n(i,j) = \Gamma(n+1) \sum_{k \geq 2} \frac{1}{{\lambda_k}^n} \Phi_k(i) \Phi_k(j) \sqrt{\frac{d_j}{d_i}} \tag{2.33}$$

or $\tau^n = \Gamma(n+1) D^{-1/2} \hat{\Phi} \hat{\Lambda}^{-n} \hat{\Phi}^T D^{1/2}$ for $\alpha = 1$ where $\hat{\Lambda}$ and $\hat{\Phi}$ indicate the removal of the 0 eigenvalue and the corresponding eigenvector. Further,

$$\tau^{\alpha n}(i,j) = \frac{\Gamma(n+1)}{\alpha^n} \sum_{k \geq 1} \frac{1}{(\beta + \lambda_k)^n} \Phi_k(i) \Phi_k(j) \sqrt{\frac{d_j}{d_i}} \tag{2.34}$$

or $\tau^{\alpha n} = \Gamma(n+1) \alpha^{-n} D^{-1/2} \Phi (\beta + \Lambda)^{-n} \Phi^T D^{1/2}$ for $0 < \alpha < 1$. From these equations, it is easy to verify that the symmetric matrix $D^{1/2} \tau^{\alpha n} D^{-1/2}$ is positive-definite and so is the matrix $\tau^{\alpha n} (diag(\pi))^{-1}$ obtained by pre and post multiplying by $(diag(\pi))^{-1/2}$, where $diag(\pi)$ is a diagonal matrix of stationary probabilities. I use this kernel to define kernel



(a) Diffusion rate at time=0.4 (b) Diffusion rate at time=2 (c) Matching to green sphere (d) RHKS of anchor points

(e) Diffusion at time=0.4 (f) Diffusion at time=2 (g) Matching to green sphere (h) HKS of anchor points

Figure 2.5. *Comparison of Rate of Diffusion (Top) vs Diffusion (Bottom). (a)-(b),(e)-(f) show isocontours and color plots of the magnitude of diffusion/ rate of diffusion emanating from red sphere at different time values. Magnitude decreases as color changes from red to blue. At small times isolines are localized and at large times the color plot is intuitive for rate of diffusion. (c),(g) display color plots of difference between scaled signatures for point marked by green sphere and other points on model for same time range $[t_1, t_2]$. Difference increases as color changes from red to blue. Signatures derived from diffusion rate are informative. (d),(h) displays scaled HKS (diffusion) and RHKS (rate of diffusion) vs logarithmic time scale for 5 color coded points on model in (c)/(g). RHKS is discriminative at all scales.*

distances, as other mesh based distance metrics naturally arise from it. Note,for geometric manifolds D is a diagonal matrix with elements $d_i = \Upsilon A_i$.

Green's function: I define the derived kernels in terms of standard operators and show the generic nature of my method in reference to other popular kernels. As stated previously, the fundamental matrix $\mathcal{Z}^0$ for continuous time random walk is defined as

$$\mathcal{Z}^0_(i,j) = \int_0^\infty (p_t(i,j) - \pi_j)dt \tag{2.35}$$

The Green's function introduced in 1828 by George Green and its discrete counterpart associated with the Laplacian, introduced in a 2000 paper [104], have received much attention. Formally, it is the pseudo-inverse of the defined Laplace operator and can be more explicitly stated as

$$G_{ij}\Delta_{M(ij)} = \delta_{ij} - \frac{d_j}{vol} \tag{2.36}$$

$$\mathcal{G}_{ij}\mathcal{L}_{ij} = \delta_{ij} - \frac{\sqrt{d_i d_j}}{vol} \tag{2.37}$$

where G and $\mathcal{G}$ are the left inverse operators of Δ_M and $\mathcal{L}$ respectively. It can be verified that $\mathcal{Z}^0$ and G reduce to the same spectral form as τ^1 (see [104] and page 107 [105]). Hence the relationship between the defined kernel, fundamental matrix and Green's function using $G^n = \mathcal{G}^n\sqrt{\frac{d_j}{d_i}}$ can be verified to be: $\tau^n = \Gamma(n+1)(\mathcal{Z}^0)^n = \Gamma(n+1)G^n$. From here on, in order to maintain consistency, I use the Green's function for representation.

Distance computation: I use the kernel distance to evaluate pairwise distances between points on a mesh. The kernel distance is defined for positive definite matrices as $(\mathcal{D}_{ij}^{\alpha n})^2 = (\kappa_{ii} - 2\kappa_{ij} + \kappa_{jj})$, where κ is a positive definite kernel. Hence, the general expressions using the discrete Green's function for distance between two vertices u and v for the generated kernels are

$$(\mathcal{D}_{ij}^{\alpha n})^2 = vol^n\left(\frac{\mathcal{G}_{ii}^{\alpha n}}{d_i} + \frac{\mathcal{G}_{jj}^{\alpha n}}{d_j} - 2\frac{\mathcal{G}_{ij}^{\alpha n}}{\sqrt{d_i d_j}}\right) \tag{2.38}$$

$$(\mathcal{D}_{ij}^{\alpha n})^2 = vol^n\left(\frac{G_{ii}^{\alpha n}}{d_i} + \frac{G_{jj}^{\alpha n}}{d_j} - \frac{G_{ij}^{\alpha n}}{d_j} - \frac{G_{ji}^{\alpha n}}{d_i}\right) \tag{2.39}$$

where $vol^n = \Gamma(n+1)\alpha^{-n}vol$, $\mathcal{G}^{\alpha n} = \Phi(\beta+\Lambda)^{-n}\Phi^T$ or $\mathcal{G}^n = \hat{\Phi}\hat{\Lambda}^{-n}\hat{\Phi}^T$ as required, based on the parameters n and α. I use the relation $G^{\alpha n} = \mathcal{G}^{\alpha n}\sqrt{\frac{d_j}{d_i}}$ to derive the second relation from the first. In terms of spectral representation this can be written as

$$(\mathcal{D}_{ij}^{\alpha n})^2 = vol^n\sum_{j\geq 1}\frac{1}{(\beta+\lambda_j)^n}\left(\frac{\Phi_j(i)}{\sqrt{d_i}} - \frac{\Phi_k(j)}{\sqrt{d_j}}\right)^2 \tag{2.40}$$

Note that the sum starts from $i = 2$ for $\mathcal{D}_{ij}^{n}$ and $i = 1$ for $\mathcal{D}_{ij}^{\alpha n}$. Thus to calculate the exact pairwise distances, I need to find the full set of eigenvectors and corresponding eigenvalues for the normalized Laplacian. Using the decreasing smoothness property of the eigenvectors, multiscale distances can be approximated using only the first K eigenvectors and pro-

vide a trade-off between accuracy and computational time. I shall return to computational complexity later.

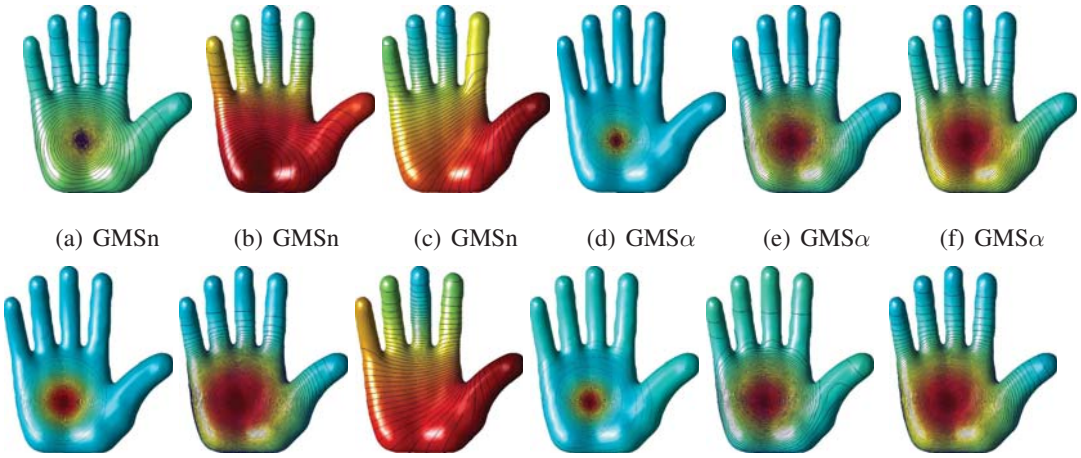
Kernel embedding: Having derived the kernel distances, I see that the embedding that preserves the n^{th} distance moment with parameter α is given by:

$$\Theta^{\alpha n} = \sqrt{vol^n}(\beta + \Lambda)^{-n/2}\Phi^T D^{-1/2} \tag{2.41}$$

Note that the $D^{-1/2}$ scaling appears due the scaling of the conformal Laplacian by area terms. To drive home this point, if the kernels were derived starting from the conformal Laplacian instead of the normalized Laplacian, the embedding would be: $\tilde{\Theta}^{\alpha n} = \sqrt{vol^n}(\tilde{\beta} + \tilde{\Lambda})^{-n/2}\tilde{\Phi}^T$, where $\tilde{\lambda}$ and $\tilde{\Phi}$ are the eigenvalues and eigenvectors of L_c and $\tilde{\beta}$ is the corresponding influence parameter. A similar embedding has recently been proposed in [16] for understanding manifolds, though with a very different approach. The difference in the embedding is due to the scaling of the operator. This can be understood by noting the relation between the Green's function of the induced subgraphs of $L_c, \mathcal{L}$ and Δ_M is $\bar{G} = GD^{-1} = D^{-1/2}\mathcal{G}D^{-1/2}$ where $\bar{G}$ is the Green's function of the conformal Laplacian [104]. As the Green's function of the induced subgraphs differ only by a scaling matrix, it is not that surprising that the two embeddings are equivalent upto a scaling matrix. Next I state properties of the kernels, distances, embeddings and show related applications.

Theoretical properties of kernels on manifolds: The formulated kernels are isometry, scale and tessellation invariant, can be made shape aware, insensitive to partial objects and noise based on the moment function and an additional influence parameter. Additionally, the corresponding kernel distances and embeddings have all the important properties, i.e., are metrics, convergent and fast to compute. I give a brief summary of the properties.

Multiscale: The central theme of the last section was the dual formulation of the multiscale operator using parameter α and moment function t^n. An intuitive interpretation of the α value is that the remainder $(1-\alpha)M$ probability translates to a random surfer model, where with probability $(1-\alpha)$, a random walker leaves the current node and teleports to another node, without regard to mesh connectivity. Hence, random contributions due to the teleportation cancel out in the kernel evaluation, resulting in increasingly local affinities with decrease in α. It is interesting to note that the α value is a fundamental component



(a) GMSn (b) GMSn (c) GMSn (d) GMSα (e) GMSα (f) GMSα

(g) HKS (h) HKS (i) HKS (j) Biharmonic (k) Biharmonic (l) Biharmonic

Figure 2.6. *Comparison of four multiscale kernels- moment, influence, heat and biharmonic kernels with local (left) to global (right) support for point at center of palm.*

in the celebrated PageRank algorithm and precisely performs the same function, albeit for ranking search queries [106]. On the other hand, t^n penalizes pairwise nodes which have a slow rate of convergence to the equilibrium distribution, i.e., nodes separated by longer path lengths as a function of time. Thus the kernels are dual multiscale, (loosely) over distance as well as over time scales with different multiscale behaviour, and is a key differentiation from other multiscale kernels which define multiscale behaviour over either time (heat kernel) or distance scales (multiscale biharmonic kernel). Figure 2.13 shows the multiscale behavior over moment n, influence α, the heat kernel over time and multiscale biharmonic kernel over the lasso constraint. I observe that the multiscale behavior over α is similar to the multiscale biharmonic kernel and the behavior over moment n is similar to the behavior over t in heat kernel. Multiscale over n has an interesting behavior. For large moments, the maximum affinity from the center of hand becomes skewed towards the thumb, which has an intuitive explanation. The operator weighs the change in transition probability and visual inspection of affinity gradient indicates that the diffusion from center of palm towards the thumb occurs over short time interval, whereas a random walker in the

central regions and other four fingers tend to get '*lost*', and hence diffusion occurs over a large time interval. It also indicates there are shorter pathways connecting the palm point to the thumb and relatively longer pathways connecting to the rest of the hand. The flip in the least affinity (blue color) from the middle finger to ring finger can also be explained similarly.

Scale, isometry and tessellation invariant: For scale invariance, the kernel weighting function on the eigenvectors must satisfy $K(\gamma/\omega^2) = \omega^{2-2/p}K(\gamma)$ [107]. It can be verified that the weighting function $K(\beta + \lambda) = (\beta + \lambda)^{-1/p}$ satisfies this property and hence the kernels are scale invariant. In Figure 3.29 the teddy model is scaled 100 times, however the coloring on the mesh is unaffected, validating this property. As the transition/rate matrices are derived from the discrete Laplace-Beltrami operator, calculated using the isometry invariant curvature normals, the derived kernels are also isometry invariant. Isometric deformation of the Victoria model in Figure 3.29 does not change the isocontours confirming this theoretical property. The defined transition (M) and rate matrices (Δ_M) implicitly account for area terms and hence tessellation invariant. The bunny model in Figure 3.29 shows the invariance of the isolines of the kernel as the model is simplified from 35000 to 1500 vertices. Additionally, invariance to noise is indicated in the dolphin model by adding gaussian noise (200%) proportional to the average edge length. The partial and full centaur models in Figure 3.29 display similar isocontours from fixed anchor point on the top of the human head. The partial centaur model was created by spectral bipartition into 2 segments, which is more natural than passing a cutting plane through the object.

Convergence: The convergence of the multiscale kernels over finite graphs for t^n where n is any positive real number can be verified using the property that the integral $\int_0^\infty e^{-t}t^{z-1}dt$ (Gamma function) is absolutely convergent for all z with $Re(z) > 0$ after doing a variable change $t' = t/\lambda_i$ (Note $\lambda_i > 0$). As each term in the spectral representation converges and the representation is finite, it follows that the kernel converges. However for continuous state space, the Green's function of a 2D surface would have logarithmic singularity along the diagonal and not defined [14].

Metric: The kernel $\tau^{\alpha n}(diag(\pi))^{-1}$ is positive definite as $D^{1/2}\tau^{\alpha n}D^{-1/2}$ is positive-definite and I pre and post multiply with a diagonal matrix $(diag(\pi))^{-1/2}$ which is a positive diagonal matrix. Because the multiscale distances are derived using a strictly positive definite kernel, it satisfies all the necessary conditions for it to be a valid metric, i.e., it is non-negative and the diagonal values vanish. The fact that it is symmetric follows from the observation that interchanging u and v does not change the distance calculation. Also $\mathcal{D}^{\alpha n}_{uv} = 0$ *iff* $u = v$ because if this were not true then $\Phi_i(u) = \Phi_i(v)\ \forall i$, as the eigenvectors form an orthonormal basis, all functions f over the surface would take the same value, reaching a contradiction.

Complexity: The multiscale kernels (Green's function) and hence distances can be computed either using the complete eigen spectrum $\mathcal{O}(|V|^2)$ or using the first K eigenvectors $\mathcal{O}(|V|^{3/2})$). An alternative approach is proposed in [14] for calculating on a set of vertices by solving a set of linear equations($\mathcal{O}(S|V|)$ where S is the size of the subset of vertices). The subset calculation is exact and the complexity can be reduced to almost linear time for a small subset ($\mathcal{O}(|V|)$ when $S << V$). However, such an approach suffers from the disadvantage that I require independent computations to find multiscale kernels with a different set of parameters. The eigenvectors and eigenvalues of the Laplace operator offer a common orthonormal basis and all multiscale kernels can be computed by changing the weightage (exponent and parameter β) of eigenvalues. I use the spectral approach and use the first 300 eigenvectors to approximate the kernel matrix. Hence the computation time is the same as for the approximate heat kernel evaluation.

2.1.4 Image segmentation using a random walk kernel

Random walk based approaches have emerged as a power tool for solving wide range of graph based problems, spanning manifold learning [75], ranking [108], shape analysis [109], etc. Diffusion can be viewed as a generalization of markov chains (or random walks) [76]. Random walk approaches and its variants have successfully been applied for seeded image segmentation [110], clustering [111] etc. Different from diffusion based

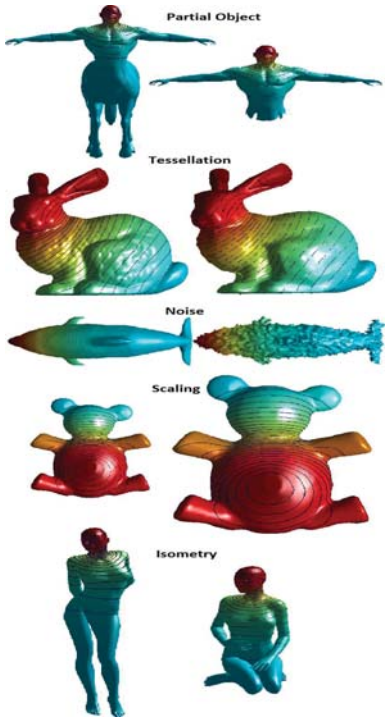


Figure 2.7. *Insensitivity of Multiscale Kernels to partial objects, tessellation, noise ,scaling and deformation for different mesh models from fixed anchor point.*

approaches, I use the dual paradigm of the rate of diffusion to motivate a novel global criterion based on *expected rate of diffusion*, which induces partitions due to its structural properties. I first introduce preliminaries and then derive its functional form, which I term the diffusion modulus criterion, $\mathcal{D}^n$. It is very similar in spirit to the kernels derived for networks and meshes. It is mathematically represented as:

$$\mathcal{D}^n = \sum_{uv} \left[\sum_{i \geq 2}^{N} \frac{1}{{\lambda_i}^n} \Phi_i(u) \Phi_i(v) \sqrt{d_v d_u} \right] \delta_{uv} \tag{2.42}$$

Where $\mathcal{D}^n$ is the expected rate of diffusion value to be maximized, n is a resolution parameter, N is the number of pixels/regions, u, v denote the pixels/region, λ_i is the i^{th} eigenvalue, $\Phi_i(u)$ is the u^{th} component of the i^{th} eigenvector of Laplacian $\mathcal{L}$ for graph $\mathcal{G}$, δ_{uv} is a delta function which is 1 if u, v are placed in the same segment and d_u is the degree of u^{th} pixel,

i.e., $d_u = \sum_{v=1}^{|V|} w(u,v)$ where $w(u,v)$ are the local affinities in the original graph G. In matrix terms, this can also be represented as $\mathcal{D}^n = H^T[D^{1/2}\hat{\Phi}\hat{\Lambda}^{-n}\hat{\Phi}^T D^{1/2}]H$, where D is a diagonal matrix of degrees d_u, H is a size $N \times K$ indicator matrix for K segments (Note each row of H sums to 1), and $\hat{\Lambda}$ and $\hat{\Phi}$ indicate the removal of the smallest eigenvalue and the corresponding eigenvector.

My first contribution is the formulation of the diffusion modulus criterion $\mathcal{D}^n$ motivated by the physical analogy of the rate of heat diffusion and its moments. $\mathcal{D}^n$ is a single parameter criterion over $n(\geq 0)$, akin to the single parameter normalized cut criterion (number of segments K). It naturally infers resolution (for lack of a better word and avoid confusion with scale in multi-scale frameworks). The expected rate of diffusion kernel, D_r^n (term in parenthesis in Equation (2.42)) as proved previously, can be viewed as generalization of the commute time and biharmonic kernel. I establish the structural equivalence of the D_r^n and modularity matrix in community detection [112] and prove that the 2-way normalized cut is a special case of the criterion.

Expected rate of diffusion or diffusion modulus: Recall that Δ governs diffusion, given by the equation $\frac{\delta H_t}{\delta t} = -\Delta H_t$ (H_t is the heat diffusion kernel). The solution to the diffusion equation is the exponential kernel $H_t = e^{-t\Delta}$ at time t [76].Using its spectral decomposition, the elements of the heat kernel can be spectrally represented as $H_t = D^{-1/2}\Phi e^{-t\Lambda}\Phi^T D^{1/2}$ or $H_t(u,v) = \sum_{i\geq 1} e^{-t\lambda_i}\Phi_i(u)\Phi_i(v)\sqrt{\frac{d_v}{d_u}}$ where Λ=diag$(\lambda_1, \lambda_2, ...\lambda_{|V|})$ is a diagonal matrix with the ordered set of eigenvalues, i.e., $0 = \lambda_1 \leq \lambda_2 ... \leq \lambda_{|V|} \leq 2$, $\Phi = (\Phi_1|\Phi_2|...|\Phi_{|V|})$ is a matrix with corresponding eigenvectors as columns. Formally, each element of the heat kernel $e^{-t\Delta}$ can be expanded using the Maclaurin series of $e^{-t(I-M)}$ as: $e^{-t\Delta}(u,v) = e^{-t}\sum_{k=0}^{\infty} M^k(u,v)\frac{(t)^k}{k!}$. Here $M^k(u,v)$ is the probability of transition from node u to v in k steps. As k spans from 0 to ∞ all path lengths between two nodes are considered in the probability calculation. The heat kernel conserves probability, i.e., $\sum_v H_{uv}^t = 1\ \forall v \in V, \forall t \geq 0$ [76]. *'Faithful'* partitions can be obtained by applying measures like the ratio or normalized cut to H_t as affinities are implicity learnt by considering all connecting pathways between two nodes. For example, ratio cut on H_t can be interpreted as the bipartition that minimizes the probability of a random walker switching

groups at time t. It is proved in [113] that Ncut minimizes the probability of a random walker evading a set (and its compliment) in its first step starting from the stationary distribution π ($\pi = \mathbf{1}D/vol$ where $\mathbf{1}$ is a vector with all coordinates 1 and $vol = \sum_u d_u$ called the volume of the graph). Using H_t naturally generalizes it to all steps for a given time t. Intuitively, time t controls the resolution and large t discovers long diffusion pathways between two nodes.

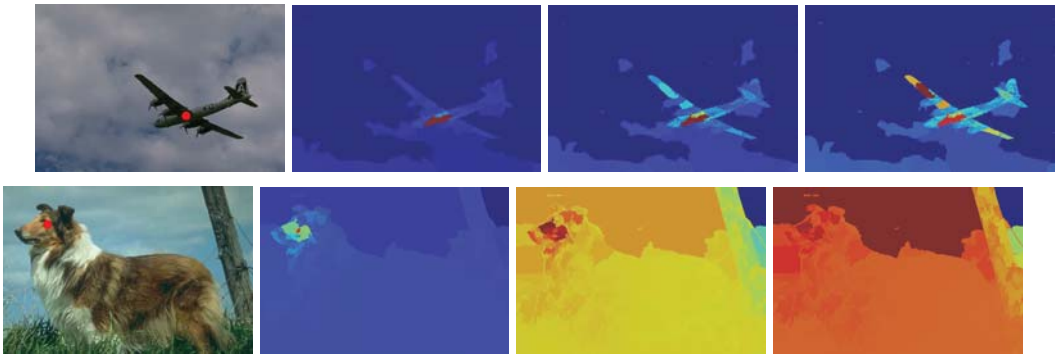


Figure 2.8. *Scale interpretation of parameter n. Affinity between exemplar region (red) in original image (left) and all other regions, for 3 values of n. n increases from left to right and similarity increases as color changes from blue to red.*

The physical analogy of the diffusion equation is that heat flows from a high temperature region to a low temperature region until equilibrium and flow rate is dependent on the negative temperature gradient. It follows the conservation of energy, i.e., heat is neither created nor destroyed. This information is captured by the differential of the heat operator $-\frac{\delta H_t}{\delta t}$ which is equal to $\Delta e^{-t\Delta}$, and I term it the rate of heat diffusion kernel RH_t. At $t = 0$ it becomes the random walk Laplacian Δ in which all rows sum to 0, i.e., flow is conserved. Indeed, flow conservation holds for all $t \geq 0$. To see this, write the net heat flow in time dt, $\Delta_M e^{-t\Delta_M} dt$, as $(e^{-t\Delta} - e^{-(t+dt)\Delta})$. All rows of $e^{-t\Delta}, e^{-(t+dt)\Delta}$ sum to 1 by probability conservation, and hence, flow conservation follows. Using $\Delta = D^{-1/2}\Phi e^{-t\Lambda}\Phi^T D^{1/2}$ and $e^{-t\Delta} = D^{-1/2}\Phi e^{-t\Lambda}\Phi^T D^{1/2}$, spectral representation of RH_t is $D^{-1/2}\Phi\Lambda e^{-t\Lambda}\Phi^T D^{1/2}$. Pre-multiplying it by D scales each row of RH_t to bring it to a favourable symmetric form

$RH_t = D^{1/2}\Phi\Lambda e^{-t\Lambda}\Phi^T D^{1/2}$. At $t = 0$ only the diagonal terms in Δ are positive. As t increases nearby nodes reach a local temperature equilibrium with positive $RH_t(u, v)$ and long scale flow comes into play. Hence, the intuition behind using the rate of heat diffusion for segmenting nodes is to minimize the rate of heat flow ($\frac{\delta H_t}{\delta t}$) within a group or equivalently maximize the negative temperature gradient ($-\frac{\delta H_t}{\delta t}$), i.e., associate nodes with positive affinities (in $\Delta e^{-t\Delta}$). The special structure of RH_t implicitly maximizes the rate of heat flow between groups, i.e., dissociates nodes with negative affinities and not explicitly included in the criterion. Thus the derived $\mathcal{D}^n$ from RH_t is not a *'cut'* criterion that seeks to *minimize the dissociation between* groups but a *'modulus'* criterion that seeks to *maximize the association within groups*. In principle RH_t can be directly used as a segmentation criterion. However, $\Lambda e^{-t\Lambda} \rightarrow \mathbf{0}$ as $t \rightarrow \infty$ where $\mathbf{0}$ is a matrix of all zeros, and hence, introduces the difficulty of choosing an appropriate time range in which segmentation is meaningful. I overcome this problem by defining the multi-resolution kernel, $\mathcal{D}_r^n$, as the n^{th} moment of $RH_t(u, v)$ with respect to time (see Theorem 1). Mathematically,

$$\mathcal{D}_r^n = \int_{t=0}^{\infty} t^n [D^{1/2}\Phi\Lambda e^{-t\Lambda}\Phi^T D^{1/2}]dt \tag{2.43}$$

This transformation using t^n scales the rate of diffusion and in effect aggregates the diffusion rate at all time scales. The effect of scaling is to penalize the rate of heat flow at long time periods. For example, $n = 2$ weights the diffusion rate between two nodes at large times more heavily than say $n = 1$ over the integral, whereas $n = 0$ weights all rates equally. Hence the moment extracts scale (see Figure 2.8). To simplify $\mathcal{D}_r^n$, rewrite it as $\mathcal{D}_r^n = \int_{t=0}^{\infty} D^{1/2} t^n [e^{-t\mathcal{L}} - e^{-(t+dt)\mathcal{L}}] D^{1/2}$. $e^{-t\mathcal{L}}$ can be expanded in spectral form to $\Phi_1(u)\Phi_1(v) + \sum_{i\geq 2} e^{-t\lambda_i}\Phi_i(u)\Phi_i(v)$. $\Phi_1(u)\Phi_1(v)$ cancels out leaving the equivalent spectral form $\int_{t=0}^{\infty} D^{1/2} t^n [\hat{\Phi}\hat{\Lambda} e^{-t\hat{\Lambda}}\hat{\Phi}^T] D^{1/2} dt$. Evaluating the integral, and using orthogonality of eigenvectors, I get the desired spectral form $\mathcal{D}_r^n(u, v) = \Gamma(n+1) \sum_{i\geq 2} {\lambda_i}^{-n}\Phi_i(u)\Phi_i(v)\sqrt{d_v d_u}$ where Γ is the gamma function. This can be expressed in matrix form as $\mathcal{D}_r^n = \Gamma(n + 1) D^{1/2}\hat{\Phi}\hat{\Lambda}^{-n}\hat{\Phi}^T D^{1/2}$. The exponent n can be viewed a frequency filter, with high n suppressing higher frequency eigenvalues [108]. Recall, the eigenvalues are arranged in in-

creasing order, and hence, the eigenvectors are in decreasing smoothness order. As a result, the eigenvectors associated with the K smallest eigenvalues of the normalized Laplacian are considered for K-way normalized cut partition. My formulation of the diffusion modulus transforms the K parameter in normalized cut to the n parameter in $\mathcal{D}^n$. The benefits of $\mathcal{D}^n$ include 1) The spectral decomposition is calculated once and varying n outputs segments at different resolutions unlike spectral approaches where decomposition is based on the number of segments, i.e., $\mathcal{D}^n$ is a more formal criterion for scale selection 2) The $\mathcal{D}_r^n$ kernel is motivated using the physical analogy of rate of heat diffusion, RH_t. RH_t subsumes the entire path length distribution over nodes, and hence, *'learns'* affinities between all node pairs. 3) RH_t is integrated over the entire time domain making the kernel robust to noise. 4) Pre and post multiplying D_r^n by the diagonal matrix $D^{-1/2}$ and assigning $n = 1$ results in the commute-time kernel (also known as Green's function [79, 108]) and $n = 2$ results in the biharmonic kernel [114]. This is better understood by using the discrete formulation of $\mathcal{D}_r^n$

$$\mathcal{D}_r^n = \sum_{t=1}^{\infty} Dt^n [M^t - M^{t+1}] \tag{2.44}$$

This is true because the term in parenthesis can be written as $[e^{-t\Delta} - e^{-(t+dt)\Delta}]$ for the continuous case [1]. Taylor series expansion of the derived $I - e^{-\Delta dt}$ term in $e^{-t\Delta}(I - e^{-\Delta dt})$ results in the rate of diffusion. D is premultiplied to symmetrize the kernel as in the continuous case. Intuitively, M^t denotes the transition matrix after time t. Hence, $M^t - M^{t+1}$ denotes the change in transition probability after one time step, i.e., the rate of probability change. Newman uses the functional in Equation (2.44) for $n = 1$ in a random-walk setting to derive the commute time kernel (see page 159 [21]). Hence, the multi-resolution term can be viewed as a generalization of several graph based kernels. The calculation of the exact diffusion modulus kernel using the entire spectral decomposition of the Laplacian has complexity $\mathcal{O}(N^3)$. It can be reduced to $\mathcal{O}(N^2)$, for sparse graphs using the efficient Lanczos method [79]. It is further reduced to $\mathcal{O}(kN)$ by trading accuracy for efficiency, and calculating the first (smallest) $k(k \ll N)$ eigenvalues and corresponding eigenvectors. The

[1] I use the proof in [76] where M^t or $(I - \Delta)^t$ converges to $e^{-t\Delta}$ for continuous time random walks or heat diffusion.

summand in Equation (2.42) can be approximated using these values and the error bound is (generally) linear in $1/k$ [114]. The larger eigenvalues correspond to local oscillations and provide little improvement in final segmentation. Note, the embedding Θ that preserves the kernel $\mathcal{D}_r^n$ is, $\Theta_n = (\hat{\Lambda})^{-n/2}\hat{\Phi}^T D^{1/2}$ as $\mathcal{D}_r^n = \Theta_n^T\Theta_n$.

Theoretical properties:

Theorem 2.1.2 *Let $\mathcal{D}_r^n$ be the matrix defined by $D^{1/2}[\hat{\Phi}\hat{\Lambda}^{-n}\hat{\Phi}^T]D^{1/2}$ (drop constant $\Gamma(n+1)$); then $\mathcal{D}_r^n$ is symmetric and positive semidefinite. $\mathcal{D}_r^n \neq \mathbf{0}$ $\forall n$. All rows, columns of $\mathcal{D}_r^n$ (also $\mathcal{D}^n$) sum to zero, and sum of all elements of $\mathcal{D}_r^n$ equals zero.*

Proof. $\mathcal{D}_r^n$ is symmetric as interchanging u, v does not change the representation of Equation (2.42). As all eigenvalues are strictly non-negative, $\mathcal{D}_r^n$ is positive semidefinite, and hence, a valid kernel. The possibility of $\mathcal{D}_r^n \to \mathbf{0}$ arises as $n \to \infty$. The eigenvalues are arranged in increasing order and as $n \to \infty$, $\mathcal{D}_r^{n\to\infty} \to D^{1/2}[\Phi_2{\lambda_2}^{-n}{\Phi_2}^T]D^{1/2}$. $\lambda_2 \leq 1$, $\therefore$ ${\lambda_2}^{-n} \geq 1$, and hence, it is easy to verify that $\mathcal{D}_r^n \neq \mathbf{0}$. Note, λ_2 or the spectral gap is related to the mixing time of a random walk and provides an upper bound to the graph conductance [115]. Intuitively, cliques have perfect mixing after one time step, and hence, have $\lambda_2 = 1$. The eigenvectors form an orthonormal basis, i.e., $\Phi_i(u)\Phi_j(v) = \delta_{ij}$. $\Phi_1(u) = \sqrt{d_u/vol}$ [79], and hence, $\sum_u \sqrt{d_u}\Phi_j(v) = 0, \forall j \neq 1$. Consider, the contribution of the j^{th} eigen-pair, (Φ_j, λ_j) to the i^{th} row of $\mathcal{D}_r^n$. The contribution in spectral form is: $\sqrt{d_i}\frac{\Phi_{ij}}{\lambda_j^n}\big(\sqrt{d_1}\Phi_{1j} + \sqrt{d_2}\Phi_{2j} \ldots \sqrt{d_N}\Phi_{Nj}\big)$. The term in parentheses equals zero, and this is true for every eigen-pair excluding (Φ_1, λ_1), which does not appear in D_r^n. Hence, each row sums to zero. Note, the sum equals zero for any $2 \leq K \leq N$ eigen-pairs. As the matrix is symmetric, all columns sum to zero, and hence, the sum of all elements in $\mathcal{D}_r^n$ equals zero. Intuitively, flow conservation holds for $\mathcal{D}_r^n$ and the trivial partition of grouping all nodes together leads to $\mathcal{D}^n$ value 0. Also, maximizing the association implicitly minimizes the disassociation as (association+dissociation=0), i.e., $\mathcal{D}^n$ is an unbiased measure.

Theorem 2.1.3 *Let $\mathcal{D}^n$ represent the diffusion modulus criterion, $H^T D^{1/2}[\hat{\Phi}\hat{\Lambda}^{-n}\hat{\Phi}^T]D^{1/2}H$. The criterion splits the graph into two segments in the limit $n \to \infty$ and places each node*

in a separate segment for $n = 0$. The 2-way segmentation is identical to the normalized cut partition and $\mathcal{D}^n$ is equivalent to the modularity metric in the discrete sense.

*Proof.*All regions are placed in separate segments, only if merging two regions leads to an inferior diffusion modulus value. Hence for the case $n = 0$, it suffices to prove that all non-diagonal elements are negative, i.e., $K = N$, *iff* $\mathcal{D}_r^0(u, v) < 0, \forall u \neq v$. As $\mathcal{L}e^{-t\mathcal{L}} = \frac{-\delta(e^{-t\mathcal{L}})}{\delta t}$, the definite integral in Equation (2.43) for $n = 0$ reduces to $D^{1/2}[-e^{-t\mathcal{L}}]_0^\infty D^{1/2}$. $\lim_{t\to\infty} e^{-t\mathcal{L}} = \frac{\sqrt{d_u d_v}}{vol}$ as all eigenvalues, apart from $\lambda_1 = 0$ are suppressed, and the u^{th} element of corresponding eigenvector equals $\sqrt{d_u/vol}$. Hence, $\mathcal{D}_r^0(u, v) = -\frac{d_u d_v}{vol}, \forall u \neq v$ proving the necessary and sufficient condition. For $n \to \infty$, barring degenerate eigenvalues, the diffusion modulus criterion is dominated by the normalized fielder vector, i.e., $\lim_{n\to\infty} \mathcal{D}_r^{n\to\infty}(u, v) = \lim_{n\to\infty} {\lambda_2}^{-n} \Phi_2(u)\Phi_2(v)\sqrt{d_v d_u}$. The diffusion modulus increases *iff*, $sign[\Phi_2(u)] = sign[\Phi_2(u)]$, and hence criterion is maximized by two-way partition based on signs of Fielder vector ($\lambda_2 > 0$). Normalized cuts partitions the graph based on the Fielder vector for the generalized eigenvalue system $(D - \mathcal{A})\acute{\Phi} = \acute{\lambda} D\Phi$. This system has eigenvalues Λ and eigenvectors $D^{-1/2}\Phi$. Hence, bipartition by signs of the Fielder vector for the normalized cut is the same as $\mathcal{D}^{n\to\infty}$. Using the discrete formulation in Equation (2.44) for $n = 0$ results in $\mathcal{D}_r^0 = \sum_{t=1}^{\infty} D[(M)^t - (M)^{t+1}]$. This converts to $D_r^0 = D[M - M^\infty]$. Using $M^\infty = \mathbf{1}\pi^T$ and $\pi = \mathbf{1}D/vol$ I recover the original definition of modularity, i.e., $[\mathcal{A} - \frac{\kappa\kappa^T}{vol}]$ where $\kappa = diag(D)$.

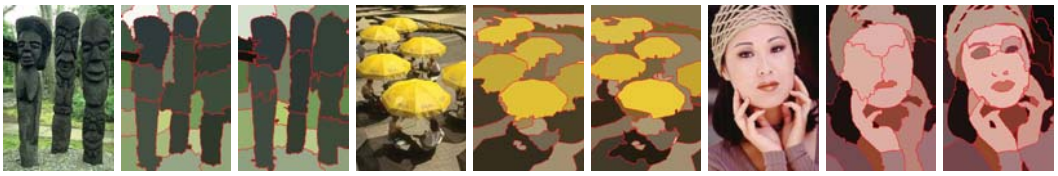


Figure 2.9. *(Left to Right in sets of 3): Original image, segmentation using diffusion modulus (DM) and multi-scale DM respectively.*

Image segmentation techniques: Here I develop 2 graphical models suitable for the diffusion modulus and provide efficient optimization techniques. I also extend my method for hierarchical image segmentation.

Multi-scale graph segmentation: I only connect edges lying in the 4-neighbourhood of the pixelated image and define the weights of the matrix $\mathcal{A}$ as $\mathcal{A}(u,v) = exp(-\theta_g \parallel g_u - g_v \parallel)$ if $(u,v) \in E$, where g represent pixel value in *Lab* color space and θ_g is a constant that controls edge strength. The reason for this construction is 2-fold: 1) As $\mathcal{D}^n$ uses a fixed number of eigenvectors to segment an image, I stress on sparsity over region consistency for efficient spectral decomposition and 2) All pairwise affinities to other pixels are implicitly learnt by the kernel $\mathcal{D}_r^n$. I proved that each row and column of the $\mathcal{D}_r^n$ sums to 0 and the sum of all elements in $\mathcal{D}_r^n$ is 0 in Theorem 1, akin to the modularity metric for community detection [112]. Hence, an efficient Louvain-like modularity maximization algorithm can be leveraged for diffusion modulus optimization [116]. The Louvain algorithm iteratively proceeds in two phases. The algorithm starts by placing all pixels in individual segments, and sequentially places each pixel in another pixel/region, which leads to maximum improvement in criterion, till no further improvement (i.e., local maximization of diffusion modulus criterion). The identified segments are subsequently merged into *'meta-regions'*, and the algorithm iterates till maximum global diffusion modulus value is reached. Although the exact complexity of Louvain is unknown, the method seems to run in time $\mathcal{O}(NlogN)$ and first level optimization requires the highest effort [116]. Applying the Louvain for first level optimization is prohibitive for even moderately sized images as $\mathcal{D}_r^n$ is a dense matrix. One way to overcome this limitation is to run it sequentially on a partitioned $\mathcal{D}_r^n$ matrix. However, this strategy scales poorly. Instead, I introduce an approximation by uniformly seeding the original image and only consider pixels within a certain neighbourhood for label assignment. The neighbourhood is set to be sufficiently large so that each pixel is a neighbour to multiple seeded pixels. I calculate the D_r^n affinity between each pixel and other neighbouring seeded pixels and group it with the seeded pixel with which it shares the highest affinity. The intuition behind this approximation is that perceptual regions consist of pixels that are *'close'* in the spatial space as well as the embedded

space. After this first level optimization, the number of *'meta-regions'* is sufficiently small for directly executing Louvain on the reduced $\mathcal{D}_r^n$.

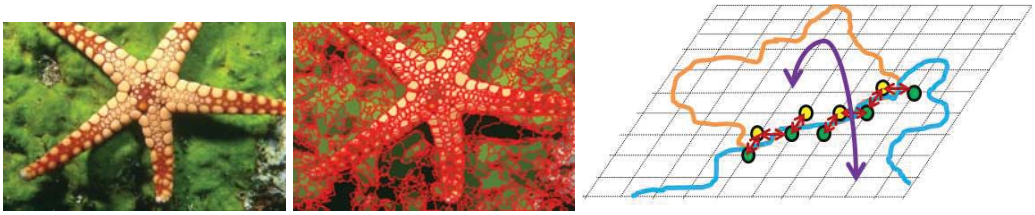


Figure 2.10. *Graph construction (left to right): Original image, over segmented image using meanshift and graph model. Region similarity (purple arrow) and boundary similarity (red edges between yellow and green pixels in blue, orange regions.)*

Rate of diffusion underscores the paradigm of learning affinities using sparse local connections. A natural extension is to use it in conjunction with multi-scale graphs. Towards this end, super-pixelated regions are generated from the original image to encode long-range connections. In principle, such regions can be achieved by either mean-shift, Felz-Hutt or other superpixel algorithms [117]. The obvious advantage of mean-shift algorithm it that it balances speed and region quality. I propose the use of mean-shift because of its deep connection with modularity, and hence, the diffusion modulus. Modularity can be reexpressed a $D(M - \pi)$ where π is the stationary density distribution, and implicitly calculates the global expectation above the average density. Mean-shift on the contrary finds the modes of a density function within a specified window, and therefore, can be understood as a more localized version of modularity. These methods compliment each other and as modularity can be derived from the $\mathcal{D}^n$ (Theorem 2), mean-shift is a natural choice for over-segmentation. Contrary to other approaches, I use the over-segmented image to construct a region graph. This is beneficial as it naturally creates a smaller graph for efficient spectral decomposition. In addition, the over-segmented regions using mean-shift can be viewed as an improvement over the approximation introduced in Louvain for first-level

optimization. A simplistic graphical model is employed using only color cues. The weights of the adjacency matrix $\mathcal{A}$ are calculated as:

$$\mathcal{A}(u,v) = \begin{cases} exp(-\theta_g \parallel \bar{g}_u - \bar{g}_v \parallel) & (u,v) \in E_r \\ \sum_C exp(-\theta_g \parallel g_{u^*} - g_{v^*} \parallel) & (u^*,v^*) \in C \\ 0 & otherwise \end{cases} \tag{2.45}$$

Where $\bar{g}$ represents the mean color of the region, E_r represents the region neighborhood and C represents all pixel pairs (u^*, v^*) that belong to different regions (u, v) but lie within 3 pixel radius. The sum over all pixel pairs in C enforces region smoothness (see Figure 2.10). Rate of diffusion optimization follows Louvain algorithm on $\mathcal{A}$. Figure 2.9 compares the segmentations of the single-scale and the multi-scale diffusion modulus criterion. Observe that the multi-scale version respects image boundaries better.

Hierarchical graph segmentation: The number of segments in an image is result of perceptual scale, and hence, real-world images are unlikely to possess an optimal number of segments. This claim in instantiated in the ground truth segmentations of the Berkeley image database (BSDS)[2] where the same image is grouped into different number of segments by different people. In order to supplement this observation, weighted boundaries are created using a set of multi-resolution segmentations over logarithmically spaced parameter n in a range. As increasing n results in increasingly global segmentations, boundary pixels are weighted by n, and summed over entire range to provide a weighted pixel value. Mathematically, $w_p(u) = \sum_{i=1}^{R} n_i \delta_{ui}$, where $w_p(u)$ is weight of pixel u, R is number of discrete n values in logarithmically spaced domain, n_i is the i^{th} value in range, and δ_{ui} is delta function equal to 1 if u lies on boundary for i^{th} segmented image using n_i. Thresholding the boundary pixel weights, outputs desired hierarchical segments, similar to [66] (see Figure 2.11).

In this section, I discussed how random walks can be used to interpret networks, geometric meshes and derive segmentation on images. I discuss Gauss's law in the context of community detection next.

[2]http://www.cs.berkeley.edu/projects/vision/bsds

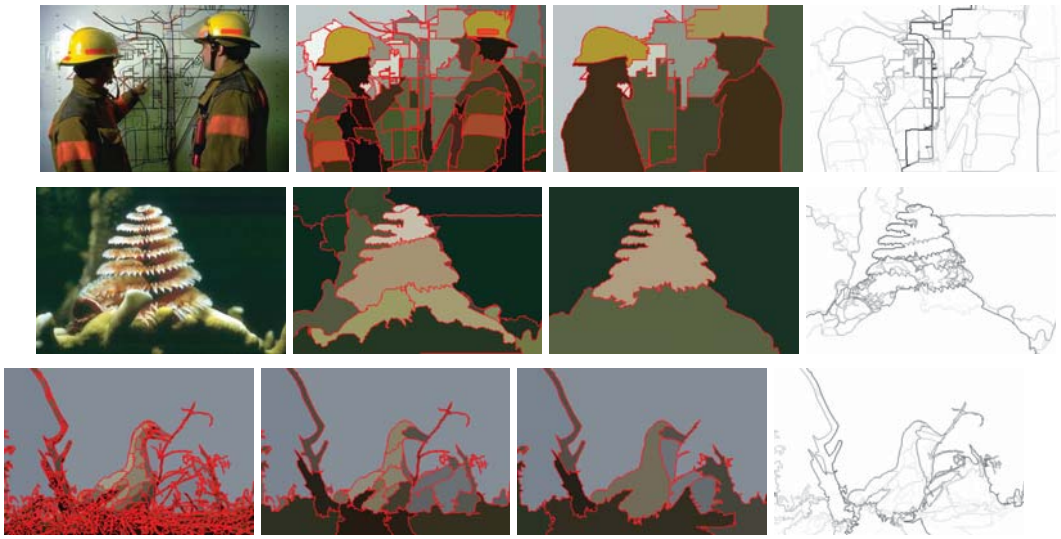


Figure 2.11. *Hierarchical segmentations (left to right): Image, 2 segmentations by increasing n and weighted boundary image.*

2.2 Gauss's Law

Although there is no consensus, a common abstraction is that communities are subgraphs with strong intra-subgraph cohesion and weak inter-subgraph cohesion, where cohesive strength is measured either in terms of direct connections or more sophisticated connectivity measures [38]. Relationships between boundaries and their contents are common in physics such as how the well-known Gauss's law relates the electric flux through a surface boundary to the charge enclosed within the surface (Figure 2.12 A). My approach to detect communities boundaries uses a network analog of Gauss's law to relate a flux through a subgraph's boundary to the cohesion among the nodes of a subgraph (Figure 2.12 B). The method works with respect to a general measure of connectivity (or similarity) between all pairs of nodes encoded in matrix S. The element $S(i,j)$ either details the adjacency $A(i,j)$ of nodes i and j in the network, or the extended similarity of node i to j evaluated using a reference property such as common neighbors, path connectivity, etc [118]. I call $S(i,j)$ the connectivity potential of i,j. Given S, the flux through a sub-

graph boundary for a node, termed connectivity flux, mirrors the electric flux through a surface for an electrostatic potential field, albeit in a discrete setting (See Figure 2.12C). According to classical electrodynamics, the electric field at a point in space is the gradient of the electrostatic potential at the corresponding point. Given the connectivity potentials between all pairs of nodes, the connectivity flux for a node i through an edge E_b is then the symmetric difference of node i's connectivity potentials with respect to the two nodes constituting the edge, E_b (say m and n) (Figure 2.12C, Top)

$$S^i(E_b) = \big(S(i,m) + S(m,i)\big) - \big(S(i,n) + S(n,i)\big). \tag{2.46}$$

As the flux through a surface is the area integral of the electric field, the net connectivity flux of node i through a boundary is the sum of the connectivity flux through all edges constituting the subgraph boundary, E_B, i.e., $\sum_{b\in B} S^i(E_b)$ (Figure 2.12C, Bottom).

The boundary comprises links that separate the subgraph from the rest of the network. It induces two disjoint multisets of internal and external boundary nodes where every internal boundary node has an external counterpart. Consequently, the connectivity flux for a node i with respect to subgraph V^k's boundary is equal to the difference of two terms, one measuring the connectivity of the node i to internal boundary nodes and one to external boundary nodes. I call them the internal $S^i(V^k_{in})$ and external $S^i(V^k_{out})$ boundary cohesion, respectively.Consider the subgraph induced by a subset of nodes V^k. Let E^k_B be the boundary links of this subgraph and V^k_B be the multiset of nodes induced by the boundary links. Then the multiset of internal nodes is represented as $V^k_{in} = \{m \mid m \in V^k \text{ and } m \in V^k_B\}$, and the multiset of external nodes is represented as $V^k_{out} = \{n \mid n \notin V^k \text{ and } n \in V^k_B\}$. The cohesion of node i with respect to a general multiset V is:

$$S^i(V) = \sum_{j\in V} \big(S(j,i) + S(i,j)\big), \tag{2.47}$$

where the summation includes all the repetitions in the multiset. The internal boundary cohesion of node i with respect to V^k is then $S^i(V^k_{in})$ and the external boundary cohesion is $S^i(V^k_{out})$. This decomposition enables me to write the connectivity flux of node i through the boundary as $\sum_{b\in E^k_B} S^i(E_b) = S^i(V^k_{in}) - S^i(V^k_{out})$ (Figure 2.12D). Note that

because of the multisets, each cohesion expression has exactly the same number of terms (See Figure 2.12 D). The net connectivity flux through subgraph V^k's boundary for the set of nodes in V^k constituting community, C is then equal to $\sum_{i \in V^k} \big(S^i(V^k_{in}) - S^i(V^k_{out})\big)$, which I write as $S^C(V^k_{in}) - S^C(V^k_{out})$. The decomposition of connectivity flux into internal and external components allows me to characterize communities in both a strong and weak sense [45] (described later). High connectivity flux through the subgraph boundary, $S^C(V^k_{in}) - S^C(V^k_{out})$ suggests a difference in the strength of internal and external cohesion of subgraph V^k, and consequently, a close-knit community (Figure 2.12E).

The relationship between flux and communities motivates an additive quality function to evaluate a partition of nodes:

$$Q = \sum_{V^k} (S^C(V^k_{in}) - S^C(V^k_{out})). \tag{2.48}$$

Exactly optimizing Q, requires a combinatorial search over all sets of boundary links which is prohibitive for large networks. My main result is a reformulation of Q such that it can be optimized using computational heuristics available for modularity maximization in order to reveal the boundary links all at once (Figure 2.12F):

$$Q = \sum_{i,j} [LS + SL]\delta(i,j). \tag{2.49}$$

Here $\delta(i,j)$ is 1 if i and j belong to the same community and 0 otherwise, L is a matrix known as the Laplacian, the discrete analogue of the Laplace operator, and Q possesses the key properties of the modularity quality function [49] (derived later). Hence, I call my quality function, the *Laplacian modularity* and the matrix $B = LS + SL$ the Laplacian modularity matrix. The Laplacian modularity, like modularity, is an unbiased metric for any S (described subsequently). However, unlike modularity, it does not suffer from the traditional resolution limit [119] if S is local (derived later).

An alternate interpretation of the Laplacian modularity matrix is that the ij^{th} term is the curvature of a link:

$$B(i,j) = d^i \left(S(i,j) - \frac{1}{d^i} \sum_{k \in N_1(i)} S(k,j) \right) + d^j \left(S(i,j) - \frac{1}{d^j} \sum_{k \in N_1(j)} S(i,k) \right). \tag{2.50}$$

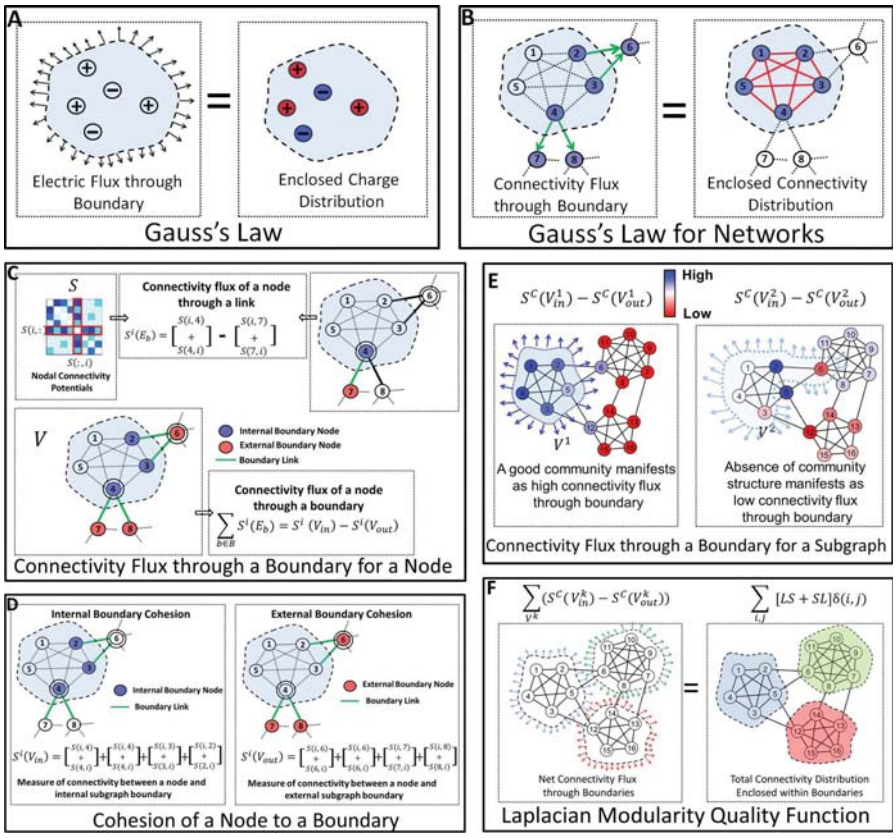


Figure 2.12. *Community detection using Gauss's Law. A: Classical Gauss's law. B: Gauss's law for networks implicitly measures the enclosed connectivity distribution as the connectivity flux through the boundary. C: Evaluating connectivity flux through a boundary for a node in the network. D: Connectivity flux quantified as measures of cohesion. E: Connectivity flux through a boundary for a subgraph: High and low values of this connectivity flux indicate good communities or absence of community structure, respectively. The darker region of blue indicate higher connectivity flux. The color of nodes (blue to red) highlight the magnitude of connectivity flux (positive to negative) through the boundary. F: Laplacian modularity quality function derived from the cumulative connectivity flux for a network partition.*

Here $d^i = d^i(V^k) + d^i(\bar{V}^k)$ is the total degree of node i and $N_1(i)$ indicates the immediate neighbors of node i. The terms in the parenthesis are the similarity of node i and j minus the average similarity over the local neighborhoods of nodes i and j, respectively, i.e., it

measures the rate at which the connectivity of a link deviates from the average connectivity value over a local neighborhood. Thus, each term in B encodes a curvature-like metric and collectively, a subgraph is a community as per the Laplacian modularity if the internal connectivity is higher than the average connectivity calculated over the local neighborhood of nodes in the subgraph.

In network science, the identification of relevant hierarchical communities in the presence of overlap remains an open problem [38]. I resolve this problem by using the heat kernel of a network, H_t, as the connectivity potential matrix S in order to holistically characterize overlapping and hierarchical communities. The heat kernel, H_t, has a parametric dependence on the time scales, t, of a Markovian process on the network, and hence, is intrinsically multiscale with a natural interpretation of the network resolution under study. I write the quality function where S is H_t, as Q_t. Although Q_t measures non-overlapping partition, overlapping nodes can be identified by evaluating the nodal flux associated with each community boundary (see Figure 2.13 A, B). Overlapping nodes are simply nodes with net positive connectivity flux to multiple communities for given connectivity potentials S. Mathematically, the positive terms in the matrix $X^T L(S + S^T)$ resolve overlap, where X is a 0-1 indicator matrix for the hard partition of dimensions $|V| \times C$ (derived later). The parametric quantity Q_t also gives me a way to pick t. The element $H_t(i, j)$ in the heat kernel represents the probability of a random walker to hop from node i to j after a Poisson number of jumps with mean t. It suffices to only consider the term LH_t for $S = H_t$ in Equation (2.140) because LH_t is symmetric, unbiased and has an intuitive diffusion dynamical interpretation. I represent LH_t as B_t, in this discussion. Substituting $H_t = e^{-t\mathbb{L}}$ in B_t results in $D\mathbb{L}e^{-t\mathbb{L}}$, where $\mathbb{L}$ is the random walk Laplacian which is physically interpreted as the negative rate of diffusion ($-\frac{dH_t}{dt}$) scaled by D. It follows from thermodynamic principles that the partition which maximizes Q_t (the trace of the clustered matrix, $X^T B_t X$), minimizes the net diffusive tendency within subgraphs, or the heat diffusion reaches a local equilibrium at time t before it proceeds to the global equilibrium. Intuitively, a partition optimal over a long time span corresponds to a stable equilibrium, or that the local equilibrium reached at time t persists over competing equilibria until time

(say) $\tilde{t} > t$. Using this intuition in conjunction with monotonicity of Q_t, I define a stability curve, $r(t)$ to be the maximum value of the diffusion modularity over the space of all partitions X, i.e,

$$r(t) = \max_X \text{trace} \left(\frac{1}{2|E|} X^T B_t X \right), \tag{2.51}$$

where the trivial constant $2|E|$ ensures that the measure lies in $[0, 1]$ and $|E|$ is number of links. By viewing this quantity as a stability curve (Figure 2.13 C) [120], I can hierarchically organize network partitions, from fine to coarse as t increases. Sustained temporal stability of partitions reveal topological scales or *good* organizational levels of the network.

I first give an overview of my algorithm and discuss each element in detail over the next few subsections. I use the Heat kernel, H_t as S in all my experiments and use the Louvain method [121] to optimize Laplacian modularity, Q_t at each time t . The hierarchical community organization of the network can be explored using the stability curve. However, only the partition with maximal persistence over a logarithmic time range is picked for an equitable comparison to other methods. In order to discern overlap in this optimal partition, I first calculate the fuzzy overlap membership of each node to all subgraphs at discrete time values in the persistent basin. This corresponds to entries of the matrix $X_o^T B_t$, where X_o is the overlap indicator matrix. I then introduce a threshold parameter ϵ that converts fuzzy overlap membership over the time range into a hard overlap. The choice of ϵ is guided by common intuition where $\epsilon = 1$ outputs a hard partition (no overlap), and $\epsilon = 0$ identifies minute structural overlap of communities, especially relevant for networks with incomplete information or missing links. I set $\epsilon = 1$ for networks with known communities because the ground-truth is a hard partition, and $\epsilon = 0$ for networks with unknown community assignment where I expect pervasive overlap.

2.2.1 Method details for Gauss's law to reveal community boundaries

Having given an overview of Gauss's law for community detection, I proceed to describe each element in great detail. In discuss additional details regarding constructing and optimizing the Laplacian modularity. Specifically, I provide a formal definition of commu-

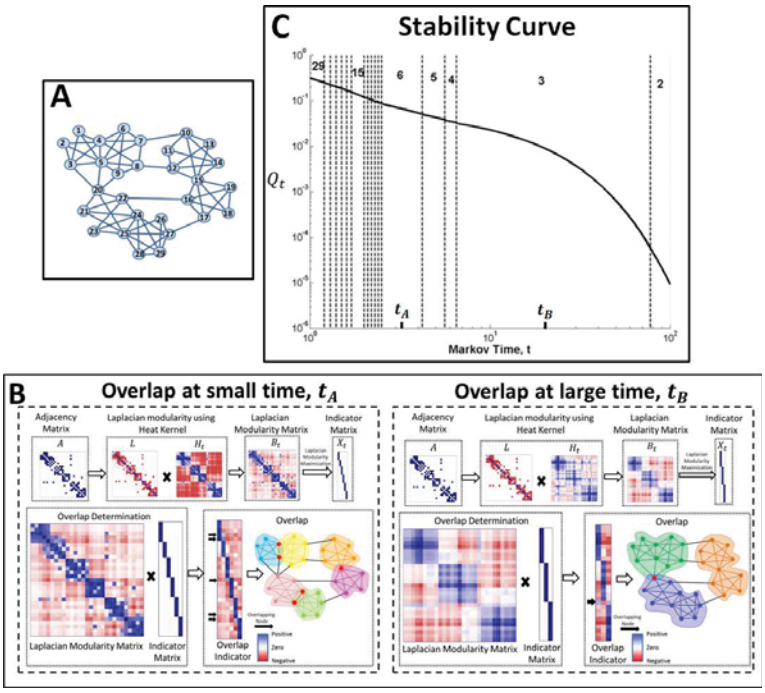


Figure 2.13. *Overlapping and hierarchal communities using Laplacian modularity and the heat kernel: The heat kernel depends on a parameter t that controls the resolution of the communities (small t is fine, and large t is coarse). A: Network of 29 nodes. B: (Left) Six identified communities at a small time, t_A with overlapping nodes highlighted in red in the network representation. The rows and columns in the matrix representation follow a linear ordering indexed by node number. Blue matrix cells indicate positive values and red cells indicate negative values. (Right) Increasing the time shows three identified communities at large time, t_B with overlapping nodes in red. Observe the overlapping node at large time t_B is not overlapping at the small time t_A, implying it has a weak connection with the individual yellow and blue communities identified at time t_A, but a strong connection with them merged into one (green community at time t_B), that is, overlap emerges with time t. C: The stability curve derived using the heat kernel. The stability curve for the network shows two large persistent time-spans enclosed within dotted lines corresponding to the 3-way partition and 6-way partition.*

nities, prove key properties of the Laplacian modularity, provide implementation details for optimizing the quality function and revealing overlap using the Laplacian modularity ma-

trix, examine the resolution limit of my approach, and provide a multi-resolution extension. I also provide an interpretation of the Laplacian modularity using resistor networks.

Strong and weak definitions of community: One intuitive formalization of a community in terms of direct connections is the LS-set, wherein each node has more internal connections than with the rest of the network [122]. For a network with adjacency matrix, A where $A(i,j) = 1$ if there is a direct connection between nodes i and j and 0 otherwise, a subgraph V^k is a LS-set if $d^i(V^k) > d^i(\bar{V}^k), \forall i \in V^k$. $d^i(V^k) = \sum_{j \in V^k} A(i,j)$ is the internal degree of a node i, and $d^i(\bar{V}^k) = \sum_{j \in \bar{V}^k} A(i,j)$ is its external degree to the rest of the network, $\bar{V}^k$ [45]. An alternate approach is to construct a hierarchical clustering tree by agglomerating a node-to-node connectivity matrix, S, and subsequently cut branches off the dendrogram to output a network partition [123]. However, there exist several partitions of a network into LS-sets or cutoff values in hierarchical clustering. Hence, the algorithmic detection of an optimal partition is most popularly premised on comparing the internal degree of a node to its expectation under a null model, $d^i(V^k) > \langle d^i(V^k) \rangle$, as in the modularity quality function [49]. The null model implicitly assumes a node randomly interacting with all other nodes, however, interactions in real-world networks are preferential within a local horizon [119, 124]. I define strong and weak communities in terms of the two components of the connectivity flux guided by the rationale that a community member is more cohesive to fellow members inside the community boundary than those outside.

Strong Community: The subgraph V^k is a community in a strong sense if:

$$S^i(V^k_{in}) > S^i(V^k_{out}), \forall i \in V^k. \tag{2.52}$$

Here $S^i(V^k_{in})$ and $S^i(V^k_{out})$ are the internal and external boundary cohesion respectively. In a strong community, the internal boundary cohesion of each node is greater than its external value for subgraph, V^k, similar to the internal degree of each node is greater than its external degree in an LS-set.

Weak Community: The subgraph V^k is a community in a weak sense if:

$$S^C(V^k_{in}) > S^C(V^k_{out}). \tag{2.53}$$

The superscript C represents sum over all nodes in V^k. In a weak community, the sum of internal boundary cohesion over all nodes in V^k is greater than the sum of external boundary cohesion, for subgraph, V^k, in analogy with the sum of internal degrees exceeds the sum of external degrees of nodes in a weak community [45]. It is apparent that a community in the strong sense is also a community in a weak sense, whereas the converse is not true. As in a LS-set, these definitions of a community account for intra-subgraph and inter-subgraph cohesion. However, unlike LS-sets, they circumvent bias to subgraph size because for every internal boundary node there is an external counterpart.

Quality function: I discuss the Laplacian modularity quality function in terms of the two disjoint components of the connectivity flux, i.e., the internal and external boundary cohesion as described in the main manuscript. Let the set of boundary links for subgraph V^k be E^k such that the node pair $[m, n]$ associated with each link satisfy $m \in V^k, n \notin V^k$. Note m is an internal node ($m \in V^k_{in}$) and n is an external node ($n \in V^k_{out}$). As described in the main manuscript, the mutual connectivity of a node i, $S^i(V)$, to a multiset of nodes, V, based on connectivity matrix, S is assessed as:

$$S^i(V) = \sum_{j \in V} \big(S(j,i) + S(i,j)\big). \tag{2.54}$$

In matrix form I have the equivalent representation:

$$S(V) = \mathcal{V}^T S + \mathcal{V} S. \tag{2.55}$$

$\mathcal{V}$ is the membership vector, where the element $\mathcal{V}^i$ details the cardinality of node i's occurrence in multiset V. $S^T\mathcal{V}$ models the connectivity of nodes in the network to multiset V, whereas $S\mathcal{V}$ models the reciprocal connectivity of nodes in multiset V to nodes in the network. A node participating in x boundary links (internal or external) is seeded x times, so as to maintain the total cardinality of the two membership vectors. Such a node inherently mediates greater control over the measure of cohesion relative to a node participating in one or no boundary link, akin to the personalization vector in PageRank [125]. The two components of the connectivity flux are then represented as:

$$S^i(V^k_{in}) = S^T\mathcal{V}^k_{in} + S\mathcal{V}^k_{in} = \sum_{E^k} \big(S(m,i) + S(i,m)\big) \tag{2.56a}$$

$$S^i(V^k_{out}) = S^T\mathcal{V}^k_{out} + S\mathcal{V}^k_{out} = \sum_{E^k}\big(S(n,i) + S(i,n)\big). \tag{2.56b}$$

The similarity of node i to the multiset V^k_{in} or V^k_{out} is equal to the sum of connectivity potentials of node i to induced internal boundary nodes and external boundary nodes, respectively. Note that I use the term similarity and connectivity potential interchangeably in this entire document. The authors in [126] motivate a sophisticated measure of cohesion based on triangle density in order to determine communities. In contrast to their definition, my measure of cohesion is generally applicable to any similarity matrix, S encoding connectivity potentials.

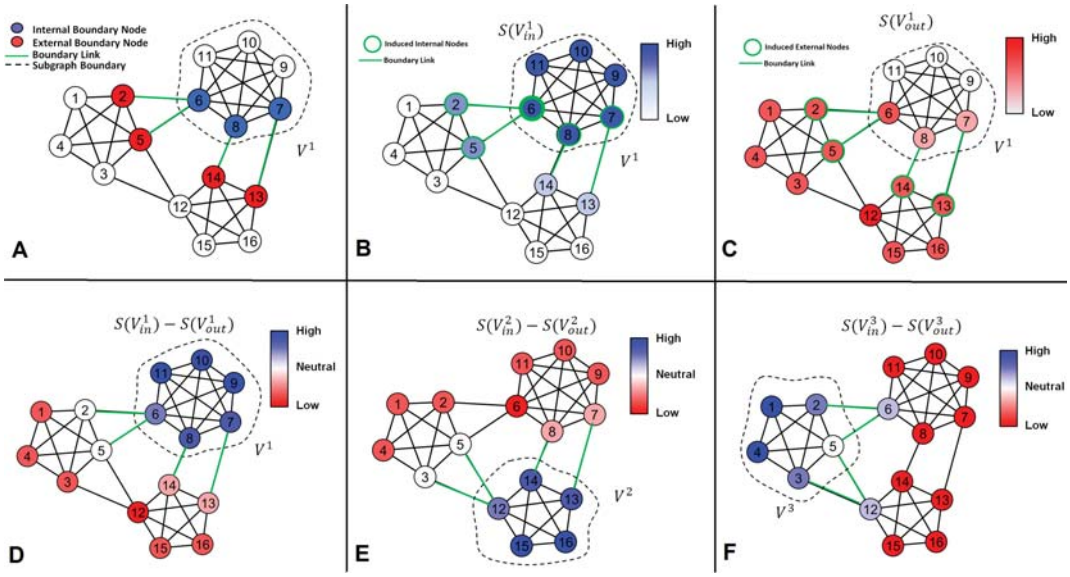


Figure 2.14. *Cohesion to subgraph boundary. A:Internal (blue) and external (red) boundary nodes induced by boundary links (green) by dotted subgraph boundary of V^1. B:Cohesion of nodes in network to internal boundary nodes of V^1. Higher values of internal boundary cohesion are displayed as darker shades of blue. C: Cohesion of nodes in network to external boundary nodes of V^1. Higher values of external boundary cohesion are displayed as darker shades of red. D: Difference of internal and external boundary cohesion due to V^1. E:Difference of internal and external boundary cohesion due to V^2. F:Difference of internal and external boundary cohesion due to V^3. Color legends for $S(V^k_{in}) - S(V^k_{out})$ values are displayed to the right in subplots D,E,F.*

Figure 2.14 A highlights the internal and external boundary nodes of subgraph V^1 in a synthetic network of 16 nodes comprising of 3 cliques. The boundary links (green edges) induce the multiset ([6,6,8,7]) and set ([2,5,14,13]) to be the internal (blue) and external (red) boundary nodes, respectively. Figure 2.14 B,C graphically illustrates the calculation of internal and external boundary cohesion for the subgraph V^1, by setting $S = A + I$. I is an identity matrix which assigns self-similarity to each node. The sum of all elements in vectors $\mathcal{V}_{in}^k$ as well as $\mathcal{V}_{out}^k$ is equal to the total number of boundary links to the subgraph, i.e., four for V^1, and the multiplicity of node 6 is two in $\mathcal{V}_{in}^1$ as it connects to two boundary links.

If a subgraph, V^k is a community in the weak sense, I have $S^C(V_{in}^k) - S^C(V_{out}^k) > 0$, An additive quality function over subgraphs derived from the weak definition of a community is then given by:

$$\begin{aligned} Q &= \sum_{V^k} (S^C(V_{in}^k) - S^C(V_{out}^k)) \\ &= \sum_{V^k} \sum_{i \in V^k} (S^i(V_{in}^k) - S^i(V_{out}^k)). \end{aligned} \tag{2.57}$$

Here V^k is the subgraph, $S^i(V_{in}^k)$ and $S^i(V_{out}^k)$ are the internal and external boundary cohesion, respectively. My main result is that optimizing quality function Q in Equation (2.57) is equivalent to optimizing the function:

$$Q = \sum_{i,j} [LS + SL]\delta(i,j) = \sum_{i,j} [(D - A)S + S(D - A)]\delta(i,j). \tag{2.58}$$

Here Q is the Laplacian modularity quality function, S is the node-to-node connectivity matrix, L is the Laplacian given as $L = D - A$, D is a diagonal matrix of node degrees, A is the adjacency matrix, and $\delta(i,j)$ is a delta function wherein $\delta(i,j) = 1$ if both i and j belong the same community and 0 otherwise. I prove that:

$$\sum_{V^k} \sum_{i \in V^k} (S^i(V_{in}^k) - S^i(V_{out}^k)) = \sum_{i,j} [(D - A)S + S(D - A)]\delta(i,j). \tag{2.59}$$

My second result in this subsection is that my quality function is an unbiased measure at all resolutions. Formally, I prove that:

$$Q^{ass} + Q^{diss} = 0. \tag{2.60}$$

Here Q_{ass} and Q_{diss} are the associative and dissociative quality functions, respectively. In the main manuscript I formulate an additive quality function. The expectation is that the best assignment maximizes the difference of $S^C(V^k_{in})$ and $S^C(V^k_{out})$ over all subgraphs, V^k. The derived quality function is the difference of internal and external cohesion of nodes with respect to its own subgraph, and hence, represents association within subgraphs and denoted as Q_{ass}:

$$Q^{ass} = \sum_{V^k} (S^C(V^k_{in}) - S^C(V^k_{out})). \tag{2.61}$$

An alternate strategy to find the best assignment is to minimize the difference of $\sum_{V^{k*}} S^C(V^{k*}_{in})$ and $\sum_{V^{k*}} S^C(V^{k*}_{out})$ for the set of all subgraphs V^k. Here, the sum over V^{k*} represents the sum over all subgraphs but V^k. This is the difference of internal and external cohesion of nodes with respect to all but its own subgraph, and hence, represents the net disassociation between V^k and set of V^{k*}. The overall dissociation between all sets of subgraphs, i.e., between all V^k and V^l such that $k \neq l$ is then given by:

$$Q^{diss} = \sum_{V^k} \sum_{V^{k*}} (S^C(V^{k*}_{in}) - S^C(V^{k*}_{out})). \tag{2.62}$$

This quality function represents the disassociation between subgraphs and I denote it as Q^{diss}. Let me denote the matrix $[(D-A)S]$ as B_1 and the matrix $[S(D-A)]$ as B_2. Before proceeding to the proofs of Equations (2.59, 2.60), I state few properties of the matrices B_1 and B_2.

Property 1 *For any node i in subgraph V^k:*

$$(S^i(V^k_{in}) - S^i(V^k_{out})) = \sum_{j \in V^k} (B_1(j,i) + B_2(i,j)). \tag{2.63}$$

Here $B_1(i,j)$ indicates the $(i,j)^{th}$ element of the matrix $[(D-A)S]$ and $B_2(i,j)$ indicates the $(i,j)^{th}$ element of the matrix $[S(D-A)]$.

Proof $\sum_{j \in V^k} B_1(j,i)$ sums over the elements of the i^{th} column of the matrix product $[(D-A)S]$ for all j belonging to V^k. Similarly, $\sum_{j \in V^k} B_2(i,j)$ sums over the elements of the i^{th} row of the matrix product $[S(D-A)]$ for all j belonging to V^k. Without loss

of generality, assume that the subgraph V^k is a set of k nodes. Using property of matrix multiplication, the $(j,i)^{th}$ element of B_1 corresponds to the vector dot product of j^{th} row of $(D-A)$ and i^{th} column of S. Also, $(i,j)^{th}$ element of B_2 corresponds to the vector dot product of i^{th} row of S and j^{th} column of $(D-A)$. It is easy to verify that $\sum_{j\in V^k} B_1(j,i)$ is of the form $\sum_m c_m^1 S(m,i), m \in V$ where $[c_1^1, c_2^1, \cdots c_m^1 \cdots c_N^1]$ is a $N \times 1$ vector of coefficients corresponding to B_1. Note that it only contains terms corresponding to the i^{th} column of S. Similarly, $\sum_{j\in V^k} B_2(i,j)$ is of the form $\sum_m c_m^2 S(i,m), m \in V$ where $[c_1^2, c_2^2, \cdots c_m^2 \cdots c_N^2]$ is a $N \times 1$ vector of coefficients corresponding to B_2. It only contains terms corresponding to the i^{th} row of S. Further, the coefficients c_m^1 are given by the column sum of the $k \times N$ matrix $(D-A)^r_{V^k}$ where $(D-A)^r_{V^k}$ represents the k rows corresponding to nodes in V^k and the coefficients c_m^2 are given by the row sum of the $N \times k$ matrix $(D-A)^c_{V^k}$ where $(D-A)^c_{V^k}$ represents the k columns corresponding to nodes in V^k. The column and row sums of entire matrix $(D-A)$ is 0 by construction. The following hold for column sums of the matrix $(D-A)^r_{V^k}$:

- The m^{th} column sum is positive if and only if $m \in V^k$ and node m has an edge connecting to at least one node $n \notin V^k$. The coefficient c_m^1 is then equal to number of links connecting $m \in V^k$ to $n \notin V^k$. This follows from observing that the only positive entries in $(D-A)^r_{V^k}$ correspond to the diagonal entries in matrix $(D-A)$ and the column sum of entire matrix $(D-A)$ is 0.

- The m^{th} column sum is 0 if $m \in V^k$ and node m has no edges connecting to nodes $n \notin V^k$ (the positive entry exactly cancels out all negative entries), or $m \notin V^k$ and has no edges connecting to nodes $n \in V^k$ (all entries are 0).

- The n^{th} column sum is negative if and only if $n \notin V^k$ and node n has an edge connecting to at least one node $m \in V^k$. The coefficient c_n^1 is then equal to number of links connecting $n \notin V^k$ to $m \in V^k$. This is true because the positive diagonal entries do not appear in the columns for $n \notin V^k$ in matrix $(D-A)^r_{V^k}$, and hence, all entries are negative and correspond to edges connecting it to nodes $m \in V^k$.

Analogously, the following hold for row sums of the matrix $(D-A)^c_{V^k}$:

- The m^{th} row sum is positive if and only if $m \in V^k$ and node m has an edge connecting to at least one node $n \notin V^k$. The coefficient c_m^2 is then equal to number of links connecting $m \in V^k$ to $n \notin V^k$. This is true as the only positive entries in $(D-A)^c_{V^k}$ correspond to the diagonal entries in matrix $(D-A)$ and the row sum of entire matrix $(D-A)$ is 0.

- The m^{th} row sum is 0 if $m \in V^k$ and node m has no edges connecting to nodes $n \notin V^k$ (the positive entry exactly cancels out all negative entries), or $m \notin V^k$ and has no edges connecting to nodes $n \in V^k$ (all entries are 0).

- The n^{th} row sum is negative if and only if $n \notin V^k$ and node n has an edge connecting to at least one node $m \in V^k$. The coefficient c_n^1 is then equal to number of links connecting $n \notin V^k$ to $m \in V^k$. This is true because the positive diagonal entries do not appear in the columns for $n \notin V^k$ in matrix $(D-A)^c_{V^k}$, and hence, all entries are negative and correspond to edges connecting it to nodes $m \in V^k$.

Now consider the term $(S^i(V_{in}^k) - S^i(V_{out}^k))$ on LHS in Equation (2.63). By definition:

$$S^i(V_{in}^k) = \sum_{E^k}(S(m,i) + S(i,m)). \tag{2.64a}$$

$$S^i(V_{out}^k) = \sum_{E^k}(S(n,i) + S(i,n)). \tag{2.64b}$$

Here E^k is the set of boundary or interfacing links for subgraph V^k such that the node pair $[m,n]$ associated with each link satisfy $m \in V^k, n \notin V^k$. $(S^i(V_{in}^k) - S^i(V_{out}^k))$ can be rewritten as $\sum_{m\in V}(\tilde{c}_m^1 S(m,i) + \tilde{c}_m^2 S(i,m)), m \in V$ where $\tilde{c}_m^1, \tilde{c}_m^2$ are positive or negative depending on $m \in V_k$ or $m \notin V_k$ respectively, and the magnitude is given by the number of boundary links in E^k which contain m in its node pair. Hence, it is clear that $c_m^1 = \tilde{c}_m^1, c_m^2 = \tilde{c}_m^2 \forall m$ and $(S^i(V_{in}^k) - S^i(V_{out}^k))$ represents $\sum_{j\in V^k} B_1(j,i) + \sum_{j\in V^k} B_2(i,j)$. ■

If I were to put all nodes into the same community, verify that all coefficients c_m^1 (given by the column sums of the $N \times N$ matrix $(D-A)$ and c_m^2 (given by the row sums of the $N \times N$ matrix $(D-A)$) are trivially 0. As a corollary, it immediately follows from that a $\sum_j B_1(j,i) = 0, \forall i$, i.e., all column sums of the matrix B_1 are equal to zero. Hence, the

sum of all elements in B_1 equals zero. Similarly, $\sum_j B_2(i,j) = 0, \forall i$, i.e., all row sums of the matrix B_2 are equal to zero. Hence, the sum of all elements in B_1 equals zero.

$$\begin{aligned} \sum_{i,j} B_1(i,j) &= \sum_{i,j} [(D-A)S](i,j) = 0 \\ \sum_{i,j} B_2(i,j) &= \sum_{i,j} [S(D-A)](i,j) = 0. \end{aligned} \tag{2.65}$$

Also, it is easy to verify that Equation (2.63) is independent of the subgraph under consideration and is generally true for any subgraph V_k . Therefore,

$$(S^i(V_{in}^{k*}) - S^i(V_{out}^{k*})) = \sum_{j \in V^{k*}} (B_1(j,i) + B_2(i,j)). \tag{2.66}$$

Here j belongs to V^{k*}, i.e., j belongs to a subgraph in the set of all subgraphs but V^k.

Theorem 2.2.1 *My main theorem in the manuscript is reformulation of aggregate flux flowing through community boundaries into the Laplacian modularity quality function:*

$$\sum_{V^k} \sum_{i \in V^k} (S^i(V_{in}^k) - S^i(V_{out}^k)) = \sum_{i,j} [(D-A)S + S(D-A)]\delta(i,j). \tag{2.67}$$

Proof Using Property 1 and summing over all nodes $i \in V^k$, I get:

$$\sum_{i \in V^k} (S^i(V_{in}^k) - S^i(V_{out}^k)) = \sum_{i \in V^k, j \in V^k} (B_1(j,i) + B_2(i,j)). \tag{2.68}$$

As $\delta(i,j)$ is 1 only if i and j belong to the same subgraph and 0 otherwise, the above equation can be rewritten as:

$$\sum_{i \in V^k} (S^i(V_{in}^k) - S^i(V_{out}^k)) = \sum_{i \in V^k, j} [(D-A)S + S(D-A)]\delta(i,j). \tag{2.69}$$

Note, this holds for any subgraph V^k. As the quality function is additive over all (hard-partitioned) subgraphs, I get:

$$\sum_{V^k} \sum_{i \in V^k} (S^i(V_{in}^k) - S^i(V_{out}^k)) = \sum_{i,j} [(D-A)S + S(D-A)]\delta(i,j). \tag{2.70}$$

This proves the theorem. ■

Theorem 2.2.2 *The relationship between the associative and dissociative quality function is:*

$$Q^{ass} + Q^{diss} = 0. \tag{2.71}$$

Proof Recall the formulation of Q^{ass}, and Q^{diss}:

$$Q^{ass} = \sum_{V^k} \sum_{i \in V^k} (S^i(V^k_{in}) - S^i(V^k_{out})). \tag{2.72}$$

$$Q^{diss} = \sum_{V^k} \sum_{V^{k*}} \sum_{i \in V^k} (S^i(V^{k*}_{in}) - S^i(V^{k*}_{out})). \tag{2.73}$$

Using Equation (2.66) and summing over all nodes $i \in V^k$, I get:

$$\sum_{i \in V^k} (S^i(V^{k*}_{in}) - S^i(V^{k*}_{out})) = \sum_{i \in V^k, j \in V^{k*}} (B_1(j,i) + B_2(i,j)). \tag{2.74}$$

The value $\sum_{i \in V^k, j \in V^{k*}} (B_1(j,i) + B_2(i,j))$ represents the disassociation between subgraph V^k and one of the remaining subgraphs V^{k*}. Adding over all subgraphs V^{k*}, I get:

$$\sum_{V^{k*}} \sum_{i \in V^k} (S^i(V^{k*}_{in}) - S^i(V^{k*}_{out})) = \sum_{i \in V^k, j \notin V^k} (B_1(j,i) + B_2(i,j)). \tag{2.75}$$

Here the RHS follows from $j \in V^{k*} \forall V^{k*} \equiv j \notin V^k$, i.e., a node $j \in V^{k*}$ over the set of all V^{k*} implies node $j \notin V^k$. Let me define an operator $\varrho(i,j)$ which is 0 if node i and j belong to the same subgraph and 1 otherwise. Then the above equation can be rewritten as:

$$\sum_{V^{k*}} \sum_{i \in V^k} (S^i(V^{k*}_{in}) - S^i(V^{k*}_{out})) = \sum_{i \in V^k, j} [(D - A)S + S(D - A)]\varrho(i,j). \tag{2.76}$$

Further, adding over all subgraphs V^k I get:

$$\sum_{V^k} \sum_{V^{k*}} \sum_{i \in V^k} (S^i(V^{k*}_{in}) - S^i(V^{k*}_{out})) = \sum_{i,j} [(D - A)S + S(D - A)]\varrho(i,j). \tag{2.77}$$

The term on LHS is precisely Q^{diss}. Using Theorem 2.4.1 and definition of Q^{ass} I have:

$$Q^{ass} = \sum_{i,j} [(D - A)S + S(D - A)]\delta(i,j). \tag{2.78}$$

Adding Equations (2.77) and (2.78) I get:

$$Q^{ass}+Q^{diss}=\sum_{i,j}[(D-A)S+S(D-A)](\delta(i,j)+\varrho(i,j))=\sum_{i,j}[(D-A)S+S(D-A)]. \tag{2.79}$$

$\delta(i,j)+\varrho(i,j)=1$ because nodes i and j can either belong to the same community or different communities and the sets are mutually exclusive. Using Equation (2.65) I have:

$$Q^{ass}+Q^{diss}=0. \tag{2.80}$$

This proves the theorem. Note the result holds independent of metric of similarity S, or in other words Q is an unbiased quality measure. Also, if A and S are both symmetrical, using elementary matrix algebra I get that $(D-A)S+S(D-A)$ is also symmetrical. As a result, it suffices to calculate $B=(D-A)S$ and obtain $(D-A)S+S(D-A)$ as $B+B^T$.

■

In this section, I proved that the sum of all elements of the Laplacian modularity matrix, $(D-A)S+S(D-A)$ is zero independent of S. Hence, the Laplacian modularity matrix shares the key feature of unbiasedness with the modularity matrix. Also, it is possible to use a host of existing modularity optimization techniques for Laplacian modularity optimization and is the topic of Section 2.2.1.

Overlap: Optimizing Q outputs a non-overlapping partition which are communities in the weak sense by construction. In the main manuscript I state that the positive terms in matrix $X^TL(S+S^T)$ resolve overlap, where X is a 0-1 indicator matrix for the (hard) partition of dimensions $|V|\times C$. $|V|$ is the number of nodes, C is the number of communities, and each row of X is all zeros except for a 1 indicating the node's membership to a community. I now provide a formal definition of overlapping nodes with respect to the strong and weak definitions of a community.

If a node associates with weak communities other than the one it is assigned to in the optimal partition, i.e., a node assigned to V^k satisfies the strong criterion, Equation (2.52) for some other subgraphs in the set $V^{k*}, k*=1,2,\cdots\backslash k$, then the node is classified as an overlapping node with membership to these communities. Figure 2.14 D,E,F displays the difference of internal and external boundary cohesion for the three communities,

V^1, V^2, V^3 identified by optimizing the Laplacian modularity. Nodes 6 and 12 are correctly assigned to their respective cliques in the optimal partition, however, have multiple connections to the subgraph V^3, and emerge as overlapping nodes with positive affinity (shaded blue) in the vector $S(V^3_{in}) - S(V^3_{out})$ (figure 2.14F).

I now prove that $C\times \mid V \mid$ matrix $X^T B_1 + (B_2 X)^T$ can efficiently resolve communities in the strong sense by checking for positive terms where X is a $[0-1]$ indicator matrix. $X^T B_1 + (B_2 X)^T$ is equivalent to $X^T(B_1 + B_2^T)$. Given a partition of nodes into *weak* communities, strong or weak membership, and hence, overlap of node i is determined by checking if the node satisfies the criterion of a *strong* community for a subgraph other than the one it is assigned to by the optimal partition, i.e., I check:

$$S^i(V^{k*}_{in}) \leq S^i(V^{k*}_{out}), \forall V^{k*}. \tag{2.81}$$

Equivalently, $S^i(V^{k*}_{in}) - S^i(V^{k*}_{out}) \leq 0, \forall V^{k*}$ or $(S^i(V^{k*}_{in}) - S^i(V^{k*}_{out})) \leq 0, \forall V^{k*}$. Using Equation (2.66), this translates to checking if:

$$\sum_{j \in V^{k*}} (B_1(j,i) + B_2(i,j)) \leq 0, \forall V^{k*}. \tag{2.82}$$

Now consider the $(k,i)^{th}$ term of the matrix $X^T(B_1 + B_2^T)$. Using elementary matrix multiplication properties, it can be verified that:

$$(X^T(B_1 + B_2^T))(k,i) = \sum_{j \in V^k} (B_1(j,i) + B_2(i,j)). \tag{2.83}$$

If node i belongs to community V^k in the strong sense if $(X^T(B_1 + B_2^T))(u,i) < 0, \forall u \in k*$ where $k*$ are the indices of subgraphs V^{k*}. It follows that presence of o positive terms in the i^{th} column implies that node i is classified as an overlapping node with membership to o subgraphs V^u corresponding to the positive indices u, s.t., $(X^T(B_1 + B_2^T))(u,i) > 0$. Thus, $X^T(B_1 + B_2^T)$ encodes all information about overlap between different subgraphs and overlapping nodes can be efficiently identified by checking for positive terms. $X^T(B_1 + B_2^T)$ can be equivalently written as $X^T(LS + (SL)^T)$ or $X^T(LS + LS^T)$ as described in the main manuscript. Using Equation (2.63), it follows that each element in $(X^T(B_1 + B_2^T))$ encodes the net connectivity flux for all nodes through each subgraph boundary, and

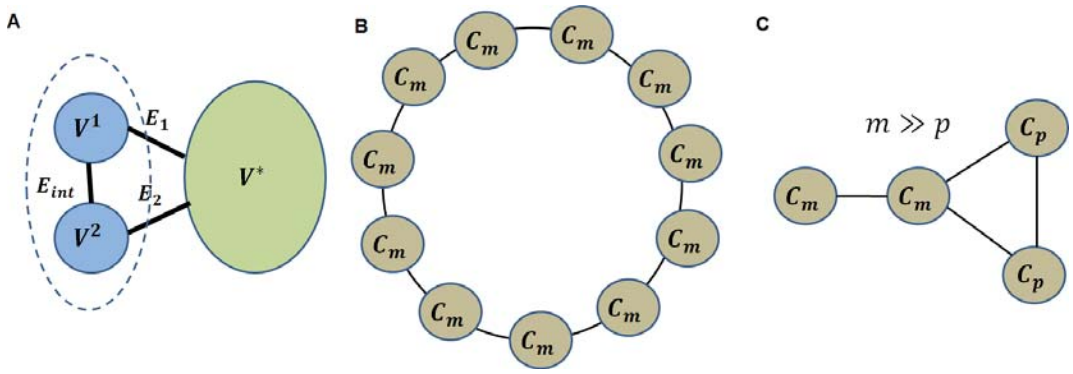


Figure 2.15. *Resolution limit. A: The Laplacian modularity framework does not suffer from the traditional resolution limit, i.e., V^1 and V_2 are not merged by the Laplacian modularity quality function if V_1 and V_2 are strong communities. B: Ring of cliques, C_m C: Pairwise identical cliques, C_m and C_p.*

hence, overlapping nodes are simply nodes with net positive connectivity flux to multiple communities as described in the main manuscript.

Optimization of Laplacian modularity: Modularity optimization is provably NP-hard and the structural similarity of the Laplacian modularity and modularity matrix makes it unlikely that an efficient algorithm exists for Laplacian modularity optimization. Several heuristics and approximate algorithms exist for modularity optimization such as spectral optimization [127], simulated annealing [128] and greedy agglomerative techniques [129]. These methods can be directly adapted for optimizing the Laplacian modularity quality function. I use the Louvain method [121] to optimize Laplacian modularity, Q, in all my experiments due to its speed and accuracy.

Louvain: The Louvain method is the current state-of-the-art method for optimizing modularity [121]. The generalized version of the method adapted for optimizing my quality function Q is a greedy optimization method that attempts to optimize Q in two steps. First, the method looks for small communities by optimizing Q locally. Second, it aggregates nodes belonging to the same community and builds a new network whose nodes are the communities. These steps are repeated iteratively until a maximum of Q is attained.

Resolution limit: I provide an analysis of resolution limit for general similarity matrices, S and discuss the resolution limit in reference to the heat kernel in the next section. Consider two partitions of the network, Y and Z. These two partitions are identical in all subgraphs represented as V^*, except in Z two subgraphs V^1 and V^2 are merged, whereas in Y they are considered as separate subgraphs (see Figure 2.15 A). Because, Q is expressed as a sum over subgraphs, the corresponding value of Q for these two partitions are then given as:

$$Q^Y = \sum_{V^*} (S^C(V^*_{in}) - S^C(V^*_{out})) + (S^C(V^1_{in}) - S^C(V^1_{out})) + (S^C(V^2_{in}) - S^C(V^2_{out})) \quad (2.84)$$

$$Q^Z = \sum_{V^*} (S^C(V^*_{in}) - S^C(V^*_{out})) + (S^C(V^{1,2}_{in}) - S^C(V^{1,2}_{out})). \quad (2.85)$$

The contribution to Q^Y and Q^Z over subgraphs V^* is the same. Then, the difference in quality of these two partitions is $\Delta Q = Q^Y - Q^Z$.

$$\Delta Q = (S^C(V^1_{in}) - S^C(V^1_{out})) + (S^C(V^2_{in}) - S^C(V^2_{out})) - (S^C(V^{1,2}_{in}) - S^C(V^{1,2}_{out})). \quad (2.86)$$

Expanding S^C as a sum over nodes I get:

$$\begin{aligned} \Delta Q = &\sum_{i \in V^1} (S^i(V^1_{in}) - S^i(V^1_{out})) + \sum_{i \in V^2} (S^i(V^2_{in}) - S^i(V^2_{out})) \\ &- \sum_{i \in V^{1,2}} (S^i(V^{1,2}_{in}) - S^i(V^{1,2}_{out})). \end{aligned} \quad (2.87)$$

To simplify the calculations, I only consider the contributions from $(D-A)S$ in the Laplacian modularity matrix and the counterpart $S(D-A)$ can be analyzed similarly. I distinguish three types of links: between V^1 and V^2 (E_{int}), between V^1 and the rest of the network V^* (E_1) and between V^2 and the rest of the network V^* (E_2). Using the property that S^i is a sum over links, I separate terms in ΔQ as follows:

$$\sum_{i \in V^1} (S^i(V^1_{in}) - S^i(V^1_{out})) = \sum_{i \in V^1} \left(\sum_{E_1} (S(m,i) - S(n,i)) + \sum_{E_{int}} (S(m,i) - S(n,i)) \right) \quad (2.88a)$$

$$\sum_{i \in V^2} (S^i(V^2_{in}) - S^i(V^2_{out})) = \sum_{i \in V^2} \left(\sum_{E_2} (S(m,i) - S(n,i)) + \sum_{E_{int}} (S(m,i) - S(n,i)) \right) \quad (2.88b)$$

$$\sum_{i\in V^{1,2}}(S^i(V^{1,2}_{in})-S^i(V^1_{out}))=\sum_{i\in V^{1,2}}\Bigg(\sum_{E_1}(S(m,i)-S(n,i))+\sum_{E_2}(S(m,i)-S(n,i))\Bigg). \tag{2.88c}$$

I now analyze ΔQ over two cases. Let $\hat{E}$ be the set of nodes induced by links E. In the first case there are no common nodes induced by links $[E_1, E_2]$, $[E_1, E_{int}]$ and $[E_2, E_{int}]$, i.e., $\hat{E}_1 \cap \hat{E}_2 = \phi$, $\hat{E}_1 \cap \hat{E}_{int} = \phi$ and $\hat{E}_2 \cap \hat{E}_{int} = \phi$. Second I analyze for the case where nodes may be common in the sets. I consider S of the form $S(i, j) > 0$ only if there is a direct connection between i and j.

Case 1: If there are no common induced nodes, then it follows that for $\sum_{E_1}$, $S(m, i) > 0, S(n, i) > 0$ only if $i \in V^1$. Similarly, for $\sum_{E_2}$, $S(m, i) > 0, S(n, i) > 0$ only if $i \in V^2$. Hence, I have:

$$\sum_{i\in V^{1,2}}\Bigg(\sum_{E_1}(S(m,i)-S(n,i))\Bigg)=\sum_{i\in V^{1}}\Bigg(\sum_{E_1}(S(m,i)-S(n,i))\Bigg) \tag{2.89a}$$

$$\sum_{i\in V^{1,2}}\Bigg(\sum_{E_2}(S(m,i)-S(n,i))\Bigg)=\sum_{i\in V^{2}}\Bigg(\sum_{E_2}(S(m,i)-S(n,i))\Bigg). \tag{2.89b}$$

Therefore these contributions cancel out in ΔQ and I am left with:

$$\Delta Q=\sum_{i\in V^1}\sum_{E_{int}}(S(m,i)-S(n,i))+\sum_{i\in V^2}\sum_{E_{int}}(S(m,i)-S(n,i)). \tag{2.90}$$

For E_{int}, m and n are complimentary for the subgraphs V^1 and V^2, i.e., an internal node to V^1 is an external node to V^2 and vice versa. Without loss of generality, let me assume m belongs to V_1 and n belongs to V_2. If I assume a constant value of $S(m, i)\forall i$, say $S(m)$, then the sum over V_1 is $d^m(V^1)S(m)$, where $d^m(V^1)$ is the number of connections of m to subgraph V^1. Similarly, sum over V_2 is $d^m(\bar{V}^1)S(m)$ where $d^m(\bar{V}^1)$ is the number of external connections of m to V^2 (in this case the external degree of node m). Hence, Equation (2.90) reduces to:

$$\Delta Q=\sum_{E_{int}}\Big((d^m(V^1)-d^m(\bar{V}^1))S(m)+(d^n(V^2)-d^n(\bar{V}^2))S(n)\Big). \tag{2.91}$$

Now under the assumption that V^1 and V^2 are LS-sets or strong communities as per the definition of Radicchi [45], $(d^m(V^1) - d^m(\bar{V}^1))$ and $(d^n(V^2) - d^n(\bar{V}^2))S(n)$ are always

greater than 0, and hence $\Delta Q > 0$. In other words, the Laplacian modularity does not suffer from the traditional resolution limit.

Case 2: In the case where some induced nodes may be common, the Equation (2.89) does not hold, and hence, no simplification occurs. So I investigate the Laplacian modularity on a per node basis so as to detail the resolution constraints on a module without any assumption on the number of common induced nodes. Let $S(i, V^k) = \sum_{j \in V^k} \big(S(i,j) + S(j,i)\big)$, i.e., $S(i, V^k)$ is the reciprocal sum of connectivity coefficients of a node i to nodes in a subgraph. Then I have:

$$\sum_{i \in V^k} (S^i(V^k_{in}) - S^i(V^k_{out})) = \sum_{i \in V^k} d^i(\bar{V}^k) S(i, V^k) - \sum_{j \in V^{k*}} d^j(V^k) S(j, V^k). \tag{2.92}$$

Here $d^i(\bar{V}^k)$ indicates the external degree of node i and $d^j(V^k)$ indicates the external degree contribution of node j to subgraph V^k. Equation (2.92) constitutes subgraph V^k's contribution to Q. Then node i's total contribution to Q, $\mathcal{S}^i(Q)$ is:

$$\mathcal{S}^i(Q) = d^i(\bar{V}^k) S(i, V^k) - \sum_{p \in k*} d^i(V^p) S(i, V^p). \tag{2.93}$$

This is because node i's positive contributions to Q occur as an internal boundary node of subgraph V^k and negative contributions occur as an external boundary node to some subgraphs in V^{k*}. I have an additional constraint that $d^i(\bar{V}^k) = \sum_{p \in k*} d^i(V^p)$, i.e., the total external connections of a node i is equal to the sum of (external) degree contributions to all other subgraphs. So for $\mathcal{S}^i(Q) \geq 0$, I get the relationship:

$$S(i, V^k) \geq \frac{\sum_{p \in k*} d^i(V^p) S(i, V^p)}{\sum_{p \in k*} d^i(V^p)}. \tag{2.94}$$

This equation intuitively reflects that the net internal connectivity of a node to a subgraph should be greater or equal to the weighted sum of net external connectivity to all other subgraphs. If the above equation holds for all nodes in a subgraph, then it is considered a module. If I consider S to be equal to the adjacency matrix A, I have $S(i, V^k) = 2d^i(V^k)$ and $S(i, V^p) = 2d^i(V^p)$. Equation (2.94) then reduces to:

$$d^i(V^k) \geq \frac{\sum_{p \in k*} (d^i(V^p))^2}{\sum_{p \in k*} d^i(V^p)}. \tag{2.95}$$

Using Cauchy-Schwarz inequality, $x_1^2 + x_2^2 \dots x_n^2 \geq (x_1 + x_2 \dots x_n)^2/n$, I get:

$$d^i(V^k) \geq \frac{\sum_{p \in k*} (d^i(V^p))^2}{\sum_{p \in k*} d^i(V^p)} \geq \frac{\left(\sum_{p \in k*} (d^i(V^p))\right)^2}{|k| \sum_{p \in k*} d^i(V^p)}. \tag{2.96}$$

Here k is the number of subgraphs node i has connections to. Using the relation $d^i(\bar{V}^k) = \sum_{p \in k*} d^i(V^p)$ I get:

$$d^i(V^k) \geq \frac{d^i(\bar{V}^k)}{|k|}. \tag{2.97}$$

Instead of requiring that the internal degree exceeds the external degree as in a strong community, the Laplacian modularity quality function for $S = A$ requires that the internal degree of a node exceeds the average external degree over subgraphs the node connects to. Further, if all nodes in a subgraph have external connections to a maximum of one other subgraph then $s = 1$ and I recover the strong definition of a community. Under these conditions, the Laplacian modularity mitigates the traditional resolution limit.

Figure 2.15 demonstrates the efficacy of my construct against the traditional resolution limit [119] for $S = A$. On a ring of cliques C_m of size m, my approach always identifies the correct partition with highest Q_{LS} irrespective of size of m for $m > 2$. In contrast, modularity provably always merges cliques and I observed that Infomap merged cliques for small m. On pairwise identical cliques C_m, C_p of size m and p respectively connected as shown in Figure 2.15, Q_{LS} never merged the smaller cliques C_p irrespective of the difference in sizes, unlike modularity which merges the smaller cliques C_p into one subgraph.

Multi-resolution methods: I briefly mention multi-resolution extension of the Laplacian modularity framework akin to multi-resolution modularity [130]. I differentiate multi-resolution from hierarchy, i.e., multi-resolution methods output partitions at all resolutions whereas I contend hierarchy to be in reference to topological scales of the network. The multi-resolution extension of the Laplacian modularity has a straight-forward extension using resolution parameter ω as:

$$Q = \sum_{i,j} [(D - \omega A)S + S(D - \omega A)]\delta(i,j) = \sum_{i,j} [L_\omega S + S L_\omega]\delta(i,j). \tag{2.98}$$

Where L_ω is the resolution dependent Laplacian. $\omega = 0$ outputs the trivial partition where all nodes are grouped into one community. Varying ω in the range $[0, 1]$ outputs multi-resolution partitions. In order to determine the relevant hierarchies among the set of partitions output by varying, ω, I construct a stability curve using the equation:

$$r(\omega) = max_X trace[X^T B_\omega X]. \tag{2.99}$$

where B_ω is the matrix given by $[L_\omega S + S L_\omega]$. Smaller values of ω favor coarser partitions. However, the ω lacks an intuitive interpretation, unlike time t, in the Laplacian modularity using the heat kernel. I shall demonstrate the stability curve for alternate definitions of connectivity potentials in results section.

Another strategy to account for resolution is to consider averaging over longer connected neighborhoods by considering the Laplacian corresponding to powers of A, i.e., L corresponding to $A^2, A^3, \cdots$. For example, the quality function corresponding to A^2 is:

$$Q = \sum_{i,j} [(D_2 - A^2)S + S(D_2 - A^2)]\delta(i, j). \tag{2.100}$$

Here D_2 are the row or column sums of the symmetric matrix, A^2. Similarly, I can consider higher powers of the Laplacian with the same structural properties of L, i.e., the row and column sums of L^n are 0. The corresponding quality functions are of the form:

$$Q_n = \sum_{i,j} [L^n S + S L^n]\delta(i, j). \tag{2.101}$$

Where Q_n is the quality function corresponding to the n^{th} power of the Laplacian. However, there is no way to determine relevant hierarchies from the partitions output by the Laplacian corresponding to different powers of A, Equation (2.100), or by considering higher order Laplacians, L^n of varying n, Equation (2.101). As a result, I advocate the Laplacian modularity using the heat kernel as a principled way to determine relevant hierarchies using the stability curve, the topic of my next section.

Resistor network interpretation of Laplacian modularity: A large connectivity flux through a community boundary is encoded as high connectivity density among the internal nodes in the Laplacian modularity matrix due to my network analog of Gausss law. This can

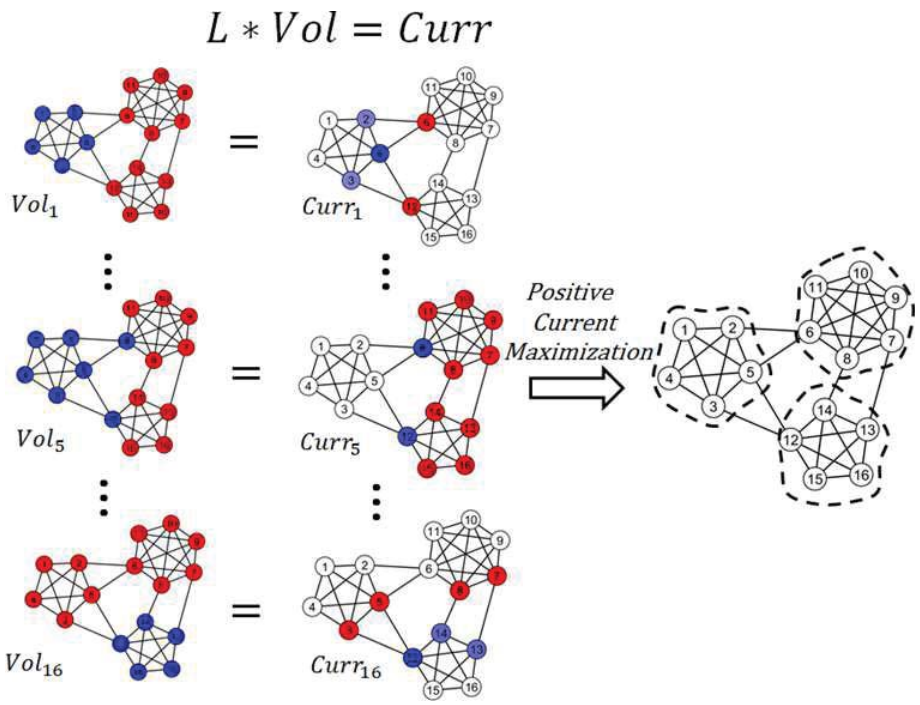


Figure 2.16. *Resistor network interpretation of Laplacian modularity: The plots to the left show the voltage applied to each node in the network. Blue indicates a unit positive potential applied to a node and its neighbors, whereas red indicates a grounded node. The plots in the middle show the net current flow through each node in the network as a result of the applied voltages on nodes in the network. The current flow changes from positive to negative as the node colors change from blue to red. The plot on the right indicates that I am in effect maximizing the net positive current in the network to identify modules in my Laplacian modularity framework.*

also be interpreted using resistor networks. The well-known relationship between current vector $Curr$, voltage vector Vol and the Laplacian L in resistor networks is [131]:

$$L * Vol = Curr. \tag{2.102}$$

Now if I were to sequentially apply voltages to each node, where the i^{th} voltage vector corresponds to the connectivity values (the i^{th} row and column of S), then the Laplacian modularity matrix is equal to the net current flow matrix, i.e., the i^{th} column of B_1 is the current induced by applying voltages equal to i^{th} column of S at the nodes and the i^{th}

column of B_2 is the current induced by applying voltages equal to i^{th} row of S at the nodes. Maximizing the Laplacian modularity matrix effectively translates to finding subgraphs such that the positive current flow is maximized. Note that the net current flow in the entire network is zero. This is graphically illustrated in Figure 2.16. The plots on the left show the applied voltage and the induced current at the different nodes due to Equation (2.102). All induced currents are succinctly represented by the Laplacian modularity matrix, and optimizing Q results in a partition wherein the positive currents are maximized (Plot to the right in Figure 2.16). This is intuitively interpreted as maximizing the charge distribution within modules using Gauss' law.

2.2.2 Laplacian modularity using heat kernel

In this section I discuss the heat kernel, the solution to the diffusion equation in context of random walks and Gauss's law for community boundary identification. I mention properties of the Laplacian modularity using the heat kernel and implementation details for identifying the relevant hierarchies from the stability curve.

Diffusion based Laplacian modularity: If I take $S = H_t$, then the resulting Laplacian modularity matrix is $(D-A)H_t + H_t(D-A)$. However, I only consider $B_t = (D-A)H_t$ for analysis as B_t has a precise physical interpretation as the rate of diffusion matrix. In all my subsequent discussions, I term the corresponding quality function, Q_t as the diffusion based Laplacian modularity abbreviated as DLM. It is worthwhile to note that the random walk Laplacian hints at an alternate strategy to construct the normalized version of the Laplacian modularity quality function, given as:

$$\tilde{Q} = \sum_{i,j}[(I - D^{-1}A)S + S(I - D^{-1}A)]\delta(i,j). \tag{2.103}$$

Where $I - D^{-1}A$ is the row normalized random walk Laplacian. The column normalized counterpart, $I - AD^{-1}$ can also be used instead of $I - D^{-1}A$ in the above equation. The properties of the Laplacian modularity quality function, Q, such as unbiasedness hold true for the normalized Laplacian modularity quality function, $\tilde{Q}$. If I take $S = H_t$ in the above equation, it can be proved that $(I - D^{-1}A)H_t = H_t(I - D^{-1}A)$ (see Property 2). Hence,

the node-to-boundary cohesion matrix, $H_t(I - D^{-1}A)$ contains no additional information. It suffices to only consider the symmetric form $(D - A)H_t$ in the DLM. The definitions of strong and weak communities under the DLM construct are discussed next.

Random walks have been used extensively for community detection [48, 132, 133], all following the common intuition of a random walker getting trapped in densely connected parts corresponding to communities. I demonstrate that the DLM quality function has a precise probabilistic interpretation in terms of random walks, an alternate viewpoint to the cohesion to subgraph boundaries as discussed in the main manuscript. The basic quantity to consider is the probability $p_t(i, j)$ of a random walker to hop from node i to j in time t. This corresponds to the probability after t discrete jumps or Poisson number of jumps with mean t for discrete and continuous time random walks respectively [134]. Intuitively, a random walker tends to visit nodes in its own subgraph more often than nodes belonging to another subgraph. I formalize this intuition by comparing the frequency of a random walker starting from a node to visit nodes in its own subgraph relative to the frequency of visiting the rest of the nodes at time t. In order to facilitate pairwise comparison, I consider contributions from random walkers starting from two disjoint sets of internal V_{in}^k and external V_{out}^k nodes belonging to the interfacing links or edge-cuts, i.e., links connecting the subgraph to the rest of the network. Formally, $H_t^i(V_{in}^k)$ and $H_t^i(V_{out}^k)$ are the sum of probabilities of independent random walks starting from the internal and external nodes of subgraph V^k to reach node i in time t, and can be viewed as random-walk counterparts of intra-community and inter-community link counts. Let the set of interfacing links for subgraph V^k be E^k such that the node pair $[m, n]$ associated with each link satisfy $m \in V^k, n \notin V^k$. Note m is an internal node ($m \in V_{in}^k$) and n is an external node ($n \in V_{out}^k$). Mathematically, then the two contributions are represented as:

$$H_t^i(V_{in}^k) = \sum_{E^k} p_t(m, i) \tag{2.104a}$$

$$H_t^i(V_{out}^k) = \sum_{E^k} p_t(n, i). \tag{2.104b}$$

Intuitively, the two quantities represent the sum of probabilities of two sets of $|E^k|$ independent random walkers starting from internal and external nodes to reach the node being eval-

uated for. In the case of the time-dependent heat kernel, boundary cohesion as described in the main manuscript is synonymous with consensus among networked multi-agent systems, where higher values indicate greater cooperation of a node with seeded agents at time t [96]. Equipped with these definitions, I proceed to define communities in the strong and weak sense.

Community in the strong sense The subgraph V^k is a community in a strong sense if:

$$H_t^i(V_{in}^k) > H_t^i(V_{out}^k), \forall i \in V^k. \tag{2.105}$$

In a strong community, the sum of probabilities of the set of random walkers starting from internal nodes to visit each node of the subgraph V^k is greater than the sum of probabilities probability of the set of random walkers starting from external nodes at time t. From an opinion dynamical perspective, in a strong community there is consensus among internal agents respecting the belongingness of each agent at time t.

Community in the weak sense The subgraph V^k is a community in a weak sense if:

$$\sum_{i \in V^k} H_t^i(V_{in}^k) > \sum_{i \in V^k} H_t^i(V_{out}^k). \tag{2.106}$$

In a weak community, the sum of probabilities of the set of random walkers starting from internal nodes of the subgraph V^k to collectively visit the nodes of subgraph V_k is greater than the sum of probabilities of the set of random walkers starting from the external nodes. Overall, these definitions model coordinated consensus among internal agents in all subgraphs with regard to community membership, at time t. The optimal partition satisfying the above definition is the DLM quality function:

$$Q_t = \sum_{i,j} [(D - A)H_t]\delta(i, j). \tag{2.107}$$

Where I represent $(D - A)H_t = B_t$ and term it the DLM matrix. I now state a few additional properties of the DLM matrix, B_t.

Property 2 *Although the matrix H_t is asymmetrical in general, the DLM matrix is symmetrical, i.e., for all node pairs (i, j) I have:*

$$B_t(i, j) = B_t(j, i). \tag{2.108}$$

Proof I prove the above property using spectral decomposition of the DLM matrix. I rewrite B_t as:

$$\begin{aligned} B_t &= (D - A)H_t \\ &= D(I - M)H_t \\ &= D^{1/2}(I - N)D^{1/2}H_t \\ &= D^{1/2}\mathcal{L}D^{1/2}H_t. \end{aligned} \tag{2.109}$$

I have used $M = D^{-1}A$ and $N = D^{1/2}MD^{-1/2}$ and $\mathcal{L} = I - N$. H_t is given as $e^{-t\mathbb{L}}$. Substituting the spectral representations I get:

$$\begin{aligned} B_t &= D^{1/2}\mathcal{L}D^{1/2}e^{-t\mathbb{L}} \\ &= D^{1/2}\Phi\Lambda\Phi^T D^{1/2}D^{-1/2}\Phi e^{-t\Lambda}\Phi^T D^{1/2}. \end{aligned} \tag{2.110}$$

Using the orthogonality of the eigenvectors $\Phi^T\Phi = I$, the final spectral representation for B_t is $D^{1/2}\Phi\Lambda e^{-t\Lambda}\Phi^T D^{1/2}$. Taking the transpose of the derived matrices, I verify that the matrix is symmetrical:

$$\begin{aligned} B_t^T &= (D^{1/2}\Phi\Lambda e^{-t\Lambda}\Phi^T D^{1/2})^T \\ &= (D^{1/2})^T(\Phi^T)^T(e^{-t\Lambda})^T\Lambda^T\Phi^T(D^{1/2})^T \\ &= D^{1/2}\Phi e^{-t\Lambda}\Lambda\Phi^T D^{1/2} \\ &= B_t. \end{aligned} \tag{2.111}$$

Note, $D, e^{-t\Lambda}, \Lambda$ are all diagonal matrices. Symmetry implies the row sums, like the column sums of B_t are all 0. I also prove the same for discrete random walks where the similarity matrix at time t is given as M^t. The discrete random walk helps understand the relation between DLM and modularity as follows:

$$\begin{aligned} B_t^{discrete} &= D^{1/2}\mathcal{L}D^{1/2}M^t \\ &= D^{1/2}\Phi\Lambda\Phi^T D^{1/2}D^{-1/2}\Phi\Lambda'^t\Phi^T D^{1/2}. \end{aligned} \tag{2.112}$$

The spectral representation for $B_t^{discrete}$ is $D^{1/2}\Phi\Lambda(I-\Lambda)^t\Phi^T D^{1/2}$ and I have:

$$\begin{aligned}(B_t^{discrete})^T &= (D^{1/2}\Phi\Lambda(1-\Lambda)^t\Phi^T D^{1/2})^T \\ &= (D^{1/2})^T(\Phi^T)^T((1-\Lambda)^t)^T\Lambda^T\Phi^T(D^{1/2})^T \\ &= D^{1/2}\Phi(I-\Lambda)^t\Lambda\Phi^T D^{1/2} \\ &= B_t^{discrete}.\end{aligned} \tag{2.113}$$

The spectral decomposition can be used to prove $(I-D^{-1}A)H_t = H_t(I-D^{-1}A)$ as follows:

$$\begin{aligned}(I-D^{-1}A)H_t &= D^{-1/2}(I-N)D^{1/2}H_t \\ &= D^{-1/2}\mathcal{L}D^{1/2}e^{-t\mathbb{L}} \\ &= D^{-1/2}\Phi\Lambda\Phi^T D^{1/2}D^{-1/2}\Phi e^{-t\Lambda}\Phi^T D^{1/2} \\ &= D^{-1/2}\Phi\Lambda e^{-t\Lambda}\Phi^T D^{1/2}.\end{aligned} \tag{2.114}$$

$$\begin{aligned}H_t(I-D^{-1}A) &= H_t D^{-1/2}(I-N)D^{1/2} \\ &= e^{-t\mathbb{L}}D^{-1/2}\mathcal{L}D^{1/2} \\ &= D^{-1/2}\Phi e^{-t\Lambda}\Phi^T D^{1/2}D^{-1/2}\Phi\Lambda\Phi^T D^{1/2} \\ &= D^{-1/2}\Phi\Lambda e^{-t\Lambda}\Phi^T D^{1/2}.\end{aligned} \tag{2.115}$$

■

In the dissertation I showed that the DLM matrix is a scaled derivative of the diffusion matrix, i.e., $(D-A)H_t$ is $-D$ times the derivative of $e^{-t\mathbb{L}}$. This interpretation enables me to make an implicit connection with diffusion which underscores the dynamics governing community exploration. The physical analogy of the diffusion equation is that a commodity (heat, current) flows from a high density region to a low density region until equilibrium and flow rate is dependent on the negative of the density gradient [21]. Consider a network with N node on which I independently administer a substance of d_i units to node i at time $t = 0$ and the ground the rest of the nodes. I do this sequentially for all nodes and observe the dynamics of the diffusion process over time. Then the matrix $-(D-A)e^{-t\mathbb{L}}$ precisely encapsulates the rate of flow information for all node pairs of this physical

process. Tailoring Equation (2.5) so as to account for the amount of substance administered at each node, I get:

$$\begin{aligned} H_t &= [e^{-t\mathbb{L}}]^T D H_0 \\ &= [e^{-t\mathbb{L}}]^T D^T H_0 \\ &= [De^{-t\mathbb{L}}]^T H_0. \end{aligned} \tag{2.116}$$

Differentiating, I have:

$$\frac{dH_t}{dt} = [-D\mathbb{L}e^{-t\mathbb{L}}]^T H_0. \tag{2.117}$$

As time grows tightly connected nodes reach a local equilibrium with and long scale flow comes into play [135]. Hence, the intuition behind using the rate of diffusion for finding community structure is to minimize the rate of flow $\frac{dH_t}{dt}$ within well connected groups or equivalently maximize the negative of the diffusion flux matrix $D\mathbb{L}e^{-t\mathbb{L}}$. Hence, I term the matrix $D\mathbb{L}e^{-t\mathbb{L}}$ to be the DLM matrix, B_t which seeks maximize the association within groups, i.e., is a *'modularity'* criterion akin to modularity. The property of the all columns of B_t summing to zero physically follows from flow conservation in diffusion systems.

Overlap: $C\times \mid V \mid$ matrix $X^T B_t$ can efficiently resolve communities in the strong sense and hence overlap for DLM, as in the Laplacian modularity framework. A distinctive feature of my approach to determine overlap in networks using DLM is that it progressively considers contribution from paths of all lengths in contrast to a local edge density evaluation as in [41, 136]. This has the desirable effect of uncovering overlapping nodes dynamically with time in contrast to all other approaches. In the main manuscript I defined overlapping nodes as those which individually satisfy the strong criterion of a community. In order to gain an intuition of the dynamic process revealing overlap in the DLM approach, I provide

an alternate viewpoint in terms of path length distribution. Consider each element of DLM matrix for continuous time random walks expressed in terms of path length probabilities:

$$\begin{aligned} B_t(i,j) &= [D-A]H_t \\ &= D(I-M)e^{-t\mathbb{L}} \\ &= D(I-M)[e^{-t}\sum_{k=0}^{\infty} M^k(i,j)\frac{(t)^k}{k!}] \\ &= De^{-t}\left(\sum_{k=1}^{\infty}[\frac{(t)^k}{k!}-\frac{(t)^{k-1}}{(k-1)!}]M^k(i,j)+I\right) \\ &= \sum_{k=1}^{\infty} f_{ij}(t,k)M^k(i,j). \end{aligned} \tag{2.118}$$

Here each element of$e^{-t\Delta_{\alpha M}}$ is expanded using Maclaurin series expansion of $e^{-t(I-M)}$ to get $B_t(i,j)$ in terms of $M(i,j)$. $f_{ij}(t,k)$ is a function of time t which absorbs the constants and the final expression validates that each term of $B_t(i,j)$ is a combination of all path lengths from 1 to ∞. The membership of a node j to subgraph V^k is decided by the value of $\sum_{i\in V^k} B_t(i,j)$ which again is a dynamic function of time:

$$\sum_{i\in V^k} B_t(i,j) = \sum_{i\in V^k}\sum_{k=1}^{\infty} f_{ij}(t,k)M^k(i,j). \tag{2.119}$$

Consequently, overlap emerges dynamically based on probabilistic paths of all length from nodes in V_k to node j, a feature unique to my method. As I consider a weighted sum over all possible path lengths instead of shortest path length, overlap determination is robust to small topological changes in the network. This is desirable when the network has spurious or missing links as common in many biological networks.

Hierarchy: My formalism demonstrates that overlap and hierarchy are complimentary to each other in the sense that hierarchical communities are completely overlapping communities at a coarser scale of analysis. Although this viewpoint reconciles the antagonism between overlap and hierarchy, it is important to determine which partitions are relevant over the time range $t = [0,\infty]$. In the main text, I introduce the stability curve $r(t)$ to determine persistent partitions over range of analysis. It is given by:

$$r(t) = max_X trace[R_t]. \tag{2.120}$$

Where R_t is the clustered matrix for partition X given by $R_t = \frac{1}{2|E|} X^T[(D-A)H_t]X$. Although my stability function shares several features of the stability function described in [120], the dynamics governing the clustering process are fundamentally different. I state the similarity between these stability functions here and later describe the differences in Section 2.2.2.

The stability curve organizes a hierarchy of fine to coarse communities as time increases, with the trivial N way partition at time $t = 0$ and 2-way partition at time $t = \infty$ which I prove next.

Property 3 *Let B_t represent the DLM matrix, $(D-A)H_t$. The associated quality function Q_t partitions the graph into two communities in the limit $t \to \infty$ and places each node in a separate community for $t = 0$.*

All nodes are placed in individual communities, if and only if merging two nodes leads to an inferior quality of partition resolved by value $\tilde{Q}$. Hence for the case $t = 0$, it suffices to prove that all non-diagonal elements are negative, i.e., $C = N$, *iff* $[D-A]H_0(i,j) \leq 0$, $\forall i \neq j$. H_t is equal to the identity matrix I at time $t = 0$ for both continuous and discrete time random walk. Hence, $B_0 = [D-A]$. As D is a diagonal matrix with positive entries and $[D-A]$ considers the negative of the adjacency matrix A with all positive entries, it is clear that the only positive entries in $[D-A]$ lie on its trace. Hence, all non-diagonal entries are negative leading to $C = N$ and $\tilde{Q}_0 = 1$. I resort to the spectral decomposition of B_t in order to show $\tilde{Q}_t$ outputs a 2-partition as $t \to \infty$. As $t \to \infty$, barring degenerate eigenvalues, the DLM matrix is dominated by the normalized fielder vector given by:

$$\begin{aligned}
\lim_{t\to\infty} B_t(i,j) &= \lim_{t\to\infty} [D-A]H_{t\to\infty}(i,j) \\
&= \lim_{t\to\infty} D(I-M)e^{-t\Delta_M} \\
&= \lim_{t\to\infty} D(D^{-1/2}\Phi\Lambda\Phi^T D^{1/2})(D^{-1/2}\Phi e^{-t\Lambda}\Phi^T D^{1/2}) \\
&= \lim_{t\to\infty} D^{1/2}\Phi\Lambda e^{-t\Lambda}\Phi^T D^{1/2} \\
&\approx D^{1/2}\Phi_2(\lambda_2 e^{-t\lambda_2})\Phi_2^T D^{1/2} \\
&= \sqrt{d_i d_j}\lambda_2 e^{-t\lambda_2}\Phi_{2,i}\Phi_{2,j}.
\end{aligned} \tag{2.121}$$

The approximation of $D^{1/2}\Phi\Lambda e^{-t\Lambda}\Phi^T D^{1/2}$ is $D^{1/2}\Phi_2(\lambda_2 e^{-t\lambda_2})\Phi_2^T D^{1/2}$ at long times because the eigenvalue λ_2 dominates all eigenvalues asymptotically, including λ_1 ($\lambda_i e^{-t\lambda_i} = 0$ for $\lambda_1 = 0$). Let $r(t, X)$ be the stability curve for partition X. Now, consider the finest partition into N communities with X_f equal to an identity matrix of size N. Then the asymptotic value of stability curve for X_f as $t \to \infty$ is given by:

$$r(t, X_f) \approx \frac{\lambda_2 e^{-t\lambda_2}}{2 \mid E \mid} \sum_i d_i \Phi_{2,i}^2. \tag{2.122}$$

The value for the partition X_{nf} where I merge 2 nodes i and j into a single community and assign the rest of the nodes independent communities is:

$$r(t, X_{nf}) \approx \frac{\lambda_2 e^{-t\lambda_2}}{2 \mid E \mid} \big(\sum_{k \neq i,j} d_k \Phi_{2,k}^2 + (\sqrt{d_i}\Phi_{2,i} + \sqrt{d_j}\Phi_{2,j})^2 \big). \tag{2.123}$$

The difference of these values is:

$$r(t, X_{nf}) - r(t, X_f) \approx \frac{\lambda_2 e^{-t\lambda_2}}{\mid E \mid} \sqrt{d_i d_j}\Phi_{2,i}\Phi_{2,j}. \tag{2.124}$$

Which only increases if $sign[\Phi_{2,i}] = sign[\Phi_{2,j}]$ as $\Phi_{2,i}\Phi_{2,j}$ becomes positive. Inductively, the quality function $\tilde{Q}$ is maximized by two-way partition based on signs of Fielder vector (eigenvector corresponding to λ_2) as $t \to \infty$. Although, I proved the property for continuous walks, the proof for discrete walks is analogous.

An important property of the DLM matrix is the monotonicity of the diagonal terms. This enables me to analyze stability without resorting to the minimum over time as in the stability formalism of [120]. The monotonicity of the diagonal terms $B_t(i, i)$ follows from spectral decomposition:

$$\begin{aligned} B_t(i, j) &= \sqrt{d_i d_j} \sum_{k \geq 2} \lambda_k e^{-t\lambda_k} \Phi_{k,i}\Phi_{k,j} \\ B_t(i, i) &= d_i \sum_{k \geq 2} \lambda_k e^{-t\lambda_k} \Phi_{k,i}^2. \end{aligned} \tag{2.125}$$

I have $\Phi_{k,i}^2 > 0$ and $\lambda_k e^{-t\lambda_k} > 0$ and in addition $\lambda_k e^{-t\lambda_k}$ is monotonic over time for $0 < \lambda_k < 2$. Hence, the diagonal terms decrease over time and diagonal becomes the zero vector as $t \to \infty$. The monotonicity directly implies that Q_t is maximum at time, $t = 0$

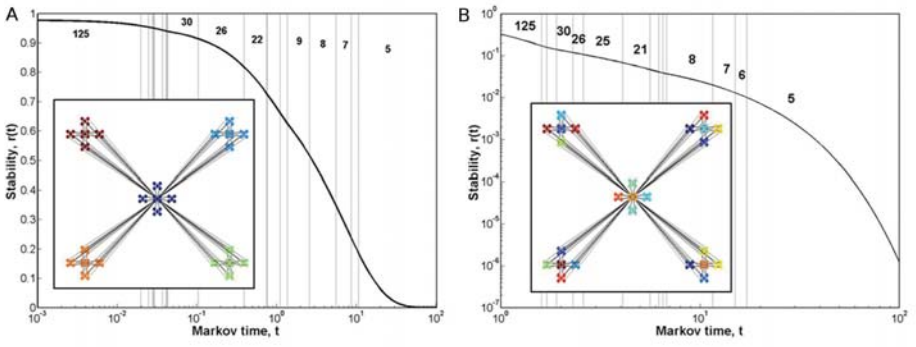


Figure 2.17. *Comparing stability curves. A: Stability curve using clustered autocovariance [120] for a hierarchical, scale-free graph proposed in [137] with 125 nodes. The curve identifies the natural 5-way partition (inset) but fails to identify the finer 25-way partition for all values of the resolution parameter t. The 26-way partition which is identical to the 25-way partition with 5 nodes in all but one community, in which the central node is assigned to an independent community. The stability value for this 26-way partition exceeds that of the 25-way partition for all t, and hence, the autocovariance stability function fails to identify the relevant hierarchical organization of the network. B: My stability function identifies the 25-way partition (inset) along with the 5-way partition to be the persistent over relatively longer time spans.*

with magnitude 1 (the trace of the clustered matrix $X_f^T D - AX_f = D - A$ is equal to $2|E|$).

A compelling validation of my approach to identify hierarchy is displayed in Figure 2.17. My stability curve for a hierarchical (figure 2.17A), scale-free graph [137] with 125 nodes identifies the 25-way partition along with the 5-way partition to be the persistent over relatively long time spans. Figure 2.17B shows the stability curve for the clustered autocovariance [120], a generalization of the modularity to identify topological scales, optimized over entire time range. I observe that it identifies the natural 5-way partition but fails to identify the finer 25-way partition for all values of the resolution parameter t.

Automatic community identification: For the purpose of constructing the stability curve $r(t)$, I first discretize t over the range $[0, t_{max}]$ into say τ distinct values. I run Louvain for each value of t and obtain a set of τ partitions. I extract the set of unique partitions from

these τ partitions as some (persistent) partitions naturally occur more than once. I draw a stability curve $r(t, X)$ for each unique partition and the optimal stability curve follows Equation (2.120).

The thrust of my framework to identify communities is based on the assumption that there is no best partition in real world networks but several valid community assignments which I discern using the parametric stability curve, the parameter being time, t. In the main body of the dissertation, I establish that partitions optimal over long time windows of the stability curve correspond to the set of relevant partitions of the network. However, for fair comparison to other parameter free community detection methods, I provide a method to obtain the most relevant or best partition from the stability curve in the continuous time case. This partition is used to compare my approach to the other methods in the main manuscript. A naive analysis based on length of persistent time window of partitions in the stability curve will always output the 2-way partition to be most relevant as it persists over the time range $[t_2, \infty]$ where t_2 is the smallest time value for which the 2-partition becomes optimal. Hence, it is imperative to discern whether the 2-way partition is a sound partition of the network in a practical context or an artifact of the stability curve. For this purpose, I explore my network over a limited time horizon $[t_{min}, t_{max}]$ with the expectation that communities identified at $t < t_{min}$ and $t > t_{max}$ are second fiddle to the overall network analysis. Recall that at $t = 0$, the number of identified communities is the trivial N-way partition, assigning each node to its own community. This partition is ignored by setting t_{min} to be equal to the smallest value of t for which the number of identified communities is less that N. Naively fixing t_{max} to a constant value without regard to graph topology could lead to missing vital information present in the analysis beyond t_{max}. Hence, it is reasonable to assert that the time t_{max} should be dependent on a global property of the graph and guarantee random walkers have explored a *sufficient* horizon of the graph. There exists several possible choices, but a heuristic that works well in practice is to take t_{max} to be equal to the relaxation time of the Markov chain or when the Markov chain is close to the steady state distribution. Mathematically, the relaxation time is the inverse of the asymptotic rate of convergence to the stationary distribution and equal to the inverse of

the spectral gap, $1/\lambda_2$ for continuous time chains and $1/(\lambda_2')$ for discrete time chains [93]. Thus, I first discretize t with logarithmic spacing in the time range $[t_{min}, t_{max}]$ and then run the Louvain algorithm on the corresponding DLM matrices for these values of t. I set the number of discrete values of t to be 200 for all networks. Next, the partitions are used to construct the stability curve and subsequently ranked based on the length of the persistent time window. The length of time window is set to be equal to the difference of $log(t_{end})$ and $log(t_{int})$ where t_{int} and t_{end} are the start and end values of the time t over which a given partition is optimal. I choose the logarithmic time scale because the DLM matrix at time t is an exponential function of t and the logarithm is the reciprocal function of exponentials, i.e., the exponentially weighted time t scales linearly on a logarithmic scale for comparison between partitions. The partition with maximum $log(t_{end}) - log(t_{int})$ value is chosen to be the most relevant hierarchical partition.

After extracting the hard maximally persistent partition over the time window $[t_{int}, t_{end}]$ in the stability curve, I obtain overlapping communities at each discrete value of t falling in the interval $[t_{int}, t_{end}]$ using the method described in Section 2.2.1. These overlapping partitions obtained for different values of $t \in [t_{int}, t_{end}]$ may not be mutually consistent, as overlap may dynamically emerge or disappear based on path length distribution over the time interval. I resolve this conflict using the magnitude of $X_o^T B(k,i)$ as it is indicative of the strength of association of an overlapping node i with the subgraph V_k it overlaps with, where X_o is the indicator matrix for an overlapping partition. Hence, the positive values indicate the relative membership strength of an overlapping node with its corresponding subgraph, i.e., leads to *fuzzy* overlap. I evaluate the fuzzy overlap of each overlapping node in the range $t \in [t_{int}, t_{end}]$. This dynamically varying fuzzy overlap is transcribed to hard overlap by thresholding the normalized magnitude of association as follows:

$$(i \in V^u) \iff \big(\frac{X_o^T B(u,i)}{X_o^T B(k,i)} \geq \epsilon\big) \forall t \in [t_{int}, t_{end}]. \tag{2.126}$$

Where ϵ is a threshold parameter which discerns the minimum strength of relative association of a node i with subgraph V^u (normalized by association strength with the subgraph

V^k, node i was assigned in the hard partition) necessary for *hard* overlap. It is easy to verify that $\epsilon = 1$ recovers the hard partition as $\frac{X_o^T B(u,i)}{X_o^T B(k,i)}$ is 1 only for $u = k, \forall t \in [t_{int}, t_{end}]$.

The steps for identifying the most relevant communities are as follows:

- Discretize time t over time range $[t_{min}, t_{max}]$, where t_{min} is the value of time t such that the number of identified communities is less than number of nodes in the network and $t_{max} = 1/(\lambda_2')$.
- Run the Louvain algorithm on the DLM matrices corresponding to the discretized values of t and obtain a set of partitions.
- Calculate the stability values, and hence, the stability curve for the network using the set of partitions.
- Extract the most relevant partition from set of partitions optimal over different time windows using the length of persistence, i.e., partition with maximum $log(t_{end}) - log(t_{int})$ value.
- Find the overlapping nodes for a given threshold value ϵ using Equation (2.126)

Figure 2.18 demonstrates DLM on a synthetic network of 29 nodes as shown in the main manuscript. The stability curve for the network in Figure 2.18 B shows two persistent basins enclosed within dotted lines corresponding to the 3-way partition and 6-way partition. The 3 and 6 communities along with the corresponding overlap at $\epsilon = 0.5$ are shown Figure 2.18 C,D. I next discuss the computational complexity of my method and provide approximate techniques for constructing the DLM matrix for very large graphs.

Computational complexity and approximation: I analyze the complexity of constructing the DLM matrix. In order to calculate the B_t exactly, I need to find H_t. For discrete time random walks, H_t is directly computable in $\mathcal{O}(cEt)$, or estimated in $\mathcal{O}(Kt)$ with accuracy $\mathcal{O}(c/\sqrt{K})$ through K random walks of length t [133]. For continuous time walks, calculating matrix exponential $e^{-t\mathbb{L}}$ unfortunately has complexity $\mathcal{O}(N^3)$. Instead of recalculating $e^{-t\mathbb{L}}$ for different values of t, I decompose the normalized Laplacian $\mathcal{L}$ into its eigenvalues and eigenvectors. The eigenvectors serve as a common orthonormal basis, and

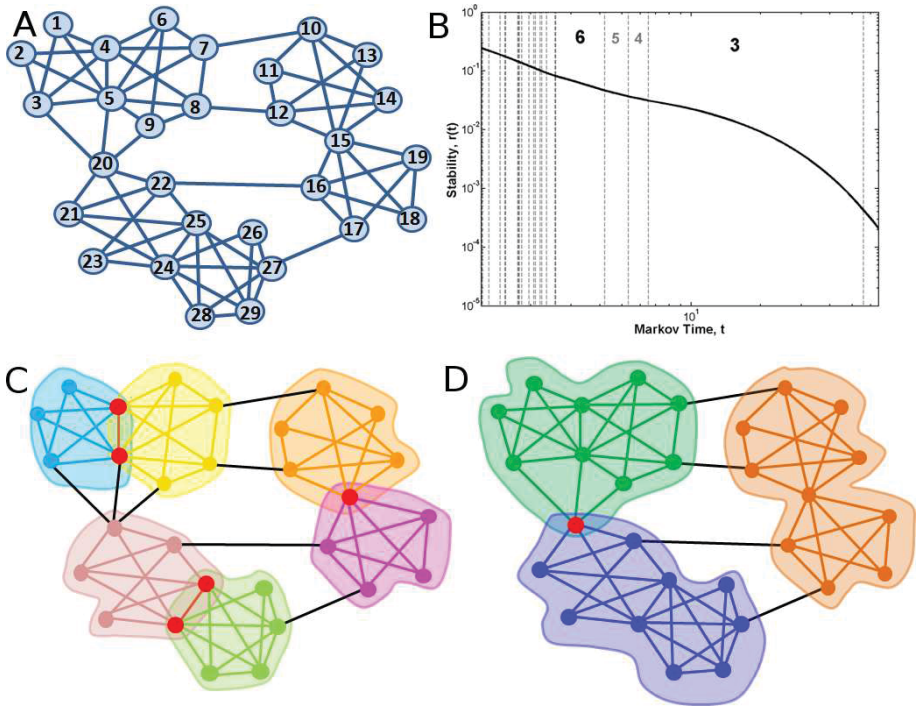


Figure 2.18. *Community detection using DLM. A: Network of 29 nodes. B: Stability curve as derived by DLM. C: 6 identified communities with overlapping nodes highlighted in red at threshold $\epsilon = 0.5$. D: 3 identified communities with overlapping nodes in red at threshold $\epsilon = 0.5$.*

hence, $e^{-t\mathbb{L}}$ is efficiently calculated by changing the weighting function of the eigenvalue matrix, i.e., $e^{-t\Lambda}$ in $D^{-1/2}\Phi e^{-t\Lambda}\Phi^T D^{1/2}$. Calculating the full spectral decomposition of the Laplacian has complexity $\mathcal{O}(N^3)$. However, as most real world networks are sparse, complexity reduces to $\mathcal{O}(N^2)$ using the efficient Lanczos method [18]. It is approximated in $\mathcal{O}(kN)$ by calculating the first (smallest) K eigenvalues and corresponding eigenvectors. The error bound is (generally) linear in $1/K$ [2] due to the decreasing smoothness property of the eigenvectors. An alternative approach to speed up calculations is to use the technique proposed in [138] to sequentially compute each column of the exponential matrix with sublinear time complexity. Eigen decomposition is prohibitive for very large graphs, and hence, discrete random walks are more suitable for large networks. The sec-

ond part to my method is optimization using Louvain for τ distinct values of t. Although the exact computational complexity of the method is not known, the method seems to run in time $O(NlogN)$ with most of the computational effort spent on the optimization at the first level [121]. I demonstrate the applicability of DLM to very large networks using the discrete time random walks in results.

Resolution limit: Modularity compares the number the edges inside a community relative to the number of edges inside a community expected at random. This comparison is expressed succinctly using my representation as $DM > DM^{\infty}$ where M^{∞} represents the matrix of stationarity distributions and multiplying by D preserves the degree distribution and volume of the graph while randomly rewiring the edges. The corresponding quality function $DM - DM^{\infty}$ measures the excess of edges relative to those expected at random. The resolution limit of modularity [119] stems from implicitly assuming all node pairs interact with each other in the null model. On the contrary, my notion of a community using discrete time Markov process translates to $DM^t > DM^{t+1}$ and unlike modularity, I only consider node pairs at resolution $t+1$ to be interacting while resolving the community structure at resolution t. The horizon of node interactions is further suppressed in the continuous case because node pairs at resolution $t+dt$ implicitly interact to resolve the community structure at resolution t, where dt is an infinitesimally small period of time. This can be verified as:

$$\begin{aligned}
[D-A]e^{-t\mathbb{L}} &= D\mathbb{L}e^{-t\mathbb{L}} \\
&= \frac{D\mathbb{L}e^{-t\mathbb{L}}dt}{dt} \\
&= \lim_{dt\to 0}\frac{D\mathbb{L}e^{-t\mathbb{L}}dt}{dt} \\
&= \lim_{dt\to 0}\frac{D(e^{-t\mathbb{L}} - e^{-(t+dt)\mathbb{L}})}{dt}.
\end{aligned} \tag{2.127}$$

I have used the Maclaurin series expansion to rewrite $\mathbb{L}e^{-t\mathbb{L}}dt$ as $(e^{-t\mathbb{L}} - e^{-(t+dt)\mathbb{L}})$.

My representation to quantify interacting node pairs also provides a mechanism to define null models conditional on horizon of interaction, i.e., DM^t wires edges based on random paths of length t or less emanating from each node. Such a formulation of nulls

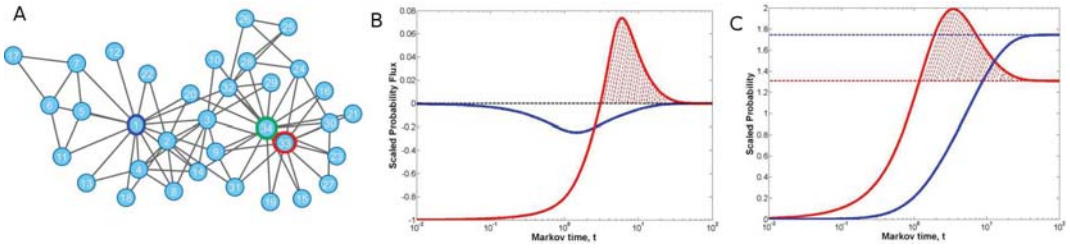


Figure 2.19. *Comparing diffusion dynamics and rate of diffusion dynamics. A: Karate network B: The scaled rate of diffusion dynamics over time for two pairs of nodes. The the rate of diffusion between node pair [34,33] is shown by red curve. The the rate of diffusion between node pair [34,1] is shown by blue curve. Two nodes are considered for coarse graining only if the probability flux is positive. The probability flux between pair [34,33] is positive for some time t (shown by shaded red region), whereas the probability flux between pair [34,1] is never positive. C: The diffusion dynamics over time for same pairs of nodes. The horizontal dotted lines indicate the baseline value for the node pairs in the null model. Two nodes are considered for coarse graining only if the probability over time exceeds the baseline value. The probability flux between pair [34,33] is positive for some time t (shown by shaded red region), whereas the probability flux between pair [34,1] is never positive. It follows from elementary calculus that the time at which probability value for node pair [34,33] is maximum (red curve) corresponds to the time at which the probability flux value for node pair [34,33] is 0 in B. Hence, the gain by coarse graining two nodes is dependent on extrinsic null model for diffusion dynamics, whereas is dependent on intrinsic network topology for rate of diffusion dynamics.*

models based on path lengths instead of degrees, has remained elusive in community detection literature [38], and I contend to be an important step towards precise definition of nulls models. It provides an alternate approach to overcome the resolution limit of traditional modularity optimization. In contrast to ad-hoc resolution parameters weighing standard configuration null model, I can instead define modularity as $DM > DM^t$ with some t less than or equal to the diameter of the graph. DM^t progressively restricts the horizon of interacting node pairs, and for a suitable choice of t overcomes the resolution limit.

Considering the rate of diffusion dynamics for mitigating the resolution limit has a decisive advantage over modularity and expected diffusion dynamics in general. This is high-

lighted by investigating the question: At what specific instant of time do the two dynamic approaches coarse-grain a pair of nodes into one? For diffusion dynamics it is equivalent to choosing a choosing a sufficiently large t such that $d_i H_t(i,j) > d_i H_\infty(i,j)$, i.e., the scaled probability of a random walker to reach node j starting from i after t steps exceeds its stationary probability scaled by d_i. For rate of diffusion dynamics, I choose t such that $d_i \frac{-dH_t(i,j)}{dt} > 0$ or equivalently any value of time t after the probability curve $H_t(i,j)$ has reached a maximum (the derivative is 0 at the peak). Using the Kolmogorov's backward equation, $d_i \frac{-dH_t(i,j)}{dt}$ can also be expressed as $d_j H_t(i,j) - \sum_{k \in N_1(j)} H_t(i,k)$ where N_1 indicates the 1-neighborhood. It follows from Markov chain theory that, the gain by coarse graining a pair of nodes is positive if the probability of a random walker to reach node j after t steps from node i exceeds the average probability of reaching node j's neighbours, $H_t(i,j) > \frac{\sum_{k \in N_1(j)} H_t(i,k)}{d_j}$. Thus, the decision to coarse grain is independent of the stationary distribution but instead dependent on the collective dynamics at the same time resolution over the 1-neighborhood (see Figure 2.19). At the network level, this translates into exploring community structure by performing a local comparison of probability measures where the precise formalism of local evolves dynamically with time. This is in stark contrast to expected diffusion dynamics which resolves communities by analyzing the local to global probability measures against the global stationary distribution, and hence, invariably involves a global comparison.

Other methods: I discuss other popular method for community detection next.

Modularity Modularity maximization is the most popular method for non-overlapping community detection [49]. Modularity measures the excess of links relative to a baseline, usually taken to be the configuration model. The modularity quality function initially proposed to detect non-overlapping communities in unweighted and undirected networks, has been extended to weighted [139], directed [140], overlapping [141] and multislice networks [142]. Despite its wide appeal, modularity suffers from the resolution limit and numerous approaches have been proposed to overcome this limit [143–145]. Modularity optimization is provably NP-hard [146], so I use the popular Louvain method to assess the performance of modularity maximization in a practical context [121].

Infomap The Infomap algorithm detects communities by minimizing the code length of map equation encoded using information theoretic principles [48]. Recent experiments on benchmarks demonstrate the Infomap algorithm to be the one of the most accurate method for community finding [38]. Although the Infomap does not suffer from the resolution limit, it has the tendency to overpartition a network (the field-of-view limit) [147]. I use the implementation provided by the authors to minimize the code length (out of 100 restarts) and derive the final community structure.

OSLOM: OSLOM, an acronym for 'order statistics local optimization method', is one of the most versatile methods of community detection capable of detecting overlapping and hierarchical communities in weighted, directed and dynamic networks [136]. OSLOM like modularity uses the configuration model to estimate statistical significance of communities. In order to overcome the resolution limit posed by this null model, it performs an iterative cluster search within the detected communities, and consequently, limits the horizon of the network under exploration.

Link Clustering Link clustering defines a community to be a group of links instead of tightly connected nodes and builds an agglomerative link-dendrogram to uncover overlap and hierarchy [41]. The link dendrogram is cut at the level which maximizes the partition density in order to obtain the most relevant community structure. Although with its own set of merits, the unconventional approach of grouping links is symmetric to the traditional approaches grouping nodes, and hence, not *a priori* better as argued in [38]. I contend that the Laplacian Modulus can be used to determine the overlapping nodes in the line graph, or equivalently the boundary links in the original graph and overcome the limitations of link clustering.

Clique Percolation The clique percolation method is one of the earliest approach to detect overlapping communities in networks [40]. The method proceeds by first finding all complete subgraphs or cliques of size k and nodes that belong to multiple disjoint (or non-percolating) cliques are classified as overlapping nodes. The parameter k provides an intuitive way to define the strength of communities and acts as an effective resolution parameter. However, the rigid definition of clique communities makes it undesirable for ex-

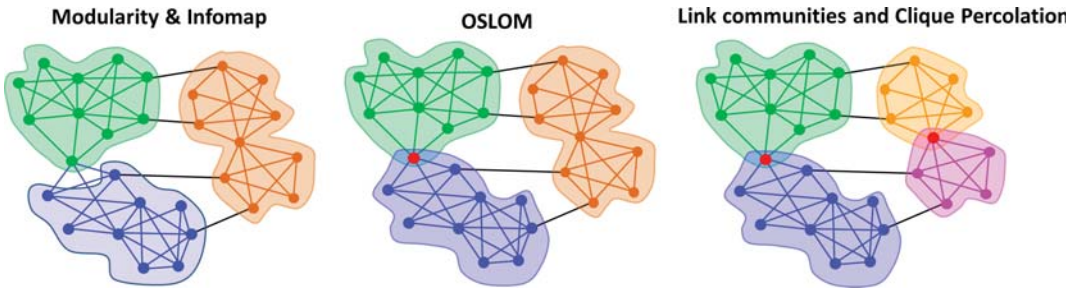


Figure 2.20. *Results of other community detection methods on synthetic graph. The identified communities for the same synthetic network of 29 nodes by different community detection methods; Left:Modularity and Infomap, Center: OSLOM, Right: Link communities and clique percolation. Observe no other method identifies the 6 communities as in Figure 3 in the main manuscript.*

tremely dense or sparse networks leading to giant components or absence of communities, respectively for all values of k [41]. I scan over the entire range of k and use the optimal k to demonstrate its performance.

2.2.3 Extension of Laplacian modularity to other network types

In this section, I extend the Laplacian modularity framework to other network types encountered in network science analysis like weighted, directed, signed and multiplex networks.

Weighted Networks: I denote a weighted graph by the triple $G = (V, E, A)$ where V is the set of nodes, $E \subseteq V \times V$ is the set of edges and A is the weighted adjacency matrix:

$$A(i,j) = \begin{cases} w(i,j) & \text{if} \quad (i,j) \in E \\ 0 & \text{otherwise.} \end{cases} \tag{2.128}$$

Here $w(i,j)$ is the weight on the edge $(i,j) \in E$. Let $D = diag(d_i)$ be the diagonal weighted degree matrix with elements given by the degree of the nodes $d_i = \sum_j w(i,j)$. The weighted Laplacian matrix takes the same form as in the unweighted case, i.e., $L = D - A$. The definition of strong and weak communities follow those of the unweighted

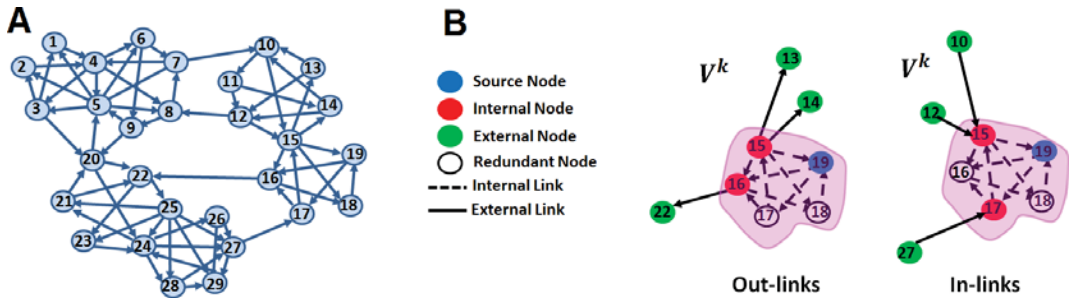


Figure 2.21. *Laplacian modularity for a directed network. A: A strongly connected directed network of 29 nodes B: Defining communities in the strong sense for a node in subgraph V_k with respect to out-links and in-links. The strong community criterion for the blue node ([19]) in subgraph, V^k, shaded pink (nodes [15,16,17,18,19]) over the 3 interfacing out-links with internal nodes ([15,16]) in red and external nodes ([13,14,22]) in green is:* $2S(19,15)+S(19,16) > S(19,13)+S(19,14)+S(19,22)$. *The strong community criterion for the blue node ([19]) in subgraph, V^k, shaded pink (nodes [15,16,17,18,19]) over the 3 interfacing in-links with internal nodes ([15,17]) in red and external nodes ([10,12,27]) in green is:* $2S(15,19)+S(17,19) > S(10,19)+S(12,19)+S(27,19)$.

graph, except that the membership vector includes the weights of the induced links, i.e., in the equation:

$$S(V) = S^T\mathcal{V} + S\mathcal{V}. \tag{2.129}$$

Here $\mathcal{V}$ is the membership vector, where the element $\mathcal{V}^i$ is equal to the sum of edge weights node i is connected to, instead of the cardinality as in unweighted graphs. The resulting quality function is identical in form. The quality function Q_{wgt} is mathematically represented as:

$$Q_{wgt} = \sum_{i,j}[(D-A)S + S(D-A)]\delta(i,j). \tag{2.130}$$

I resort to the same algorithms to determine overlap as in the unweighted case.

Directed Networks: I naturally extend my Laplacian modularity framework to directed networks. Let G be a strongly connected directed graph with N nodes and E be a set of directed edges. Strong connectivity ensures that the dynamics over time is ergodic useful for DLM. Edge directionality changes the calculation of boundary cohesion. Specifically,

instead of the entire set of interfacing links E^k for subgraph V^k as in undirected communities, I consider two sets of interfacing links, in-links E^k_{in} (those that point towards the subgraph) and out-links E^k_{out} (those that point away from the subgraph). Formally, if node pair associated with a directed edge is $[m, n]$ where m is the origin node and n the destination node, then E^k_{in} is the set of edges with $m \notin V^k, n \in V^k$ and E^k_{out} is the set of edges with $m \in V^k, n \notin V^k$. The two sets of contributions for internal and external nodes used in the boundary cohesion calculation are modified to account for in-links and out-links separately (see Figure 2.21). Mathematically, $S^i(V^k_{in})$ and $S^i(V^k_{out})$ are represented as:

$$S^i(V^k_{in}) = \sum_{E^k_{out}} S(m, i) + \sum_{E^k_{in}} S(i, m) \tag{2.131a}$$

$$S^i(V^k_{out}) = \sum_{E^k_{out}} S(n, i) + \sum_{E^k_{in}} S(i, n). \tag{2.131b}$$

Following a similar derivation, the quality function to be optimized becomes:

$$Q^{dir} = \sum_{i,j} [(D_{out} - A_{dir})S + S(D_{in} - A_{dir})]\delta(i, j). \tag{2.132}$$

Here A_{dir} is the adjacency matrix with the $A_{dir}(i, j)^{th}$ entry denoting a directed edge from node i to j,D_{out} is a diagonal matrix of out-degrees, i.e., $D_{out} = diag(d_{out})$ where d^i_{out} is the i^{th} element corresponding to node i of the out-degree vector given by $d^i_{out} = sum_j A_{dir}(i, j)$, D_{in} is a diagonal matrix of in-degrees, i.e., $D_{in} = diag(d_{in})$ where d^i_{in} is the i^{th} element corresponding to node i of the out-degree vector given by $d^i_{in} = sum_j A_{dir}(j, i)$, S is the connectivity matrix, and $\delta(i, j)$ is 1 if i and j belong to the same community and 0 otherwise.

I also introduce the directed counterpart of the random walk Laplacian so as to be able to use eigen decomposition for efficient optimization of the DLM quality function. The probability transition matrix M_{dir} is given by $M_{dir} = D^{-1}_{out} A_{dir}$, indicating the probability of leaving a node is evenly split among its adjacent neighbors. Adjacent neighbors of a node i in a directed network are those connected with out-links starting from node i. Naive pre and post multiplication by an exponent of the degree matrix doesn't convert M into a symmetric normalized matrix, as in the undirected case. Instead, the normalized symmetric

matrix N_{dir} for directed graph is given by $N_{dir} = \Pi^{1/2} M_{dir} \Pi^{-1/2}$ where Π is a diagonal matrix with elements corresponding to the stationary distribution vector π for the directed graph [94]. The stationary distribution vector is well defined for a strongly connected graph and is given by the eigenvector corresponding to eigenvalue 1 of M_{dir}. The random walk Laplacian $\Delta_{M_d ir}$ and its normalized counterpart $\mathcal{L}_{dir}$ are given by $\Delta_{M_d ir} = I - M_{dir}$ and $\mathcal{L}_{dir} = I - N_{dir}$ where I is an identity matrix of size N. Let the spectral decomposition of the normalized Laplacian be again given by $\mathcal{L}_{dir} = \Phi \Lambda \Phi^T$ as in the undirected case. The eigen decomposition of the normalized transition matrix is then given by $N_{dir} = \Phi \Lambda' \Phi^T$ with same set of eigenvectors and eigenvalues related as $\Lambda' = I - \Lambda$ as in the undirected case. Using the above relations, the probability transition matrix M^t_{dir} for discrete directed random walks at time t is:

$$M^t(i,j) = \sum_{k \geq 1} \lambda_k'^t \Phi_k(i) \Phi_k(j) \sqrt{\frac{\pi_j}{pi_i}}. \tag{2.133}$$

The exponential matrix $e^{-t\Delta_{M_{dir}}}$, denoting the probability transition at time t for continuous time directed random walks is given by:

$$H_t(i,j) = \sum_{k \geq 1} e^{-t\lambda_k} \Phi_k(i) \Phi_k(j) \sqrt{\frac{\pi_j}{\pi_i}}. \tag{2.134}$$

In both the above equations, the $(i,j)^{th}$ term is the probability of starting at i and reaching j in time t. It is important to note that all formulaes for undirected networks reconcile with the ones for directed networks in the Laplacian modularity as well as DLM frameworks. Thus, one efficacy of my formalism is that undirected networks are a special case of directed networks. However, the Laplacian and DLM matrices for directed networks are not symmetric in general.

Another appealing feature of my approach using random walks is that it provides a rigorous formalism of modularity for directed networks. Modularity Q^{mod}_{dir} for directed networks is defined as:

$$Q^{mod}_{dir} = \frac{1}{\mid E \mid} \sum_{i,j} [A - \frac{d_{in} d^T_{out}}{\mid E \mid}] \delta(i,j). \tag{2.135}$$

Here d_{in} and d_{out} are $N \times 1$ length vectors of the in-degree and out-degree respectively ($d^i_{in} = \sum_j A_{dir}(j,i)$) [140]. However, this definition of modularity has critical limitations

as expressed in [148], particularly the inability to coherently account for edge directionality but instead favor partitions with high edge densities. In order to circumvent this problem, the modification proposed in [148] is to instead define modularity as:

$$\tilde{Q}_{dir}^{mod} = \sum_{i,j}[\Pi M_{dir} - \pi\pi^T]\delta(i,j). \tag{2.136}$$

However, as $\Pi M_{dir} \neq A$ in general, the above definition introduces an undesirable transformation to the original adjacency matrix, i.e., I introduce structural changes to the topology of the original network. I overcome this limitation as well as limitation of original modularity for directed networks with my own version which appears naturally from the properties of random walks. It is given by:

$$\hat{Q}_{dir}^{mod} = \frac{1}{\mid E \mid}\sum_{i,j}[A - d_{out}\pi^T]\delta(i,j). \tag{2.137}$$

This definition stems from defining modularity matrix as $D_{out}[M_{dir} - M_{dir}^{\infty}]$ where M_{dir}^{∞} is the matrix of stationarity distribution. M_{dir}^{∞} is given by $\mathbb{1}\pi^T$ where $\mathbb{1}$ is a $N \times 1$ vector with all coordinates 1. As $D_{out}M_{dir} = A$ and $D_{out}M_{dir}^{\infty} = d_{out}\pi^T$, $D_{out}[M_{dir} - M_{dir}^{\infty}]$ simplifies to $[A - d_{out}\pi^T]$ as in Equation (2.137). Note that the modularity matrix reduces to $D[M - M^{\infty}]$ for undirected networks. I use the relation $M^{\infty} = \mathbb{1}\pi^T$ where $\pi = \mathbb{1}D/vol$ and $DM = A$ to recover the original modularity matrix given by $[A - \frac{d^T d}{vol}]$. Overall, I contend that Equation (2.137) offers a more coherent formalism of modularity for directed networks than those proposed previously.

Signed networks: In signed networks, there are both positive and negative links. I perform a separate analysis for positive and negative links:

$$\begin{aligned} S_+(V_+) &= S_+^T\mathcal{V}_+ + S_+\mathcal{V}_+ \\ S_-(V_-) &= S_-^T\mathcal{V}_- + +S_-\mathcal{V}_-. \end{aligned} \tag{2.138}$$

Here $\mathcal{V}_-, \mathcal{V}_+$ are the membership vectors indicating boundary nodes for the positive and negative networks with connectivity potentials S_+, S_-. Following a similar analysis, the quality function for signed networks is:

$$Q_{sgn} = \sum_{i,j}[(D_+ - A_+)S_+ + S_+(D_+ - A_+) - (D_- - A_-)S_- - S_-(D_- - A_-)]\delta(i,j). \tag{2.139}$$

Here D_+, D_- are the positive and negative degrees, A_+, A_- are the positive and negative adjacency matrices. In terms of the Laplacian for the positive and negative adjacency network, the equation becomes:

$$Q_{sgn} = \sum_{i,j}[L_+S_+ + S_+L_+ - L_-S_- - S_-L_-]\delta(i,j). \tag{2.140}$$

Here A is the signed adjacency matrix and D is sum of signed diagonal degrees. Note that when the resulting matrix is asymmetrical, I symmetrize it, in order to apply the Louvain algorithm.

Multislice Networks: Another class of networks receiving a lot of attention are multislice networks that are defined by coupling multiple adjacency matrices. Each slice encodes connections that may vary over time or across different attributes of the network. [142] generalizes modularity for multislice networks by making a connection between Laplacian dynamics and null models. The general nature of my framework enables extending my definition of communities to multislice networks with the additional feature of identifying overlap unlike [142].

A natural feature of a multislice network is that it introduces replicates of each node in every slice. Hence I get a set of $N \times Y$ nodes, V_{multi} for which I wish to explore the community structure where Y is the number of slices. Let me represent nodes in unweighted undirected multislice networks as i_s where the subscript s indicates the slice under consideration for node i. The edges in the network $E_{multislice}$ are broadly characterized into two types, E^{intra}_{multi} indicating an edge between node pair $[i_s, j_s]$ in the same slice and E^{inter}_{multi} indicating an edge between the same node i in different slices r and s (node pair $[i_r, i_s]$). I construct the multislice adjacency matrix A_{multi} in order to quantify random walks on the network:

$$A_{multi}(i_s, j_s) = \begin{cases} 1 & \text{if} \quad (i_s, j_s) \in E^{intra}_{multi} \\ \omega & \text{if} \quad (i_r, i_s) \in E^{inter}_{multi} \\ 0 & \text{otherwise.} \end{cases} \tag{2.141}$$

Here the parameter ω controls the coupling between slices. The diagonal matrix is given by $D_{multi} = diag(d_{i_s})$ where $d_{i_s} = \sum_{j_r} w(i_s, j_r)$. I account for intra- and inter-slice

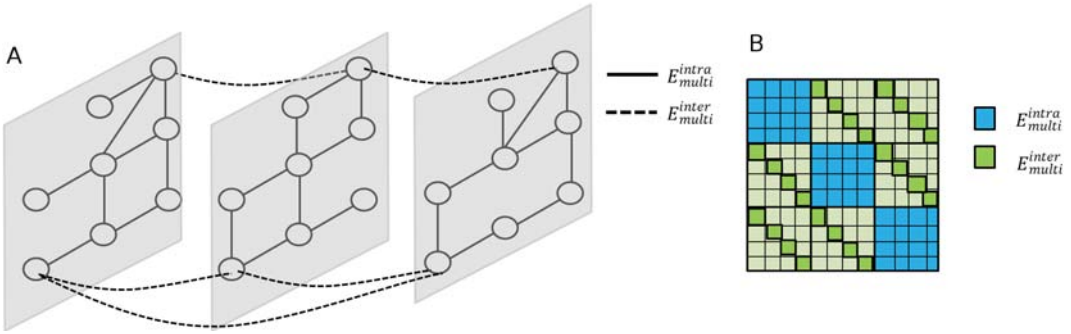


Figure 2.22. *Laplacian modularity for a multislice network. A: An illustrative multislice network with dashed inter-slice links and solid intra-slice links. B: Constructing the adjacency matrix A_{multi} for the multislice network. The blue areas correspond to intra-slice links and green blocks correspond to the placement of inter-slice links.*

interfacing links separately for internal and external nodes (see Figure 2.22). Let the set of intra-slice interfacing links for subgraph V^k be E^k_{intra} and the set of inter-slice interfacing links be E^k_{inter}. Mathematically, the list of the four contributions are:

$$S^{i_s}(V^k_{in\cap intra}) = \sum_{E^k_{intra}} (S(m_s, i_s) + S(i_s, m_s)) \tag{2.142a}$$

$$S^{i_s}(V^k_{in\cap inter}) = \sum_{E^k_{inter}} (S(i_r, i_s) + S(i_s, i_r)) \tag{2.142b}$$

$$S^{i_s}(V^k_{out\cap intra}) = \sum_{E^k_{intra}} (S(n_s, i_s) + S(i_s, n_s)) \tag{2.142c}$$

$$S^{i_s}(V^k_{out\cap inter}) = \sum_{E^k_{inter}} (S(i_r, i_s) + S(i_s, i_r)). \tag{2.142d}$$

The net internal boundary cohesion is then calculated as $(S^{i_s}(V^k_{in\cap intra}) + S^{i_s}(V^k_{in\cap inter}))$. Similarly, the net external boundary cohesion is calculated as $(S^{i_s}(V^k_{out\cap intra}) + S^{i_s}(V^k_{out\cap inter}))$. For notational simplicity, let me denote $(S^{i_s}(V^k_{in\cap intra}) + S^{i_s}(V^k_{in\cap inter}))$ as $S^{i_s}(V^k_{in})$, and represent $(S^{i_s}(V^k_{out\cap intra}) + S^{i_s}(V^k_{out\cap inter}))$ as $S^{i_s}(V^k_{out})$. Then the definitions of strong and weak communities are given by:

Community in the strong sense: The subgraph V^k is a community in a strong sense if:

$$S^{i_s}(V^k_{in}) > S^{i_s}(V^k_{out}), \forall i \in V^k. \tag{2.143}$$

In a strong community, the internal boundary cohesion of all (intra or inter-slice) nodes is greater than its external value for subgraph, V^k.

Community in the weak sense: Then the subgraph V^k is a community in a weak sense if:

$$\sum_{i_s \in V^k} S^{i_s}(V^k_{in}) > \sum_{i_s \in V^k} S^{i_s}(V^k_{out}). \tag{2.144}$$

In a weak community, the sum of internal boundary cohesion over all (intra or inter-slice) nodes in V^k is greater than the sum of external boundary cohesion, for subgraph, V^k.

Using these sets of definition, the quality function to be optimized takes the familiar form:

$$\tilde{Q}_{multi} = \sum_{i_s, j_r} [(D_{multi} - A_{multi})S + S(D_{multi} - A_{multi})]\delta(i_s, j_r). \tag{2.145}$$

The spectral representations for S can be derived by following the procedure in Section 2.2. The multislice quality function in Equation (2.146) is can identify overlap in multislice networks using the same principles derived for single slice networks.

An alternate way is to neglect the cohesion from inter-slice coupling and consider it as a free parameter. In order words, only the intra-slice edges are considered while calculating the average similarity over the 1-neighborhood. In such a scenario, the quality function takes the form of the quality function introduced in [142], and mathematically represented as:

$$\tilde{Q}_{multi} = \sum_{i_s, j_r} \Bigg([(D^s_{multi} - A^s_{multi})S^s + S^s(D^s_{multi} - A^s_{multi})]\delta_{sr} + \delta_{ij}(C_{jsr} + C_{isr}) \Bigg) \delta(i_s, j_r) \tag{2.146}$$

Here superscript s indicates the slice under consideration, and C_{jsr} is the parameter representing the inter-slice weights.

Overall, in this section I show that the Laplacian modularity framework is one of the most complete architectures to determine community structure in various network types.

2.3 Deep Neural Networks

Convolutional neural networks (CNNs) are machine learning models inspired by biological neural networks. Unlike traditional deep architectures, CNN's have the attractive property of weight sharing reducing the number of variables to be learned. The principle of weight sharing in convolutional filters is especially relevant [25] to image processing where same atomic features reappear at different parts of the image. This is applicable to 3D shapes as well. I use CNN's for two applications: (1) Understanding 3D meshes by introducing a novel technique for surface parametrization, so that, 3D meshes can be analyzed similar to 2D images , and (2) Extend CNN's for regression by introducing a technique called deep matrix completion and apply it to a prototypical example of hand tracking from depth images.

2.3.1 Learning geometry image of shapes using CNN's

The choice of spherical parametrization has these critical advantages:

1. The cut is defined *post* parametrization, in contrast to other surface parameterizations (e.g., Floater) wherein the cuts are defined *pre* parametrization. This allows me to efficiently sample additional cuts in order to achieve rotational invariance of the geometry images to the cut location for learning shapes using CNN's. Geometry images encoding intrinsic properties for different cuts are pose invariant, in the sense they are either translated or rotated. Note, max pooling in CNNs are known to achieve translational invariance.

2. Spherical symmetry allows creating a continuous geometry image without edge discontinuities, and hence, implicitly inform the CNN about the spherical domain via padding. A cut defined *pre* parametrization won't have this symmetry.

3. Spherical parametrization. results in geometry image with a fixed square boundary within which all pixels encode shape information. Iterative cuts will either create

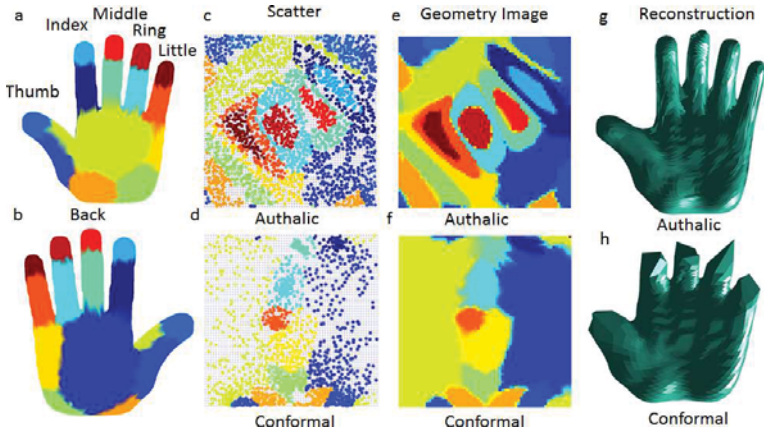


Figure 2.23. *Authalic vs Conformal: 2500 vertices of the hand mesh are color coded (a,b) and mapped onto a square domain using authalic (c) and conformal (d) spherical parametrization. Conformal parametrization compress high curvature points to dense regions. Hence, finger tips are all mapped to a very small regions in (d). The 64× 64 geometry image is created by uniformly sampling the square domain (blue dots in c,d), and then interpolating the nearby feature values. (e) Authalic geometry image encodes all tip features. (f) The resolution of geometry image is insufficient to capture the tip feature colors. This is validated by reconstructing shape from geometry images encoding x, y, z locations (g-h).*

an atlas losing connectivity information between subregions, or create a free boundary increasing memory complexity as many pixels outside the boundary contained in geometry image don't contain shape information. It also increases the learning complexity as CNNs have to abstract the mask of inside/outside shape boundary.

Hence, spherical parametrization is the most natural setting for learning shapes using CNNs. Next, the necessity of authalic parametrization arises from the fact that the number of training samples and parameters in the CNN limit the input resolution of the geometry image. Under the constraint of resolution, authalic geometry images encode more information about the shape as compared to conformal geometry images (see Figure 2.23).

Spherical parametrization: I briefly overview authalic spherical parametrization techniques and point the readers to [149] for literature on general surface parametrization. I

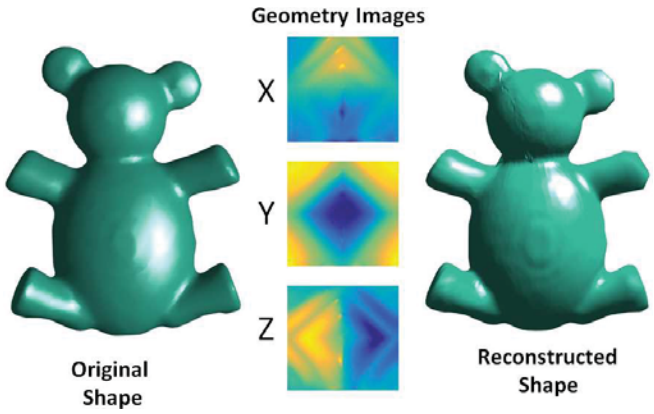


Figure 2.24. *Shape reconstruction using geometry images: The original teddy model to the left is reconstructed (right) using geometry images corresponding to the X, Y and Z coordinates (center).*

also discuss prevalent deep learning paradigms and state the distinctiveness of my approach to current methodologies

Although, methods for conformal (angle preserving) mesh parametrization abound [150–152], there is relatively less work on authalic (area preserving) mesh parametrization. This is because a conformal parametrization preserves local shape, which is useful to the graphics community for feature oriented applications such as texture mapping. However, an authalic parametrization of a shape is more compatible with the notion of convolving surface patches with constant size (equi-areal) filters. Also, conformal parametrization induces severe distortion to elongated shape structures common in deformable shape models [153]. Note that a mapping that is both conformal and authalic is isometric, and must have zero Gaussian curvature everywhere. This is rare in the context of general 3D mesh models and one must choose one or the other.

There exist several approaches to authalically parameterize a mesh with disk like topology [154] but deformable shape surfaces usually admit the topology of a 2-sphere and are compact manifolds. There exist only a handful of methods in literature that authalically parameterize a shape on a spherical domain. Dominitz and Tannenbaum [155] and Zhao et al. [153] use optimal transport for area-preserving mapping. Although efficient to im-

plement, these methods introduce smoothing and sharp edges get lost [156]. This is a critical drawback for CAD-like objects which contain several sharp edges. A method that implicitly corrects area distortion by penalizing large triangle sizes is proposed in [157]. However, my experiments indicate that this approach fails to work in a practical setting. A method similar in spirit to mine uses Lie advection to iteratively minimize the planar areal distortion of a parametrization [158]. The proposed extension of their approach to the 3D spherical domain introduces several singularities and triangle flips, highly undesirable for coherent 3D shape representation and analysis. Instead, the gradient vectors derived from the area restoring diffeomorphic flow are mapped on the original mesh domain using barycentric mapping to determine the corresponding vertex displacement coordinates in the spherical domain.

As stated earlier, current methods employing deep learning for 3D shape analysis either appropriate for rigid transformations such as ShapeNets [22], VoxNet [23], DeepPano [24] or fail to utilize the power of CNN's to learn hierarchical abstractions from raw input features by introducing transformations to the convolutional filters [31] or the shape representation [30]. Another critical bottleneck in voxel based approaches is that the 3^{rd} extra dimension introduces a large computational overhead. Consequently, the voxel grid is restricted to a relatively low resolution ($30 \times 30 \times 30$ in ShapeNets and $32 \times 32 \times 32$ in Voxnet). Also, active voxels interior to the shape are not useful indicators of the overall shape, as long as the shape boundary is well defined and does not contain internal voids. Indeed, they are deterrent for analyzing shapes undergoing isometric transformations.

In contrast to all these approaches, I only represent the shape surface as a geometry image wherein each pixel can encode any global or local feature of the shape, or the point coordinates themselves. In my experiments, I encode intrinsic properties of points on the surface as pixel intensities in the geometry image which serves as input to the CNN model. The CNN then automatically learns discriminative abstractions of the 3D shape, useful for shape classification or retrieval.

Authalic Parametrization of 3D shapes: In this section I first discuss preprocessing steps to transform erroneous or high genus mesh models into a genus zero topology. Next I

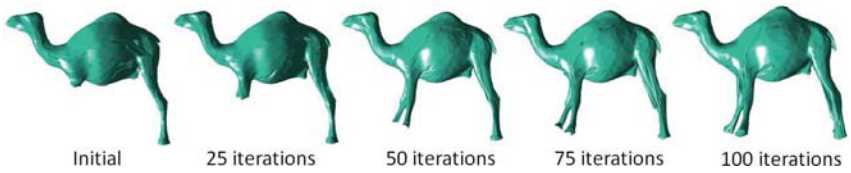


Figure 2.25. *Progression of my authalic spherical parametrization algorithm: Individual plots display the shape reconstructed from the geometry image corresponding to a spherical parameterizations. The area distortion associated with the geometry image, and hence the spherical parametrization, progressively decreases with iterations given an initial spherical parametrization.*

state my novel approach for authalic parametrization. Note that I employ notation and terminologies in the context of discrete differential geometry, unless stated otherwise.

Mesh preprocessing: A surface mesh, M is represented as V, F, E wherein V is the set of vertex coordinates, F the set of faces and E the set of edges constituting all faces. With abuse of notation, I term mesh models following the Euler characteristic to be accurate, given by:

$$2 - 2m = |V| - |E| + |F| \tag{2.147}$$

where $|x|$ indicates the cardinality of feature x and m is the genus of the surface. Unfortunately, most mesh models do not strictly follow the Euler characteristic, such as those in the Princeton ModelNet benchmark. In such a scenario, I first find *active points* in a grid structure with suitable resolution which are interior to or on the boundary of the mesh model. This is akin to the process of voxelization. I then find the α-shape with radius equal to the resolution of the grid. An α-shape is a generalization of the convex hull with radius parameter r such that varying r from 0 to ∞ returns a range of shapes between the empty shape and convex hull, respectively. I find the largest connected region of the resulting α-shape and use the corresponding boundary vertices and facets for subsequent procedures. The resulting shape is usually accurate barring degeneracies in the original model. In my experiments I perform the above procedure only for models in the Princeton ModelNet [22] benchmark.

If the genus of an accurate mesh model is evaluated to be non-zero, I then find the medial axis of the shape and sequentially fill all identified loops until the genus becomes zero. I keep track of the imputed points which constitute the topological mask of the shape. If this fails to return a genus-zero surface, I progressively increase the radius r of the α-shape till the genus becomes zero. I mostly get a high fidelity representation of the original shape after this procedure. Additional details for making a mesh accurate and transforming a high genus surface to a surface topologically equivalent to a sphere are provided next.

I discuss the process for creating a topological mask for non genus zero surfaces for models in Princeton ModelNet benchmark. As discussed, the procedure to convert a high genus surface to a genus zero surface is to first fill topological holes identified using medial axis, and then progressively increase the radius r of the α-shape till the genus becomes zero. The additional points that fill the holes identified by the medial axis are indexed 0, whereas the points on the original shape are indexed 1. A geometry image, C_{idx} created from these points indexed 0 or 1 serves as a topological mask. However, the holes filled by increasing the radius r of the α-shape does not introduce additional points onto the shape, but instead introduces new faces. For such shapes, I calculate the point-wise Hausdorff distance between points in the geometry image (X,Y and Z coordinates) corresponding to the genus-zero shape and points on the original α-shape with r equal to the grid resolution. It is expected that the Hausdorff distance be small (or equal to 0) for a point in the geometry image which is contained in the original shape and be large for points lying on new faces in the transformed α-shape. Hence, I encode the Hausdorff distance, D, as a similarity value in the range [0; 1] such that values close to 0 indicate points on new faces and values close to 1 indicate points in the original shape.

The topological mask, C_{top} is then calculated as the Hadamard product of C_{idx} and C_{dist}, where C_{idx} corresponds to the geometry image of points on the shape indexed with values 0 or 1. C_{top} then serves as a topological mask for the shape. The overall procedure to derive C_{top} for higher genus surfaces is to first calculate the medial axis and then calculate C_{idx} corresponding to the 0; 1 points. If the genus of the resulting shape is 0 then C_{idx} directly serves as C_{top} for subsequent shape analysis. Else if the genus of the resulting shape

is higher than 0, then I progressively increase r of the α-shape till the genus becomes zero and calculate C_{dist}. This genus-zero shape serves as input to the authalic parametrization procedure.

Authalic Parametrization: My method for authalic spherical parametrization takes as input any spherically parameterized mesh and iteratively minimizes the areal distortion in 3 steps described in detail below and outputs a bijective map onto the surface of a sphere. I use the spherical parametrization suggested in [159] for initialization due to its speed and ease of implementation. I evaluated different initial parameterizations [152] and my experiments indicate that the final authalic parametrization is robust to initialization. I now detail the 3 steps:

(1) At every iteration I first evaluate a scalar harmonic field corresponding to the areal distortion ratio of vertices in the original mesh and spherical mesh by solving a Poisson equation. Mathematically, I solve

$$\nabla^2 g = \delta h \tag{2.148}$$

where g is a function defined on the vertex set V, ∇^2 transforms to the Laplacian operator for a closed mesh surface [160], and δh is the areal distortion ratio wherein each element of the vector is defined as

$$\delta h_u = \frac{A_u^s}{A_u} - 1 \tag{2.149}$$

A_u^s is the spherical triangular area associated with the Voronoi region around vertex u and A_u is the triangular area associated with vertex u on the mesh model. The elements of the discretized version of the Laplacian operator with cotangent weights are calculated as:

$$L(u,v) = \begin{cases} \sum_v \frac{cota_{u,v}+cotb_{u,v}}{2} & \text{if}, v = u \\ -\frac{cota_{u,v}+cotb_{u,v}}{2} & \text{if}, v \subseteq N_1(u) \\ 0 & \text{otherwise} \end{cases}$$

where $\mathcal{N}_1(u)$ is the set of 1-ring neighbours of vertex u, $a_{u,v}, b_{u,v}$ are the two angles supporting the edge connecting vertices u and v and E_B indicates a boundary edge. I assume that the triangulation is regular, i.e., $a_{u,v}+b_{u,v} \leq \pi$ so that all weights are positive. Equation (2.152) now becomes

$$Lg = \delta h \tag{2.150}$$

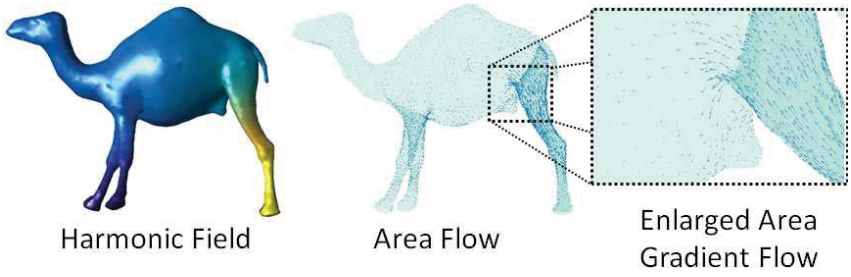


Figure 2.26. *Left: Harmonic field corresponding to area distortion on sphere displayed on the original mesh. Center: Area restoring flow on the spherical domain mapped onto the original mesh as a quiver plot. Right: Enlarged plot of area restoring flow.*

The scalar field g is evaluated using the above equation at every iteration for the vector δh (see Figure 2.26) . Due to the sparsity of L, Equation (2.150) can be efficient evaluated at every iteration using the preconditioned bi-conjugate gradient method. However, I precalculate the pseudoinverse of L once, and use it for every iteration. This saves the overall computational time. Note, k-rank approximation ($k \approx 300$) of the pseudoinverse when the $|V|$ is large does not noticeably affect the final result. (2) I then evaluate the gradient field of the of the harmonic function on the original mesh. This field is indicative of the required vertex displacements on the spherical mesh so as to decrease the areal distortion ratio. Consider a face f_{uvw} in the original mesh with its three corners lying at u, v, w. Let n be a unit normal vector perpendicular to the plane of the triangle. The gradient vector ∇g for each face is solved as [161]:

$$\begin{bmatrix} v-u \\ w-v \\ n \end{bmatrix} \nabla g = \begin{bmatrix} g_v - g_u \\ g_w - g_v \\ 0 \end{bmatrix}$$

A unique gradient vector for each vertex is obtained as weighted mean of incident angle of each face at the vertex and the corresponding gradient value as done in [158] :

$$\nabla g_u = \frac{1}{\sum_{f_{uvw}} c^u_{vw}} \sum_{f_{uvw}} c^u_{vw} \nabla g(f_{uvw}) \tag{2.151}$$

f_{uvw} are the faces in the one ring neighborhood of vertex u and c^u_{vw} is the angle subtended at vertex u by the edge vw. Figure 2.26 shows the gradient low field using a quiver plot on the mesh model.

(3) In contrast to [158] which operates directly on the spherical mesh domain, I displace the vertices on the original mesh and then map these displacements onto the spherical mesh using barycentric mapping, i.e., vertex displacements on the original mesh serve as proxy to determine the corresponding displacements on the spherical mesh. Barycentric mapping is possible because the original and spherical mesh have the same triangulation. Each vertex in the original mesh is (hypothetically) displaced by:

$$v = v + \rho \nabla g_v \tag{2.152}$$

where ρ is a small parameter value. A large value of ρ leads to a large displacement of the vertex and may displace it beyond the its 1-neighborhood. This causes triangle flips and the error propagates through iterations. However, a small value of ρ leads to large convergence time. I empirically set ρ equal to 0.01 in all my experiments which achieves the right tradeoff between number of iterations to convergence and accuracy. The barycentric coordinates of displaced vertices are evaluated with respect to triangles in the one-ring, and the triangle with all coordinates less than 1 is naturally chosen as the destination face. The vertex in the spherical mesh is then mapped to the corresponding destination face with the same barycentric weights. The purpose of this indirect mapping procedure is twofold (1) The vertex displacements minimizing areal distortion are constrained to be on the input mesh, which in turn ensure the mapped displacements onto the spherical domain are well behaved. (2) Additionally, the constraint that the vertices remain on the mesh model minimize triangle flips and alleviate the need for an expensive retriangulation procedure after each iteration. The iterations continue until convergence. In practice I stop the iterations after the all areal distortion ratios fall below a threshold or the maximum number of iterations has been reached. The maximum number of iterations is set to 100.

Learning shapes using geometry images: In this section I briefly discuss the creation of a geometry image and intrinsic properties of a surface relevant to learning 3D shapes. I

also discuss my CNN architecture for learning rigid as well as deformable shapes for shape classification and retrieval.

Geometry image The spherical parametrization maps the surface of the mesh onto a sphere. I then project this spherical surface onto an octahedron and cut it to obtain a square, thus creating a geometry image. The advantage of mapping the surface onto an octahedron over other regular polyhedra such as a tetrahedron or cube is that the signals can be linearly interpolated onto a regular square grid with minimum stretch [35]. For brevity, I skip details on the spherical area sampling for projecting points on the sphere onto an octahedron and refer readers to [35] for details. The edges of the octahedron cut to flatten the polyhedron are shown in Figure 2.27. Observe the reflective symmetry of the geometry image along the vertical, horizontal and diagonal axes shown in Figure 2.27. Due to this symmetry, I can create replicates without any discontinuities along any edge or corner of the image (see Figure 2.29). This property is useful for implicitly informing a deep learning model about the warped mesh the image represents. Additional discussion of this property is provided ahead. The procedure of creating the geometry image is visually elucidated in Figure 2.27.

Intrinsic properties of shapes: Having obtained a geometry image from a mesh model, I next discuss encoding the pixel values with intrinsic descriptors. There exist several possibilities which I enumerate:

1. Principal curvatures: The two principal curvatures, κ_1 and κ_2 measure the degree by which the surface bends in orthogonal directions at a point. They are in effect the eigenvalues of the shape tensor at a given point.

2. Gaussian Curvature: The Gaussian curvature κ is defined as the product of the principal curvatures at a point on the surface, $\kappa = \kappa_1\kappa_2$. The sign of Gaussian curvature indicates whether a point is elliptic ($\kappa > 0$), hyperbolic ($\kappa < 0$) or flat ($\kappa = 0$)

3. Mean curvature: The mean curvature, H is the average of the principal curvatures at a point on the surface, $H = \frac{1}{2}(\kappa_1 + \kappa_2)$.

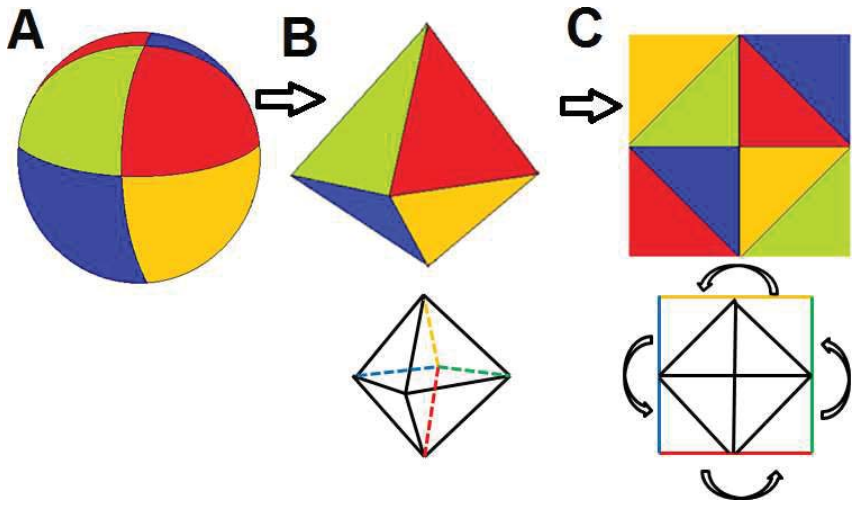


Figure 2.27. *Explanation of geometry image construction from a spherical parametrization: The spherical parametrization (A) is mapped onto an octahedron (B) and then cut along edges (4 colored dashed edges in line plot below) to output a flat geometry image (C). The colored edges share the same color coding as the one in the octahedron. Also the half-edges on either side of the midpoint of colored edges correspond to the same edge of the octahedron.*

4. Heat kernel signature [25]: The heat kernel, h_t is the solution to the differential equation $\frac{\delta h_t}{\delta t} = -\Delta h_t$ (h_t is the heat kernel). The heat kernel signature (HKS) at the point is the amount of untransferred heat after time t , given by

$$h_t(u,u) = \sum_{i \geq 0} e^{-t\lambda_i} \Phi_i(u) \Phi_i(u) \tag{2.153}$$

Where λ and Φ are the eigenvalues and eigenvectors of the Laplace-Beltrami operator. The heat kernel is invariant under isometric transformations and stable under small perturbations to the isometry, such as small topological changes or noise. Additionally, the time parameter t in the HKS controls the scale of the signature with large t representing increasingly global properties, i.e., its a multiscale signature. Variants of the heat kernel include the GMS [162], GPS [163]which differ in the weighting of the eigenvalues.

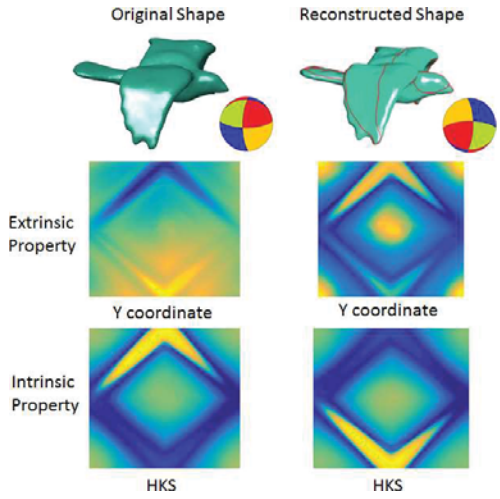


Figure 2.28. *Intrinsic vs. Extrinsic properties of shapes. Top left: Original shape. Top Right: Reconstructed shape from geometry image with cut edges displayed in red. The middle and bottom rows show the geometry image encoding the y coordinates and HKS, respectively of two spherical parameterizations (left and right). The two spherical parametrization are symmetrically rotated by 180 degrees along the Y-axis. The geometry images for Y-coordinate display an axial as well as intensity flip. Whereas, the geometry images for HKS only display an axial flip. This is because HKS is an intrinsic shape signature whereas point coordinates on a shape surface are not.*

Figure 2.28 discusses the difference between intrinsic and extrinsic shape properties in the context of geometry images. I discuss the pros and cons of each intrinsic property in the scope of deep learning using CNN next.

Convolutional Neural Net: CNNs is a machine learning model inspired by biological neural networks. Unlike traditional deep architectures, CNN's have the attractive property of weight sharing reducing the number of variables to be learned. The principle of weight sharing in convolutional filters is especially relevant [25] to image processing where same atomic features reappear at different parts of the image. This is applicable to 3D shapes as well. From the purview of intrinsic learning of shapes using CNN, the principal curvatures are the most atomic property of a point on the surface. The Gaussian and Mean curvature are functions of the principal curvatures, and hence, are expected to provide little additional

cues for learning shapes. The heat kernel signature too can be interpreted as an extension to curvature, wherein the curvature of a point is estimated over balls of larger radius controlled by the time parameter t [25]. The heat kernel is useful for enforcing consistency to geometry images when local curvature calculations are affected by noise. I discuss this point to a greater detail in the experiments section.

I discuss the salient features of my CNN based learning of geometry images Different from traditional image based CNNs. First, I pad the geometry image with replicates of the original image to implicity inform the CNN about the warped structure of the original mesh. Owing to the octahedral symmetry, there are no edge and corner discontinuities if the 4 connected replicates to a geometry image are rotated by 180 degrees (or flipped once along the x-axis and y-axis each). This is visually illustrated for the geometry images encoding the $x, y,$ and z coordinates of the mesh model in Figure 2.29. No subsequent layer in the CNN is padded so as to not distort this information. Second, the orientation of the geometry image is dependent on the poles of the spherical parametrization, which are ambiguous independent of shape orientation due to my flow based authalic parametrization. When the shapes are known to be upright, the z-axis ambiguity is naturally remedied by incorporating an additional feature map in the geometry image encoding the angle between a vertex normal and the z-axis [29]. This works better in practice than say realigning the north pole of the derived spherical parametrization with the highest point along the centroid axis. However, note that this is an extrinsic representation. The x and y axis ambiguities can be then be approximately resolved by rotating the spherical parametrization in equal intervals about the z-axis for each shape. This however is not applicable to general databases. In such a case, additional samples are created by fixing a predetermined number of points to be the north pole of the spherical parametrization as discussed later. The pixel mean is subtracted from each feature map before feeding the geometry image into a CNN.

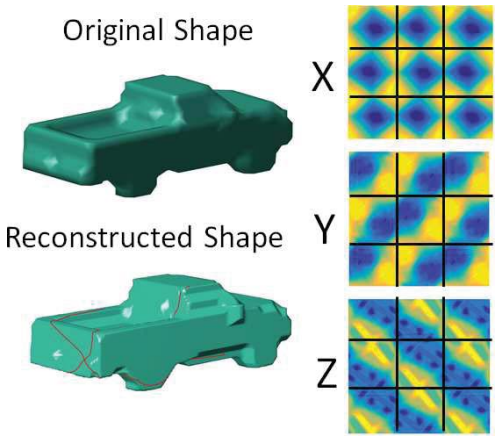


Figure 2.29. *Padding structure of geometry images: The geometry images for the 3 coordinates are replicated to produce a 3× 3 grid. The center image in each grid corresponds to the original geometry image. Observe no discontinuities exist along the grid edges.*

2.3.2 Deep learning for matrix completion

The easy availability of commercial depth cameras marked the advent of real time solutions to the human pose estimation problem [164]. The success of hand tracking naturally depends on synthesizing my knowledge of the hand (e.g., geometric shape, constraints on pose configurations) and latent features of the RGB-D data stream (e.g., region of interest, key feature points like finger tips, and temporal continuity). In this dissertation, I propose a novel method to achieve this synthesis by drawing on collaborative filtering approaches for recommender systems. My main insight is that a recommender system (e.g., Netflix) is very similar to a pose tracking system. Both systems have some *intrinsic* and *extrinsic* information about its constituent objects, the users in a recommender system and individual poses in a tracking system. The *intrinsic* knowledge of the hand in a tracking system corresponds to known user ratings in a recommender system. Similarly, the *extrinsic* RGB-D point cloud information corresponds to the metadata available about users (e.g., geographical locations, background, and interest). Specifically, the hand pose estimation problem is analogous to the cold-start problem in recommender systems. The cold-start problem in

recommender systems is to suggest personalized items to a new user with unknown preferences. In analogy to a tracking system, the hand pose estimation problem is to evaluate the unknown pose parameters of the kinematic hand model for a new point clouds appearing at every instant of time via a RGB-D sensor. A common technique to alleviate the cold-start problem is to suggest items to a new user based on recommendations available for like-minded users. The like-mindedness or similarity between users is evaluated using metadata such as age, gender, geographical location, interests, etc.

I propose a deep matrix completion approach to estimate the 3D pose of a hand using depth data from commercial 3D sensors. I discriminatively train convolutional neural networks to output a low dimensional activation feature given a depth map. This activation feature vector is representative of the global or local joint angle parameters of a hand pose. I efficiently identify 'spatial' nearest neighbors to the activation feature, from a database of features corresponding to synthetic depth maps, and store some 'temporal' neighbors from previous frames. My matrix completion algorithm uses these 'spatio-temporal' activation features and the corresponding known pose parameter values to estimate the unknown pose parameters of the input feature vector. My database of activation features supplements large viewpoint coverage and my hierarchical estimation of pose parameters is robust to occlusions. I show that my approach compares favorably to state-of-the-art methods while achieving real time performance ($\approx$ 32 FPS) on a standard computer.

Preliminaries: In this section, I briefly describe my 3D hand model and discuss my method to extract the region of interest corresponding to the hand which serves as input to my hand pose estimation method.

Hand model: I use a kinematic hand model with 21 degrees of freedom (DOF), represented as $\mathcal{H}(\boldsymbol{\theta}, \boldsymbol{\phi})$, as standard in hand pose estimation literature (see Figure 1.7d). $\boldsymbol{\theta}$ denotes the set of 18 joint angle parameters and $\boldsymbol{\phi}$ is the set of 3 global translation parameters (x, y and z) of the hand.

Region of interest extraction: Unlike the body, the hand occupies a relatively small region in the overall depth image obtained from the 3D depth camera. Hence, I preprocess the depth image to only include values that lie in the range of [50, 500] mm under the premise

that the hand lies within this range. I then do a largest blob detection as an indicator of the hand segment, followed by by median filtering for noise removal, depth normalization so that values lie in the range [0, 255], and finally resize the image while maintaining the aspect ratio to obtain a 64×64 depth image.The centroid of the blob in the original image marks the global position, ϕ. In more extreme settings (for ranges upto 2000 mm), I use a colored wristband as a simple indicator of the hand region as done in [88, 165]. Even in a close range scenario, the wristband helps removing extraneous pixels like those below the wrist, leading to better performance.

Dimensionality Reduction using Deep Learning: It is well known that the activation features from the intermediate hidden layers of a ConvNet can be re-purposed across domains. This suggests that the activation feature of a depth image itself contains discriminative cues about its overall shape and form of the hand, in the context of hand pose estimation. The thrust of my approach relies on the contention that a pool of nearby activation features is better able to reach consensus about the hand's orientation and shape. This introduces two challenges (1) The activation features in the population should conform to the activation features obtained from different individuals in diverse real settings. Additionally, they should be accurately annotated with their ground truth labels (joint angles or positions) (2) The population of activation features must be large enough to provide robust nearest neighbors to any input activation feature, however should be efficiently retrievable and consume limited memory. A straightforward approach is to directly use the depth data gathered from 3D sensors to train a ConvNet and store the corresponding activation features. However, creating a such database of hand poses to cover full range of hand articulations with accurate ground truth labels is a tedious task. In this section, I describe how I generate such a population of activation features from synthetic dataset, reflective of real data.

Synthetic population of realistic hand poses: I generate synthetic depth maps by first imposing static (e.g., range of motion, joint length, location) and dynamic (e.g., among joints and fingers) constraints listed in [166][3]. I then uniformly sample each of the 18 joint param-

[3]The availability of rigourous constraints in terms of joint angles is the main reason I choose angles over joint position in my hand pose method.

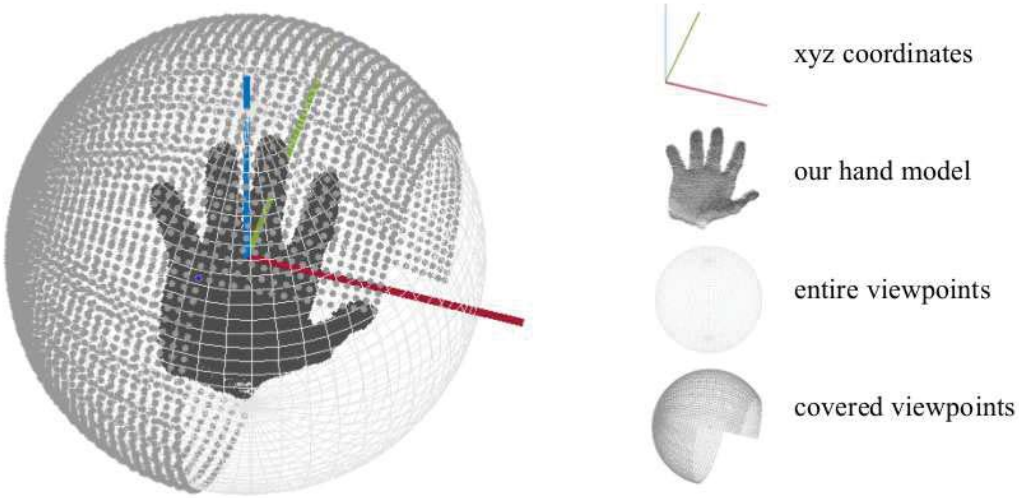


Figure 2.30. *The viewpoint coverage of the hand model. Under the assumption that the hand appears closest to the camera, my system covers nearly the entire viewpoints.*

eters in this restricted configuration space. This ensures that the depth maps are reflective of real poses covering a wide range of hand articulations. However, data from 3D sensors are prone to noise, distortion and additional artifacts. Hence, I add gaussian noise $N(0, \sigma^2)$ to the synthetic depth maps wherein the standard deviation σ is chosen from a range of [0, 2] by uniform sampling. The viewpoint coverage (coverage due to the 3 wrist rotation angles $\boldsymbol{\theta^W} = \{\theta_r^W, \theta_p^W, \theta_y^W\}$) is illustrated in Figure 2.30. My large coverage ensures the robustness my method to camera viewpoint changes and not restricted to near frontal poses. I discuss the size of the synthetic population in context to ConvNets in the next subsection.

Activation features using ConvNet: ConvNet and its variants are the current state of the art architecture for numerous classification tasks such as object detection, scene recognition, texture recognition and fine grained classification. However, hand tracking is effectively a regression task. My preliminary experiments with deep learning indicated that ConvNets do not adapt to regression as well as they do for classification. Consequently, my activation

Table 2.1.
Accuracy and memory comparison of global pose initialization.

Model	Accuracy	Memory	Settings
RF	57.45 %	1.30 GB	22 Depth, 70 Trees
	59.04 %	1.87 GB	22 Depth, 100 Trees
ConvNet	71.01 %	2.12 MB	20 Epochs
	72.30 %	2.12 MB	25 Epochs
PCA	5.72 %	None	

features are computed using ConvNet for classification instead of regression. These activation features feed into my matrix completion method which implicitly regresses and outputs the estimated joint angle parameters. The classification of joint angles into quantized bins, and hence, calculation of the activation feature in the penultimate layer, is performed by the ConvNet architecture. Observe that the penultimate layer corresponding to the activation feature is a 32 dimensional vector so as to reduce memory usage in storing the population of activation features.

There are two extremal strategies for quantization. The first strategy is to quantize each joint angle separately for a total of 21 ConvNets. However, this is inefficient both in terms of speed and memory. The second is to use an all-in-one strategy to train all joint angle parameters simultaneously. However, it would be impossible to learn an accurate classifier in such a high dimensional space even with a nominal number of bins. Hence, I use a 2-stage hierarchical strategy which satisfactorily balances computational time, memory requirement and classification accuracy.

In *Stage 1* the activation feature associated with the 3 global rotation angles, $\boldsymbol{\theta^W} = \{\theta_r^W, \theta_p^W, \theta_y^W\}$ is calculated and input into the matrix completion method along with a pool of nearest neighbors. The output of the matrix completion method is used to infer the correct rotation bin. For each rotation bin, five ConvNets are trained to output the activa-

tion feature associated with each of the five fingers. The ConvNets in *Stage 2* are trained on images within the bin to simplify learning and also on images in adjacent bins to prevent boundary errors. I used 200K images for *Stage 1* global regression (see Figure 1.7c) wherein the roll, pitch, yaw angles were quantized into 144 bins. Subsequently, 5 Convnets for each of 144 bins were trained on 10K images within the bin and 10K randomly chosen images in adjacent bins. Training converged after 20 Epochs for the global bin and approximately 10 Epochs for the local rotation bins. The discrete quantization over the joint angle values for each finger is as follows: thumb (144), index (144), middle (36), ring (144), and little (144).

The activation feature associated with the global rotation is critical to the overall accuracy of my approach because this step influences all subsequent ones. To demonstrate the efficacy of ConvNet relative to other approaches, I detail the classification accuracy of ConvNet for global rotation relative to PCA [85] and random forest (RF) [89]. I used 100K depth images because of RF's memory constraints. Table 2.1 shows that ConvNet achieves a very high accuracy with minimal memory requirement.

Deep Matrix Completion: The matrix completion algorithm runs 6 times: once for the 3 global rotation angles and 5 times for estimating the 15 joint angle parameters associated with the fingers. An iterative approach as the one in [91] is inefficient. Instead I evaluate the unknown parameters in a single shot by assuming a low rank matrix. I discuss the details of my nearest neighbor retrieval to create a pool of activation features followed by the matrix completion method below.

Extracting pool of activation features: My deep matrix completion method takes spatio-temporal nearest neighbors as input. Acquiring temporal nearest neighbors are trivial as they are simply the activation features from the previous frames. However, brute force nearest neighbor evaluation from say the 200K global activation vectors introduces a computational bottleneck unsuitable for realtime application. My solution to alleviate this problem is to use the top classes predicted by the softmax function in CNN to first reduce the search space. I then use highly efficient product quantization based nearest neighbor approximation [167] with 8 subquantizers to retrieve the desired number of nearest neighbors. Details

Table 2.2.
Overall architecture of my convolutional network. (Conv: convolutional layer, Pmax: max pooling layer, ReLU: rectified linear units layer, Smax: softmax layer).

	Layers	# Kernels	Filter size	Stride	Pad
1	Conv	16	5×5×1	1	2
2	Pmax			2	0
3	ReLU				
4	Conv	32	5×5×16	1	2
5	ReLU				
6	Pmax			2	0
7	Conv	32	5×5×32	1	2
8	ReLU				
9	Pmax			2	0
10	Conv	64	5×5×32	1	2
11	ReLU				
12	Pmax			2	0
13	Conv	128	4×4×64	1	0
14	ReLU				
15	Conv	32	1×1×128	1	0
16	ReLU				
17	Conv	144	1×1×32	1	0
18	Smax				

of product quantization are skipped for brevity. In practice, I found retrieving a higher fraction of approximate nearest neighbors by product quantization and then selecting the desired number of nearest neighbors using brute force search from this reduced subset to be more robust than direct retrieval.

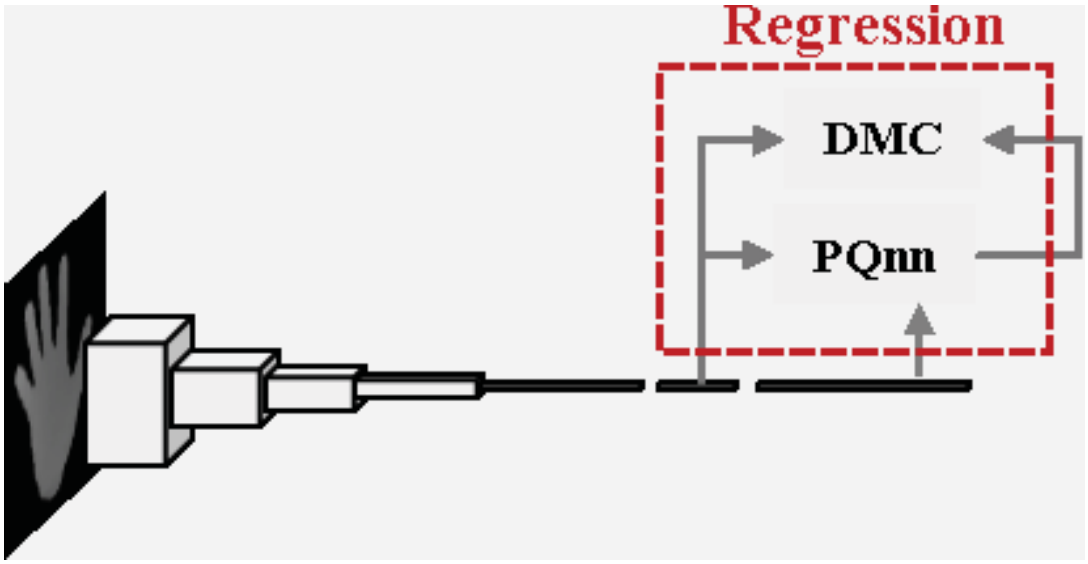


Figure 2.31. *The low dimensional activation features are learned from the sixth convolutional layer. I use these activation features in a collaborative spatio-temporal fashion to estimate pose parameters using efficient nearest neighbor search and my novel DMC model.*

The DMC model: Let n be number of spatial nearest neighbors, $\mathbf{D_1} \in \mathbb{R}^{n \times r}$ be the r dimensional activation vectors and $\mathbf{P_1} \in \mathbb{R}^{n \times m}$ be the m desired joint angle parameters being estimated of the n neighbors. In addition, let vector $\mathbf{d_2} \in \mathbb{R}^{1 \times r}$ be the r dimensional activation feature output from ConvNet. Let vector $\mathbf{p_2} \in \mathbb{R}^{1 \times m}$ be the unknown parameters.

$$\mathbf{M} = \begin{bmatrix} \mathbf{D_1} & \mathbf{P_1} \\ \mathbf{d_2} & \mathbf{p_2} \end{bmatrix} \tag{2.154}$$

My task is to estimate $\mathbf{p_2}$ given the other 3 block matrices. Assuming a low rank structure of matrix M this reduces ro solving:

$$\mathbf{p_2} = \mathbf{d_2}(\mathbf{D_1})^{-1}\mathbf{P_1}, \tag{2.155}$$

The proof of the above result is detailed next.

I present a mathematical derivation of the DMC model which estimates the unknown pose vector $\mathbf{p_2} \in \mathbb{R}^{1\times m}$ from the activation feature matrix $\mathbf{D} = [\mathbf{D_1}; \mathbf{d_2}]$ and the parameter matrix $\mathbf{P_1}$. I first search n spatially nearest hand poses to the input hand from my database, and then their activation features and annotated parameters are respectively fed into the matrix $\mathbf{D}$ and $\mathbf{P}$. Additionally, t numbers of previous frames are taken as temporal neighbors to fill the corresponding matrices. Next, I kernelize the feature matrix $\mathbf{D}$ in the form of radial basis function (RBF):

$$\mathbf{K}(\mathbf{D}, \mathbf{D}) = \exp\left(-\frac{\|\mathbf{D}^T\mathbf{D}\|^2}{2\sigma^2}\right), \tag{2.156}$$

where σ denotes a variance of the dataset (σ=200). The auxiliary knowledge about neighbors (i.e, kernelized similarity matrix $\mathbf{K}$) directly corresponds to joint parameters $\mathbf{P}$, and thus the estimation problem can be simplified. Now I suppose that $\mathbf{p_2}$ is a submatrix of the matrix $\mathbf{X}$.

$$\mathbf{X} = \begin{bmatrix} \mathbf{K_1} & \mathbf{P_1} \\ \mathbf{k_2} & \mathbf{p_2} \end{bmatrix} \tag{2.157}$$

where $\mathbf{K_1} \in \mathbb{R}^{(n+t)\times r}$, $\mathbf{k_2} \in \mathbb{R}^{1\times r}$, and $\mathbf{P_1} \in \mathbb{R}^{(n+t)\times m}$.

Property 4 *Suppose that the matrix* $\mathbf{X}$ *of rank* k *is partitioned as in Equation (2.157) and the matrix* $\mathbf{p_2}$ *also has rank* k*. Then*

$$\mathbf{p_2} = \mathbf{k_2}(\mathbf{K_1})^{+}\mathbf{P_1}, \tag{2.158}$$

where + denotes the Moore-Penrose pseudo-inverse.

Proof The proof of Equation (2.158) is very similar to that of Lemma 1 in [168]. The matrix $\mathbf{X}$ follows an SVD of rank k, thus $\mathbf{X} = U\Sigma V'$ where $\Sigma = diag(\sigma_1, \sigma_2, ..., \sigma_k)$, $U \in \mathbb{R}^{(n+t+1)\times k}$, and $V \in \mathbb{R}^{(r+m)\times k}$. Assume $U_1 \in \mathbb{R}^{(n+t)\times k}$ and $U_2 \in \mathbb{R}^{1\times k}$, subsequently $V_1 \in \mathbb{R}^{r\times k}$ and $V_2 \in \mathbb{R}^{m\times k}$. Then, I can rewrite $\mathbf{K_1} = U_1\Sigma V_1', \mathbf{P_1} = U_1\Sigma V_2', \mathbf{k_2} = U_2\Sigma V_1'$,

and $\mathbf{p_2} = U_2 \Sigma V_2'$. Let $\mathbf{K_1} = LR$, where $L = U_1 S$ and $R = S V_1'$ for $S = \sqrt{\Sigma}$. From MacDuffee's theorem,

$$\begin{aligned} \mathbf{K_1}^+ &= R^+ L^+ \\ &= R'(RR')^{-1}(L'L)^{-1}L' \\ &= V_1 S (S V_1' V_1 S)^{-1} (S U_1' U_1 S)^{-1} S U_1' \\ &= V_1 (V_1' V_1)^{-1} \Sigma^{-1} (U_1' U_1)^{-1} U_1' \end{aligned} \tag{2.159}$$

Consequently, $\mathbf{K_2}(\mathbf{K_1})^+ \mathbf{P_1}$ is

$$\begin{aligned} \mathbf{K_2}(\mathbf{K_1})^+ \mathbf{P_1} &= (U_2 \Sigma V_1') V 1 (V_1' V_1)^{-1} \Sigma^{-1} \\ &\qquad (U_1' U_1)^{-1} U_1' U_1 \Sigma V_2' \\ &= U_2 \Sigma V_2' \\ &= \mathbf{p_2}. \end{aligned} \tag{2.160}$$

■

In practice, I observed that kernelizing the feature matrix and regularizing it by adding a small constant, c to the diagonal, in the spirit of ridge regression makes the output more robust. This parameter c is set to 0.001 in all my experiments. I use the RBF kernel with sigma equal to the variance of the dataset ($\sigma = 200$). Therefore, the solution is given by:

$$\mathbf{p_2} = \mathbf{k_2} \left(\mathbf{K_1} + c\mathbf{I} \right)^{-1} \mathbf{P_1}, \tag{2.161}$$

A straightforward extension beyond including just the spatial neighbors is to also include t temporal neighbors from previous frames. This reduces jitter and improves the final quality of my solution. I use 60 nearest neighbors and 16 temporal neighbors for the global parameter estimation. For the 15 local angles, I use 24 nearest neighbors and 4 temporal neighbors.

2.4 Feedback Loops in Systems

Collaborative filtering is a popular technique to infer users' preferences on new content based on the collective information of all users preferences. Recommender systems then

use this information to make personalized content suggestions to users. When users accept these recommendations it creates a feedback loop in the recommendation system, and these loops iteratively influence the collaborative filtering algorithm's predictions over time. In this dissertation, I investigate whether it is possible to identify items affected by these feedback loops. I state sufficient assumptions to deconvolve the feedback loops while keeping the inverse solution tractable. I furthermore develop a metric to unravel the recommender system's influence on the entire user-item rating matrix. I use this metric on synthetic and real-world datasets to (1) identify the extent to which the recommender system affects the final rating matrix, (2) rank frequently recommended items, and (3) distinguish whether a user's rated item was recommended or an intrinsic preference. My results indicate that it is possible to recover the ratings matrix of intrinsic user preferences segregated from items recommended to the user by the recommender system using only a single snapshot of the ratings matrix without any temporal information.

2.4.1 Types of feedback

User feedback is commonplace in most recommender systems based on collaborative filtering. User feedback in collaborative filtering systems is categorized as either explicit feedback which includes input by users regarding their interest in products [169], or implicit feedback such as purchase and browsing history, search patterns, etc. [170]. There has been a considerable amount of work on incorporating the information from these two types of user feedback mechanisms in collaborative filtering algorithms in order to improve and personalize recommendations [171, 172]. Both types of feedback affect the item-item or user-user similarities used in the collaborative filtering algorithm for predicting future recommendations [173]. In this work I do not focus on improving collaborative filtering algorithms for recommender system by studying user feedback, but instead, my thrust is to recover the each user's intrinsic preference of an item by separating the rating bias introduced by the recommender system due to feedback.

Another line of work based on user feedback in recommender systems is related to understanding the exploration and exploitation tradeoff [174] associated with the training feedback loop in collaborative filtering algorithms. Solutions to this dilemma are commonly used to maximize clicks or advertising revenue [175]. This line of research evaluates *'what-if'* scenarios such as evaluating the performance of alternative collaborative filtering models to the one used for data collection or, adapting the algorithm based on user-click feedbacks to maximize a notion of reward. Popular approaches towards this end include the multi-armed bandit setting [176,177] in reinforcement learning and counterfactual learning systems [178]. In contrast to these methods which find out actions or recommendations in order to maximize payoff in an alternate model or dynamically evolving scenarios, I tackle the problem of recovering the true ratings matrix if feedback loops were absent.

My mechanism to deconvolve feedback effects in recommender systems is similar in spirit to deconvolution algorithms introduced in the network science literature which aim to separate direct effects from indirect dependencies in biological networks [179, 180]. Indeed, my approach can be viewed as a generalization of their methods for general rectangular matrices. I now provide my model for a recommender system and describe an algorithm to deconvolve the feedback loops.

2.4.2 Deconvolving feedback loops

In this section, I first state equations for a simplified recommender system with feedback loops and then derive an implicit relationship between the true rating matrix and the observed rating matrix. I state the assumptions under which the true rating matrix is recoverable (or deconvolvable) from the observed matrix, and provide an algorithm to deconvolve the observed matrix using the singular value decomposition (SVD).

A model recommender system: Consider a ratings matrix $\boldsymbol{R}$ of dimension $m \times n$ where m is the number of users and n is the number of items being rated. Users are denoted by subscript u, and items are denoted by subscript i, i.e., $\boldsymbol{R}_{u,i}$ denotes user u's rating for item i. As stated in (1.3), the observed rating matrix is a convolution of the true user ratings and

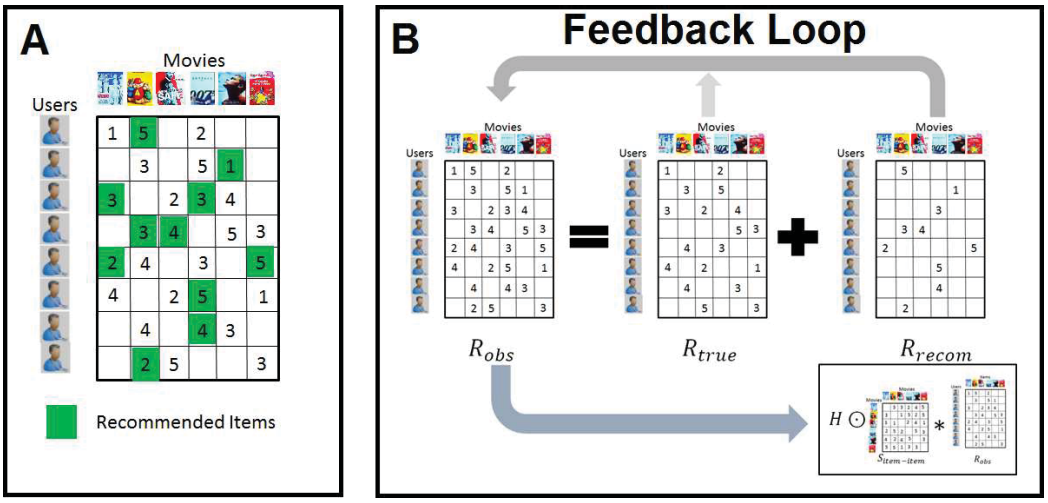


Figure 2.32. *Figure A: A ratings matrix with recommender induced ratings and true ratings; Figure B: Feedback loop in recommender system wherein the observed ratings is a function of the true ratings and ratings induced by a recommender system.*

ratings as a consequence of the recommender system. My objective is to decouple $\boldsymbol{R}_{\text{true}}$ from $\boldsymbol{R}_{\text{recom}}$ given the matrix $\boldsymbol{R}_{\text{obs}}$. Although this problem seems intractable, I list a series of assumptions under which a closed form solution of $\boldsymbol{R}_{\text{true}}$ is deconvolvable from $\boldsymbol{R}_{\text{obs}}$ alone.

Assumption 1: The feedback in the recommender occurs through the iterative process involving the observed ratings and an item-item similarity matrix $\boldsymbol{S}$:

$$\boldsymbol{R}_{\text{obs}} = \boldsymbol{R}_{\text{true}} + \boldsymbol{H} \odot (\boldsymbol{R}_{\text{obs}}\boldsymbol{S}). \tag{2.162}$$

Recall that $\odot$ indicates Hadamard, or entrywise product, given as: $(\boldsymbol{H} \odot \boldsymbol{R})_{u,i} = \boldsymbol{H}_{u,i} \cdot \boldsymbol{R}_{u,i}$. This assumption is justified because in many collaborative filtering techniques, $\boldsymbol{R}_{\text{recom}}$ is a function of the observed ratings $\boldsymbol{R}_{\text{obs}}$. The matrix $\boldsymbol{H}$ is an indicator matrix over a set of items where the user followed the recommendation and agreed with it. This matrix is essentially completely unknown and is essentially unknowable without direct human interviews.

The model recommender system Equation (2.162) then iteratively updates $\boldsymbol{R}_{\text{obs}}$ based on commonly rated items by users. This key idea is illustrated in Figure 2.32. The recursion progressively fills all missing entries in matrix $\boldsymbol{R}_{\text{obs}}$ starting from $\boldsymbol{R}_{\text{true}}$. Mathematically, I get

$$\begin{aligned}\boldsymbol{R}_{\text{obs}} &= \boldsymbol{R}_{\text{true}} + \boldsymbol{H}^{(1)} \odot \Big(\big(\boldsymbol{R}_{\text{true}} + \boldsymbol{H}^{(2)} \odot (\boldsymbol{R}_{\text{obs}} \boldsymbol{S}) \big) \boldsymbol{S} \Big) \\ &= \boldsymbol{R}_{\text{true}} + \boldsymbol{H}^{(1)} \odot \boldsymbol{R}_{\text{true}} \boldsymbol{S} \\ &\quad + \boldsymbol{H}^{(1)} \odot (\boldsymbol{H}^{(2)} \odot (\boldsymbol{R}_{\text{true}} \boldsymbol{S})) \boldsymbol{S} + \cdots \end{aligned} \tag{2.163}$$

Thus, I see that $\boldsymbol{R}_{\text{obs}}$ is an implicit function of $\boldsymbol{R}_{\text{true}}$ and $\boldsymbol{S}$.

Assumption 2: I approximate the Hadamard product with a probability parameter $\alpha \in (0, 1]$.

I briefly justify this assumption. Note that $\boldsymbol{R}_{\text{obs}}$ becomes denser with every iteration, and hence the higher order Hadamard products in the series fill fewer missing terms. In my simplified system, I model the selection matrix H and it's Hadamard problem in expectation and replace the successive matrices H with independent Bernoulli random matrices with probability α. Taking the expectation allows me to replace the matrix H with the probability parameter α itself. The effect, however, is similar. Higher powers of α produce successively less of an impact, just as in the true model. This function is instead carried out by α in the reformulated equation, wherein higher powers of α tend to 0 and progressively have lesser effect on the summand:

$$\begin{aligned}\boldsymbol{R}_{\text{obs}} &= \boldsymbol{R}_{\text{true}} + \alpha \boldsymbol{R}_{\text{true}} \boldsymbol{S} + \alpha^2 \boldsymbol{R}_{\text{true}} \boldsymbol{S}^2 + \alpha^3 \boldsymbol{R}_{\text{true}} \boldsymbol{S}^3 + \cdots \\ &= \boldsymbol{R}_{\text{true}} (I + \alpha \boldsymbol{S} + \alpha^2 \boldsymbol{S}^2 + \alpha^3 \boldsymbol{S}^3 + \cdots) \end{aligned} \tag{2.164}$$

It is now possible to recover $\boldsymbol{R}_{\text{true}}$ by inverting the matrix $(I + \alpha \boldsymbol{S} + \alpha^2 \boldsymbol{S}^2 + \alpha^3 \boldsymbol{S}^3 + \cdots)$, but $\boldsymbol{S}$ is *apriori* unknown. Thus, my next assumption.

Assumption 3: The item-item similarity used in the recommender system is induced by $\boldsymbol{R}_{\text{true}}$.

Although it is ideal to represent $\boldsymbol{S}$ as a dynamic function of $\boldsymbol{R}_{\text{obs}}$, the temporal dependency of $\boldsymbol{S}$ on $\boldsymbol{R}_{\text{obs}}$ makes the problem more intricate. Instead, I represent $\boldsymbol{S}$ in terms of $\boldsymbol{R}_{\text{true}}$, which decouples the interaction effects between the recommender system and users by only considering the direct relationships in the similarity calculation.

I am now faced with the task of constructing a valid similarity metric using $\boldsymbol{R}_{\text{true}}$. Towards this end, I make my final assumption. *Assumption 4*: The user mean $\bar{\boldsymbol{R}}_u$ in the observed and true matrix are roughly equal: $\bar{\boldsymbol{R}}_u^{(\text{obs})} \approx \bar{\boldsymbol{R}}_u^{(\text{true})}$. The Euclidean item norms $\|\boldsymbol{R}_i\|$ are also roughly equal: $\|\boldsymbol{R}_i^{(\text{obs})}\| \approx \|\boldsymbol{R}_i^{(\text{true})}\|$.

These assumptions are justified because ultimately I are interested in relative preferences of items for a user and unbiased relative ratings of items by users. These can be achieved by centering users and the normalizing item ratings, respectively, in the true and observed ratings matrices. The induced similarity metric then becomes:

$$\boldsymbol{S}(i,j) = \frac{\sum_{u\in U}(\boldsymbol{R}_{u,i} - \bar{\boldsymbol{R}}_u)(\boldsymbol{R}_{u,j} - \bar{\boldsymbol{R}}_u)}{\sqrt{\sum_{u\in U}(\boldsymbol{R}_{u,i} - \bar{\boldsymbol{R}}_u)^2}\sqrt{\sum_{u\in U}(\boldsymbol{R}_{u,j} - \bar{\boldsymbol{R}}_u)^2}} \tag{2.165}$$

This metric is known as the adjusted cosine similarity, and preferred over traditional cosine similarity because it mitigates the effect of rating schemes over users [181]. Using the relations:

$$\tilde{\boldsymbol{R}}_{u,i} = \boldsymbol{R}_{u,i} - \bar{\boldsymbol{R}}_u; \text{and} \tag{2.166}$$

$$\hat{\boldsymbol{R}}_{u,i} = \frac{\tilde{\boldsymbol{R}}_{u,i}}{\|\tilde{\boldsymbol{R}}_i\|} = \frac{\boldsymbol{R}_{u,i} - \bar{\boldsymbol{R}}_u}{\sqrt{\sum_{u\in U}(\boldsymbol{R}_{u,i} - \bar{\boldsymbol{R}}_u)^2}} \tag{2.167}$$

the governing expression of my simplified recommender (2.164) becomes

$$\begin{aligned}\hat{\boldsymbol{R}}_{\text{obs}} = \hat{\boldsymbol{R}}_{\text{true}}(I + \alpha\hat{\boldsymbol{R}}_{\text{true}}^T\hat{\boldsymbol{R}}_{\text{true}} + \alpha^2(\hat{\boldsymbol{R}}_{\text{true}}^T\hat{\boldsymbol{R}}_{\text{true}})^2 \\ + \alpha^3(\hat{\boldsymbol{R}}_{\text{true}}^T\hat{\boldsymbol{R}}_{\text{true}})^3 + \cdots)\end{aligned} \tag{2.168}$$

I see that the centering and normalizing both sides of the equation results in $\hat{\boldsymbol{R}}_{\text{obs}}$ being explicitly represented in terms of $\hat{\boldsymbol{R}}_{\text{true}}$ and α. I discuss the mechanism of solving this equation in my main theorem.

The algorithm for deconvolving feedback loops

Theorem 2.4.1 *Assuming the recommender system follows* (2.168) *and the singular value decomposition of the observed rating matrix is,* $\hat{\boldsymbol{R}}_{obs} = \boldsymbol{U}\boldsymbol{\Sigma}_{obs}\boldsymbol{V}^T$*, the deconvolved matrix* $\boldsymbol{R}_{true}$ *of true ratings is given as* $\boldsymbol{U}\boldsymbol{\Sigma}_{true}\boldsymbol{V}^T$*, where the* $\boldsymbol{\Sigma}_{true}$ *is a diagonal matrix with elements:*

$$\sigma_i^{true} = \frac{-1}{2\alpha\sigma_i^{obs}} + \sqrt{\frac{1}{4\alpha^2(\sigma_i^{obs})^2} + \frac{1}{\alpha}} \tag{2.169}$$

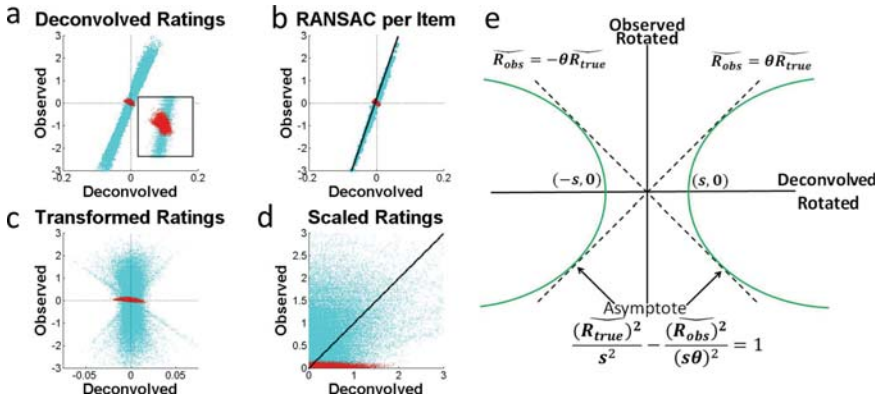


Figure 2.33. *(a) to (e): My procedure for scoring ratings based on the deconvolved scores with true initial ratings in cyan and ratings due to recommender in red. (a) The observed and deconvolved ratings. (b) The RANSAC fit to extract straight line passing through data points for each item. (c) Rotation and translation of data points using fitted line such that the scatter plot is approximately parallel to y-axis and recommender effects are distinguishable along x-axis. (d) Scaling of data points used for subsequent score assignment. (e) Score assignment using the vertex of the hyperbola with slope $\theta = 1$ that passes through the data point.*

where α is between 0 and 1.

Proof Both $\boldsymbol{U}, \boldsymbol{V}$ are orthogonal matrices of dimension $m \times m$, and $n \times n$, respectively and $\boldsymbol{\Sigma}_{\text{true}}$ is a non-negative diagonal matrix of singular values. Then, the eigenvalue decomposition of $\boldsymbol{S}$ is given as:

$$\begin{aligned} \boldsymbol{S} &= \hat{\boldsymbol{R}}^T \hat{\boldsymbol{R}} \\ &= (\boldsymbol{U}\boldsymbol{\Sigma}_{\text{true}}\boldsymbol{V}^T)^T \boldsymbol{U}\boldsymbol{\Sigma}_{\text{true}}\boldsymbol{V}^T \\ &= \boldsymbol{V}\boldsymbol{\Sigma}_{\text{true}}^2\boldsymbol{V}^T. \end{aligned} \tag{2.170}$$

By applying the Taylor series summation in Equation (2.168), I have:

$$\hat{\boldsymbol{R}}_{\text{obs}} = \hat{\boldsymbol{R}}_{\text{true}}(\boldsymbol{I} - \alpha\boldsymbol{S})^{-1}. \tag{2.171}$$

Note that this is valid if $\alpha(\sigma^{\text{true}})_i^2 < 1$ for all i in order for the series to converge. Below, I show that this holds for the choice of σ^{true} in the theorem. In spectral form, I then have

$$(\boldsymbol{I} - \alpha\boldsymbol{S})^{-1} = \boldsymbol{V}(\boldsymbol{I} - \alpha\boldsymbol{\Sigma}_{\text{true}}^2)^{-1}\boldsymbol{V}^T. \tag{2.172}$$

Hence,

$$\begin{aligned}\hat{\boldsymbol{R}}_{\text{obs}} &= \boldsymbol{U}\boldsymbol{\Sigma}_{\text{true}}\boldsymbol{V}^T\boldsymbol{V}(\boldsymbol{I} - \alpha\boldsymbol{\Sigma}_{\text{true}}^2)^{-1}\boldsymbol{V}^T \\ &= \boldsymbol{U}\boldsymbol{\Sigma}_{\text{true}}(\boldsymbol{I} - \alpha\boldsymbol{\Sigma}_{\text{true}}^2)^{-1}\boldsymbol{V}^T.\end{aligned} \tag{2.173}$$

Let the singular value decomposition of the observed rating matrix $\hat{\boldsymbol{R}}_{\text{obs}}$ be

$$\hat{\boldsymbol{R}}_{\text{obs}} = \boldsymbol{U}\boldsymbol{\Sigma}_{\text{obs}}\boldsymbol{V}^T \tag{2.174}$$

Using the Equations (2.173) and (2.174) I find

$$\boldsymbol{\Sigma}_{\text{obs}} = \boldsymbol{\Sigma}_{\text{true}}(\boldsymbol{I} - \alpha\boldsymbol{\Sigma}_{\text{true}}^2)^{-1} \quad \text{i.e.} \tag{2.175}$$

$$\sigma_i^{\text{obs}} = \frac{\sigma_i^{\text{true}}}{1 - \alpha(\sigma_i^{\text{true}})^2} \quad \forall i. \tag{2.176}$$

Solving the quadratic in σ_i^{true} in terms of σ_i^{obs}, and considering only the positive value, I get:

$$\sigma_i^{\text{true}} = \frac{-1}{2\alpha\sigma_i^{\text{obs}}} + \sqrt{\frac{1}{4\alpha^2(\sigma_i^{\text{obs}})^2} + \frac{1}{\alpha}} \tag{2.177}$$

What remains to show is that this choice of σ_i^{true} satisfies the convergence assumption above. This can be inferred through the following inequalities:

$$\begin{aligned}&\sqrt{\frac{1}{4\alpha^2(\sigma_i^{obs})^2} + \frac{1}{\alpha}} < \sqrt{\frac{1}{\alpha}} + \sqrt{\frac{1}{4\alpha^2(\sigma_i^{obs})^2}} \\ &\sqrt{\frac{1}{4\alpha^2(\sigma_i^{obs})^2} + \frac{1}{\alpha}} < \sqrt{\frac{1}{\alpha}} + \frac{1}{2\alpha\sigma_i^{obs}} \\ &\frac{-1}{2\alpha\sigma_i^{obs}} + \sqrt{\frac{1}{4\alpha^2(\sigma_i^{obs})^2} + \frac{1}{\alpha}} < \sqrt{\frac{1}{\alpha}} \\ &\sigma_i^{true} < \sqrt{\frac{1}{\alpha}} \\ &\alpha(\sigma_i^{true})^2 < 1\end{aligned} \tag{2.178}$$

■

In practical applications, the feedback loops are deconvolved by taking a truncated-SVD (low rank approximation) instead of the complete decomposition. In this process, I naturally concede accuracy for performance. I consider the matrix of singular values $\tilde{\boldsymbol{\Sigma}}_{\text{obs}}$ to only contain the k largest singular values (the other singular values are replaced by zero). The algorithm is simple and easy to compute as it just involves a singular value decomposition of the ratings matrix.

3. APPLICATIONS AND RESULTS

In this chapter I discuss the applications of my theoretical framework of physical laws to networks, shapes and images in the order they were introduced.

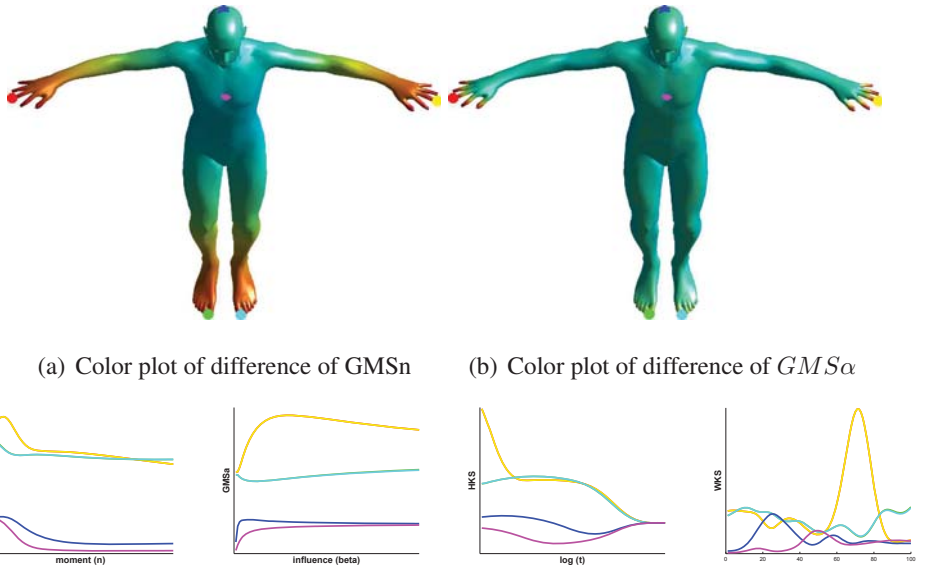
3.1 Random Walk Based Results on Shapes and Images

The use of multiscale kernels is discussed for shape analysis, and results of the diffusion modulus criterion are shown on standard benchmarks.

3.1.1 Application of multiscale kernels to shape analysis

The multiscale kernels consolidate popular metrics under a single framework and I discuss possible applications.

Multiscale Kernel: In the last chapter, I stressed that the intuition driving the formulation was that of using the rate of change instead of the probability values, and hence, is fundamentally different from the heat kernel. However, the resulting kernel is functionally equivalent to heat kernel, i.e., exponential with parameter n (instead of t) with logarithmic weighting of eigenvalues ($\lambda^{-n} = e^{-n \ln \lambda}$ with $\lambda > 0$). My first observation is that as I average over entire time, the resulting kernel possesses robust multiscale characteristics. Second, the logarithmic weighting of eigenvalues (instead of linear in heat kernel) makes the multiscale kernel discriminative at all scales, unlike the heat kernel which converges to a uniform distribution. To see this, logarithm of eigenvalues in the interval $(0, 1)$ are negative, hence contribute exponentially to the Green's function and for very large scales, i.e, as $n \to \infty$, the sum will be dominated by the normalized Fielder vector and second eigenvalue. Specifically, the multiscale kernel embedding will tend to $\sqrt{vol^n}\lambda_2^{-n/2}\Phi_2^T D^{-1/2}$ in the limit . Inspired by the remarkable informative property of the heat kernel and these observations, I define the dual Green's Mean Signature (GMS) of vertex u to be $|(\frac{\mathcal{G}^n_{uu}}{\sum_v \mathcal{G}^n_{vv}})|$



(a) Color plot of difference of GMSn (b) Color plot of difference of $GMS\alpha$

(c) Green's over n (d) Green's over β (e) Heat Kernel (HKS) (f) Wave Kernel (WKS)

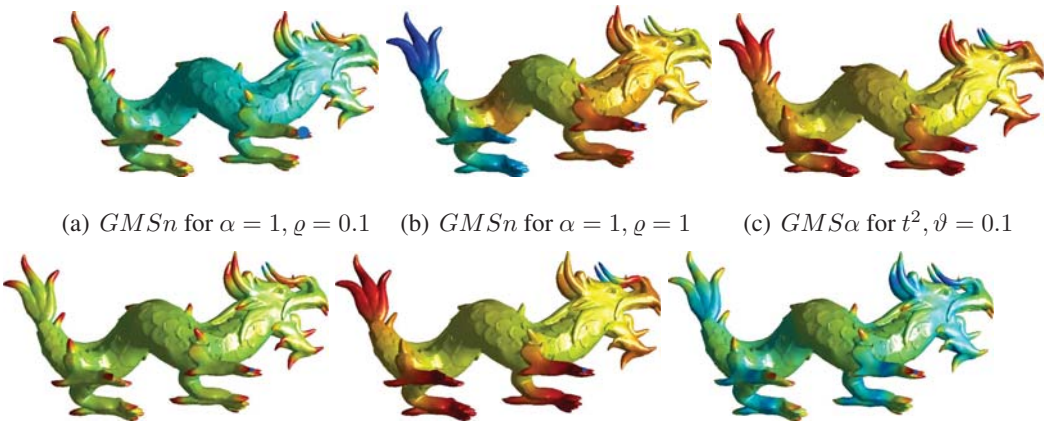
Figure 3.1. *Green's Mean Signature difference ($GMSn$ and $GMS\alpha$) between finger and other points on David model (top) based on entire range of parameters. In color plot red indicates similar and cyan indicates dissimilar, $\alpha = 1$ for $GMSn$ and $n = 2$ for $GMS\alpha$. (a)-(d) represent the 4 signatures for 6 anchor points. Signatures for 2 fingers and 2 feet are indistinguishable because of symmetry for all signatures. Note $GMSn$ and $GMS\alpha$ are discriminative at all scales (unlike HKS) and multiscale (unlike WKS).*

and $|(\frac{\mathcal{G}^{\alpha}_{uu}}{\sum_v \mathcal{G}^{\alpha}_{vv}})|$, and name them $GMSn$ and $GMS\alpha$, to indicate the multiscale nature over moment and influence parameter respectively. The α parameter used in my multiscale kernels has a direct semblance with the parameter t used to derive multiscale biharmonic kernels. Note the parameter α (hence β) performs a frequency shift of the eigenvalues, and hence, indirectly enforces partial support of the eigenfunctions. Interestingly, the wave kernel signature (WKS) [13] can be linked to random walks, specifically maximal entropy random walks (MERW) [182]. The transition matrix (M_Ψ) for MERW is given by

$M_\Psi(u, v) = \frac{\bar{M}(u,v)\bar{\Phi}_{0v}}{\bar{\lambda}_0\bar{\Phi}_{0u}}$

where $\bar{M} = DM = D - L_c$ and $\bar{\lambda}, \bar{\Phi}$ are associated eigenvalues and eigenvectors respectively. The unique stationary distribution is $(\bar{\Phi}_{0i})^2$ and can be interpreted as the prob-

ability of finding a particle in the ground state of the operator $(-\bar{M})$ with wave function $\bar{\Phi}_{0i}$ [183]. The particular choice of log-normal energy distribution in WKS is derived from a perturbation-theoretical analysis. $GMSn$ emphasizes on global features and has higher '*specificity (detect false negative errors)*' while $GMS\alpha$ suppresses large-scale features and has higher '*sensitivity (detect false positive errors)*', similar to HKS and WKS respectively, as argued in [184]. Having shown the interrelationship to HKS, WKS and multiscale biharmonic kernels, I now discuss computation and compare my signatures. As mentioned earlier, I find the 300 smallest eigenvalues and associated eigenvectors of $\mathcal{L}$, or alternatively solve for the generalized eigenvalue problem $L_c\Phi = D\lambda\Phi$. Using the formulae in the previous sections, I calculate the $GMSn$ or $GMS\alpha$ as required by fixing the range of n to $[0, \varrho log(1/\lambda_2)]$ and range of β to $[0, \vartheta\lambda_{300}]$ where ϱ, ϑ are scale tuning parameters with 100 linear increments over the range. $1/\lambda_2$ is the mixing time of a random walk and as GMSn kernels are logarithmically weighted over eigenvalues, hence transforming the logarithmic time scaling in heat kernel to linear scaling over n, I empirically take the logarithm of the mixing time. Figure 3.1 shows the comparison between $GMSn, GMS\alpha, HKS, WKS$ for David model and I confirm the theoretical claim that the set of GMS signatures are discriminative at all scales unlike the HKS, which decay to equilibrium distribution. Also, the ϱ, ϑ parameters provide a multiscale setting which is not possible with WKS. Figure 3.1 shows matching between the node on finger marked by red sphere and the rest of the shape, by calculating the L_2-norm of the difference between point signatures. It is represented using color plot, with similarity decreasing as color changes from red to cyan and $\varrho = 1, \vartheta = 1$. I observe that the $GMSn$ provides a global understanding of the shape while the $GMS\alpha$ provides a local understanding hence matching the fingers. These characteristics are further exemplified using dragon model in Figure 3.10 where I show multiscale matching by varying the parameters ϱ and ϑ. The specificity of $GMSn$ in matching the legs and local sensitivity of $GMS\alpha$ in identifying protrusions, can be coherently combined in a learning framework to construct optimal signatures, using the approach in [184], which I leave to future work.



(a) $GMSn$ for $\alpha = 1, \varrho = 0.1$ (b) $GMSn$ for $\alpha = 1, \varrho = 1$ (c) $GMS\alpha$ for $t^2, \vartheta = 0.1$

(d) $GMS\alpha$ for $t^2, \vartheta = 1$ (e) HKS (f) WKS

Figure 3.2. *Signature similarity to point marked on front feet with blue sphere in (a) with difference increasing as color ranges from red to cyan to blue. (a)-(b) GMSn signature for two values of ϱ. (c)-(d) $GMS\alpha$ signature for two values of ϑ. Increasing ϱ produces global signature while increasing ϑ produces local signature. (e) HKS and (c); (f) WKS and (a) have similar color plots.*

Multiscale Distance: The 0^{th} moment operator is $I - \pi \mathbf{1}^T$ and corresponding distance is $vol(\frac{1}{d_u} + \frac{1}{d_v})$. This can be understood as $\frac{vol}{d_u}$ is the distance from any node on the graph to vertex u, because the expected value for a random walker would depend only on the degree (scaled area) of the destination vertex and by symmetry. I only recover local connectivity, thus naturally at the lowest scale in my set of multiscale kernels. The commute time distance is the expected time for a random walker to travel from one vertex to another and back, i.e., it is the first moment. The commute distance derived in [18] is precisely the multiscale distance for $n = 1$ and $\alpha = 1$. Considering $n = 2$ and $\alpha = 1$, the distance metric is equivalent to biharmonic distance [14] upto a constant scaling factor. The biharmonic distance is thus the second moment of the rate of heat diffusion which I discern to be more intuitive, i.e., it weights diffusion rate between 2 nodes by t^2 and hence is a more global metric relative to the commute time distance. The diffusion distance between 2 nodes separated by time t is defined as $(\mathcal{D}_{uv}^{Diff})^2 = \sum_{i \geq 1} e^{-t\tilde{\lambda}_i} \left(\tilde{\Phi}_i(u) - \tilde{\Phi}_i(v)\right)^2$. These

multiscale distances can be viewed as performing a Mellin transform on $e^{-t\Lambda}$ (Λ is a constant for given mesh). It replaces the exponential over t weighting scheme by zeta function ($\zeta(n) = \sum_{\lambda_i \neq 0} \lambda_i^{-n}$) which is exponential over moment n and infers scale akin to time. Such an approach has been applied to the trace of the heat kernel to successfully extract graph characteristics [185]. Briefly, I mention that the geodesic distance for a graph is the shortest distance between 2 points and translates into finding the smallest t for which p_{uv}^t is non-zero. Hence it is connected to the floor function of the multiscale kernel distances. Thus, these kernels can be viewed as generalization of existing distance metrics and with additional parameter β, opens up a host of possibilities. I introduce scale sensitive variant of Shepard interpolation [186] demonstrated in [14]. The Shepard interpolant with zeroth order precision for point u on the mesh is defined as $f(u) = \frac{\sum_i w_i(u) f_i}{\sum_i w_i(u)}$ where I choose $w_i(u)$ to be scale sensitive weights proportional to $1/\mathcal{D}_{uv_i}^{\alpha n}$. Figure 3.3 shows the interpolant on the mesh along with value (ς) at fixed anchor point (red sphere) for different β values at 11 anchor points and 2 multiscale kernels ($n = 2, 4$). The β values for bottom, middle and top anchor points (blue sphere) indicated by triple in parentheses [bottom, middle, top] are constants multiplied with λ_2 and fix the scale sensitive weights w_i. The values f_i at anchor points are set at -1,0,1 for bottom, middle and top anchor points respectively. I see that changing the weighting scheme, allows the anchor points to increase or decrease dominance over the function values at other points on the mesh surface. A large value of β will only allow local influence as shown in Plot 3.3b, while reversing the set of weights as shown in Plots 3.3c,3.3e have very different effects on the value at the red anchor point. Changing n from 2 to 4 in Plots 3.3d,3.3f, affects the magnitude at the red anchor point.

Multiscale Embedding: The first moment embedding is equivalent to the GPS embedding upto scaling Υ [17]. I discuss some insights of these embeddings useful for shape segmentation. Figure 3.4 shows the segmentation of the ant model using naive kmeans for different moments. The 0^{th} moment embedding or the eigenvectors without eigenvalue scaling correctly segments the head, torso and abdomen of the ant model. However, increasing the moment results in skewed segmentations. As kmeans minimizes distortion, central regions which have highest affinity to other regions tend to be clustered together (the n^{th} embed-

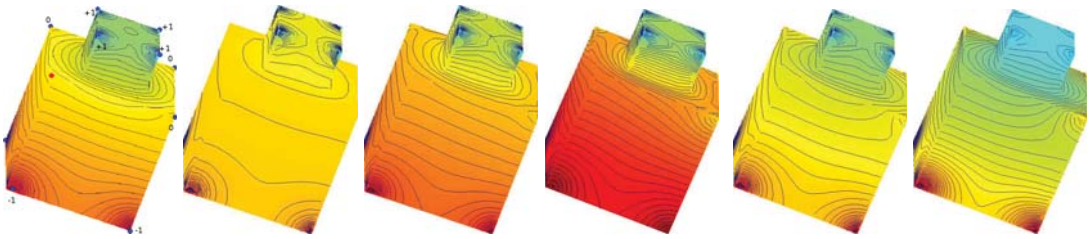


Figure 3.3. *Scale Sensitive Function Interpolation: Function values ς for point marked with red sphere with 4 bottom, 3 middle and 4 top row anchor (blue) points set at -1, 0, +1 respectively for varying n and β (hence α) weighting. ς can vary from negative to positive value based on β weighting and magnitude can be tuned using n.*

ding preserves the n^{th} distance moment). Indeed, larger values of k are required as input to kmeans at higher scales, to identify salient segments and the torso region suffers from oversegmentation. Consequently, the 0^{th} embedding of the kernel is best suited for segmentation, but it provides little information as a distance metric. As a second example, I illustrate the advantage of using scaled G2-distributions for calculating the shape similarity and subsequent MDS projection by varying β in Equation (2.41) as introduced in [17]. Figure 3.5 displays the MDS projection for 11 different shapes which include 11 isometric models of Victoria and Michael for two values of β; $\beta_1 = 0$ which corresponds to using G2-distributions ($n = 1$) and G2α-distributions for $\beta_2 = 100\lambda_2$. I see that β_2 is able to better discriminate between the models: dog, wolf, Michael and Victoria, and is also able to separate the 11 isometric models of Victoria (blue spheres) and Michael (red spheres) into two tight clusters, unlike the MDS projection using the original G2 distribution (β_1). This is because β acts as a *'scale-shift'* parameter and the embedding with β_2 is more localized, and hence, neglects contribution from distant (possibly noisy) nodes. I can create a family of distributions similar to G2 by varying n in Equation (2.41) and these distributions can have interesting yet intuitive implications based on the application.

(a) $K = 3, n = 0, \beta = 0$ (b) $K = 11, n = 0, \beta = 0$ (c) $K = 3, n = 1, \beta = 0$ (d) $K = 12, n = 1, \beta = 0$ (e) $K = 3, n = 4, \beta = 0$ (f) $K = 14, n = 4, \beta = 0$

Figure 3.4. *Sensitivity of multiscale embeddings to shape segmentation: Increasing moment from $n = 0$ (a)-(b) to $n = 1$ (c)-(d) to $n = 4$ (e)-(f), leads to oversegmentation of torso region into 1,2 and 4 segments for $n = 0, 1, 4$ respectively, while ensuring all 6 legs and 2 antennas are segmented. K is input to kmeans and $\beta = 0$.*

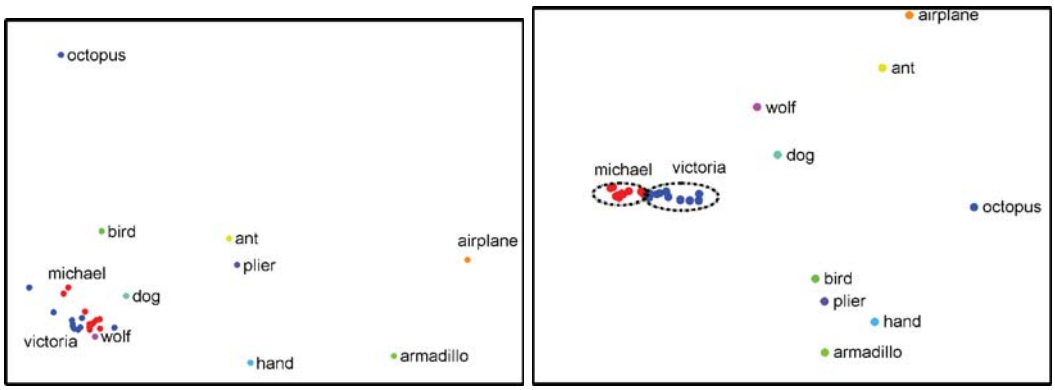


Figure 3.5. *Scale sensitive G2-distributions: Classical MDS projection for shape similarity by summing L2 distances between 36 G2α-distribution histograms with 50 dimensions for 11 shape models (see [17] for details) (b) $\beta_2 = 100\lambda_2$ is more discriminative between shapes relative to (a) $\beta_1 = 0$.*

3.1.2 Results on image segmentation using diffusion modulus

I demonstrate the efficacy of my criterion developed in the last chapter on the BSDS and MSRC object recognition database[1]. For oversegmenting the image, the parameters for mean shift algorithm are fixed to $(h_s, h_r, P) = (8, 4, 25)$ where h_s, h_r are the scale and resolution parameters and P controls the minimum number of pixels in a region. Ta-

[1] http://research/microsoft.com/en-us/projects/objectclassrecognition

Table 3.1.
Quantitative comparison to 10 other methods on BSDS. Top 3 results for each criterion highlighted in bold.

Methods/Score	PRI	VoI	GCE	BDE
NCut [67]	0.7242	2.9061	0.2232	17.15
MNCut [70]	0.7559	2.4701	0.1925	15.10
MShift [68]	0.7958	1.9725	0.1888	14.41
FH [69]	0.7139	3.3949	**0.1746**	16.67
JSEG [187]	0.7756	2.3217	0.1989	14.40
NTP [188]	0.7521	2.4954	0.2373	16.30
Saliency [189]	0.7758	1.8165	0.1768	16.24
UCM [66]	**0.85**	**1.47**	N/A	N/A
FAC [78]	0.8146	1.8545	0.1809	**12.21**
SAS [72]	**0.8319**	**1.6849**	0.1779	**11.21**
DM	0.8006	1.8742	0.2125	14.80
DM(WB)	0.8262	1.8045	**0.1679**	13.63
DM(MS)	**0.8322**	**1.6181**	**0.1757**	**13.45**

ble 3.1 lists the qualitative evaluation of my algorithm, compared to 10 other conventional methods on BSDS, which contains 300 images belonging to different categories. I use four standard measures for evaluation: Probabilistic Rand Index (PRI) which counts consistent pixel pairs between ground truth and segmented image; Variation of Information (VoI), which measures the information shared between two segmentations; Global consistency error (GCE), which measures the similarity between segmentations in terms of refinement; and Boundary displacement error, which measures the extent of boundary displacement. Large PRI or small Voi, GCE, BDE values are preferred. The top 3 results are highlighted for each measure. I use 3 variants of my algorithm: Diffusion modulus, DM,

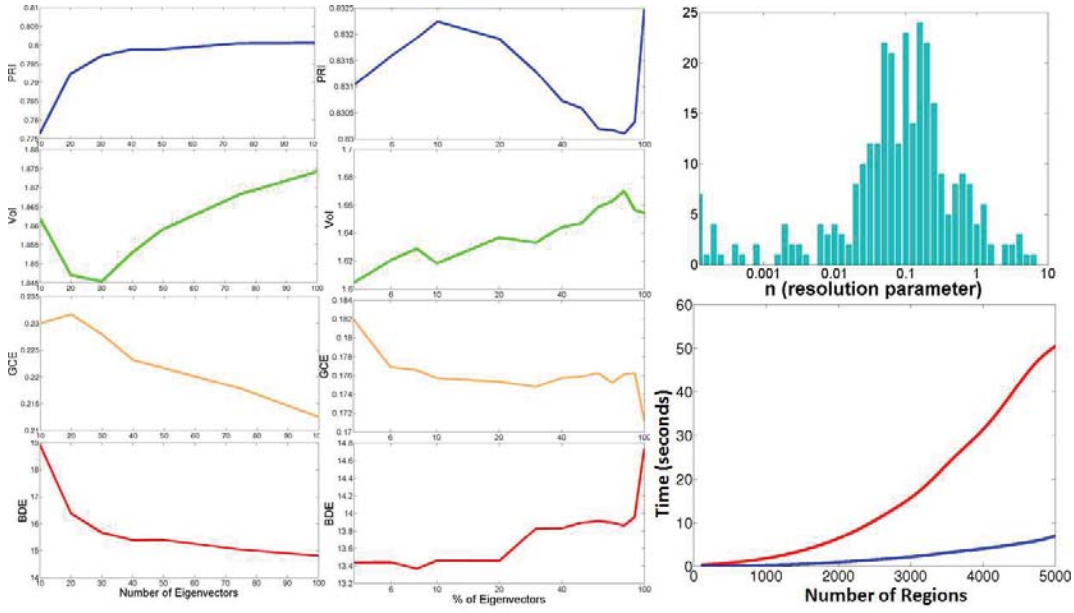


Figure 3.6. *Left: Dependence of PRI (blue), VoI (green), GCE (orange) and BDE (red) on number of eigenvectors for DM. Center: Dependence of 4 measures on percentage of eigenvectors for DM(ML). Right: Bar chart of optimal resolution parameter n on BSDS (Top) and running time for full spectral decomposition (red) and Louvain (blue) over number of regions (Bottom).*

Multi-scale diffusion modulus, DM(MS), and over weighted boundary pixels, DM(WB). DM, DM(MS) are obtained by setting the resolution n in Algorithm 1. I choose 100 eigenvectors for DM and 10% eigenvectors (10% of N) for DM(ML). The dependence on the number or $\%$ of eigenvectors is displayed in Figure 3.42. PRI value remains almost constant after 40 eigenvectors but 100 eigenvectors achieve better GCE and BDE values. Using 10% of the eigenvectors for DM(ML) achieves a tradeoff between the different measures. For DM(WB) I discretize n in the domain $[10^{-2}, 1]$ with logarithmic spacing and optimally threshold the weighted pixel values. The top right plot in 3.42 plots a bar chart of the optimal n values for the BSDS. I observe that $n = 0.1$ has the highest frequency and $n = 1$ or the commute time kernel leads to 'undersegmentation' for most images. Hence, the commute-time or biharmonic kernel are not suitable for image segmentation, and conse-

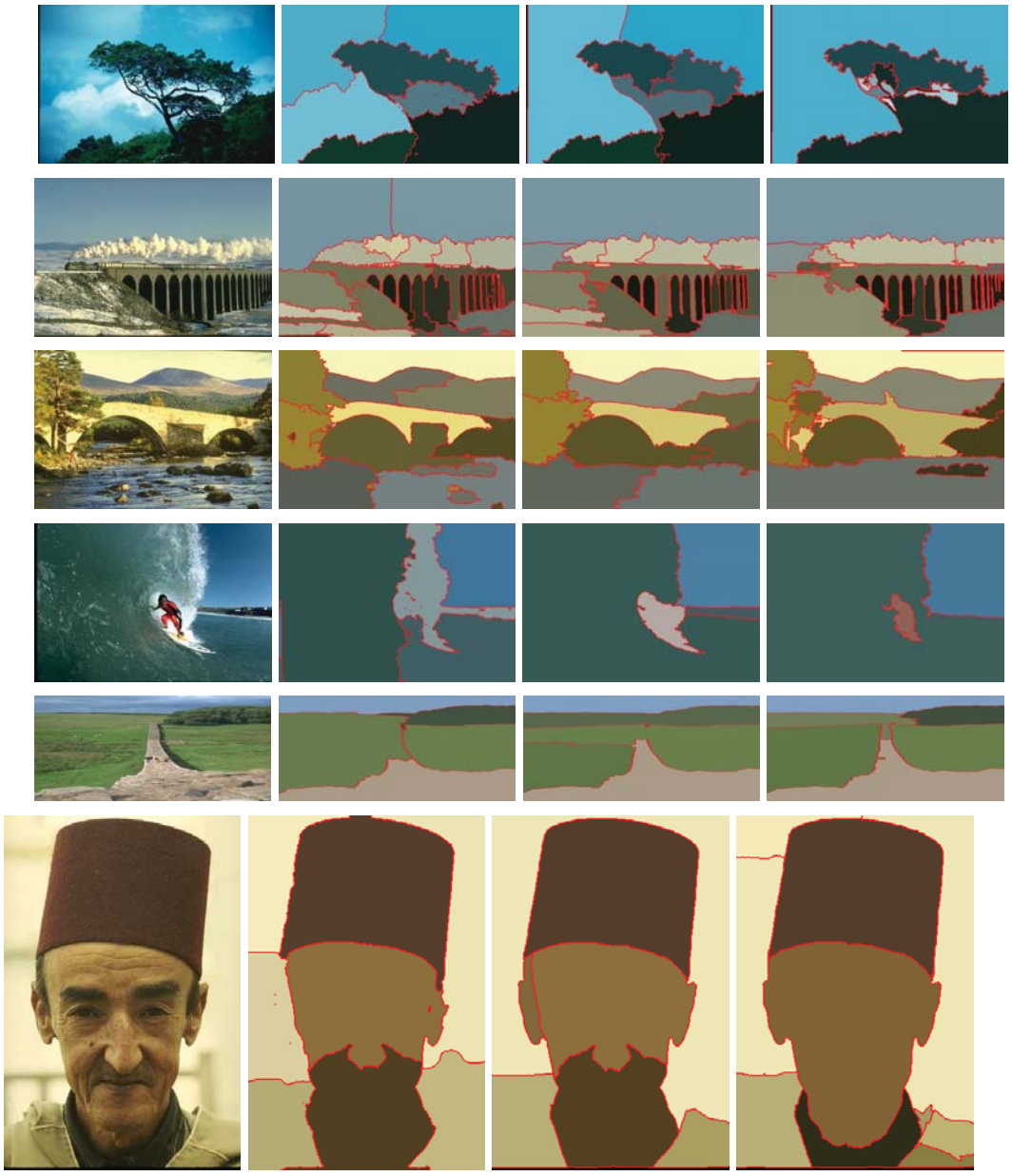


Figure 3.7. *Left to Right: Visual comparison of original image and best segmentations using MNCut, FAC and DM(MS) on BSDS.*

quently, are used conjunctly with other measures to impose partitions. The seeded image segmentation method in [110] is not effective for unsupervised image segmentation as it

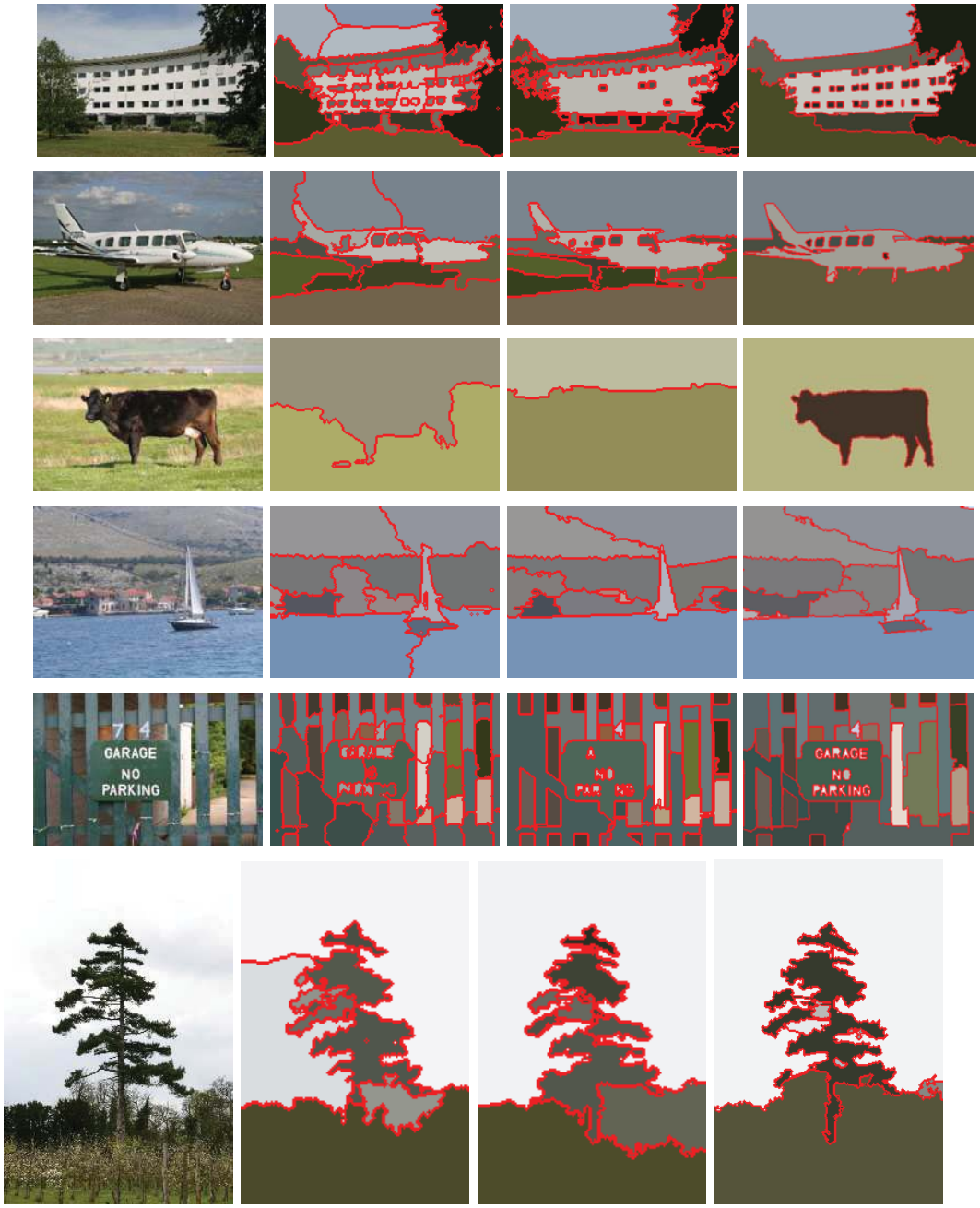


Figure 3.8. *Left to Right: Visual comparison of original image, MNCut, FAC and DM(MS) on MSRC for same number of final segments.*

considers the resistance matrix, tied to the commute kernel. My introduction of the critical exponent n in $\mathcal{D}^n$ resolves these problems.

The values in the table above are the average scores over the entire database. DM performs better than Ncut and MNCut on all but GCE measure. The best PRI and VOI values are that of UCM followed by DM(MS). DM(MS) uses only the spectral decomposition and does not benefit from parameter tuning on training images, unlike UCM. Combining DM(MS) with the mPb kernels will naturally result in better segments and is left for future work. The higher BDE values compared to SAS and FAC are reasonable because DM uses only one over-segmentation, whereas SAS and FAC use several over-segmentations to derive the final partition. Overall, it is ranks in top 3 for all measures. The superior results of DM(MS) relative to DM can be attributed to the additional information gained from the over-segmentation, and relative to DM(WB) can be attributed to the discretization error of n. However, DM(WB) achieves an overwhelming GCE value as it systematically incorporates segmentations over different resolutions to output salient weighted boundaries. Computing 100 eigenvectors for the sparse DM matrix takes ≈ 28 seconds and Louvain optimization takes < 1 second on a 3.40 GHz CPU and 8GB RAM desktop. Figure 3.42 shows the time for computing the full spectral decomposition and the subsequent optimization using Louvain for different sizes of oversegmentation. For ≈ 1000 segments, full spectral decomposition takes $\approx 1 second$, and Louvain takes $\approx 0.2 second$ using MATLAB. The total computation time, including meanshift and graph construction is < 4 seconds for given n. The computation time for weighted boundary pixel evaluation is on average ≈ 32 seconds for BSDS. Figures 3.7 and 3.8 visually compares the segmentations for a few images in BSDS and MSRC. Compared to MNCut and FAC, DM(MS) produces good-quality segments and respects object boundaries.

3.2 Gauss's Law for Finding Community Boundaries

I validate my approach using twenty networks of varying size and topology spanning diverse scientific domains. Of these, ten have known ground truth community assignments.

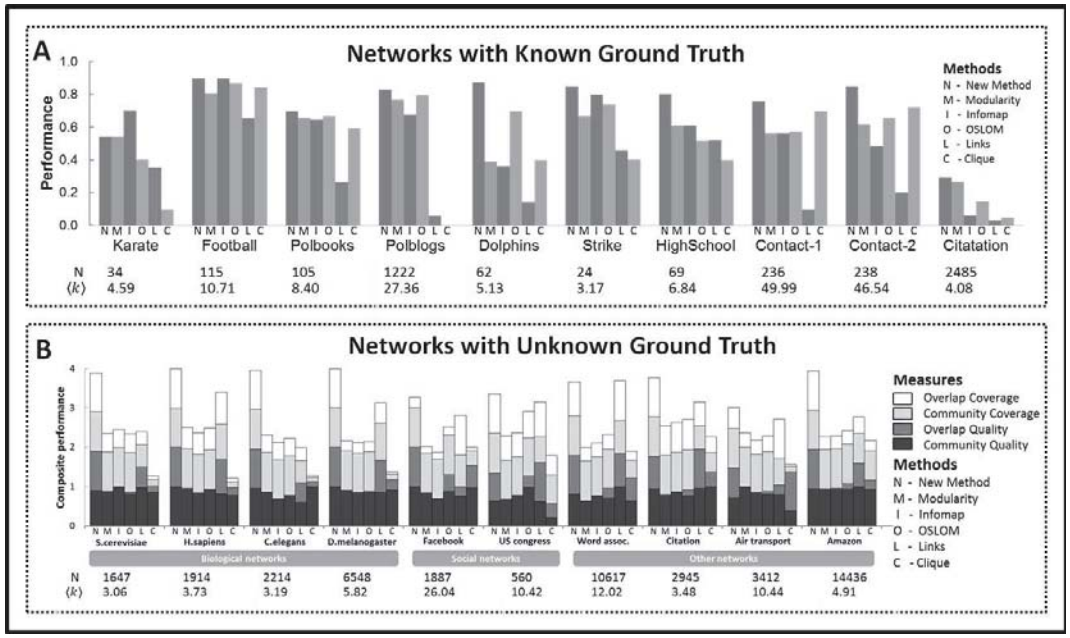


Figure 3.9. *Performance of community detection methods on real-world networks. A: Performance on ten networks with known community assignment measured with the omega index of identified communities against ground truth. B: Composite performance based on metadata on ten networks with unknown community assignment. The tables below list the number of nodes, N, and the average node degree, $\langle k \rangle$ for each network.*

On these networks, my Laplacian modularity optimization (see Methods) produces the most accurate communities (Figure 3.29 A) as measured in terms of the widely used omega index [190] for 9 of the 10 networks when compared against five prominent community detection methods: modularity optimization [49, 121], Infomap [48], OSLOM [136], link communities [41] and clique percolation [40]. For the networks with unknown community assignment, I follow the approach in [41] to derive a composite performance metric for each method from the metadata. The composite performance metric is a normalized sum of four measures: community quality, overlap quality, community coverage and overlap coverage. In Figure 3.29 B, I see that Laplacian modularity framework is the overall leader

for all networks, and the performance is especially striking for biological networks which are expected to be incomplete.

In addition, my stability curve offers a natural way to interpret relevant hierarchical communities in the network. English words have associations at many different levels and my methods identifies four levels (Figure 3.10 A). For example words associated with alcoholism and soft drinks appear as separate communities in the third level and the alcoholism subcommunity is semantically subdivided in the fourth level. The stability curve of the air transportation networks identifies partitions that correspond well to my natural conception of a community, i.e., the first level roughly identifies the large geopolitical regions of North America, South American, the Europe-Africa-Middle East, and the Asia-Pacific region, the second level identifies finer regions including Europe (alone) and Australia, and the third level subdivides continents into small country-sized regions (Figure 3.10 D) while maintaining spatial proximity of overlapping communities.

3.2.1 Results on network datasets with known community assignment

In this section I first describe the metric used to evaluate the quality of partitions, followed by brief description of each network along with the associated partition obtained by my method.

Omega Index: The omega index ω is an extension of the adjusted Rand index to overlapping communities [190]. It is defined as:

$$\omega(C_i, C_j) = \frac{\omega_u(C_i, C_j) - \omega_e(C_i, C_j)}{1 - \omega_e(C_i, C_j)}. \tag{3.1}$$

where C_i, C_j are the partitions to be compared, $\omega_u(C_i, C_j)$ is the fraction of node pairs occurring together in same number of communities in C_i and C_j, and $\omega_e(C_i, C_j)$ is the expected value of this fraction. Intuitively, $\omega_u(C_i, C_j)$ is the observed agreement between the partitions C_i and C_j, whereas $\omega_e(C_i, C_j)$ is the expected agreement of the partitions. The omega index is then observed agreement adjusted by expected agreement divided by the maximum possible agreement, i.e., 1 adjusted by expected agreement.

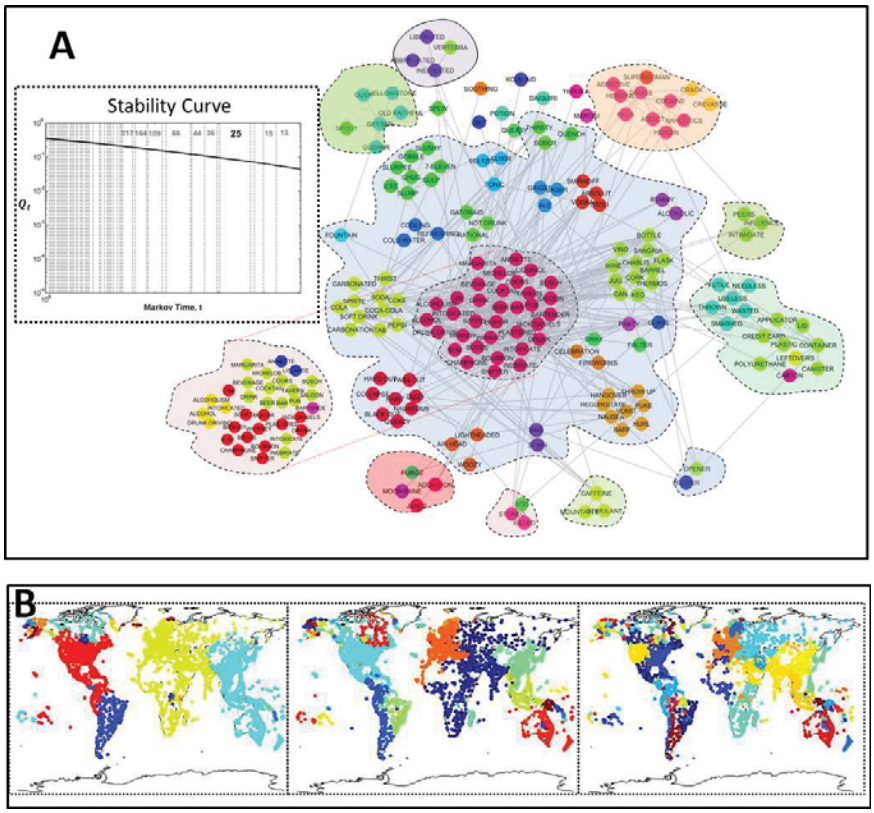


Figure 3.10. *Hierarchical communities using Laplacian modularity. A: Hierarchy in word association network. The stability curve (left) indicates that the network should be hierarchically organized into 25 and 66 communities. I also show the 164 and 217 communities information for deeper hierarchy. The subgraph shown is a community of the 25-way partition (first level). The dotted enclosed regions correspond to stable subcommunities in the 66-way partition (second level). The colors correspond to stable subcommunities in the 164-way partition (third level). Further subdivision of the red subcommunity in the 217-way partition is shown separately to the left (fourth level). B: Hierarchical structure of air transportation network, divided into 28, 76 and 133 communities from left to right as identified by the stability curve identifies geopolitically significant regions at various size-scales.*

Karate club: The karate club network has served as a proving ground for many community detection algorithms. The network was constructed by Wayne Zachary and consists of ties between 34 members of a karate club at a US university in the 1970s [191]. The club

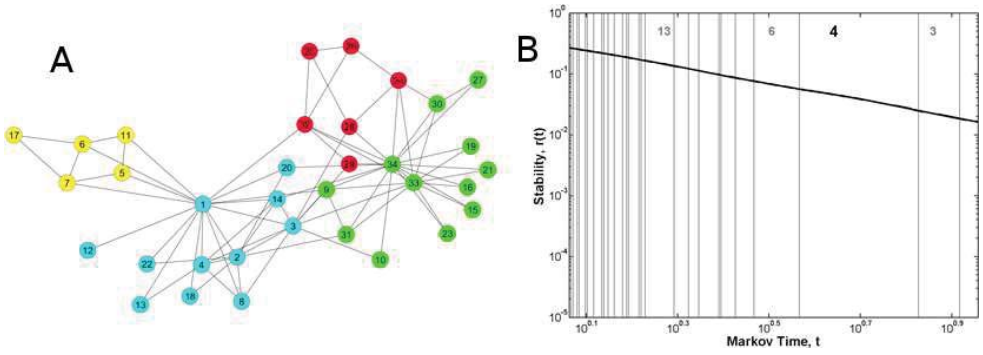


Figure 3.11. *Diffusion based Laplacian modularity applied to karate network. A: Identified communities in the karate network B: Stability curve for the karate network.*

split up into two groups following disputes among the members over the price of karate lessons. My community detection algorithm splits the network into 4 communities, the same as modularity optimization (Figure 3.11). It is interesting to note that the 2-way partition obtained in the long time limit is identical to the original community assignment by Zachary. Hence, it may be useful to analyze partitions beyond t_{end} in certain cases, for example if the number of partitions is known apriori.

Football: The football network compiled by M. Girvan and M. Newman contains the set of American football games played by Division IA colleges during regular season in Fall 2000 [37]. The ground truth community assignment is according to the conference each team belongs to. There are in total 12 conferences: Atlantic Coast, Big East,Big Ten, Big Twelve, Conference USA, Independents, Mid-American, Mountain West, Pacific Ten, Southeastern, Sun Belt and Western Athletic. My community detection algorithm correctly partitions the network into 12 groups. In Figure 3.12 I see that nine of the twelve conferences are correctly identified without any misclassifications. The 3 other conferences with partial misclassifications also reveal interesting features about the network. For example, the group of Independents comprising of 5 universities is not a conference, but institutions not affiliated with NCAA. I see that institutions in Independents all are classified in another conference, presumably with the conference it has the closest affiliation with. Louisiana

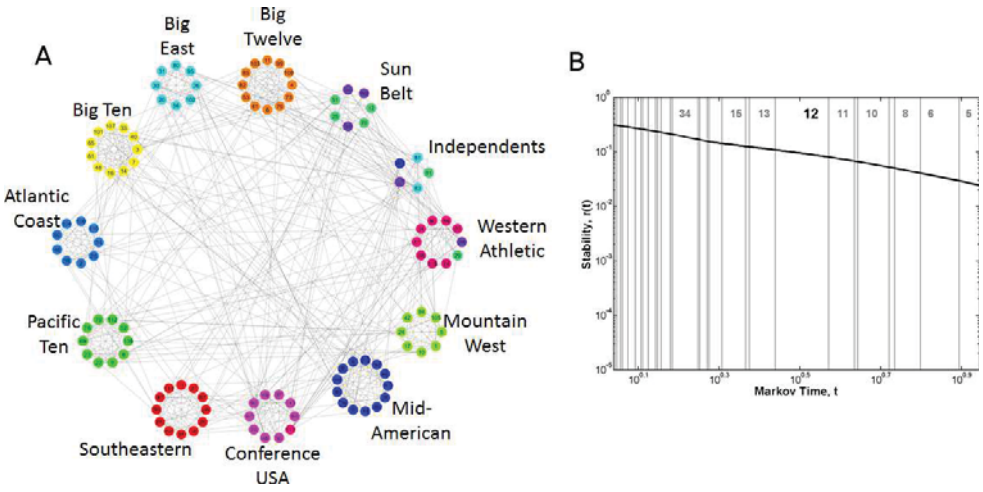


Figure 3.12. *Diffusion based Laplacian modularity applied to football network. A: Each circular layout of nodes corresponds to one ground truth community in the football network. The communities identified by my method are color coded B: Stability curve for the football network.*

Tech University shifted from the Sun Belt conference to Western Athletic conference in 2001. Consequently, node 59 corresponding to Louisiana Tech University in the western athletic conference is misclassified to be in the sun belt conference by my method.

Polbooks: The political books network compiled by Valdis Krebs represents co-purchased political books sold by Amazon.com around the time of the 2004 presidential election. The ground truth community assignment corresponds to the political philosophy advocated by the book as either conservative, neutral or liberal. In Figure 3.13 I see that my methods identifies two groups and classifies books in the ground truth neutral community to be either right-leaning or left-leaning based on the network topology. It is also interesting to note that the node 59 (Rise of the Vulcans) and node 78 (Bush at War) are written by journalists associated with the left-leaning Washington Post. The two nodes are misclassified to be liberal by my method, although the ground truth assignment is conservative. I also show the 3-way partition identified by the stability curve. The identified communities correspond well to conservative and liberal political books. I further investigate the assignment of

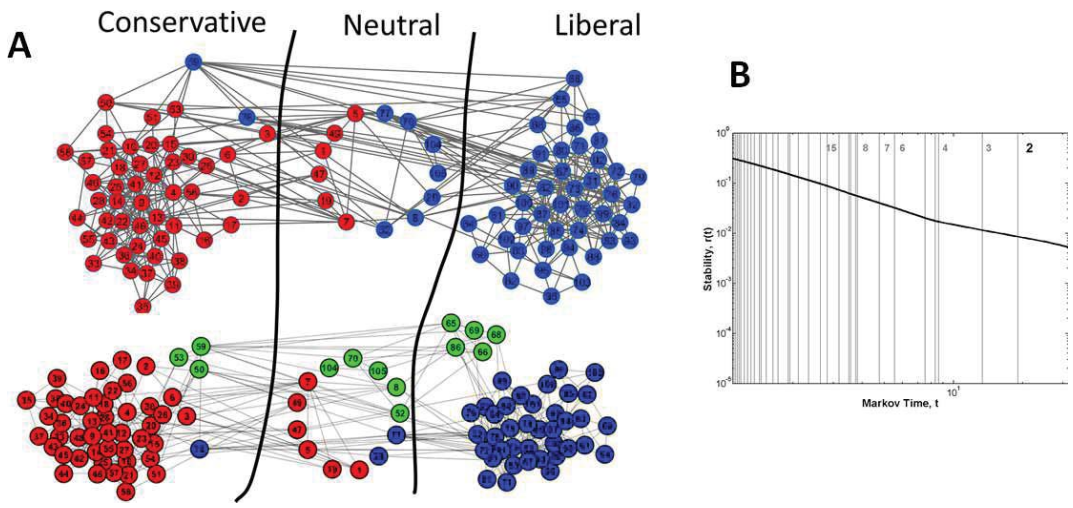


Figure 3.13. *Diffusion based Laplacian modularity applied to political books network. A:The three communities in the ground truth are separated by the two black lines. They correspond to conservative, neutral and liberal communities from left to right. The plot in top shows the two communities identified by my method. The nodes are colored red and blue. The plot in the bottom shows the 3-way partition identified by the stability curve. The nodes are colored red, green and blue. B: Stability curve for the polbooks network.*

neutral political books by my method. The liberal books classified as neutral (nodes 77 and 29) as well as the neutral political books classified as liberal (nodes 65,66,68,69,86) are all identified as overlapping nodes with membership to liberal and neutral communities at $\epsilon = 0$. This validates the relevance of the 3-way overlapping community structure identified by my method.

Polblogs: The political blogs network compiled by Lada Adamic and Natalie Glance represents the political blogosphere around the time of the 2004 presidential election [192]. Nodes are classified to be liberal or conservative. My method achieves the highest omega index among all community detection methods. The stability curve in shown in Figure 3.17.

Dolphin: The dolphin social network compiled by D. Lusseau represents frequent associations between 62 dolphins in a community living off Doubtful Sound, New Zealand [193].

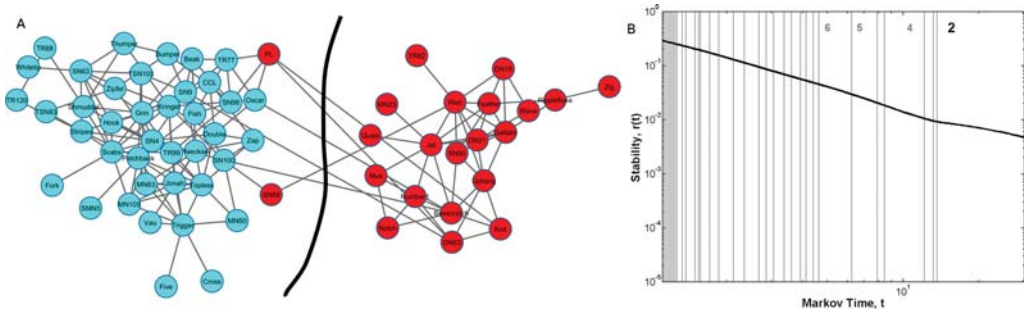


Figure 3.14. *Diffusion based Laplacian modularity applied to dolphin network. A: The two communities in the ground truth are separated by the black line. The two communities identified by my method are colored red and blue B: Stability curve for the dolphins network.*

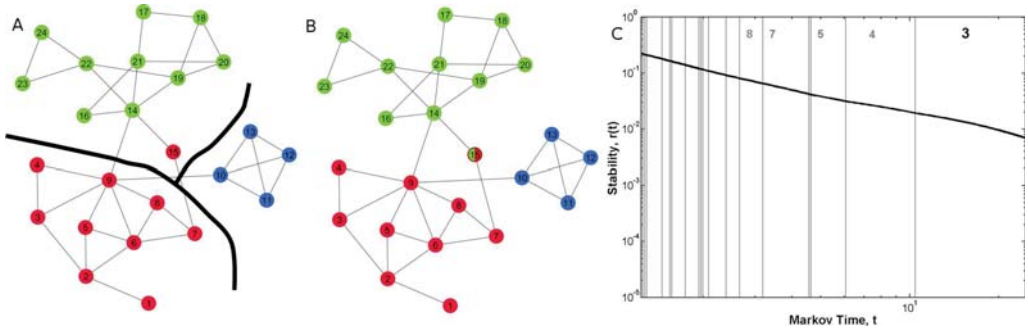


Figure 3.15. *Diffusion based Laplacian modularity applied to strike network. A: The black lines separate the network into its constituent ground truth communities. The communities identified by my method are color coded. B: The overlapping communities identified by my method at* $\epsilon = 0.5$ *B: Stability curve for the strike network.*

The ground truth community assignment into 2 groups is primarily based on the gender of the dolphins. My community detection method correctly identifies 2 clusters, correctly classifying all but 2 dolphins. The misclassified dolphins PL and SN89, structurally share more number of links with the identified group than with the other group, and hence, the identified communities correspond to a sound partition (3.14).

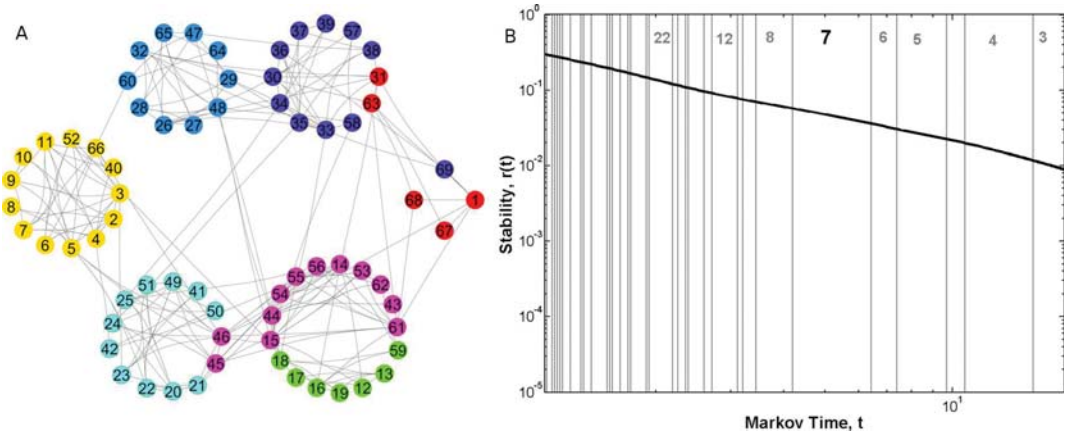


Figure 3.16. *Diffusion based Laplacian modularity applied to highschool network. A: Each circular layout of nodes corresponds to one ground truth community in the highschool network. The communities identified by my method are color coded B: Stability curve for the highschool network.*

Strike: The strike network represents the communication structure among employees in a wood processing facility during a strike [194]. The network displays fairly stringent demarcations between groups defined on age and language. The 3 ground truth communities correspond to English-speaking young employees, English-speaking old employees and finally Spanish-speaking young employees. My community detection method correctly identifies the communities and misclassifies one node (Figure 3.15). Node 15 (Ozzie) is the father of node 14 (Karl), which explains the misclassification. Interestingly, node 15 is identified as an overlapping node, indicating it shares membership with the group of old English-speaking employees and the father of a young English-speaking employee (node 14).

Highschool: The highschool network, compiled by the National Longitudinal Study of Adolescent Health, represents self-reporting of friendship between students from grades 7 to 12 [195]. The ground truth community assignment corresponds to the students' grade, with the expectation that students belonging to the same grade form stronger associations than those belonging to different grades. In Figure 3.16 I see that my community detection

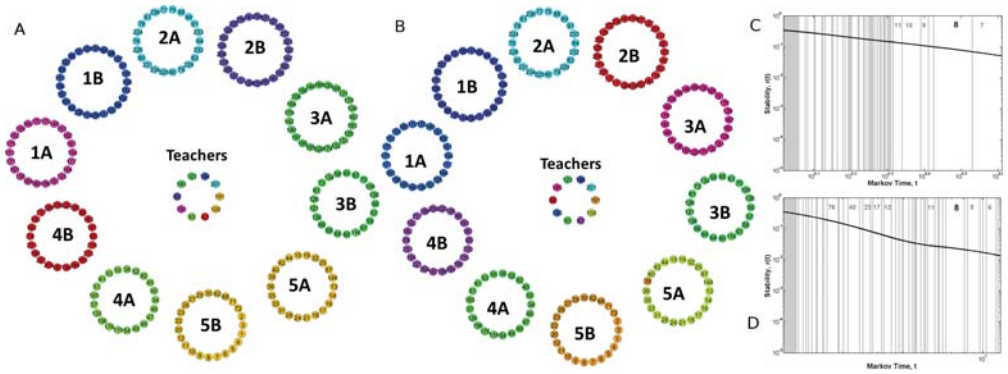


Figure 3.17. *Diffusion based Laplacian modularity applied to contact-1 network. A: Each circular layout of nodes corresponds to one ground truth community in the contact1 network. The communities identified by my method are color coded B: The communities identified by my method corresponding to the stable 10-way partition C: Stability curve for the contact1 network D: Stability curve for the polblogs network.*

method identifies 7 communities contrary to the 6 ground truth communities. Most nodes agree with the ground truth assignment in 5 of the 7 identified communities. However, the sixth ground truth community is split into 2 groups (colored pink and green) in my identified 7-way partition. Upon further investigating this community, it is revealed that it possesses hierarchical organization of friendship and the pink and green nodes correspond well with the group of white and black students, respectively. Hence, my community detection method is revealing of the community structure and the hierarchy present in the network.

Contact-1 and Contact-2: The two networks represent the face-to-face proximity between students and teachers in a primary school for 2 days [196]. Although, the original network is weighted, I analyze the unweighted version of the network. As in the highschool network, the ground truth communities correspond to the class and grade of the students along with a separate community for teachers. There are a total of 11 communities in the network, 10 communities of students and 1 of teachers. Figures 3.17 and 3.18 display the 8 and 9 communities identified by my automated method for contact-1 and contact-2 networks

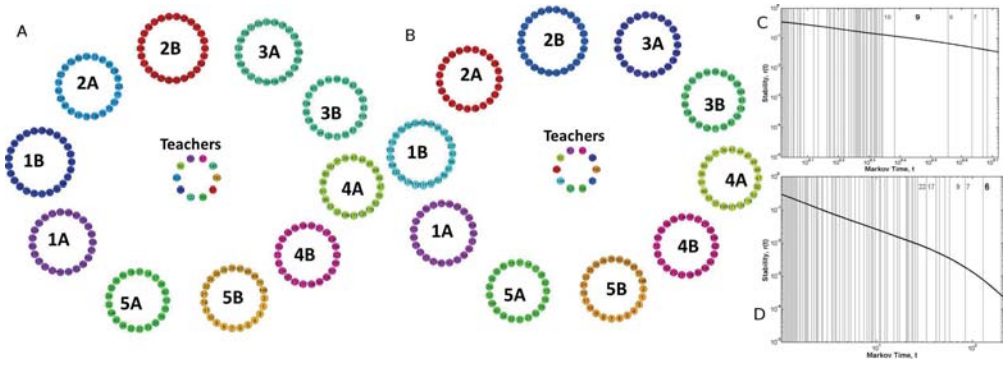


Figure 3.18. *Diffusion based Laplacian modularity applied to contact-2 network. A: Each circular layout of nodes corresponds to one ground truth community in the contact2 network. The communities identified by my method are color coded B: The communities identified by my method corresponding to the stable 10-way partition C: Stability curve for the contact2 network D: Stability curve for the citation network.*

respectively. Observe that the 3A, 3B and 5A,5B communities are merged in contact-1 network suggesting hierarchical organization of the students at the same grade level but different divisions. The 10-way partition emerges as one of the stable partitions in the stability curve and the identified communities are color coded in Figure 3.17 B. All but 1 of the 226 students are correctly classified and the community of teachers is split by the class it teaches. Similarly, the 3A, 3B communities are merged in contact-2 network indicating hierarchy. Again, the 10-way partition emerges as one of the stable partitions in the stability curve and the identified communities are color coded in Figure 3.18 B. All of the 228 students are correctly classified and the 10 teachers are assigned to the class it teaches.

Citation: The citation network comprises of computer science research papers categorized in 7 groups: neural networks, rule learning, reinforcement learning, theory, probabilistic methods, case based and genetic algorithms. The stability curve for this dataset is displayed in Figure 3.18 D.

3.2.2 Results on network datasets with unknown community assignment

In this section I first describe the four measures used to evaluate the quality of communities output by different community detection algorithms. Next, I briefly describe each of the 10 networks and the associated metadata used to evaluate the quality measures.

Measures: I follow the approach in [41] to derive a composite performance metric for the communities identified in the network. The composite performance metric consists of 4 measures: 2 quality measures that assess the relevance of the discovered memberships and 2 coverage measures that quantify the quantity of information extracted. These measures are mathematically formalized as follows:

Community Quality: Community quality is evaluated using the network metadata under the assumption that nodes belonging to the same community are similar and hence, share more metadata than nodes belonging to different communities. This is quantified as:

$$Enrichment = \frac{\langle \mu(i,j) \rangle_{\delta(i,j)}}{\langle \mu(i,j) \rangle_{\varrho(i,j)}}. \tag{3.2}$$

where $\mu(i,j)$ is metadata-based similarity between nodes i and j, $\langle \mu(i,j) \rangle$ is the mean similarity and the subscripts $\delta(i,j)$, $\varrho(i,j)$ indicate the average calculated over all node pairs belonging to the same community or different community respectively. Unlike [41] where the enrichment is calculated over the baseline similarity between all node pairs, I calculate the ratio of the average intra-community and inter-community similarity, which I discern to more discriminative. High intra-community similarity and low inter-community similarity lead to favourable values of enrichment.

Overlap Quality: To assess the quality of overlap output by a method, I calculate the mutual information between the number of memberships identified by the method and the overlap metadata of the network. Unlike [41], homeless nodes are assigned to an independent community, and hence, have one membership (to itself). This avoids artificially increasing the mutual information due to homeless nodes which do not convey any overlap information, and consequently, methods identifying non-overlapping communities always have 0 overlap quality. Note that in the absence of any overlap metadata for a node, I take the number of associated memberships to be trivially 1. In 2 of the 10 networks, I use the

average split-betweenness of overlapping nodes as a proxy for community quality instead of mutual information.

Community Coverage: Community coverage is the fraction of nodes which belong to non-trivial communities, i.e., of size 3 or higher. Community coverage hints at the quantity of useful membership information output by the community detection method.

Overlap Coverage: Overlap coverage measures the average membership of nodes belonging to non-trivial communities. Note that the community coverage and overlap coverage are identical for non-overlapping community methods.

The composite performance metric is derived by first normalizing all four measures by the corresponding best performing method, and next summing the four measures. A method which outputs the best community in terms of all 4 measures has composite performance value 4. The 4 measures are visually represented as a stacked bar chart, as in [41].

Biological networks: I analyzed the protein-protein interaction (PPI) network of Saccharomyces cerevisiae, Homo sapiens, Caenorhabditis elegans and Drosophila melanogaster. I use the Gene Ontology (GO) terms as metadata to assess the quality of communities. The quality measures are first evaluated using GO annotations for biological process, molecular function and cellular components. The overall quality is simply taken as the average of these three quality measures.

Construction The PPI network of Saccharomyces cerevisiae is constructed using the yeast two-hybrid (Y2H) dataset published in [197]. The PPI network of Homo sapiens is constructed using the high quality Y2H interactions determined in [198]. The PPI network of Caenorhabditis elegans is constructed from the dataset released in [199]. Finally, the PPI network of Drosophila melanogaster is taken from [200]. I use the largest component of each network.

Community quality Three separate directed acyclic graphs are constructed using the controlled vocabulary for biological process, molecular function and cellular components. The functional similarity between all pairs of proteins is then calculated from the directed acyclic graph using Lin's information theoretic definition [201]. I use the implementation

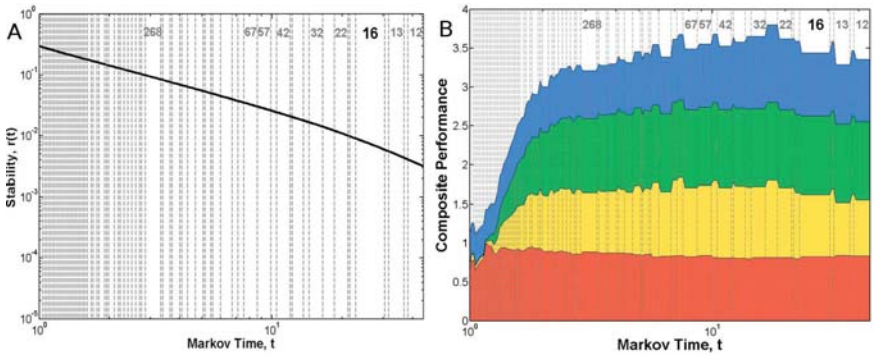


Figure 3.19. *A: Stability curve for Saccharomyces cerevisiae network B: Variation of composite performance as a function of the stable communities identified by the stability curve over time shown as an area chart. The red corresponds to community quality, yellow to overlap quality, green to community coverage and blue to overlap coverage. The number of communities in the partitions with high persistence is shown above the composite performance index curve. The number of communities in the partition used for evaluation in the main manuscript is highlighted in bold. Communities with higher composite performance index than the one used for evaluation are identified.*

provided with [202] . The corresponding enrichment values are calculated using Equation (3.2) and the overall quality is taken as the average of these enrichment values.

Overlap quality Overlap quality is calculated for biological process, molecular function and cellular components to be the mutual information between the number of associated GO terms and number of memberships. The net overlap quality is taken to be the average of the three individual overlap qualities.

Facebook network: I analyzed the Facebook network of students in an American university graduating in the same year. Communities were evaluated using biographic information such as college major/minor, highschool they attended and dorms they reside in.

Construction The full Facebook network of Michigan university published in [203] consists of friendship links between students and faculty. Only the students stating biographic information in their profile are considered. I would expect students graduating in the same year to form communities based on major/minor or residency. Hence, the friendship circles

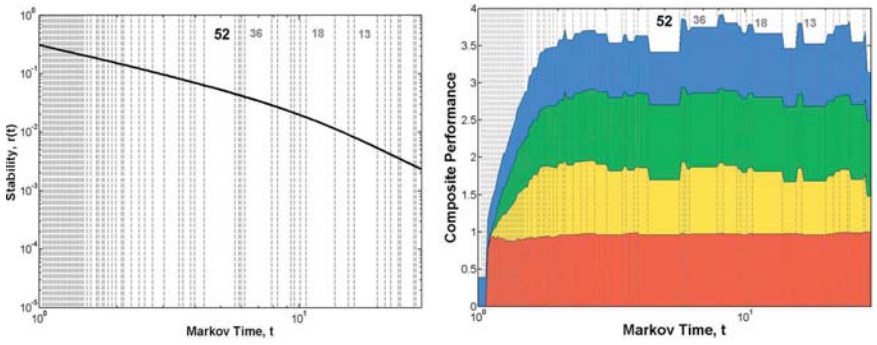


Figure 3.20. *A: Stability curve for Homo sapiens network B: Variation of composite performance as a function of the stable communities identified by the stability curve over time shown as an area chart. The red corresponds to community quality, yellow to overlap quality, green to community coverage and blue to overlap coverage. The number of communities in the partitions with high persistence is shown above the composite performance index curve. The number of communities in the partition used for evaluation in the main manuscript is highlighted in bold. Communities with higher composite performance index than the one used for evaluation are identified.*

of the students graduating in the year 2008 are used in constructing the network. I use the largest component of this network.

Community quality The metadata contains anonymized information regarding the students' major/minor, the dorms they reside in and their highschool. I build a bipartite network linking each student to its corresponding major/minor, dorm and highschool. The similarity between students is inferred from the one-mode projection onto the students of this two-mode bipartite network. The maximum similarity between two students is 3 if they share the same major/minor, dorm and highschool. The one-mode projected matrix is used to calculate the enrichment as per Equation (3.2).

Overlap quality The community quality partially reflects the overlap quality as a student participates in multiple (possibly overlapping) communities corresponding to their major/minor, dorm and highschool. However, additional factors like a student's participation in hobby-clubs, college events or sports teams may contribute to forming friendship, and hence, communities that overlap. I use the concept of split betweenness to evaluate

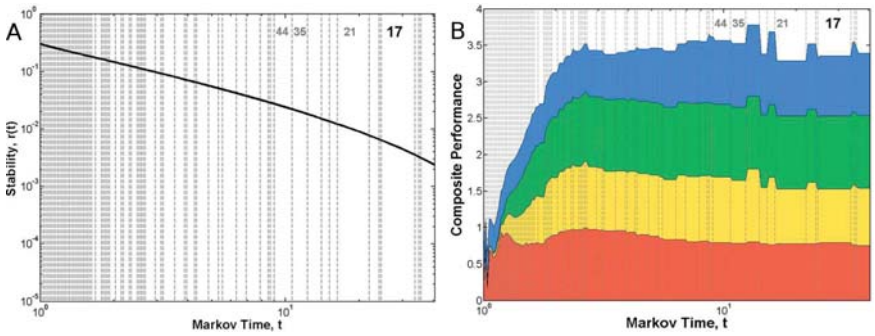


Figure 3.21. *A: Stability curve for Caenorhabditis elegans network B: Variation of composite performance as a function of the stable communities identified by the stability curve over time shown as an area chart. The red corresponds to community quality, yellow to overlap quality, green to community coverage and blue to overlap coverage. The number of communities in the partitions with high persistence is shown above the composite performance index curve. The number of communities in the partition used for evaluation in the main manuscript is highlighted in bold. Communities with higher composite performance index than the one used for evaluation are identified.*

the quality of overlapping nodes due to extraneous information not available in the metadata. Unlike [204] where the split betweenness metric is used to determine overlap, I use it to evaluate the quality of overlapping nodes. The network with overlapping nodes as determined by the community finding algorithm is first transformed into a similar network where all nodes and their linkages remain the same, but the nodes classified as overlapping. An overlapping node with membership to o communities is replicated o times in the transformed network such that each replicated node is a member to only one community, i.e., I convert the network with overlapping communities into one with no overlap by replicating overlapping nodes. The overlap quality is calculated to be the average edge betweenness of new edges introduced in the transformed network with higher values favored, i.e., a high value of average split betweenness hints at better overlap quality (see Figure 3.23).

Cosponsorship network: The cosponsorship network consists of legislative collaborations between the United States House of Representatives for the 93^{rd} to 108^{th} Congresses [205].

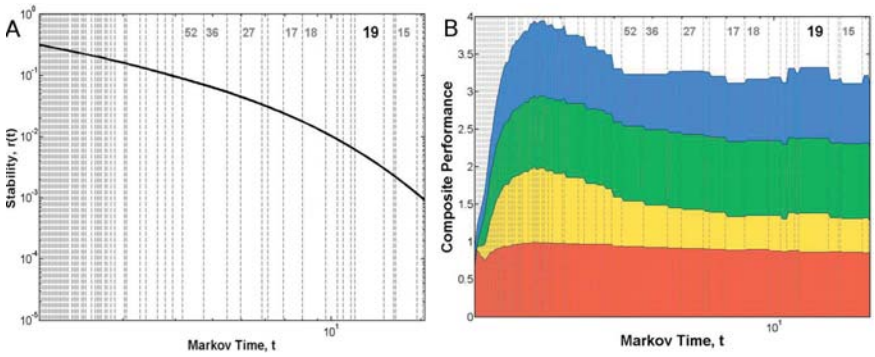


Figure 3.22. *A: Stability curve for Drosophila melanogaster network B: Variation of composite performance as a function of the stable communities identified by the stability curve over time shown as an area chart. The red corresponds to community quality, yellow to overlap quality, green to community coverage and blue to overlap coverage. The number of communities in the partitions with high persistence is shown above the composite performance index curve. The number of communities in the partition used for evaluation in the main manuscript is highlighted in bold. Communities with higher composite performance index than the one used for evaluation are identified.*

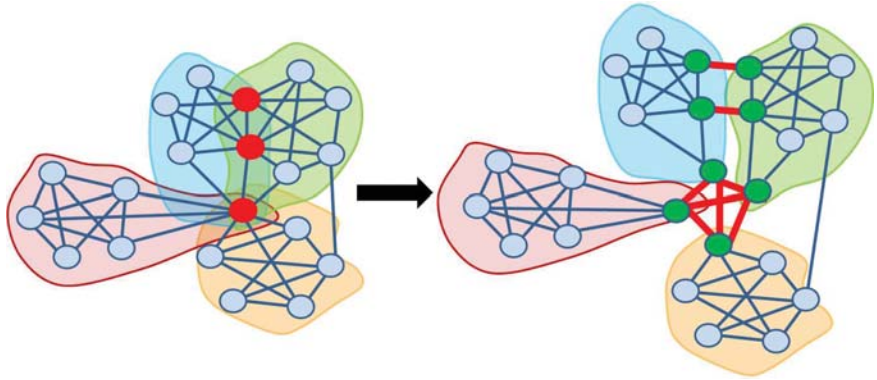


Figure 3.23. *Left network shows the communities identified by an algorithm with overlapping nodes colored red. Right network displays the transformed network used for overlap quality determination. The overlapping nodes are replicated such that the replicated nodes belong to only one community, shown in green. The overlap quality is the average edge betweenness of the edges colored red.*

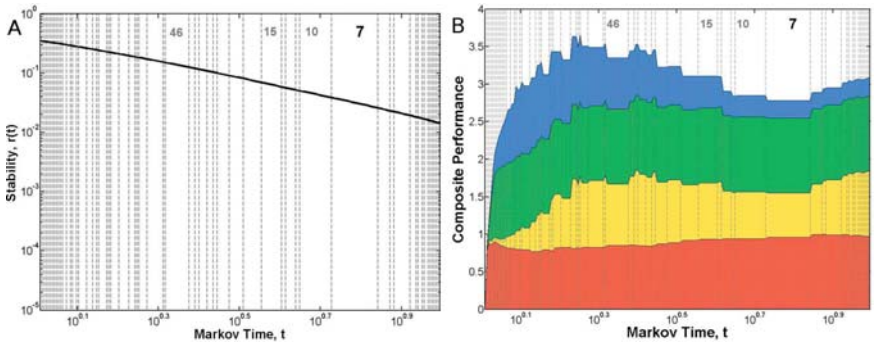


Figure 3.24. *A: Stability curve for the facebook network B: Variation of composite performance as a function of the stable communities identified by the stability curve over time shown as an area chart. The red corresponds to community quality, yellow to overlap quality, green to community coverage and blue to overlap coverage. The number of communities in the partitions with high persistence is shown above the composite performance index curve. The number of communities in the partition used for evaluation in the main manuscript is highlighted in bold. Communities with higher composite performance index than the one used for evaluation are identified.*

Construction As in [41], I construct a bipartite network of house members and the bills they (co)-sponsored. Projecting this network onto the representatives results in a very dense, nearly complete graph for which methods like clique percolation with exponential running time fail to provide any meaningful communities. To avoid this, I only consider bills that contain 10 or less (co)-sponsors in building the bipartite network, and subsequently project it onto the representatives. In order to capture the tightest working relationships I delete links with weight less that 15, i.e., the final network has an edge between two house members if they (co)-sponsor 15 or more bills that contain 10 or less (co)-sponsors total. The giant connected component of this network consists of 560 nodes.

Community quality I follow the approach in [200] to evaluate community quality using the common space score of house representatives. The common space score consists of two values between -1 and 1, wherein the first dimension represents liberal/conservative bias and the second dimension is related to civil-rights and social issues. The similarity $\mu(i, j)$ between two house members is taken to be the Euclidean distance between the common

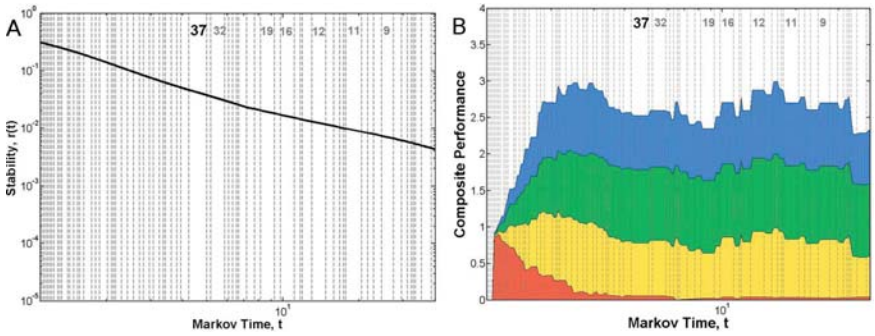


Figure 3.25. *A: Stability curve for the cosponsorship network B: Variation of composite performance as a function of the stable communities identified by the stability curve over time shown as an area chart. The red corresponds to community quality, yellow to overlap quality, green to community coverage and blue to overlap coverage. The number of communities in the partitions with high persistence is shown above the composite performance index curve. The number of communities in the partition used for evaluation in the main manuscript is highlighted in bold. Communities with higher composite performance index than the one used for evaluation are identified.*

space scores with smaller values denoting higher similarity. Hence, the overall enrichment is taken to be:

$$Enrichment = 1 - \frac{\langle \mu(i,j) \rangle_{\delta(i,j)}}{\langle \mu(i,j) \rangle_{\varrho(i,j)}} \tag{3.3}$$

Overlap quality The overlap quality is determined to be the mutual information between number of community memberships and the number of elected terms of representatives with the expectation that, representatives serving longer terms have (co)-sponsored more bills, and hence, participate in many collaborations.

Word association: The word association dataset is compiled by University of South Florida and University of Kansas [206]. It consists of three-quarters of a million free association responses to 5019 stimulus words from more than 6000 participants.

Construction Although there are 5019 stimulus words, the set of total words including responses has cardinality 10617. The edges represent discrete association of two words, i.e., the first meaningfully related word that came to a participant's mind in response to the

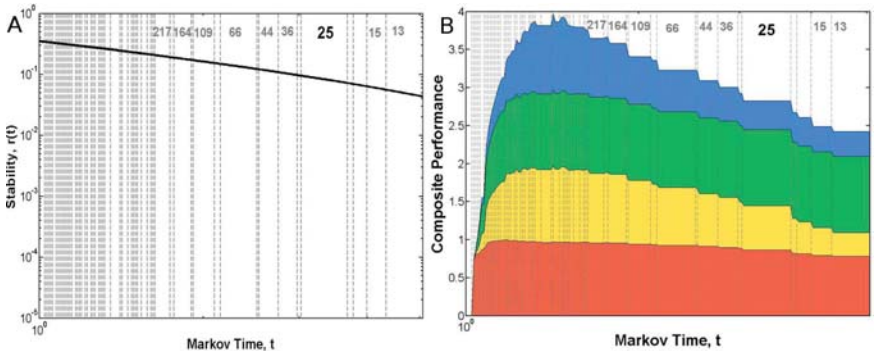


Figure 3.26. *A: Stability curve for the word association network B: Variation of composite performance as a function of the stable communities identified by the stability curve over time shown as an area chart. The red corresponds to community quality, yellow to overlap quality, green to community coverage and blue to overlap coverage. The number of communities in the partitions with high persistence is shown above the composite performance index curve. The number of communities in the partition used for evaluation in the main manuscript is highlighted in bold. Communities with higher composite performance index than the one used for evaluation are identified.*

presented word. I ignore the directionality and weights of the edges in the network, and hence, the final network for evaluation is undirected and unweighted.

Community quality I use the WordNet database to extract all synsets (or senses) associated with a word in the network. The Leacock-Chodorow similarity score [207] between two words is calculated for all combination of associated synsets, based on the shortest path distance that connects the synsets and the maximum depth of the taxonomy in which the senses occur. The similarity between two words is then taken to be equal to the maximum Leacock-Chodorow score from these combinations and further, Equation (3.2) is used to calculate the overall enrichment.

Overlap quality The more synsets a word is associated with, greater the likelihood of it participating in multiple communities. Hence, the overlap quality is determined to be the mutual information between the number of synsets and the number of community memberships for all words in the network.

PRL: The network is created using the citation information of papers published in the Physical Review Letters (PRL).

Construction The network is created using a subset of all papers published in the PRL. I consider all papers published in Volume-96 to Volume-100 to be nodes and place an undirected edge between two nodes, if any one cites the other. This network is disconnected and I take the largest connected component for my evaluation.

Community quality Each paper is associated with a general category such as, 'Atomic, Molecular, and Optical Physics', 'Gravitation and Astrophysics', etc. I define two papers to be similar if they share a paper category, i.e., $\mu(i, j) = 1$ if i and j share the same category, 0 otherwise. Equation 3.2 is used to calculate the community quality.

Overlap quality I assume that a paper with many authors is more likely to have diverse content, and hence, cited in multiple disciplines as compared to a paper with a single author. I calculate the overlap quality to be the mutual information between the number of non-trivial community memberships and number of authors of each paper in the network.

Air Transport: The air transport network represents flights connecting airports all over the world.

Construction The network was created using the flights connectivity information between different airports available at www.openflights.org. The nodes represent airports and an edge between two nodes represents a flight connecting these two airports. I take the largest connected component for network evaluation consisting of 3412 airports.

Community quality The international organization for standardization(ISO) has codes for different regions in the world. I associate each airport to three features, its ISO-region, the country it belongs to and the continent it resides in. The similarity between two airports is taken to equal to the number of features it shares, i.e., has maximum similarity value 3 if it shares the region, country and continent; 2 if it shares country and continent; 1 if the airports are in the same continent; and 0 otherwise. Equation 3.2 is used to calculate the enrichment using the similarity values.

Overlap quality An airport is expected to overlap with different communities if there are many connecting flights emerging from it. This in turn depends on the betweenness of the

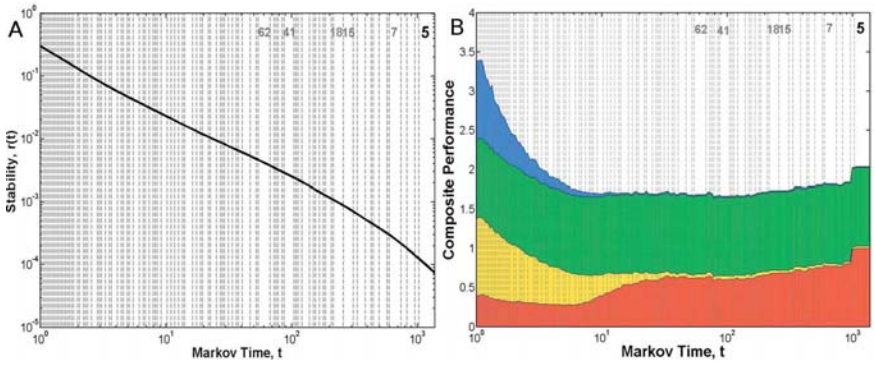


Figure 3.27. *A: Stability curve for the citation network B: Variation of composite performance as a function of the stable communities identified by the stability curve over time shown as an area chart. The red corresponds to community quality, yellow to overlap quality, green to community coverage and blue to overlap coverage. The number of communities in the partitions with high persistence is shown above the composite performance index curve. The number of communities in the partition used for evaluation in the main manuscript is highlighted in bold. Communities with higher composite performance index than the one used for evaluation are identified.*

airport, i.e., how many shortest paths (connecting flights) pass through that node (airport). Hence, I use the average split betweenness of overlapping airports as a measure of overlap quality.

Co-purchasing: This network represents the co-purchasing of DVD titles at the site Amazon.com, i.e., DVD's that are purchased simultaneously by the same customer.

Construction The network is taken from www.snap.stanford.edu which contains metadata information related to wide range of products like books, music CD's etc. I consider the information relevant to co-purchasing of DVD's for my network construction. Associated with each DVD, is a set of 5 or less similar co-purchased titles. I build a network using this information, linking two nodes if they are co-purchased. The final strongly connected network consists of 14436 DVD titles.

Community quality Each DVD is associated with a set of related categories. For eg., the DVD 'NASA' is associated with space exploration, documentary, DVD titles starting with

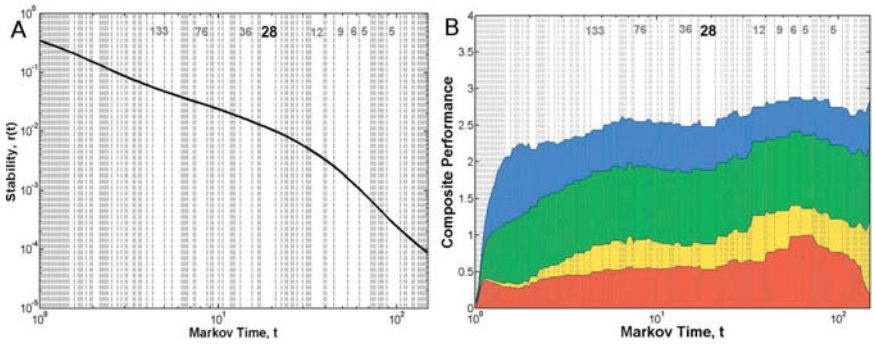


Figure 3.28. *A: Stability curve for the air transportation network B: Variation of composite performance as a function of the stable communities identified by the stability curve over time shown as an area chart. The red corresponds to community quality, yellow to overlap quality, green to community coverage and blue to overlap coverage. The number of communities in the partitions with high persistence is shown above the composite performance index curve. The number of communities in the partition used for evaluation in the main manuscript is highlighted in bold. Communities with higher composite performance index than the one used for evaluation are identified.*

'N' etc. The similarity between two nodes is evaluated to be equal to the number of categories it shares. The enrichment is then calculated using Equation (3.2)

Overlap quality I calculate the mutual information between the number of product categories and the number of non-trivial community memberships for the products.

Overlap Landscape: For each of the 10 networks, I investigate the overlapping landscape over time, i.e., I evaluate the number of nodes against strength of overlap over time. Figure 3.30 shows the overlapping landscape for each of the 10 networks and reveals characteristic patterns in the overlapping community landscape. I fit a curve to the histogram of number of overlapping nodes against the overlap strength which lies between 0 and 1. I consider at 5 values of time as multiples of the relaxation time $1/\lambda_2$: $0.1log(1/\lambda_2), 0.3log(1/\lambda_2), 0.5log(1/\lambda_2), 0.7log(1/\lambda_2), 0.9log(1/\lambda_2))$. A common pattern is that the peaks of the curve gradually shift over time to the left, i.e., at small time most nodes overlap with high strength whereas at longer times, the overlap strength is nor-

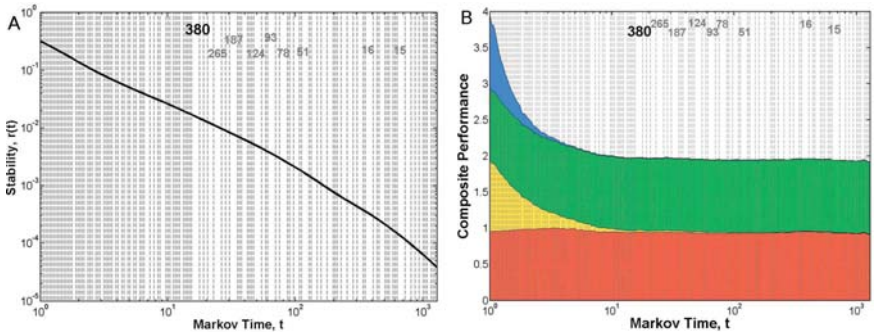


Figure 3.29. *A: Stability curve for the copurchasing network B: Variation of composite performance as a function of the stable communities identified by the stability curve over time shown as an area chart. The red corresponds to community quality, yellow to overlap quality, green to community coverage and blue to overlap coverage. The number of communities in the partitions with high persistence is shown above the composite performance index curve. The number of communities in the partition used for evaluation in the main manuscript is highlighted in bold. Communities with higher composite performance index than the one used for evaluation are identified.*

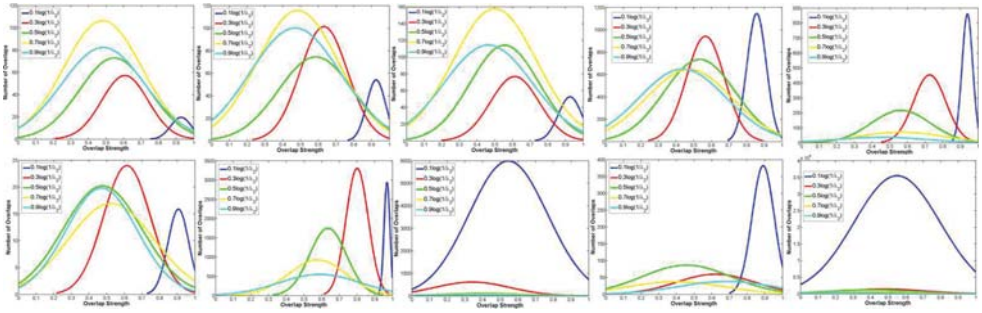


Figure 3.30. *Overlapping landscape strength over time. Top(Left to Right): S.Cerevisiae, H.sapiens, C.elegans, D.Melanogaster, Facebook networks. Bottom(Left to Right): US congress, Word assoc., Citation, Air transport, Amazon networks.*

malized towards the average of 0.5. Interestingly, the protein interaction networks have more overlaps at long time compared to small time, in contrast to other networks.

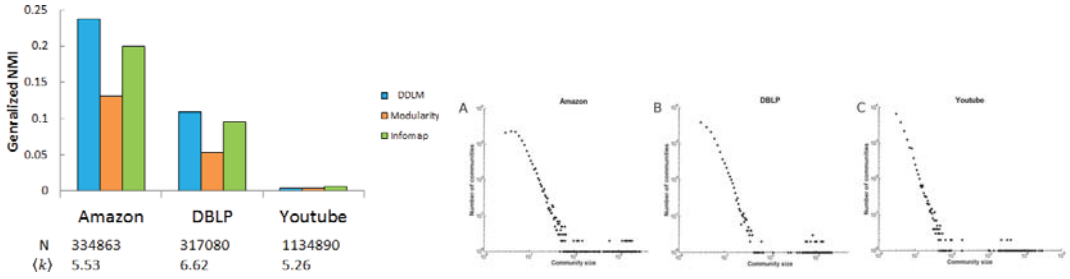


Figure 3.31. *Diffusion based Laplacian modularity applied to large networks. Left: Performance on networks measured as generalized normalized mutual information of identified communities against ground truth for the 3 large networks. Tested algorithms are discrete diffusion based Laplacian modularity (DDLM) optimization corresponding to $t = 1$ for discrete random walks; greedy modularity optimization and Infomap. Table below lists the number of nodes, N, and the average node degree, $\langle k \rangle$ for each network. Diffusion based Laplacian modularity optimization finds higher quality communities relative to other methods. Right: Community size distribution follows a scale-free law in the A: Amazon, B: DBLP, and C: Youtube networks.*

3.2.3 Results on large networks and for other similarity measures

As a concluding remark, I demonstrate the Laplacian modularity framework for some large networks and alternate definitions governing the connectivity potential, S.

Large networks: I investigate the Laplacian modularity on three large networks taken from www.snap.stanford.edu which have corresponding ground truth community information. I use transition matrix M as a measure of similarity instead of the heat kernel, for computational efficiency. This corresponds to the discrete time random walk for time, $t = 1$. I use the generalized NMI metric to evaluate community quality against the ground truth [46]. The methods tested are greedy modularity optimization, Infomap, both scalable to large networks, and finally DLM optimization at $t = 1$ for discrete random walks. The computational complexity of probability evaluation is $\mathcal{O}(cE)$, and hence, scalable to large networks like the Youtube network which contains more than a million nodes. I see in Figure 3.31 that DLM optimization is the best performing method for the Amazon and DBLP networks, while all three methods perform poorly on the Youtube network. I examine the statistics of

community size distribution in these three networks in Figure 3.31B. The community size distribution is heavy tailed for all networks, confirming the relevance of the communities detected.

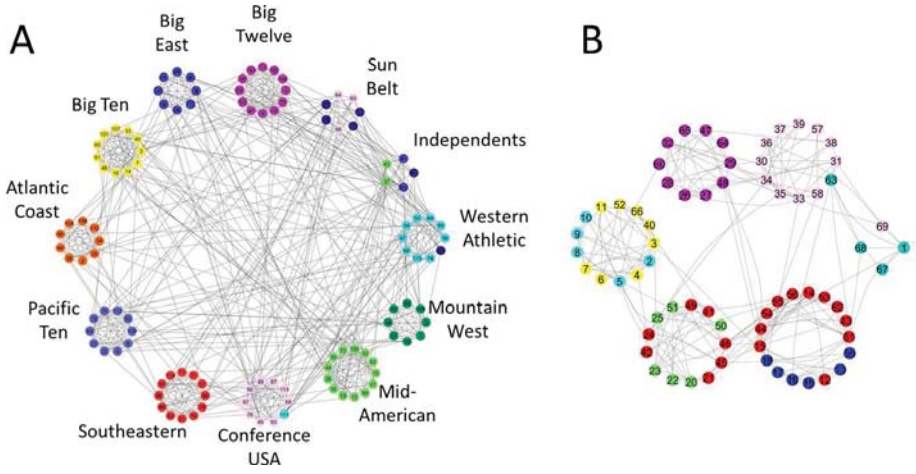


Figure 3.32. *Laplacian modularity for alternate definitions of similarity. A: Each circular layout of nodes corresponds to one ground truth community in the football network. The communities identified by setting the similarity between two nodes to be equal to the number of common neighbors in the Laplacian modularity quality function are color coded B: Each circular layout of nodes corresponds to one ground truth community in the highschool network. The communities identified by setting the similarity between two nodes to be equal to Newman-Leicht similarity in the Laplacian modularity quality function are color coded.*

Alternate definitions of connectivity potentials: I evaluate the community structure for the football network by setting the connectivity potential between two nodes to be equal to the number of common neighbors in the Laplacian modularity quality function. The similarity values are efficiently evaluated by the matrix A^2, i.e., the similarity between two nodes is $S(i,j) = A^2(i,j)$ for $i \neq j$ and I set $S(i,i) = 0 \forall i$. Figure 3.32 A shows the community structure for this similarity metric. The corresponding community quality is slightly inferior than the DLM and Infomap methods, and better than the other four community detection methods. I also evaluate the community structure of the highschool network by setting the connectivity potential between two nodes to be equal to the Newman-Leicht

similarity for $\alpha = 0.85$ [208]. Figure 3.32 B shows the community structure for this similarity metric. The quality of the corresponding communities in terms of the omega index is only slightly inferior than the communities identified by DLM, and better than the remaining five community detection methods.

Multi-resolution Laplacian: In order to validate the generality of the DLM framework to similarity measures beyond the heat kernel, I apply the multi-resolution Laplacian modularity using two alternate measures of similarity. In the contact-1 network, I use the adjacency matrix with self-loops, $S = A + I$ where I is the identity matrix and use the cosine similarity measure between nodes in the contact-2 network. The stability curves corresponding to these similarity measures is shown in Figure 3.33. The curves are constructed by varying the Laplacian resolution parameter ω in $L = D - \omega A$, i.e., the multi-resolution extension of Laplacian modularity as discussed in Section 2.2.1. ω was continuously varied from 0 till I obtain a partition wherein all nodes are assigned independent communities. The range of ω for finding the most relevant partition was empirically set to be in the range $[[\omega_{int}, \omega_{fin}]$ where ω_{int} is the value of ω for which I obtain a 2-way partition and ω_{fin} is the value of ω for which I obtain a $N/3$-way partition. N is the number of nodes in the network, and I use $N/3$ because a community is composed of atleast 3 nodes. The relevance of a k=way partition was set equal to the length of time window, i.e., equal to the difference of ω_{end} and ω_{start} where ω_{start} and ω_{end} are the start and end values of the parameter ω over which a given partition is optimal. Note that the scale is linear unlike the logarithmic time scale in the heat kernel. This validates the efficacy of Laplacian modularity to detect multi-scale communities using general similarity measures. A more thorough investigation using alternate similarity measures, such as those used for link prediction, is suggested as future work. The stability curve corresponding to $S = I + A$, identifies the 10-way partition to be maximally persistent over the range of resolution parameter, ω, and has the highest community quality in terms of the normalized mutual information, better than all other methods (including DLM). The 8-way partition as identified by the cosine similarity metric on the contact-2 network has a higher community than all methods apart from DLM.

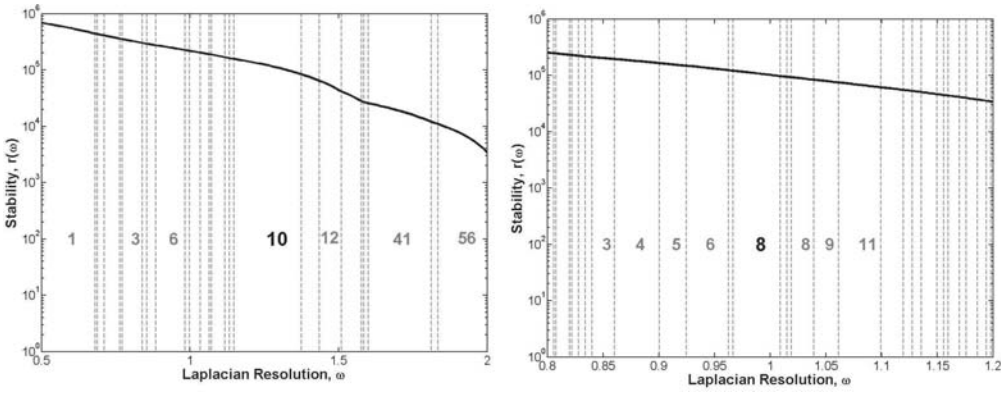


Figure 3.33. *Stability curves for multi-resolution Laplacian modularity. Left: Stability curve for the contact1 network using $S = I + A$, i.e., considering the adjacency matrix with self-loops as the node similarity matrix. Right: Stability curve for the contact2 network using cosine similarity.*

Similarity between nodes in a network is often intuitively resolved, whereas the associated community structure is non-trivial and difficult to anatomize. The power of the Laplacian modularity framework lies in efficient resolution of community structure given a metric of similarity, local or global, consistent with my notion of node-to-subgraph cohesion. I illustrate this framework using different metrics of similarity in order to uncover communities. For a general node-to-node similarity matrix, S, I also demonstrate that the multi-resolution Laplacian, $L_\omega = D - \omega A$, in lieu of the standard Laplacian, L, facilitates the identification of relevant communities. Although, I investigated the Laplacian modularity for different metrics of connectivity, its full potential using system-specific similarity metrics remains unexplored.

3.3 Deep Neural Networks Applications

I show application of deep neural networks for learning shapes using geometry images and for real time hand tracking using deep matrix completion.

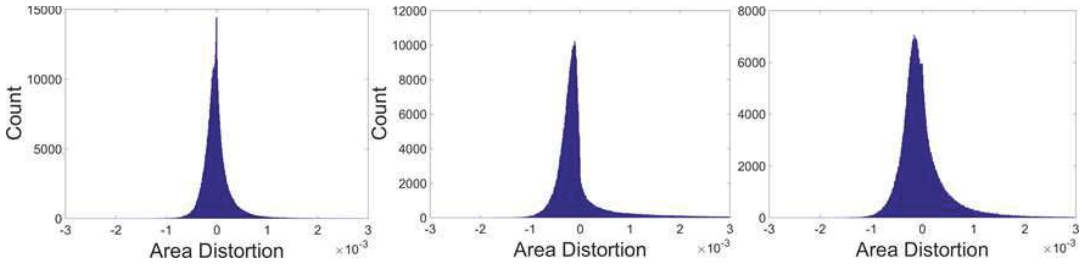


Figure 3.34. *Left to Right: Cumulative area distortion for mine, Lie advection based method in [158], Penalty based method proposed in [157]. My method has the sharpest peak and variance, 9.8e-8 compared to 5.2e-7 for [158] and 2.65e-7 for [157].*

3.3.1 Results on deep learning geometry images

In this section I first compare my spherical parametrization scheme to some other approaches. Then I demonstrate the efficacy of my framework to learn 3D shapes using geometry images on rigid as well as non-rigid shapes by comparing it with other methods.

Area distortion measure: I compare my authalic spherical parametrization scheme to other area correcting methods. I adjudge the quality of the parametrization in terms of the geometry image created from the corresponding spherical parameterizations on some prototypical meshes. The methods compared to are the lie advection based method in [158], and the penalty-term based method proposed in [157], both of which are iterative methods. For fair comparison, the maximum number of iterations was fixed to 100 for all methods along with suggested parameter settings. Figure 3.35 shows the comparison. I observe that my method is the only method to consistently complete the shape while keeping extraneous noise at a minimum. For example no method apart from mine is able to complete the bunny's ears or completely reveal all 5 fingers. This validates my approach in the context of geometry image creation and authalic spherical parametrization in general. I propose to use my method in applications requiring high throughput spherical parametrization such as brain image processing in future work. Figure 3.34 shows area distortion as a histogram over all triangles of 148 shapes in TOSCA database

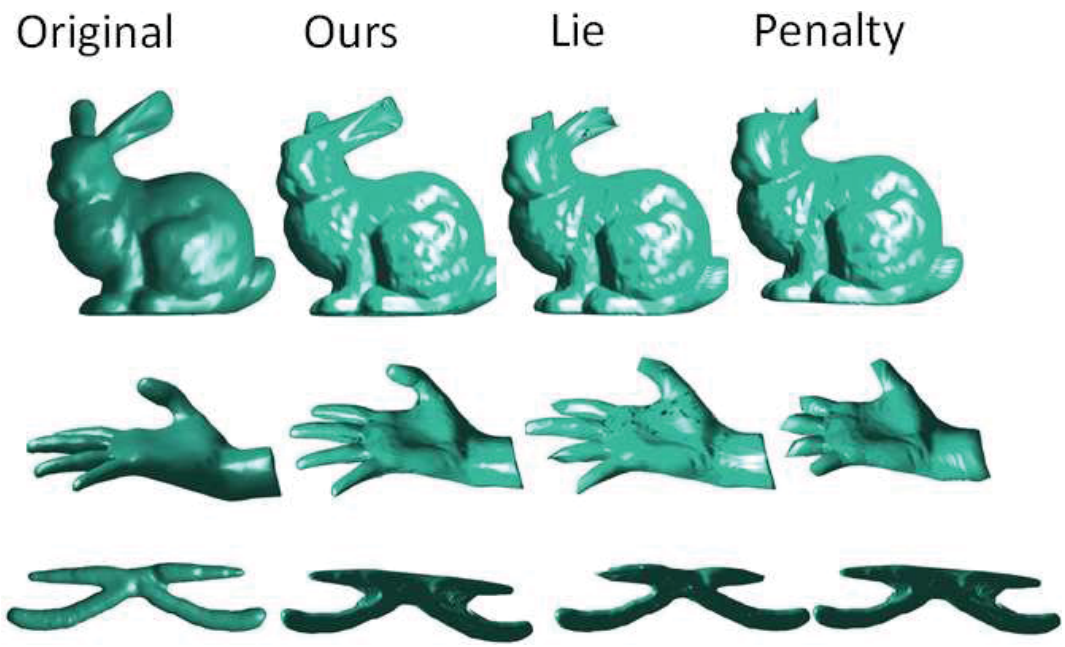


Figure 3.35. *Comparison of authalic surface parametrization methods in terms of shape reconstruction using geometry image. Left to Right: Original mesh model, My authalic parametrization, Lie advection based method in [158], Penalty based method proposed in [157].*

Non-Rigid Shapes: I evaluated my approach for surface based intrinsic learning of shapes on two datasets. I used 200 shapes from the McGill 3D shape benchmark consisting of articulated as well as non-articulated shapes from 10 classes (20 in each class). To test the robustness of my approach, I also evaluated my approach on the challenging SHREC-11 [209] database of watertight meshes consisting of 20 shapes from 30 classes (600 in total). For each of the 2 databases, I performed classification tasks on 2 splits: (1) 10 randomly chosen shapes from each class were used for training and 10 were test (2) 16 randomly chosen shapes were in the train set and the rest were test cases. Due to the small size of the database, I kept my CNN relatively shallow (3 convolutional, 1 fully connected layer and a classification layer) so as to limit the number of training parameters. In order to achieve rotational invariance, 36 geometry images were created for each spherical parametrization of a shape by (1) first fixing the intersections of the 3 coordinate axes with

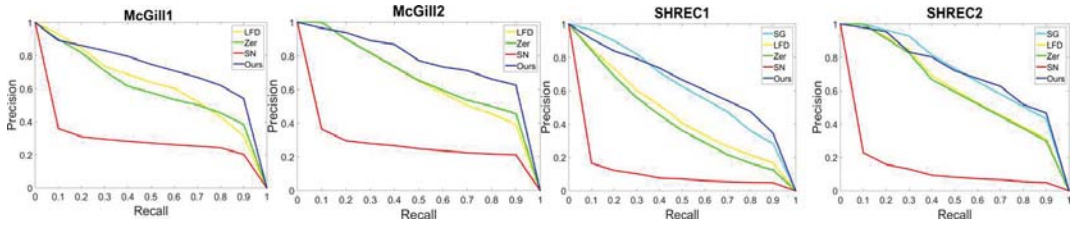


Figure 3.36. *Precision recall curves for shape retrieval on non-rigid shapes.*

the spherical domain as the center of the geometry image and then (2) incrementally rotating it by 60 degrees to cover a full circle. Images of size 56×56 were padded as described in Section 4.3 to produce a 64×64 image as input to the CNN. For features, I used HKS sampled at 5 logarithmically sampled time scales to produce a 5 dimensional feature map. Due to the small training sample, the CNN's using only principal curvatures failed to converge. Training using the HKS features converged after 30 epochs. I compare my approach to 4 other methods: State-of-art ShapeGoogle (SG) [210], Zerkine moments (Zer) [211], Light Field Descriptor (LFD) and 3DShapeNets [22] for classification and retrieval. A class was assigned to each shape in my method by simply pooling predictions from the softmax layer over the 36 views and then selecting the one with the highest overall score. I trained a SVM classifier for SG, LFD and Zer methods.[2] I see that my method significantly outperforms all other methods on both splits for the 2 databases (Table 3.2) indicating that my intrinsic representation was able to *learn* the intrinsic shape structure of each class. I use the L2 distance to measure the similarity between all pairs of testing samples and retrieval accuracy was measured in terms of mean average precision (MAP) as standard in literature. The penultimate 48-dimensional activation vector in the fully connected layer was used for measuring the retrieval accuracy of my method. I perform best in all but one dataset, i.e. 2^{nd} to SG for SHREC2, inspite my feature vector being $1/50^{th}$ the size of SG. This highlights that my method can be used to output highly informative shape signatures.

[2]Note I do not report the scores for SG on Mcgill because the author provided implementation failed on several shapes and produced spurious results

Table 3.2.
Classification (top) and retrieval (bottom) accuracy of my method compared to 4 other methods.

Database	McGill1	Mcgill2	SHREC1	SHREC2
SG [210]	NA	NA	62.6	70.8
Zer [211]	63.0	57.5	43.3	50.8
LFD [212]	75	72.5	56.7	65.8
SN [22]	65.0	57.2	52.7	48.4
Ours	**83.0**	**92.5**	**88.6**	**96.6**
SG [210]	NA	NA	**0.65**	**0.74**
Zer [211]	0.64	0.69	0.47	0.64
LFD [212]	0.67	0.68	0.50	0.65
SN [22]	0.29	0.28	0.10	0.13
Ours	**0.75**	**0.72**	**0.65**	0.72

Rigid Shapes: I evaluate my approach for surface-based learning of 3D shape classification on the two versions of the large scale Princeton ModelNet dataset: ModelNet40 and ModelNet10 consisting of 40 and 10 classes respectively. I use the principal curvatures and topological mask (intrinsic properties) along with a height field encoded as angle to the positive z-axis. Additionally, each spherical parameterizations has 12 replicates shifted by 30 degrees along the z-axis incrementally. The size and structure of the geometry image is the same as the ones used for non-rigid testing. Table 3.3 shows the results relative to 4 methods. I achieve the best accuracy on ModelNet40 dataset. This validates that intrinsic curvatures suffice for holistic learning of 3D shapes. I wish to build upon this deep insight in future work.

Table 3.3.
Classification accuracies of my method on the ModelNet40 and ModelNet10 database compared to 3 other methods.

Methods/Score	ModelNet40	ModelNet10
VoxNet	83.0	**92.0**
DeepPano	77.6	85.5
3DShapeNets	77.0	83.5
Ours	**83.9**	88.4

3.3.2 Results on real-time tracking using deep matrix completion

I along with my colleague Chiho Choi conducted comprehensive a evaluation with state-of-the-art approaches as well as self-generated baselines on the synthetic and real datasets to demonstrate the efficacy of my solution. I first describe the datasets and baselines.

Datasets: I split my evaluation into two stages. First, I use synthetic data to compare my method to baselines. This comparison validates the rationale of my specific approach against other choices. This data is generated using the same approach as described in Section 3 to generate my database, albeit continuity constraints are enforced. Two synthetic sequences are generated which are 2.5K frames long at standard rates (approximately 80 seconds each). The advantage of these synthetic sequences are that they are already labeled, avoiding tedious ground-truth assignment.

Next, for fair comparison to other methods, I evaluate the performance of my method on two publicly available datasets: Dexter1 [213] and NYU [28]. The Dexter1 dataset consists of seven gestures (i.e., adbadd, flexex1, pinch, fingercount, tigergrasp, fingerwave, and random) with high inter-gesture verifiability, however, mostly from frontal viewpoints. Hence I use the NYU dataset for a more thorough evaluation of the method. As I shall

shortly show, my method remarkably achieves state-of-art performance without fine-tuning on their training dataset.

Although the authors are aware of other datasets like ICVL [84], MSRA14 [88], or MSRA15 [85] in the literature, I do not use them for one or more of the following reasons: (1) the depth pixels of the body are included with the hand depth map. Recall I use a heuristic method for segregating the hand from the rest of the body and a wrist band under more extreme conditions. I did not find a straightforward way to segregate the data without incurring loss. (2) The hand poses are enforced using muscular labor, contrasting and modeling additional constraints accounting unnatural hand poses is plan of future work. Also note that I use the SoftKinetic's DethSense DS325 for all my real demonstrations.

Baselines for method validation: There are three salient features of my approach which I rigorously validate. First, a pool of activation features is better at estimating the hand pose than a direct regressor. Second, my matrix completion approach is better than alternative choices. Third, that a hierarchical approach is justified inspite of the computational overload it introduces. I naturally perform this validation by comparing to these three baselines: (a) *Conv-PQ* which directly estimates the pose parameters to be the nearest neighbor (b) *JMFC* which also performs a matrix update, although using computationally expensive iterations (c) *Holistic* which evaluates all parameters in an all-in-one approach using a single activation feature. The validation is done in terms of standard error metrics popular to pose estimation problems. They are: (a) the average joint angle error in degrees, (b) the average joint distance error in millimeters, (c) the maximum allowed joint angle error in terms of a threshold ε_A, and (d) the maximum allowed joint distance error in terms of a threshold ε_D. Broadly speaking, the first two metrics evaluate performance at a local joint level whereas the other measure global robustness of an approach. I employ the appropriate metric based on the context of the evaluation. Although my angle based method is particularly effective in minimizing joint angle errors, yet I choose joint distances as my error metric on public datasets to demonstrate the overall robustness of my approach.

Comparison to Baselines: In this section, I quantitatively evaluate my method with respect to the baselines on the synthetic datasets. Figure 3.37 shows that my method significantly

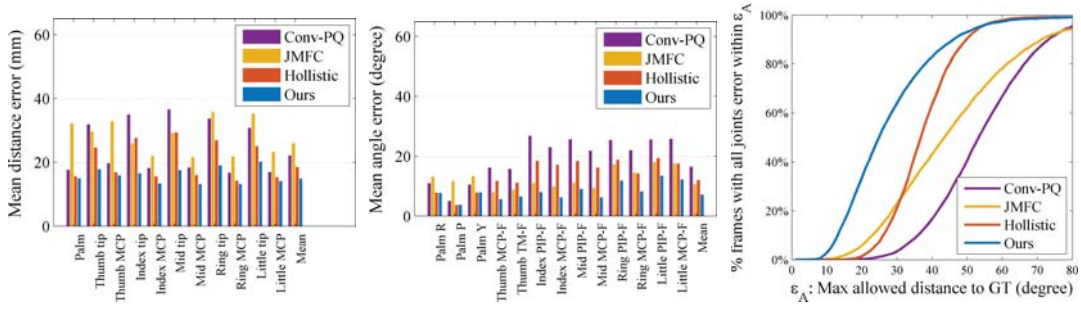


Figure 3.37. *Results of quantitative evaluation on the synthetic dataset.*

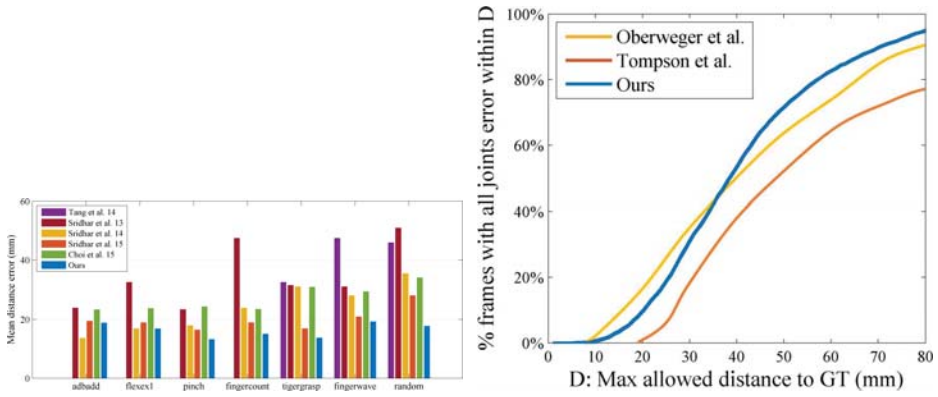


Figure 3.38. *The results of quantitative evaluation on the public dataset. Note that the accuracies are directly estimated from corresponding figures (i.e., Figure 4 in [214] and Figure 3a in [90]).*

Table 3.4.
The overall average error (mm) of the five fingertip positions on Dexter1. Mine shows the lowest error rate compared to the state-of-the-art methods.

Methods	[84]	[213]	[215]	[214]	[91]	Ours
Error	42.4	31.8	24.1	19.6	25.27	16.35

Table 3.5.
The overall average error (mm) of the joint positions on NYU. Note that 14 joint positions are used in [28] and [90], whereas I only use 12 joints (Ours-12) or 5 fingertips (Ours-5).

Methods	[28]	[90]	Ours-12	Ours-5
Error	21	20	11.39	14.25

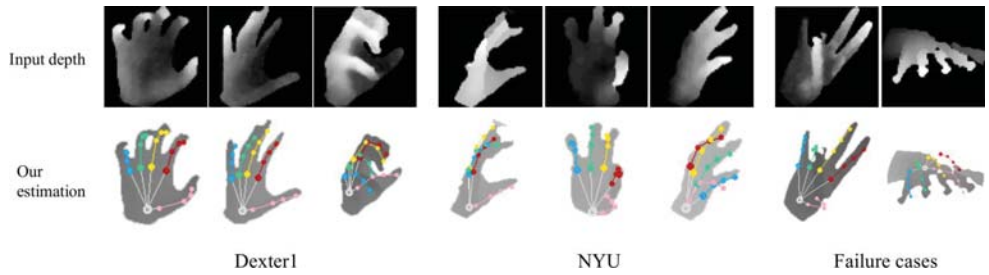


Figure 3.39. *Qualitative evaluations are conducted on two public datasets, Dexter1 and NYU. The first row shows the input depth image, and corresponding estimation is presented in the second row.*

outperforms the three baselines both in terms of local as well overall global accuracy. The performance markup over the *Conv-PQ* approach as seen in Figure 3.37 c indicates that a ConvNet by itself would do a poor job of inferring a complex articulated structure such as the hand. The performance improvement over Holistic in the zone of small angles is also intuitive. It indicates that the global activation feature contains some latent information about the local joint angles, but this information is better revealed by a hierarchical estimation procedure. This is also validated in Figure 3.3.2 and 3.3.2 where I see a significant performance improvement in terms of joint angles for finger portions that are frequently occluded such as the middle finger. It is also noteworthy to note that the similarity of these plots in terms of error ranges to plots on real hand sequences implicitly validate my data creation process.

Comparison with the state-of-the-arts: Having validated the rationale of my approach, I now compare my method to other state-of-the-art approaches [28, 84, 90, 91, 213–215] on the Dexter1 and NYU datasets.

Quantitative Analysis I measured the average distance error of five fingertips (in mm) on the Dexter1 dataset to evaluate the overall robustness of my approach. Figure 3.3.2 shows the comparison of my approach to four other methods which include both discriminative [84, 91] as well as generative [213, 215] methods. Not only does my method achieve the lowest overall error rate (see Table 3.4), I also achieve the lowest individual error rates for all but one gesture i.e. *adbadd*. This is because the particular gesture is especially hard to model in terms of joint angle constraints.

I evaluated my approach directly on the 8.2K of test depth maps from the NYU dataset. Figure 3.3.2 illustrates the maximum allowed error with respect to the distance threshold. The fact that my method performs better that [90] over a long range indicates that the activation features I get from ConvNet are general purpose and can be used across domains and sensor types [3]. This is encouraging in the context of progressively fine-tuning ConvNets with more information such as when new joint angle constraints or dynamic constraints become available.

Qualitative Analysis I do a qualitative evaluation of my algorithm with the state-of-the-art methods on some public datasets. The top row of Figure 3.39 shows cropped 64x64 depth images which are used as input to my system, and the second row shows corresponding estimates with my DMC completion method (without temporal neighbors). All estimated poses are kinematically valid and follow a natural sequence. For the sake of completion, I also show some failure cases in the last two columns of Figure 3.39. In my system this happens when some unnatural pose (driven by muscular force) appears in front of the camera or when the image is severely affected by noise or has missing parts.

[3] NYU dataset use PrimeSense to capture their data

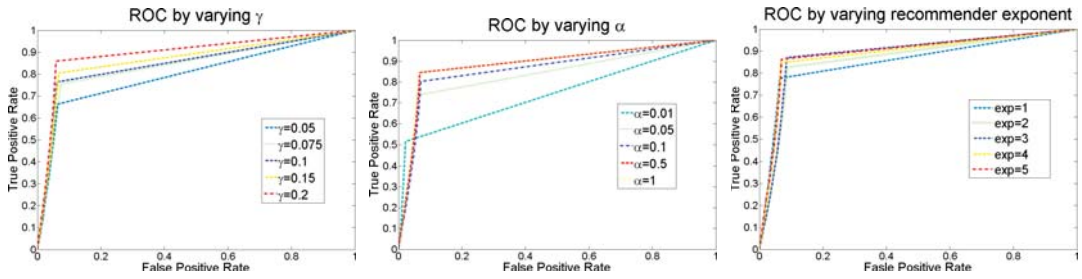


Figure 3.40. *Results for a synthetic recommender system with controllable effects. (Left to right): (a) ROC curves by varying data sparsity. (b) ROC curves by varying the deconvolution parameter α (c) ROC curves by varying feedback exponent of recommender system.*

3.4 Feedback Loops Deconvolved in Recommender Systems

I tested my approach for deconvolving feedback loops in recommender systems for synthetic datasets and designed a metric to identify the ratings most affected by the recommender. I then use the same automated technique to study real-world ratings data, and I find that the metric is able to identify items influenced to a recommender system. I first discuss my method to generate synthetic data simulating a real-world recommender system and discuss my results on real data.

3.4.1 Deconvolving feedback loops in synthetic data

I use item response theory to generate a sparse true rating matrix $\boldsymbol{R}_{\text{true}}$ using a model related to that in [216]. Let a_u be the center of user u's rating scale, and b_u be the rating sensitivity of user u. Let t_i be the intrinsic score of item i. I generate a user-item rating matrix as:

$$\boldsymbol{R}_{u,i} = L[a_u + b_u t_i + \eta_{u,i}] \tag{3.4}$$

where $L[\omega]$ is the discrete levels function assigning a score in the range 1 to 5:

$$L[\omega] = \max(\min(\text{round}(\omega), 5), 1) \tag{3.5}$$

and $\eta_{u,i}$ is a noise parameter. In my experiment, I draw $a_u \sim N(3,1)$, $b_u \sim N(0.5,0.5)$, $t_u \sim N(0.1,1)$, and $\eta_{u,i} \sim \epsilon N(0,1)$, where N is a standard normal, and ϵ is a noise parameter. I sample these ratings uniformly at random by specifying a desired level of rating sparsity γ which serves as the input, $\boldsymbol{R}_{\text{true}}$, to my recommender system. I then run a cosine similarity based recommender system, progressively increasing the density of the rating matrix. The unknown ratings are iteratively updated using the standard item-item collaborative filtering technique [217] as:

$$\boldsymbol{R}_{u,i}^{k+1} = \frac{\sum_{j \in i}(s_{i,j}^{k} \boldsymbol{R}_{u,j}^{k})}{\sum_{j \in i}(|s_{i,j}^{k}|)} \tag{3.6}$$

where k is the iteration number and $\boldsymbol{R}^0 = \boldsymbol{R}_{\text{true}}$, and the similarity measure at the k^{th} iteration is given as:

$$s_{i,j}^{k} = \frac{\sum_{u \in U} \boldsymbol{R}_{u,i}^{k} \boldsymbol{R}_{u,j}^{k}}{\sqrt{\sum_{u \in U} (\boldsymbol{R}_{u,i}^{k})^2} \sqrt{\sum_{u \in U} (\boldsymbol{R}_{u,j}^{k})^2}} \tag{3.7}$$

After the k^{th} iteration, each synthetic user accepts the top r recommendations with probability proportional to $(\boldsymbol{R}_{u,i}^{k+1})^e$, where e is an exponent controlling the frequency of acceptance. I fix the number of iterative updates to be 10, r to be 10 and the resulting rating matrix is $\boldsymbol{R}_{\text{obs}}$. I deconvolve $\boldsymbol{R}_{\text{obs}}$ as per Algorithm 1 to output $\hat{\boldsymbol{R}}_{\text{true}}$. Recall, $\hat{\boldsymbol{R}}_{\text{true}}$ is user-centered and item-normalized. In the absence of any recommender effects $\boldsymbol{R}_{\text{recom}}$, the expectation is that $\hat{\boldsymbol{R}}_{\text{true}}$ is perfectly correlated with $\hat{\boldsymbol{R}}_{\text{obs}}$. The absence of a linear correlation hints at factors extraneous to the user, i.e., the recommender. Thus, I plot $\hat{\boldsymbol{R}}_{\text{true}}$ (the deconvolved ratings) against the $\hat{\boldsymbol{R}}_{\text{obs}}$, and search for characteristic signals that exemplify recommender effects (see Figure 2.33a and inset).

A metric to assess a recommender system: I develop an algorithm guided by the intuition that deviation of ratings from a straight line suggest recommender effects (Algorithm 2). The procedure is visually elucidated in Figure 2.33. I consider fitting a line to the observed and deconvolved ratings; however, my experiments indicate that least square fit of a straight line in the presence of severe recommender effects is not robust. The outliers in my formulation correspond to recommended items. Hence, I use random sample consensus or the RANSAC method [218] to fit a straight line *on a per item basis* (Figure 2.33b). The

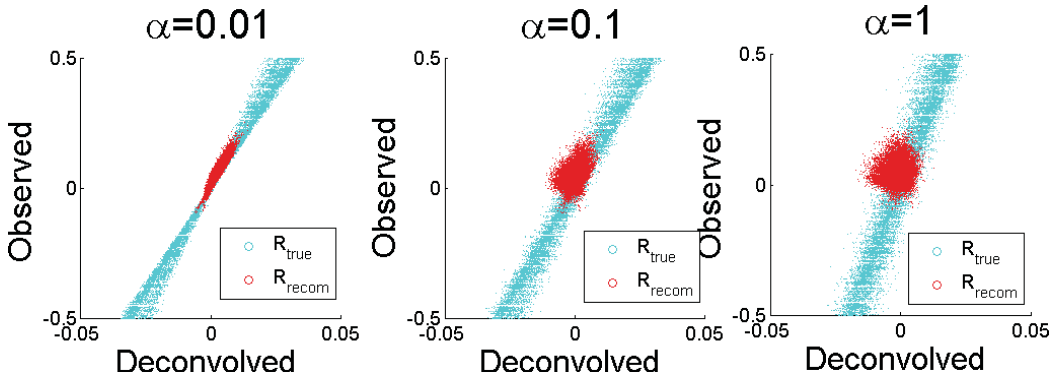


Figure 3.41. *(Left to Right): Increasing α deconvolves implicit feedback loops to a greater extent and better discriminates recommender effects as illustrated by the red points which show more pronounced deviation when $\alpha = 1$.*

pseudo code for RANSAC in Matlab is detailed in the appendix. All these straight lines are translated and rotated so as to coincide with the y-axis as displayed in Figure 2.33c. Observe that the data points corresponding to recommended ratings pop out as a bump along the x-axis. Thus, the effect of the RANSAC and rotation is to place the ratings into a precise location. Next, the ratings are scaled so as to make the maximum absolute values of the rotated and translated $\breve{\boldsymbol{R}}_{\text{true}}, \breve{\boldsymbol{R}}_{\text{obs}}$, values to be equal (Figure 2.33d).

The scores I design are to measure "extent" into the x-axis. But I want to consider some allowable vertical displacement. The final score I assign is given by fitting a hyperbola through each rating viewed as a point: $\breve{\boldsymbol{R}}_{\text{true}}, \breve{\boldsymbol{R}}_{\text{obs}}$. A straight line of slope, $\theta = 1$ passing through the origin is fixed as an asymptote to all hyperbolas. The vertex of this hyperbola serves as the score of the corresponding data point. The higher the value of the vertex of the associated hyperbola to a data point, the more likely is the data point to be recommended item. Using the relationship between slope of asymptote, and vertex of hyperbola, the score $s(\breve{\boldsymbol{R}}_{\text{true}}, \breve{\boldsymbol{R}}_{\text{obs}})$ is given by:

$$s(\breve{\boldsymbol{R}}_{\text{true}}, \breve{\boldsymbol{R}}_{\text{obs}}) = \text{real}(\sqrt{\breve{\boldsymbol{R}}_{\text{true}}^{2} - \breve{\boldsymbol{R}}_{\text{obs}}^{2}}) \tag{3.8}$$

I set the slope of the asymptote, $\theta = 1$, because the maximum magnitudes of $\breve{\boldsymbol{R}}_{\text{true}}, \breve{\boldsymbol{R}}_{\text{obs}}$ are equal (see Figure 2.33 d,e). Scores are zero if the point is inside the hyperbola with vertex 0.

Identifying high recommender effects in the synthetic system:

I display the ROC curve of my algorithm to identify recommended products in my synthetic simulation by varying the sparsity, γ in $\boldsymbol{R}_{\text{true}}$ (Figure 3.40a), varying α (Figure 3.40b), and varying exponent e (Figure 3.40c) for acceptance probability. The dimensions of the rating matrix is fixed at $[1000, 100]$ with 1000 users and 100 items. Decreasing α as well as γ has adversarial effects on the ROC curve, and hence, AUC values, as is natural. The fact that high values of α produce more discriminative deconvolution is clearly illustrated in Figure 3.41. Additionally, Figure 3.42 shows that the calculated score varies linearly with the true score as I change the recommender exponent, e, color coded in the legend. Overall, my algorithm is remarkably successful in extracting recommended items from $\boldsymbol{R}_{\text{obs}}$ without any additional information. Also, I can score the overall impact of the recommender system (see the upcoming section recommender system scores) and it accurately tracks the true effect of the recommender system.

3.4.2 Deconvolving feedback in real data

In this subsection I validate my approach for deconvolving feedback loops on real-world recommender systems. First, I demonstrate that the deconvolved ratings are able to distinguish datasets that use a recommender system against those that do not. Second, I specify a metric that reflects the extent of recommender system effects on the final ratings matrix. Finally, I conclusively validate that the score returned by my algorithm is indicative of the recommender effects on a per item basis. The α value for all my experiments is set to 1.

Datasets: Table 3.6 lists all the datasets I use to validate my approach for deconvolving recommender systems (from [219–221]). The columns detail name of the dataset, number of users, the number of items, the lower threshold for number of ratings per item (RPI)

Table 3.6.
Datasets and parameters for evaluating deconvolved effects.

Dataset	Users	Items	Min RPI	Rating	k in SVD	Score
Jester-1	24.9K	100	1	615K	100	0.0487
Jester-2	50.6K	140	1	1.72M	140	0.0389
MovieLens-100K	943	603	50	83.2K	603	0.2834
MovieLens-1M	6.04K	2514	50	975K	2514	0.3033
MovieLens-10M	69.8K	7259	50	9.90M	1500	0.3821
BeerAdvocate	31.8K	9146	20	1.35M	1500	0.2223
RateBeer	28.0K	20129	20	2.40M	1500	0.1526
Fine Foods	130K	5015	20	329K	1500	0.1209
Wine Ratings	21.0K	8772	20	320K	1500	0.1601
Netflix	480K	16795	100	100M	1500	0.2661
Yahoo	1.94M	10740	50	109M	1000	0.5101

considered in the input ratings matrix and the number of singular vectors k (as many as possible based on the limits of computer memory), respectively. I briefly discuss the datasets below.

Jester. I use two versions of the Jester-joke dataset, one collected between April 1999 - May 2003 and the other between November 2006 - May 2009. These two datasets contain ratings directly reported by the users without a recommender system interface.

MovieLens. MovieLens data sets were collected by the GroupLens Research Project at the University of Minnesota. I use three versions of the data: MovieLens-100K consisting of one hundred thousand ratings, MovieLens-1M consisting of one million ratings and MovieLens-10M consisting of 10 million ratings. I gathered from the Movielens website that Movielens-100K dataset was released on 4/1998, MovieLens 1M was released on 2/2003 and MovieLens 10M was released on 1/2009. Movielens 10M contains almost all movies in the 100K and 1M dataset. This indicates that the data corresponds to different snap-shots of the same data obtained at different times of the data collection process.

Beer, Wine, Food. The beer-rating websites BeerAdvocate and RateBeer allow users to rate beers using a five-aspect rating system. They also include reviews of pubs. I also

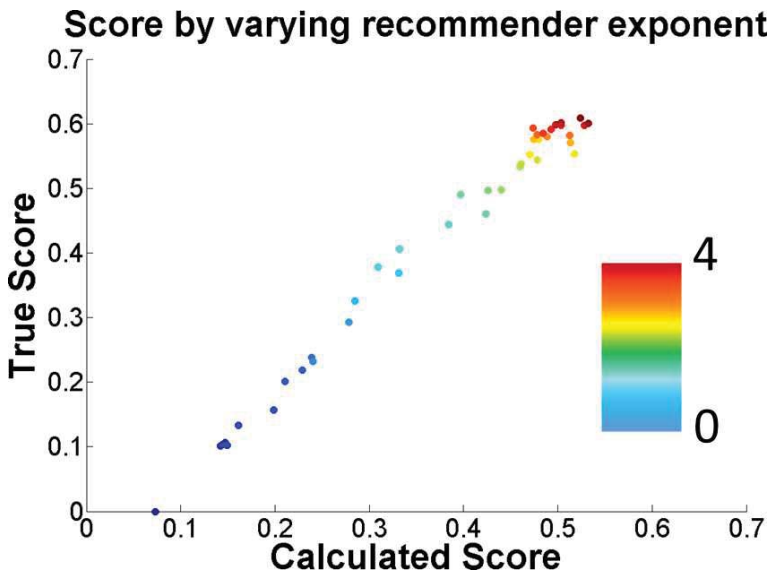


Figure 3.42. *Score assessing the overall recommendation effects as I vary the true effect.*

consider the wine review website CellarTracker, as well as reviews from the Fine Foods. Note that the rating datasets have a different scale (e.g. beers on RateBeer are rated out of 20, wines on CellarTracker are rated out of 100, etc.).

Netflix. Netflix provided a training data set of 100,480,507 ratings that 480,189 users gave to 17,770 movies for the Netflix competition [220]. This dataset had the Cinematch algorithm running on it.

Yahoo. This Yahoo! Music data represents a sample of (anonymized) Yahoo! users' ratings of musical artists around March 2004. It contains 100 million ratings of 98,211 artists by 1,948,882 anonymous users.

Classification of ratings matrix: I display the density plot of observed (y-axis) vs. deconvolved (x-axis) ratings for all datasets considered in out evaluation in Figure 3.44. Higher density is indicated by darker shades in the scatter plot of observed and deconvolved ratings. The two datasets that have no recommender system running on them are Jester-1 and Jester-2 [221]. Figures 3.44 a,b display the density plot of observed vs. decon-

volved ratings for the two datasets. The fact that this dataset did not use a recommender system is evident from the density plot, wherein I see that the observed and deconvolved ratings are linearly correlated. In contrast, the density plot of observed vs. deconvolved ratings for the other datasets (Figure 3.44 c-k) show varying levels of dispersion, and indicate that the observed and deconvolved ratings are not very correlated or that a recommender system is operating on these datasets. It is the ratings that fall in the zones above and below the straight line of linear correlation that suffer from recommender effects. I discuss these recommender effects in terms of the recommender system score, subsequently.

Recommender system scores: The recommender system scores I displayed in Table 3.6 are based on the fraction of ratings with non-zero score (using the score metric Equation (3.8)). Recall that a zero score indicates that the data point lies outside the associated hyperbola and does not suffer from recommender effect. Hence, the recommender system score is indicative of the fraction of ratings affected by the recommender. Looking back at Table 3.6, I see that the two Jester datasets have low recommender system scores validating that the Jester dataset did not run a recommender system. Interestingly, the recommender system score progressively increases for the three versions of the MovieLens datasets: MovieLens-100K consisting of hundred thousand ratings, MovieLens-1M consisting of one million ratings and MovieLens-10M consisting of 10 million ratings. This is expected as the recommender system effects would have progressively accrued over time in these datasets. Note that Netflix is also lower than Movielens, indicating that Netflix's recommender likely correlated better with users' true tastes. The recommender system scores associated with alcohol datasets (RateBeer, BeerAdvocate and Wine Ratings) are higher compared to the Fine Foods dataset. This is surprising. I conjecture that this effect is due to common features that correlate with evaluations of alcohol such as the age of wine or percentage of alcohol in beer. But this requires more investigation. Furthermore, the Yahoo dataset outputs high recommender system scores with more than 50% of the ratings affected by Yahoo's recommender system. Music preference is intrinsically complex because it possesses several intricate features beyond artist and genre classification and musical tastes are known to be highly susceptible to influence [63]. Moreover, early

online music systems surfaced many previously unknown artists. Thus, it is natural that the dataset has a high recommender system score. I next discuss the salient features of my approach with respect to the Netflix dataset.

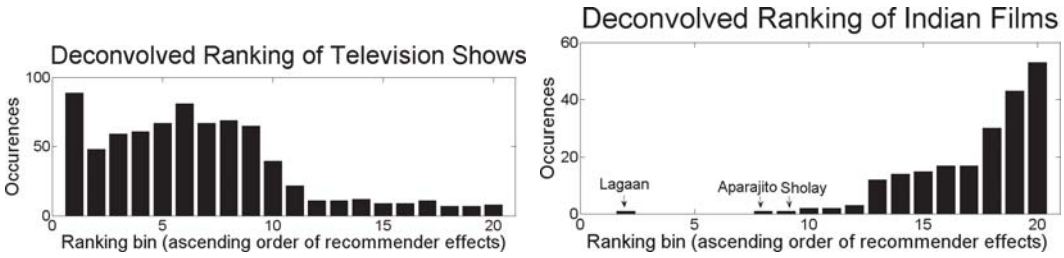


Figure 3.43. *(Left to right) (a) Deconvolved ranking as a bar chart for T.V. shows. (b) Deconvolved ranking as a bar chart for Indian movies.*

Ranking of items based on recommendation score: I associate a recommender system rating to each item as my mean score of an item over all users. All items are ranked in ascending order of recommender system score and I first look at items with low recommender system scores. The Netflix dataset comprises of movies as well as television shows. I expect that television shows are less likely to be affected by a recommender system because each season of a T.V. show requires longer time commitment, and they have their own following. To validate this expectation, I first identify all T.V. shows in the ranked list and compute the number of occurrences of a T.V. show in equally spaced bins of size 840. Figure 3.43 shows a bar chart for the number of occurrences and I see that there are ≈ 90 T.V.shows in the first bin (or top 840 items as per the score). This is highest compared to all bins and the number of occurrences progressively decrease as I move further down the list, validating my expectation. Also unsurprisingly, the seasons of the popular sitcom Friends comprised of 10 out of the top 20 T.V. seasons with lowest recommender system scores. It is also expected that the Season 1 of a T.V. show is more likely to be recommended relative to subsequent seasons. I identified the top 40 T.V shows with multiple (at least 2) seasons, and observed that 31 of these have a higher recommender system score for Season 1 relative to Season 2. The 9 T.V. shows where the converse is true are mostly comedies like

Coupling, That 70's Show etc., for which the seasons can be viewed independently of each other.

Next, I looked at items with high recommender system score. At the time the dataset was released, Netflix operated exclusively in the U.S., and one plausible use is that immigrants might use Netflix's recommender system to watch movies from their native country. I specifically looked at Indian films in the ranked list to validate this expectation. Figure 3.43b shows a bar chart similar to the one plotted for T.V. shows and I observe an increasing trend along the ranked list for the number of occurrences of Indian films. The movie with lowest recommendation score is Lagaan, the only Indian movie to be nominated for the Oscars in last 25 years. Another interesting observation is that the Primer, a low budget movie known for its cult following has a low recommender system score and is ranked 46 among 16,795 movies, indicating that users were not recommended the film but watched it out of self-interest.

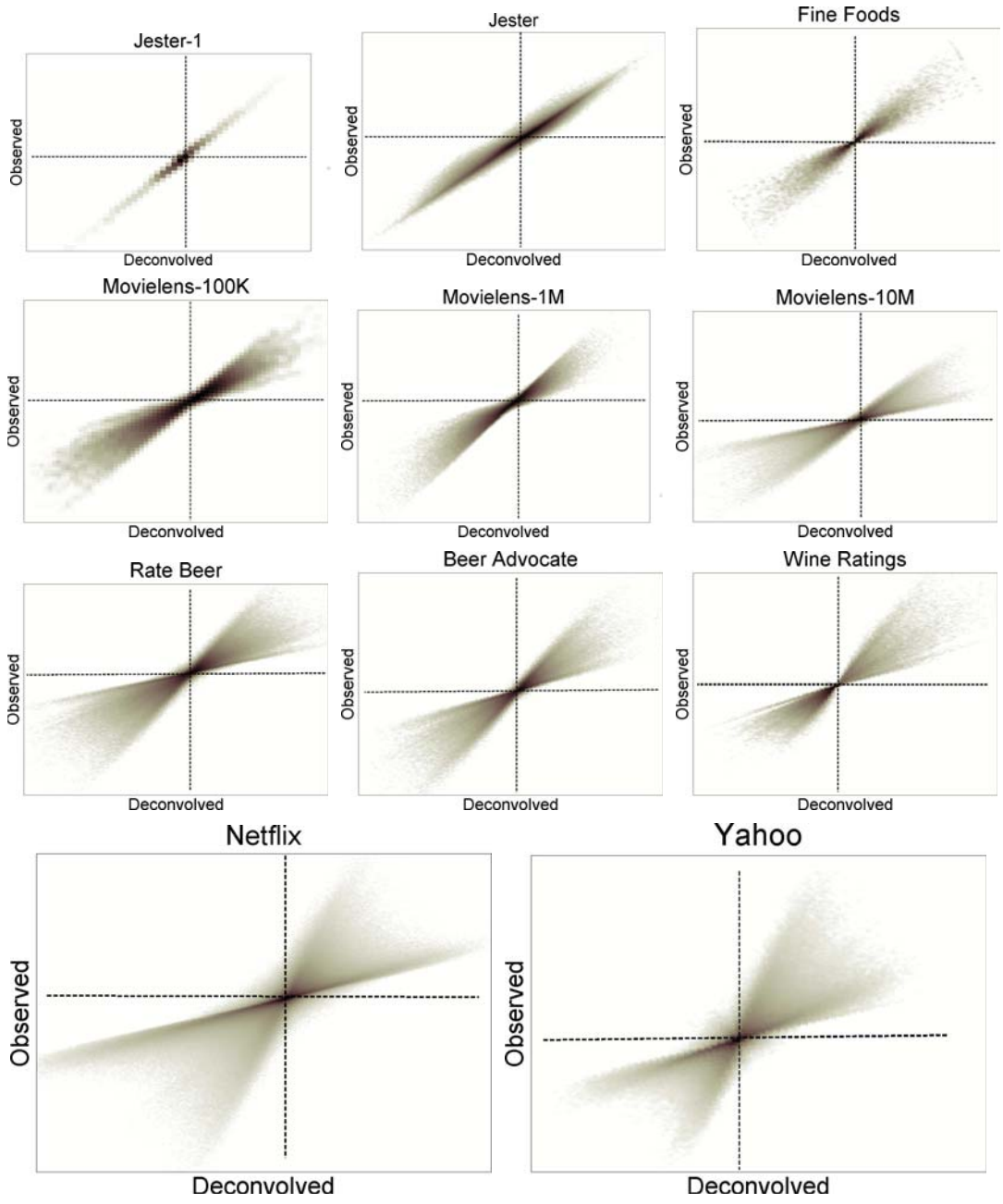


Figure 3.44. *Density plot of deconvolved vs.observed ratings for (a) Jester-1 (b) Jester-2 (c) Fine Foods (d) MovieLens-100K (e) MovieLens-1M (f) MovieLens-10M (g) RateBeer (h) BeerAdvocate (i) Wine Ratings (j) Netflix (k) Yahoo, respectively.*

4. SUMMARY

My contributions in this dissertation is the proposal of four physical or natural laws for knowledge discovery in data. I discuss each of them separately.

4.1 Random Walks

My dissertation presents a general framework to construct multiscale kernels on discrete meshes using random walks and in doing so presents a consolidated view of current state of the art shape signatures, distance metrics and embeddings. It also offers an intuitive explanation not offered by current frameworks. These kernels possess all necessary characteristics, hence immediately applicable for shape analysis including and not limited to shape retrieval, robust segmentation, correspondence propagation and symmetry detection. The notion of dual multiscale kernels is to my knowledge the first and the experiments show promising potential for GMS. One current limitation is regular triangulation of the mesh as defined in [222], so as to ensure positive weights (Property 1). Future research work can investigate continuous state space random walks, i.e., Brownian motion on Riemannian manifolds and recover the same set of intuitive interpretations. Combining the $GMSn$ and $GMS\alpha$ signatures in a coherent way so as to construct optimal descriptors is another line of work. Another line of research is to investigate different functional forms of t instead of t^n suited for different applications. My work serves as a gateway from Markovian processes to discrete geometry and it is my belief that it only constitutes the tip of the proverbial iceberg in realizing the full scope of such an integrated approach.

In the context of diffusion and random walks, I also propose the novel diffusion modulus criterion. The criterion underscores the paradigm of clustering using learnt pairwise affinities. Affinity learning is motivated by the physical analogy of the heat diffusion process and grouping is driven by flow conservation. Notably, the criterion generates the

normalized cut partition and the modularity metric based on the resolution parameter n. I empirically validated the benefit of the resolution parameter n for segmenting images over the special cases of commute-time and biharmonic kernel. Future work will focus on more diverse applications and automatically determine the optimal resolution. Overall, the diffusion modulus provides an elegant integrated approach for affinity learning and the perceptual grouping problem. It is my belief that the diffusion modulus provides a fresh perspective to the original goal of the normalized cut to extract the *'big picture'* of a scene.

4.2 Gauss's Law

I have illustrated the Laplacian modularity under the paradigm of heat diffusion in order to systematically uncover overlapping communities organized at relevant hierarchical scales of a network. My framework extends to other notions of similarity that generate the connectivity potentials and I investigate multi-resolution Laplacians, cosine similarity, and adjacency. It can scale to networks with millions of nodes when the connectivity potential is sparse. The Laplacian modularity quality function is further extensible to networks with directed, weighted and signed links, applicable to time-dependent multiplex or multislice networks, and hence, capable of identifying community structure in a broad class of networks.

My idea of identifying the community boundaries has its roots in the original Girvan-Newman algorithm which iteratively removes boundary links based on edge betweenness [37]. In comparison, optimizing the Laplacian modularity quality function reveals the boundary links all at once. The power of the Laplacian modularity framework also lies in efficient resolution of community structure given a metric of similarity, local or global, consistent with my notion of node-to-subgraph cohesion. These ideas have a strong connection to classical physics through the network analog of Gauss's law. I believe that my approach using Gauss's law to identify community boundaries, in concert with system-specific connectivity potentials, holds great promise in improving my understanding of the modular nature of complex networks.

4.3 Deep Neural Networks

I introduce geometry images for intrinsically learning 3D shape surfaces. My geometry images are constructed by combining area correcting flows, spherical parameterizations and barycentric mapping. I show the potential of geometry images to encode intrinsic properties of shape surfaces and demonstrate its efficacy for understanding both non-rigid and rigid shapes. Furthermore, my work serves as a general validation of surface based representations for shape understanding. I believe that deep learning using geometry images can potentially spark a closer communion between the 3D vision and geometry community.

I also present a novel framework for hand pose estimation using a deep convolutional neural network. Instead of using a single activation feature, I use a pool of activation features to synchronize and collectively estimate the hand configuration, all in real time. This pool is derived by training a deep ConvNet with a large database of synthetic hand poses and efficiently storing the activation feature corresponding to the penultimate fully connected layer. Careful thought was placed so that this database is reflective of real data. At runtime the pool of activation features in the spatial domain and temporal domain combine together in a hierarchical way to robustly estimate the hand pose. The derived activation features can be applied across domains and sensor types as demonstrated in my experiments. Furthermore, my method achieves state of the art performance. Although my approach is general, one limitation of my activation features is that the estimations are only valid in the joint angle domain. Future work can focus on ways such that people working in the joint angle or joint position domain can seamlessly fuse their models together to create even deeper and more robust models. Another line of future work is to investigate the deep matrix completion approach in a more general setting. The simplicity combined with its efficiency makes is a promising alternative to standard regression techniques for a wide array of machine learning tasks.

4.4 Feedback Loops

I propose a mechanism to deconvolve feedback effects on recommender systems, similar in spirit to the network deconvolution method to distinguish direct dependencies in biological networks [179]. I do so by only considering a ratings matrix at a given instant of time. My approach depends on a few reasonable assumptions that enables me to create a tractable model of a recommender system. When I evaluate the resulting methods on synthetic and real-world datasets, I find that I are able to assess the degree of influence that a recommender system has had on those ratings. This analysis is also easy to compute and just involves a singular value decomposition of the ratings matrix.

Future work can rigorously analyze the relationship between deconvolution on square matrices and general matrices. Researchers can investigate each assumption to a greater detail, and possibly relax some of the assumptions. Some derivatives of my method include setting the parameters considered unknown in my current approach with known values (such as S) if known *a priori*. Incorporating temporal information at different snapshots of time while deconvolving the feedback loops is also an interesting line of future work. From another viewpoint, my approach can serve as a supplement to the active learning community to unbias the data and reveal additional insights regarding feedback loops considered in this paper. Overall, I believe that deconvolving feedback loops opens new gateways for understanding ratings and recommendations.

5. RECOMMENDATIONS

My goal during my doctoral research was to delve deeper into physically motivated algorithms for data mining and pattern recognition in graphs. I was able to couple machine learning with physical laws to reveal patterns that govern communities and society at large. As the underlying motivation of my methods stem from the physical world around us, the resulting algorithms were simple to implement and easy to understand. Having gained unique insights during the course of my PhD, I recommend these physical and geometric constructs as possible avenues of future work in data mining.

- Poissons equation for network link prediction in analogy with Poisson surface reconstruction and mesh processing.
- Quantum random walks for overlapping cluster detection, inspired by mixed quantum states.
- Diffusion on dynamic networks to reveal patterns over time. Examine the connections between the time to equilibrium for a random walk and the temporal evolution of a network
- Resistor-capacitor-inductor circuits for network analysis using Laplace transforms. Extend known relationships between the Kirchhoff/resistance matrix and the Laplacian to general electrical networks using concepts of circuit theory.
- Partial differential equations beyond the diffusion equation for understanding big data. Use knowledge of fluid mechanics and thermodynamics in the domain of data mining.
- Gauss law for networks by exploring more advanced concepts related to divergence, gradient and Laplacian operators on networks.

Over time, I am confident the above approaches to data mining will reveal new insights from big data.

LIST OF REFERENCES

LIST OF REFERENCES

[1] Jian Sun, Maks Ovsjanikov, and Leonidas J. Guibas. A concise and provably informative multi-scale signature based on heat diffusion. *Comput. Graph. Forum*, 28(5):1383–1392, 2009.

[2] Raif M. Rustamov. Multiscale biharmonic kernels. *Comput. Graph. Forum*, 30(5):1521–1531, 2011.

[3] Alexander M. Bronstein, Michael M. Bronstein, Leonidas J. Guibas, and Maks Ovsjanikov. Shape google: Geometric words and expressions for invariant shape retrieval. *ACM Trans. Graph.*, 30(1):1:1–1:20, February 2011.

[4] P. Skraba, M. Ovsjanikov, F. Chazal, and L. Guibas. Persistence-based segmentation of deformable shapes. In *CVPR Workshop on Non-Rigid Shape Analysis and Deformable Image Alignment*, June 2010.

[5] Maks Ovsjanikov, Quentin Mérigot, Facundo Mémoli, and Leonidas J. Guibas. One point isometric matching with the heat kernel. *Comput. Graph. Forum*, 29(5):1555–1564, 2010.

[6] L. Lovász. Random walks on graphs: A survey. *Combinatorics, Paul Erdos is Eighty*, 2(1):1–46, 1993.

[7] Xianfang Sun, Paul L. Rosin, R. R. Martin, and F. C. Langbein. Random walks for mesh denoising. In *SPM*, pages 11–22, Beijing, China, 2007. ACM.

[8] Yu-Kun Lai, Shi-Min Hu, Ralph R. Martin, and Paul L. Rosin. Fast mesh segmentation using random walks. In *SPM*, pages 183–191, New York, USA, 2008. ACM.

[9] Juyong Zhang, Jianmin Zheng, and Jianfei Cai. Interactive mesh cutting using constrained random walks. *IEEE Transactions on Visualization and Computer Graphics*, 17(3):357–367, May 2011.

[10] Bruno Lévy. Laplace-beltrami eigenfunctions towards an algorithm that "understands" geometry. In *SMI*, page 13, College Station, USA, 2006. IEEE.

[11] Boaz Nadler, Stphane Lafon, Ronald R. Coifman, and Ioannis G. Kevrekidis. Diffusion maps, spectral clustering and eigenfunctions of fokker-planck operators. In *NIPS*, pages 955–962, Vancouver, Canada, 2005. MIT Press.

[12] Risi Imre Kondor and John D. Lafferty. Diffusion kernels on graphs and other discrete input spaces. In *ICML*, pages 315–322, Sydney, Australia, 2002. Morgan Kaufmann.

[13] Mathieu Aubry, Ulrich Schlickewei, and Daniel Cremers. The wave kernel signature: A quantum mechanical approach to shape analysis. In *ICCV Workshops*, pages 1626–1633. IEEE, 2011.

[14] Yaron Lipman, Raif Rustamov, and Thomas Funkhouser. Biharmonic distance. *ACM Transactions on Graphics*, 29(3), June 2010.

[15] Raif M. Rustamov. Average interpolating wavelets on point clouds and graphs. *CoRR*, abs/1110.2227, 2011.

[16] Xueyuan Zhou, Mikhail Belkin, and Nathan Srebro. An iterated graph laplacian approach for ranking on manifolds. In *KDD*, pages 877–885, San Diego, USA, 2011. ACM.

[17] Raif M. Rustamov. Laplace-beltrami eigenfunctions for deformation invariant shape representation. In *SGP*, pages 225–233, Barcelona,Spain, 2007. Eurographics.

[18] Huaijun Qiu and Edwin R. Hancock. Clustering and embedding using commute times. *IEEE Trans. Pattern Anal. Mach. Intell.*, 29(11):1873–1890, 2007.

[19] Franois Fouss, Alain Pirotte, Jean michel Renders, and Marco Saerens. Random-walk computation of similarities between nodes of a graph, with application to collaborative recommendation. *IEEE Transactions on Knowledge and Data Engineering*, 19:2007, 2006.

[20] Xiao Bai, RichardC. Wilson, and EdwinR. Hancock. Manifold embedding of graphs using the heat kernel. In *IMA*, volume 3604, pages 34–49, Loughborough, UK, 2005. Springer.

[21] Mark Newman. *Networks: An Introduction*. Oxford University Press, Inc., New York, NY, USA, 2010.

[22] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1912–1920, 2015.

[23] Daniel Maturana and Sebastian Scherer. Voxnet: A 3D convolutional neural network for real-time object recognition. *Signal Processing Letters 2015*, 2015.

[24] Baoguang Shi, Song Bai, Zhichao Zhou, and Xiang Bai. Deeppano: Deep panoramic representation for 3-D shape recognition. *Signal Processing Letters, IEEE*, 22(12):2339–2343, 2015.

[25] Jian Sun, Maks Ovsjanikov, and Leonidas Guibas. A concise and provably informative multi-scale signature based on heat diffusion. In *Proceedings of the Symposium on Geometry Processing*, SGP '09, pages 1383–1392, Aire-la-Ville, Switzerland, Switzerland, 2009. Eurographics Association.

[26] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C.J.C. Burges, L. Bottou, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012.

[27] Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. Large-scale video classi?cation with convolutional neural networks. In *Proceedings of International Computer Vision and Pattern Recognition (CVPR 2014)*, 2014.

[28] Jonathan Tompson, Murphy Stein, Yann Lecun, and Ken Perlin. Real-time continuous pose recovery of human hands using convolutional networks. *ACM Transactions on Graphics (TOG)*, 33(5):169, 2014.

[29] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jagannath Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 580–587. IEEE, 2014.

[30] Davide Boscaini, Jonathan Masci, Simone Melzi, Michael M Bronstein, Umberto Castellani, and Pierre Vandergheynst. Learning class-specific descriptors for deformable shapes using localized spectral convolutional networks. In *Computer Graphics Forum*, volume 34, pages 13–23. Wiley Online Library, 2015.

[31] Jonathan Masci, Davide Boscaini, Michael M Bronstein, and Pierre Vandergheynst. Shapenet: Convolutional neural networks on non-euclidean manifolds. *arXiv preprint arXiv:1501.06297*, 2015.

[32] Yi Fang, Jin Xie, Guoxian Dai, Meng Wang, Fan Zhu, Tiantian Xu, and Edward Wong. 3d deep shape descriptor. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2319–2328, 2015.

[33] Xianfeng Gu, Steven J. Gortler, and Hugues Hoppe. Geometry images. In *Proceedings of the 29th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '02, pages 355–361, New York, NY, USA, 2002. ACM.

[34] Bruno Lévy, Sylvain Petitjean, Nicolas Ray, and Jérome Maillot. Least squares conformal maps for automatic texture atlas generation. *ACM Trans. Graph.*, 21(3):362–371, July 2002.

[35] Emil Praun and Hugues Hoppe. Spherical parametrization and remeshing. In *ACM Transactions on Graphics (TOG)*, volume 22, pages 340–349. ACM, 2003.

[36] Mark E. J. Newman, Albert L. Barabási, and Duncan J. Watts, editors. *The Structure and Dynamics of Networks*. Princeton University Press, 2006.

[37] M. Girvan and M. E. J. Newman. Community structure in social and biological networks. *Proceedings of the National Academy of Sciences*, 99(12):7821–7826, June 2002.

[38] Santo Fortunato. Community detection in graphs. *Physics Reports*, 486(3):75–174, 2010.

[39] Gergely Palla, Albert-Laszlo Barabasi, and Tamas Vicsek. Quantifying social group evolution. *Nature*, 446(7136):664–667, April 2007.

[40] G. Palla, I. Derenyi, I. Farkas, and T. Vicsek. Uncovering the overlapping community structure of complex networks in nature and society. *Nature*, 435:814–818, 2005.

[41] Y.-Y. Ahn, J. P. Bagrow, and S. Lehmann. Link communities reveal multiscale complexity in networks. *Nature*, 466:761–764, 2010.

[42] Aaron Clauset, Cristopher Moore, and M. E. J. Newman. Hierarchical structure and the prediction of missing links in networks. *Nature*, 453:98–101, 2008.

[43] Marta Sales-Pardo, Roger Guimerà, André A. Moreira, and Luís A. Nunes Amaral. Extracting the hierarchical organization of complex systems. *Proceedings of the National Academy of Sciences*, 104(39):15224–15229, September 2007.

[44] M. E. J. Newman and M. Girvan. Finding and evaluating community structure in networks. *Physical Review E*, 69(2):026113+, August 2003.

[45] Filippo Radicchi, Claudio Castellano, Federico Cecconi, Vittorio Loreto, and Domenico Parisi. Defining and identifying communities in networks. *Proceedings of the National Academy of Sciences*, 101(9):2658–2663, March 2004.

[46] Andrea Lancichinetti, Santo Fortunato, and János Kertész. Detecting the overlapping and hierarchical community structure in complex networks. *New Journal of Physics*, 11(3):033015+, March 2009.

[47] Aaron Clauset. Finding local community structure in networks. *Physical Review E*, 72:026132, Aug 2005.

[48] Martin Rosvall and Carl T. Bergstrom. Maps of random walks on complex networks reveal community structure. *Proceedings of the National Academy of Sciences*, 105(4):1118–1123, January 2008.

[49] M. E. J. Newman. Modularity and community structure in networks. *Proceedings of the National Academy of Sciences*, 103(23):8577–8582, June 2006.

[50] Brian Karrer and M. E. J. Newman. Stochastic blockmodels and community structure in networks. *Physical Review E*, 83:016107, Jan 2011.

[51] E. Ravasz, A. L. Somera, D. A. Mongru, Z. N. Oltvai, and A. L. Barabási. Hierarchical organization of modularity in metabolic networks. *Science (New York, N.Y.)*, 297(5586):1551–1555, August 2002.

[52] J. Reichardt and S. Bornholdt. Statistical mechanics of community detection. *Physical Review E*, 74(1 Pt 2), July 2006.

[53] A Arenas, A Fernndez, and S Gmez. Analysis of the structure of complex networks at different resolution levels. *New Journal of Physics*, 10(5):053039, 2008.

[54] Santo Fortunato and Marc Barthélemy. Resolution limit in community detection. *Proceedings of the National Academy of Sciences*, 104(1):36–41, January 2007.

[55] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. In *Proceedings of the 7th International World Wide Web Conference*, pages 161–172, Brisbane, Australia, 1998.

[56] Huaijun Qiu and E. R. Hancock. Clustering and Embedding Using Commute Times. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 29(11):1873–1890, November 2007.

[57] Xueyuan Zhou, Mikhail Belkin, and Nathan Srebro. An iterated graph laplacian approach for ranking on manifolds. In *KDD*, pages 877–885, 2011.

[58] A. Sinha and K. Ramani. Multi-scale kernels using random walks. *Computer Graphics Forum*, 33(1):164–177, 2014.

[59] Muriel Mdard Manolis Kellis Soheil Feizi, Daniel Marbach. Network deconvolution as a general method to distinguish direct dependencies in networks. *Nature Biotechnology*, (8):726733, 2013.

[60] Greg Linden, Brent Smith, and Jeremy York. Amazon.com recommendations: Item-to-item collaborative filtering. *IEEE Internet Computing*, 7(1):76–80, January 2003.

[61] Francesco Ricci, Lior Rokach, Bracha Shapira, and Paul B. Kantor. *Recommender Systems Handbook*. Springer-Verlag New York, Inc., New York, NY, USA, 1st edition, 2010.

[62] Dan Cosley, Shyong K. Lam, Istvan Albert, Joseph A. Konstan, and John Riedl. Is seeing believing?: How recommender system interfaces affect users' opinions. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '03, pages 585–592, New York, NY, USA, 2003. ACM.

[63] Matthew J. Salganik, Peter Sheridan Dodds, and Duncan J. Watts. Experimental study of inequality and unpredictability in an artificial cultural market. *Science*, 311(5762):854–856, 2006.

[64] Xavier Amatriain, Josep M. Pujol, Nava Tintarev, and Nuria Oliver. Rate it again: Increasing recommendation accuracy by user re-rating. In *Proceedings of the Third ACM Conference on Recommender Systems*, RecSys '09, pages 173–180, New York, NY, USA, 2009. ACM.

[65] Robin S. Poston and Cheri Speier. Effective use of knowledge management systems: A process model of content ratings and credibility indicators. *MIS Quarterly*, 29(2):pp. 221–244, 2005.

[66] P. Arbela andez, M. Maire, C. Fowlkes, and J. Malik. Contour detection and hierarchical image segmentation. *PAMI*, 33(5):898 –916, May 2011.

[67] Jianbo Shi and Jitendra Malik. Normalized Cuts and Image Segmentation. *PAMI*, 22(8):888–905, 2000.

[68] Dorin Comaniciu, Peter Meer, and Senior Member. Mean shift: A robust approach toward feature space analysis. *PAMI*, 24:603–619, 2002.

[69] Pedro F. Felzenszwalb and Daniel P. Huttenlocher. Efficient graph-based image segmentation. *IJCV*, 59:2004, 2004.

[70] T. Cour, F. Benezit, and J. Shi. Spectral segmentation with multiscale graph decomposition. In *CVPR*, 2005.

[71] Pushmeet Kohli, L'Ubor Ladický, and Philip H. Torr. Robust higher order potentials for enforcing label consistency. *IJCV*, 82(3):302–324, May 2009.

[72] Zhenguo Li, Xiao-Ming Wu, and Shih-Fu Chang. Segmentation using superpixels: A bipartite graph partitioning approach. In *CVPR*, 2012.

[73] Xiang Bai, Xingwei Yang, L. J. Latecki, Wenyu Liu, and Zhuowen Tu. Learning Context-Sensitive Shape Similarity by Graph Transduction. *PAMI*, 32(5):861–874, May 2010.

[74] Wang Bo and Zhuowen Tu. Affinity learning via self-diffusion for image segmentation and clustering. In *CVPR*, 2012.

[75] R. R. Coifman, S. Lafon, A. B. Lee, M. Maggioni, B. Nadler, F. Warner, and S. W. Zucker. Geometric diffusions as a tool for harmonic analysis and structure definition of data: Diffusion maps. *PNAS*, 102(21):7426–7431, May 2005.

[76] Risi Imre Kondor and John D. Lafferty. Diffusion kernels on graphs and other discrete input spaces. In *ICML*, 2002.

[77] Xingwei Yang and L. J. Latecki. Affinity learning on a tensor product graph with applications to shape and image retrieval. In *CVPR*, 2011.

[78] Tae Hoon Kim and Kyoung Mu Lee. Learning full pairwise affinities for spectral segmentation. In *CVPR*, 2010.

[79] Huaijun Qiu and Edwin R. Hancock. Clustering and embedding using commute times. *PAMI*, 29(11):1873–1890, 2007.

[80] Mengtian Sun, Yi Fang, and Karthik Ramani. Center-shift: An approach towards automatic robust mesh segmentation (arms). In *CVPR*, pages 630–637, 2012.

[81] Leo Grady and Eric L. Schwartz. Isoperimetric graph partitioning for image segmentation. *PAMI*, 28:469–475, 2006.

[82] Cem Keskin, Furkan Kıraç, Yunus Emre Kara, and Lale Akarun. Hand pose estimation and hand shape classification using multi-layered randomized decision forests. In *Computer Vision–ECCV 2012*, pages 852–863. Springer, 2012.

[83] Cem Keskin, Furkan Kıraç, Yunus Emre Kara, and Lale Akarun. Real time hand pose estimation using depth sensors. In *Consumer Depth Cameras for Computer Vision*, pages 119–137. Springer, 2013.

[84] Danhang Tang, Hyung Jin Chang, Alykhan Tejani, and Tae-Kyun Kim. Latent regression forest: Structured estimation of 3d articulated hand posture. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 3786–3793. IEEE, 2014.

[85] Xiao Sun, Yichen Wei, Shuang Liang, Xiaoou Tang, and Jian Sun. Cascaded hand pose regression. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 824–832, 2015.

[86] Iason Oikonomidis, Nikolaos Kyriazis, and Antonis A Argyros. Efficient model-based 3d tracking of hand articulations using kinect. In *BMVC*, volume 1, page 3, 2011.

[87] Stan Melax, Leonid Keselman, and Sterling Orsten. Dynamics based 3d skeletal hand tracking. In *Proceedings of Graphics Interface 2013*, pages 63–70. Canadian Information Processing Society, 2013.

[88] Chen Qian, Xiao Sun, Yichen Wei, Xiaoou Tang, and Jian Sun. Realtime and robust hand tracking from depth. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 1106–1113. IEEE, 2014.

[89] Toby Sharp, Cem Keskin, Duncan Robertson, Jonathan Taylor, Jamie Shotton, David Kim Christoph Rhemann Ido Leichter, Alon Vinnikov Yichen Wei, Daniel Freedman Pushmeet Kohli Eyal Krupka, Andrew Fitzgibbon, and Shahram Izadi. Accurate, robust, and flexible real-time hand tracking. In *Proc. CHI*, volume 8, 2015.

[90] Markus Oberweger, Paul Wohlhart, and Vincent Lepetit. Hands deep in deep learning for hand pose estimation. *arXiv preprint arXiv:1502.06807*, 2015.

[91] Chiho Choi, Ayan Sinha, Joon Hee Choi, Sujin Jang, and Karthik Ramani. A collaborative filtering approach to real-time hand pose estimation. In *Computer Vision (ICCV), 2015 IEEE International Conference on*.

[92] Julian Keilson. *Markov chain models-Rarity and Exponentiality*. Number 28 in Applied mathematical sciences. Springer, New York, USA, 1979.

[93] L. Lovász. Random walks on graphs: A survey. volume 2, pages 353–398. János Bolyai Mathematical Society, Budapest, 1996.

[94] F. R. K. Chung. *Spectral Graph Theory*. American Mathematical Society, 1997.

[95] J.G. Kemeny and J.L. Snell. *Finite Markov Chains*. Undergraduate Texts in Mathematics. Springer-Verlag, 1960.

[96] R. Olfati-Saber, J.A. Fax, and R.M. Murray. Consensus and cooperation in networked multi-agent systems. *Proceedings of the IEEE*, 95(1):215–233, Jan 2007.

[97] Alex Arenas, Albert Diaz-Guilera, and Conrad J. Pérez-Vicente. Synchronization reveals topological scales in complex networks. *Phys. Rev. Lett.*, 96:114102, Mar 2006.

[98] Fan Chung. The heat kernel as the pagerank of a graph. *Proceedings of the National Academy of Sciences*, 104(50):19735–19740, 2007.

[99] E. A. Leicht, Petter Holme, and M. E. J. Newman. Vertex similarity in networks. *Phys. Rev. E*, 73:026120, Feb 2006.

[100] G.F. Lawler. *Introduction to Stochastic Processes*. Chapman and Hall/CRC Probability Series. Chapman & Hall/CRC, 2006.

[101] Mark Meyer, Mathieu Desbrun, Peter Schröder, and Alan H. Barr. Discrete differential-geometry operators for triangulated 2-manifolds. In *Proc. VisMath*, pages 35–57, 2002.

[102] Ulrich Pinkall and Konrad Polthier. Computing discrete minimal surfaces and their conjugates. *Experimental Mathematics*, 2:15–36, 1993.

[103] Yanqing Hu, Menghui Li, Peng Zhang, Ying Fan, and Zengru Di. Community detection by signaling on complex networks. *Physical Review E*, 78:016115, Jul 2008.

[104] Fan R. K. Chung and S.-T. Yau. Discrete green's functions. *J. Comb. Theory, Ser. A*, 91(1-2):191–214, 2000.

[105] Julian Keilson. *Markov chain models-Rarity and Exponentiality*. Number 28 in Applied mathematical sciences. Springer, New York, USA, 1979.

[106] Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual web search engine. *Computer Networks*, 30(1-7):107–117, 1998.

[107] Michael M. Bronstein and Alexander M. Bronstein. Shape recognition with spectral distances. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(5):1065–1071, 2011.

[108] Xueyuan Zhou, Mikhail Belkin, and Nathan Srebro. An iterated graph laplacian approach for ranking on manifolds. In *ACM SIGKDD*, pages 877–885, 2011.

[109] Jian Sun, Maks Ovsjanikov, and Leonidas Guibas. A concise and provably informative multi-scale signature based on heat diffusion. In *SGP*, 2009.

[110] Leo Grady. Random walks for image segmentation. *PAMI*, 28(11):1768–1783, November 2006.

[111] David Harel and Yehuda Koren. On Clustering Using Random Walks. In *FST TCS*, pages 18–41. November 2001.

[112] M E Newman. Modularity and community structure in networks. *PNAS*, 103(23):8577–8582, June 2006.

[113] Marina Meila and Jianbo Shi. A Random Walks View of Spectral Segmentation. In *AIS*, 2001.

[114] Yaron Lipman, Raif Rustamov, and Thomas Funkhouser. Biharmonic distance. *ACM TOG*, 29(3), June 2010.

[115] F. R. K. Chung. *Spectral Graph Theory*. American Mathematical Society, 1997.

[116] Vincent D. Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 10:P10008+, October 2008.

[117] Xiaofeng Ren and Jitendra Malik. Learning a classification model for segmentation. In *ICCV*, 2003.

[118] David Liben-Nowell and Jon Kleinberg. The link-prediction problem for social networks. *Journal of the American Society for Information Science and Technology*, 58(7):1019–1031, 2007.

[119] Santo Fortunato and Marc Barthélemy. Resolution limit in community detection. *Proceedings of the National Academy of Sciences*, 104(1):36–41, January 2007.

[120] J. C. Delvenne, S. N. Yaliraki, and M. Barahona. Stability of graph communities across time scales. *Proceedings of the National Academy of Sciences*, 107(29):12755–12760, July 2010.

[121] Vincent D. Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10008+, July 2008.

[122] F. Luccio and M. Sami. On the decomposition of networks in minimally interconnected subnetworks. *Circuit Theory, IEEE Transactions on*, 16(2):184–188, May 1969.

[123] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer Series in Statistics. Springer New York Inc., New York, NY, USA, 2001.

[124] Albert-Lszl Barabsi and Rka Albert. Emergence of scaling in random networks. *Science*, 286(5439):509–512, 1999.

[125] Glen Jeh and Jennifer Widom. Scaling personalized web search. In *Proceedings of the 12th International Conference on World Wide Web*, WWW '03, pages 271–279, New York, NY, USA, 2003. ACM.

[126] Adrien Friggeri, Guillaume Chelius, and Eric Fleury. Egomunities, exploring socially cohesive person-based communities. *CoRR*, abs/1102.2623.

[127] M. E. J. Newman. Finding community structure in networks using the eigenvectors of matrices. *Physical Review E*, 74(3):036104+, July 2006.

[128] Jian Liu and Tingzhan Liu. Detecting community structure in complex networks using simulated annealing with k-means algorithms. *Physica A: Statistical Mechanics and its Applications*, 389(11):2300–2309, February 2010.

[129] M. E. J. Newman. Fast algorithm for detecting community structure in networks. *Physical Review E*, 69:066133, Jun 2004.

[130] Jörg Reichardt and Stefan Bornholdt. Statistical mechanics of community detection. *Phys. Rev. E*, 74:016110, Jul 2006.

[131] F. Y. Wu. Theory of resistor networks: the two-point resistance. *Journal of Physics A: Mathematical and General*, 37(26):6653–6673, July 2004.

[132] Nelson A. Alves. Unveiling community structures in weighted networks. *Physical Review E*, 76:036101, Sep 2007.

[133] P. Pons and M. Latapy. Computing communities in large networks using random walks. *J. Graph Algorithms Appl.*, 10(2):284–293, 2004.

[134] László Lovász. *Random Walks on Graphs: A Survey*, pages 353–397. 1993.

[135] Hua-Wei Shen. Community structure and diffusion dynamics on networks. In *Community Structure of Complex Networks*, Springer Theses, pages 73–92. Springer Berlin Heidelberg, 2013.

[136] Andrea Lancichinetti, Filippo Radicchi, José J. Ramasco, and Santo Fortunato. Finding Statistically Significant Communities in Networks. *PLoS ONE*, 6(5).

[137] Erzsébet Ravasz and Albert-László L. Barabási. Hierarchical organization in complex networks. *Physical review. E, Statistical, nonlinear, and soft matter physics*, 67(2 Pt 2), February 2003.

[138] Kyle Kloster and David F. Gleich. A nearly-sublinear method for approximating a column of the matrix exponential for matrices from large, sparse networks. In Anthony Bonato, Michael Mitzenmacher, and Pawe Praat, editors, *Algorithms and Models for the Web Graph*, volume 8305 of *Lecture Notes in Computer Science*, pages 68–79. Springer International Publishing, December 2013.

[139] M. E. J. Newman. Analysis of weighted networks. *Physical Review E*, 70(5):056131, November 2004.

[140] E. A. Leicht and M. E. J. Newman. Community structure in directed networks. *Phys. Rev. Lett.*, 100(11):118703, March 2008.

[141] V Nicosia, G Mangioni, V Carchiolo, and M Malgeri. Extending the definition of modularity to directed graphs with overlapping communities. *Journal of Statistical Mechanics: Theory and Experiment*, 2009(03):P03024, 2009.

[142] Peter J. Mucha, Thomas Richardson, Kevin Macon, Mason A. Porter, and Jukka-Pekka Onnela. Community structure in time-dependent, multiscale, and multiplex networks. *Science*, 328(5980):876–878, 2010.

[143] A Arenas, A Fernndez, and S Gmez. Analysis of the structure of complex networks at different resolution levels. *New Journal of Physics*, 10(5):053039, 2008.

[144] Jonathan W. Berry, Bruce Hendrickson, Randall A. LaViolette, and Cynthia A. Phillips. Tolerating the community detection resolution limit with edge weighting. *Physical Review E*, 83(5):056119+, May 2011.

[145] V. A. Traag, P. Van Dooren, and Y. Nesterov. Narrow scope for resolution-limit-free community detection. *Phys. Rev. E*, 84:016114, Jul 2011.

[146] Ulrik Brandes, Daniel Delling, Marco Gaertler, Robert G?rke, Martin Hoefer, Zoran Nikoloski, and Dorothea Wagner. On modularity clustering. *IEEE Transactions on Knowledge and Data Engineering*, 20(2):172–188, 2008.

[147] Michael T. Schaub, Jean-Charles Delvenne, Sophia N. Yaliraki, and Mauricio Barahona. Markov dynamics as a zooming lens for multiscale community detection: Non clique-like communities and the field-of-view limit. *PLoS ONE*, 7(2):e32210, 02 2012.

[148] Youngdo Kim, Seung-Woo Son, and Hawoong Jeong. Finding communities in directed networks. *Physical Review E*, 81:016103, Jan 2010.

[149] Michael S. Floater and Kai Hormann. Surface parameterization: a tutorial and survey. In *In Advances in Multiresolution for Geometric Modelling*, pages 157–186. Springer, 2005.

[150] M. Desbrun, M. Meyer, and P. Alliez. Intrinsic Parameterizations of Surface Meshes. *Computer Graphics Forum*, 21, 2002.

[151] Xianfeng Gu, Yalin Wang, Tony F. Chan, Paul M. Thompson, and Shing tung Yau. Genus zero surface conformal mapping and its application to brain surface mapping. *IEEE Transactions on Medical Imaging*, 23:949–958, 2004.

[152] Li Shen and Fillia Makedon. Spherical mapping for processing of 3d closed surfaces. *Image and Vision Computing*, 24(7):743 – 761, 2006.

[153] Xin Zhao, Zhengyu Su, Xianfeng David Gu, Arie Kaufman, Jian Sun, Jie Gao, and Feng Luo. Area-preservation mapping using optimal mass transport. *Visualization and Computer Graphics, IEEE Transactions on*, 19(12):2838–2847, 2013.

[154] Olga Sorkine and Marc Alexa. As-rigid-as-possible surface modeling. In *Proceedings of the Fifth Eurographics Symposium on Geometry Processing*, SGP '07, pages 109–116, Aire-la-Ville, Switzerland, Switzerland, 2007. Eurographics Association.

[155] Ayelet Dominitz and Allen Tannenbaum. Texture mapping via optimal mass transport. *IEEE Transactions on Visualization and Computer Graphics*, 16(3):419–433, May 2010.

[156] Justin Solomon, Fernando de Goes, Pixar Animation Studios, Gabriel Peyré, Marco Cuturi, Adrian Butscher, Andy Nguyen, Tao Du, and Leonidas Guibas. Convolutional wasserstein distances: Efficient optimal transportation on geometric domains. *ACM Transactions on Graphics (Proc. SIGGRAPH 2015), to appear*, 2015.

[157] Ilja Friedel, Peter Schrder, and Mathieu Desbrun. Unconstrained spherical parameterization. *J. Graphics Tools*, 12(1):17–26, 2007.

[158] Guangyu Zou, Jiaxi Hu, Xianfeng Gu, and Jing Hua. Authalic parameterization of general surfaces using lie advection. *Visualization and Computer Graphics, IEEE Transactions on*, 17(12):2005–2014, 2011.

[159] Craig Gotsman, Xianfeng Gu, and Alia Sheffer. Fundamentals of spherical parameterization for 3d meshes. In *PROCEEDINGS OF THE 2006 SYMPOSIUM ON INTERACTIVE 3D GRAPHICS AND GAMES, MARCH 1417, 2006*, pages 28–29, 2003.

[160] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. Poisson surface reconstruction. In *Proceedings of the Fourth Eurographics Symposium on Geometry Processing*, SGP '06, pages 61–70, Aire-la-Ville, Switzerland, Switzerland, 2006. Eurographics Association.

[161] Yizhou Yu, Kun Zhou, Dong Xu, Xiaohan Shi, Hujun Bao, Baining Guo, and Heung-Yeung Shum. Mesh editing with poisson-based gradient field manipulation. *ACM Trans. Graph.*, 23(3):644–651, August 2004.

[162] A. Sinha and K. Ramani. Multi-scale kernels using random walks. *Computer Graphics Forum*, 33(1):164–177, 2014.

[163] Raif M. Rustamov. Laplace-beltrami eigenfunctions for deformation invariant shape representation. In *Proceedings of the Fifth Eurographics Symposium on Geometry Processing*, SGP '07, pages 225–233, Aire-la-Ville, Switzerland, Switzerland, 2007. Eurographics Association.

[164] Jamie Shotton, Toby Sharp, Alex Kipman, Andrew Fitzgibbon, Mark Finocchio, Andrew Blake, Mat Cook, and Richard Moore. Real-time human pose recognition in parts from single depth images. *Communications of the ACM*, 56(1):116–124, 2013.

[165] Andrea Tagliasacchi, Matthias Schroeder, Anastasia Tkach, Sofien Bouaziz, Mario Botsch, and Mark Pauly. Robust articulated-icp for real-time hand tracking. Technical report, 2015.

[166] Jintae Lee and Tosiyasu L Kunii. Model-based analysis of hand posture. *Computer Graphics and Applications, IEEE*, 15(5):77–86, 1995.

[167] Hervé Jégou, Florent Perronnin, Matthijs Douze, Javier Sanchez, Pablo Perez, and Cordelia Schmid. Aggregating local image descriptors into compact codes. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 34(9):1704–1716, 2012.

[168] Art B Owen and Patrick O Perry. Bi-cross-validation of the SVD and the nonnegative matrix factorization. *The Annals of Applied Statistics*, pages 564–594, 2009.

[169] Gediminas Adomavicius and Alexander Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Trans. on Knowl. and Data Eng.*, 17(6):734–749, June 2005.

[170] Yifan Hu, Yehuda Koren, and Chris Volinsky. Collaborative filtering for implicit feedback datasets. In *Proceedings of the 2008 Eighth IEEE International Conference on Data Mining*, ICDM '08, pages 263–272, Washington, DC, USA, 2008. IEEE Computer Society.

[171] Gawesh Jawaheer, Martin Szomszor, and Patty Kostkova. Comparison of implicit and explicit feedback from an online music recommendation service. In *Proceedings of the 1st International Workshop on Information Heterogeneity and Fusion in Recommender Systems*, HetRec '10, pages 47–51, New York, NY, USA, 2010. ACM.

[172] Li Chen, Guanliang Chen, and Feng Wang. Recommender systems based on user reviews: the state of the art. *User Modeling and User-Adapted Interaction*, 25(2):99–154, 2015.

[173] Neal Lathia, Stephen Hailes, Licia Capra, and Xavier Amatriain. Temporal diversity in recommender systems. In *Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '10, pages 210–217, New York, NY, USA, 2010. ACM.

[174] James G. March. Exploration and exploitation in organizational learning. *Organization Science*, 2(1):pp. 71–87, 1991.

[175] Benjamin Edelman, Michael Ostrovsky, and Michael Schwarz. Internet advertising and the generalized second-price auction: Selling billions of dollars worth of keywords. *American Economic Review*, 97(1):242–259, 2007.

[176] Lihong Li, Wei Chu, John Langford, and Robert E. Schapire. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th International Conference on World Wide Web, WWW 2010, Raleigh, North Carolina, USA, April 26-30, 2010*, pages 661–670, 2010.

[177] Wei Li, Xuerui Wang, Ruofei Zhang, Ying Cui, Jianchang Mao, and Rong Jin. Exploitation and exploration in a performance based contextual advertising system. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '10, pages 27–36, New York, NY, USA, 2010. ACM.

[178] Léon Bottou, Jonas Peters, Joaquin Qui nonero Candela, Denis X. Charles, D. Max Chickering, Elon Portugaly, Dipankar Ray, Patrice Simard, and Ed Snelson. Counterfactual reasoning and learning systems: The example of computational advertising. *Journal of Machine Learning Research*, 14:3207–3260, 2013.

[179] Soheil Feizi, Daniel Marbach, Muriel Medard, and Manolis Kellis. Network deconvolution as a general method to distinguish direct dependencies in networks. *Nature Biotechnology*, 31(8):726–733, July 2013.

[180] Baruch Barzel and Albert-László Barabási. Network link prediction by global silencing of indirect correlations. *Nature biotechnology*, 31(8):720–725, 2013.

[181] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th International Conference on World Wide Web*, WWW '01, pages 285–295, New York, NY, USA, 2001. ACM.

[182] Roberta Sinatra, Jesús Gómez-Gardeñes, Renaud Lambiotte, Vincenzo Nicosia, and Vito Latora. Maximal-entropy random walks in complex networks with limited information. *Phys. Rev. E*, 83:030103, Mar 2011.

[183] Jarek Duda. From maximal entropy random walk to quantum thermodynamics. *Journal of Physics: Conference Series*, 361(1):012039, 2012.

[184] Alexander M. Bronstein. Spectral descriptors for deformable shapes. *CoRR*, abs/1110.5015, 2011.

[185] Bai Xiao, Edwin R. Hancock, and Richard C. Wilson. Graph characteristics from the heat kernel trace. *Pattern Recognition*, 42(11):2589–2606, 2009.

[186] Donald Shepard. A two-dimensional interpolation function for irregularly-spaced data. In *Proceedings of the* 23^{rd} *ACM national conference*, pages 517–524, New York, USA, 1968. ACM.

[187] Yining Deng and B. S. Manjunath. Unsupervised segmentation of color-texture regions in images and video. *PAMI*, 23(8):800–810, 2001.

[188] Jingdong Wang, Yangqing Jia, Xian-Sheng Hua, Changshui Zhang, and Long Quan. Normalized tree partitioning for image segmentation. In *CVPR*, 2008.

[189] Michael Donoser, Martin Urschler, Martin Hirzer, and Horst Bischof. Saliency driven total variation segmentation. In *ICCV*, 2009.

[190] L. M. Collins and C. W. Dent. Omega: A general formulation of the rand index of cluster recovery suitable for non-disjoint solutions. *Multivariate Behavioral Research*, 23(2):231–242, 1988.

[191] Wayne W. Zachary. An information flow model for conflict and fission in small groups. *Journal of Anthropological Research*, 33(4):pp. 452–473, 1977.

[192] Lada A. Adamic and Natalie Glance. The political blogosphere and the 2004 u.s. election: Divided they blog. In *Proceedings of the 3rd International Workshop on Link Discovery*, KDD '05, pages 36–43, New York, NY, USA, 2005. ACM.

[193] David Lusseau, Karsten Schneider, Oliver J Boisseau, Patti Haase, Elisabeth Slooten, and Steve M Dawson. The bottlenose dolphin community of doubtful sound features a large proportion of long-lasting associations. *Behavioral Ecology and Sociobiology*, 54(4):396–405, 2003.

[194] Judd H. LMichael and Joseph G. Massey. Modeling the communication network in a sawmill. *Forest Products Journal*, 47(9):p25, 1997.

[195] *This work uses data from Add Health, a program project designed by Udry J., Bearman S., and Harris, Kathleen Mullan, and funded by a grant P01-HD31921 from the National Institute of Child Health and Human Development, with cooperative funding from 17 other agencies. Special acknowledgment is due Ronald R. Rindfuss and Barbara Entwisle for assistance in the original design. Persons interested in obtaining data files from Add Health should contact Add Health, Carolina Population Center, 123 W. Franklin Street, Chapel Hill, NC 275162524 (addhealth@unc.edu).*

[196] Juliette Stehl, Nicolas Voirin, Alain Barrat, Ciro Cattuto, Lorenzo Isella, Jean-Franois Pinton, Marco Quaggiotto, Wouter Van den Broeck, Corinne Rgis, Bruno Lina, and Philippe Vanhems. High-resolution measurements of face-to-face contact patterns in a primary school. *PLoS ONE*, 6(8):e23176, 08 2011.

[197] H. Yu, P. Braun, M. A. Yildirim, I. Lemmens, K. Venkatesan, J. Sahalie, Hirozane T. Kishikawa, F. Gebreab, N. Li, N. Simonis, and Others. High-Quality Binary Protein Interaction Map of the Yeast Interactome Network. *Science*, 322(5898):104, 2008.

[198] Haiyuan Yu, Leah Tardivo, Stanley Tam, Evan Weiner, Fana Gebreab, Changyu Fan, Nenad Svrzikapa, Tomoko Hirozane-Kishikawa, Edward Rietman, Xinping Yang, Julie Sahalie, Kourosh Salehi-Ashtiani, Tong Hao, Michael E. Cusick, David E. Hill, Frederick P. Roth, Pascal Braun, and Marc Vidal. Next-generation sequencing to generate interactome datasets. *Nature methods*, 8(6):478–480, June 2011.

[199] Jan O. Korbel and Charles Lee. Genome assembly and haplotyping with Hi-C. *Nat Biotech*, 31(12):1099–1101, December 2013.

[200] Clement A. Stanyon, Guozhen Liu, Bernardo A. Mangiola, Nishi Patel, Loic Giot, Bing Kuang, Huamei Zhang, Jinhui Zhong, and Russell L. Finley. A Drosophila protein-interaction map centered on cell-cycle regulators. *Genome biology*, 5(12), 2004.

[201] *An information-theoretic definition of similarity*. Morgan Kaufmann, San Francisco, CA, 1998.

[202] Haixuan Yang, Tamás Nepusz, and Alberto Paccanaro. Improving go semantic similarity measures by exploring the ontology beneath the terms and modelling uncertainty. *Bioinformatics*, 28(10):1383–1389, May 2012.

[203] Amanda L. Traud, Eric D. Kelsic, Peter J. Mucha, and Mason A. Porter. Comparing community structure to characteristics in online collegiate social networks. *SIAM Rev.*, 53(3):526–543, August 2011.

[204] Steve Gregory. An algorithm to find overlapping community structure in networks. In JoostN. Kok, Jacek Koronacki, Ramon Lopez de Mantaras, Stan Matwin, Dunja Mladenic, and Andrzej Skowron, editors, *Knowledge Discovery in Databases: PKDD 2007*, volume 4702 of *Lecture Notes in Computer Science*, pages 91–102. Springer Berlin Heidelberg, 2007.

[205] James Folwer. Connecting the Congress: A Study of Cosponsorship Networks. *Political Analysis*, 14(4):456–487, 2006.

[206] D. L. Nelson, C. L. McEvoy, and T. A Schreiber. *The University of South Florida word association, rhyme, and word fragment norms*, 1998.

[207] Claudia Leacock, Martin Chodorow, and George A. Miller. Using Corpus Statistics and WordNet Relations for Sense Identification. *Association for Computational Linguistics*, 1998.

[208] E. A. Leicht, Petter Holme, and M. E. J. Newman. Vertex similarity in networks. *Phys. Rev. E*, 73:026120, Feb 2006.

[209] H. Laga, T. Schreck, A. Ferreira, A. Godil, I. Pratikakis (editors, Watertight Meshes, Z. Lian, A. Godil, B. Bustos, M. Daoudi, J. Hermans, S. Kawamura, Y. Kurita, G. Lavou, H. V. Nguyen, R. Ohbuchi, Y. Ohkita, Y. Ohishi, F. Porikli, M. Reuter, I. Sipiran, D. Smeets, P. Suetens, and H. Tabia. Shrec11 track: Shape retrieval on non-rigid 3d, 2011.

[210] Alexander M Bronstein, Michael M Bronstein, Leonidas J Guibas, and Maks Ovsjanikov. Shape google: Geometric words and expressions for invariant shape retrieval. *ACM Transactions on Graphics (TOG)*, 30(1):1, 2011.

[211] Marcin Novotni and Reinhard Klein. Shape retrieval using 3d zernike descriptors. *Computer Aided Design*, 36:1047–1062, 2004.

[212] Ding-Yun Chen, Xiao-Pei Tian, Yu-Te Shen, and Ming Ouhyoung. On visual similarity based 3d model retrieval. *Computer Graphics Forum*, 22(3):223–232, 2003.

[213] Srinath Sridhar, Antti Oulasvirta, and Christian Theobalt. Interactive markerless articulated hand motion tracking using rgb and depth data. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, December 2013.

[214] Srinath Sridhar, Franziska Mueller, Antti Oulasvirta, and Christian Theobalt. Fast and robust hand tracking using detection-guided optimization. In *Proceedings of Computer Vision and Pattern Recognition (CVPR)*, 2015.

[215] Srinath Sridhar, Helge Rhodin, Hans-Peter Seidel, Antti Oulasvirta, and Christian Theobalt. Real-time hand tracking using a sum of anisotropic gaussians model. In *Proceedings of the International Conference on 3D Vision (3DV)*, December 2014.

[216] David F. Gleich and Lek-heng Lim. Rank aggregation via nuclear norm minimization. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '11, pages 60–68, New York, NY, USA, 2011. ACM.

[217] Mukund Deshpande and George Karypis. Item-based top-n recommendation algorithms. *ACM Trans. Inf. Syst.*, 22(1):143–177, January 2004.

[218] Martin A. Fischler and Robert C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, June 1981.

[219] Julian John McAuley and Jure Leskovec. From amateurs to connoisseurs: Modeling the evolution of user expertise through online reviews. In *Proceedings of the 22nd International Conference on World Wide Web*, WWW '13, pages 897–908. International World Wide Web Conferences Steering Committee, 2013.

[220] J. Bennett and S. Lanning. The netflix prize. In *Proceedings of the KDD Cup Workshop 2007*, pages 3–6, New York, August 2007. ACM.

[221] Ken Goldberg, Theresa Roeder, Dhruv Gupta, and Chris Perkins. Eigentaste: A constant time collaborative filtering algorithm. *Inf. Retr.*, 4(2):133–151, July 2001.

[222] Max Wardetzky, Saurabh Mathur, Felix Kälberer, and Eitan Grinspun. Discrete laplace operators: no free lunch. In *SGP*, pages 33–37, Barcelona, Spain, 2007. Eurographics.

VITA

VITA

My research aims to answer the most pressing problems of “Big Data” by combining physical laws and machine learning. I have developed fast and reliable algorithms to uncover the underlying patterns in the data. The ‘secret sauce’ in these techniques is inspired by physical laws and processes like heat flow, Gauss’s law and biological neural networks applied to graphs. To test the robustness, I have implemented these techniques to a wide variety of domains by representing data points and their pairwise relationships as a graph structure without rebuilding them for each domain. These techniques provide superior results in general pattern recognition objectives such as to retrieve similar shapes to a query from a large database of geometric mesh models, track and reconstruct 3D objects in real-time, identify the hierarchical and overlapping organization of networks, and prevent false positives in recommender systems.