Assessing a Nonlinear Dimensionality Reduction-Based Approach to Biological Network Reconstruction.

Vinodh N. Rajapakse – vinodh@math.umd.edu
PhD Advisor: Professor Wojciech Czaja – wojtek@math.umd.edu

ABSTRACT

Deeper understanding of biological processes requires a systems perspective, integrating knowledge of both component elements (genes, proteins, metabolites, etc.), and their broader interactions. Graph theory provides a rich language for describing and analyzing these sorts of complex, networked systems. But, reliable inference of biological network structure from noisy, high-dimensional (and still fundamentally limited) data remains a challenging problem. A promising avenue involves integrating nonlinear dimensionality reduction (DR) approaches with network analysis and reconstruction algorithms. Beyond basic dimensionality reduction, these approaches can potentially accentuate latent structure and organize data in ways that allow integration of diverse information sources, including prior knowledge. Software tools for performing elements of this analysis exist, but they are distributed over many packages, which often do not interact easily together.

The objective of this project was to build a basic, integrated data analysis pipeline for deriving gene association network models from gene expression data. Dimensionality reduction techniques were applied to map the input data in ways that aim to capture intrinsic structure. After this step, standard network reconstruction and analysis techniques were applied. Algorithm implementations were individually validated using well-characterized data sets and established software. Following this foundational work, the impact of dimensionality reduction on the overall network reconstruction was systematically assessed using additional validation and testing data sets. This report details work in the areas described above, and additionally presents results comparing the developed nonlinear dimensionality reduction-based approach with a leading network reconstruction algorithm. In particular, recovery of the first-neighbor network of the well-studied oncogene MYC was analyzed and compared with results obtained over the same data set using the ARACNE algorithm [1]. The developed approach, based on the Laplacian Eigenmaps technique, yielded a local network around MYC with substantial enrichment for biochemically validated MYC direct targets (23 of 61 matches to a research database). This performance did lag that of the ARACNE method, which yielded 24 of 56 matches, as well as candidate associations that were subsequently biochemically validated. Still, the results can be considered somewhat promising, in view of the avenues for improving the relatively basic approach. The report concludes with discussion of ongoing work in these areas.

INTRODUCTION

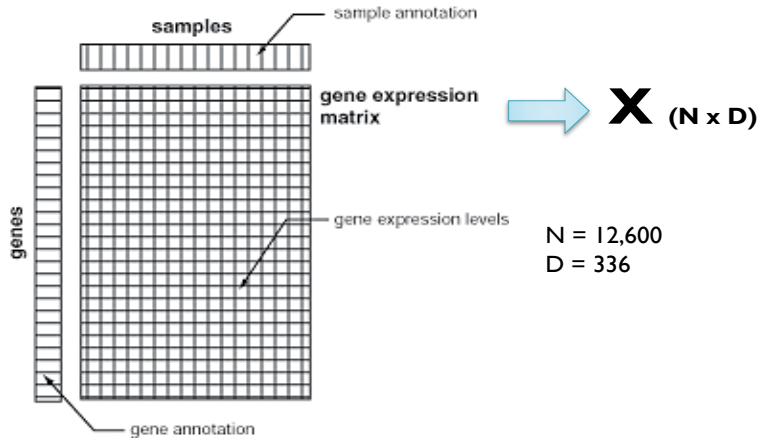

Figure 1: The basic structure of a gene expression data set. Each column of the data matrix is derived from particular biological sample (a specific experiment, patient sample, etc.). Each row captures gene expression values across the samples, and can be seen as a point in D-dimensional space. The matrix dimensions indicated are for the main experimental data set considered in this study.

The starting point for this analysis is an N x D gene expression data matrix, which we will denote by **X**. The entry $(X)_{ij}$ in **X** records the expression (relative abundance) of the ith gene (mRNA product) in the jth sample. In what follows, we will focus on gene expression profiles across samples, which are naturally organized along the rows of **X**. The expression profile of the ith gene will be designated by $\mathbf{x}_i$, and will be regarded as a point in a D-dimensional input data space. From this input data, network reconstruction will proceed in three broad steps:

(1) Starting from the N x D gene expression data matrix **X**, derive an N x d matrix **Y**, (d < D) using Laplacian Eigenmaps (or another dimensionality reduction technique).
(2) Construct an N x N matrix **W*** capturing pairwise Euclidean distances between row vectors of **Y** (which can be regarded as 'reduced gene profiles').
(3) Apply a (smallest) distance-based threshold to the elements of **W*** to obtain a network (adjacency matrix) representation.

Laplacian Eigenmaps Background

The motivation for integrating nonlinear dimensionality reduction with network reconstruction and analysis derives from the thought that gene expression data sets might be relatively constrained to lower dimensional manifolds within the high-dimensional spaces in which they reside. This notion is not implausible given the highly ordered structure of gene regulatory networks and their associated processes. Laplacian Eigenmaps (LE) aims to map input space points to a lower dimensional space in such a way that local relationships are preserved (2). These mapped space distances might approximate distances along a putative data manifold in the original space. Some initial studies have suggested that their application might support more biologically specific clustering of gene expression profiles (3). These results motivate the more thorough consideration of manifold learning-based nonlinear dimensionality reduction in support of detailed network structure recovery.

Operationally, Laplacian Eigenmaps transforms an input (N x D) data matrix (data element vectors organized along rows) to an output (N x d) matrix (d < D). There are three steps, though the first two can be combined computationally.

(1) Model Data Point Relationships: Build a graph G with nodes *i* and *j* connected if $\mathbf{x_i}$ is one of the k nearest neighbors of $\mathbf{x_j}$ or vice versa. (Euclidean distances are used, though alternatives are possible.) k is a local structure resolution parameter.
(2) Construct Weight Matrix: Form a diffusion weight matrix **W**, with entry $(W)_{ij} = \exp\{ - || \mathbf{x_i} - \mathbf{x_j} ||^2/\sigma \}$ if i and j are connected; 0 otherwise.
(3) Solve the Minimization Problem:

$$\min_{(Y^T DY=I)} \frac{1}{2} \sum_{i,j} \| y_i - y_j \|^2 W_{i,j} = \min_{(Y^T DY=I)} \operatorname{trace}(Y^T LY),$$

In the above formulation, **D** is a diagonal 'connectivity matrix', whose entries record the sum of the edge weights for each data point-derived node (as recorded along the rows or columns of **W**). **L** = **D** – **W** is the Laplacian matrix. *The key idea with the minimization is to force points that are close together in the original data space (e.g., that have larger associated edge weights) to lie close together in the mapped data space, while more loosely related points can be relatively pushed apart, drawing out (potentially complex, nonlinear) structure in the data.* The specified constraint serves to scale the output and eliminate trivial solutions. Some basic results from linear algebra show that the optimal mapping can be obtained by solving the generalized eigenvalue problem $\mathbf{Lx} = \lambda\mathbf{Dx}$ under the above constraint. In particular, the coordinates of the mapped data point $\mathbf{y_i}$ in the space of reduced dimension *d* can be extracted from the ith coordinates of the *d* eigenvectors with the smallest nonzero eigenvalues (2).

The Laplacian Eigenmaps method has two major strengths. First, the data-organizing map can be obtained by solving a standard and reasonably tractable computational problem. In addition, the approach comes with deeper theoretical assurances of optimal manifold recovery, in the limit of sufficient data. In particular, the graph-based Laplacian matrix L can be seen as a discrete analogue of the Laplacian-Beltrami operator on the underlying manifold. The eigenmaps of the latter operator can be shown to provide an optimal embedding of the manifold into a space of reduced, intrinsic dimension. Since the graph-based Laplacian converges to the manifold-based Laplace-Beltrami operator, its associated data mappings progressively inherit the corresponding manifold recovery guarantees (2).

Derivation of Network Structure

Given an N x N matrix **W*** capturing pairwise Euclidean distances between mapped gene expression profiles *in the reduced d-dimensional space*, and a target (fractional) value α, a distance threshold for deriving a network adjacency matrix from **W*** will be obtained by applying the following steps.

- Rank pairwise distances between the derived points in the mapped data space.
- Select the distance threshold that excludes the upper ($\mathbf{1 - \alpha}$) fraction of observed distance values.

While this basic approach is reasonably motivated, and can be used to advance development and testing of the larger analysis pipeline, more sophisticated approaches for network derivation are possible. In particular, a more statistically motivated approach involving edge identification based on estimated false discovery rates and associated q-values (analogous in this context to p-values) can perhaps be developed and applied in future work (13). This will require better understanding of the probability model underlying the gene expression data.

Additional Details

Eigenvalue problem formulation: Note that we have:

$\mathbf{Lx} = \lambda\, \mathbf{Dx} \quad \Leftrightarrow \quad (\mathbf{D}^{-1/2}\, \mathbf{L}\, \mathbf{D}^{-1/2})\mathbf{u} = \lambda\, \mathbf{u}$, where $\mathbf{u} = \mathbf{D}^{-1/2}\mathbf{x}$, $(\mathbf{D}^{-1/2}\, \mathbf{L}\, \mathbf{D}^{-1/2}) = \mathbf{L}_{\mathbf{sym}}$

As such, we can always obtain a solution to the described generalized eigenvalue problem by solving a standard eigenvalue problem. This can be useful with numerical packages (e.g., SciPy) that only support the latter.

Laplacian Eigenmaps parameter selection: Three parameters must be selected in the course of constructing the data map using Laplacian Eigenmaps – the target reduced dimension d, the neighborhood resolution parameter NN, and the 'kernel width' parameter σ. While in some instances, it may be possible to select the target dimension based on knowledge of the biological process under investigation, for this project a more data-driven approach was applied. In particular, the maximum of two estimates was taken to be a coarse estimate of the intrinsic dimensionality (with the maximum taken to err on the side of preserving potentially valuable information). The first was the number of principal components required to capture 95% of the variance in the data. The second was an established maximum likelihood estimate of the intrinsic dimensionality (8).

Operationally, the computing the maximum likelihood estimate (MLE) entails averaging many local estimates of the intrinsic dimensionality to obtain an overall estimate. In particular, let $T_k(x)$ denote the Euclidean distance from a fixed point x to its kth nearest neighbor in the data set of size N. For a particular k, we construct the following local (with respect to point x_i) estimate of the intrinsic dimensionality d:

$$\widehat{d_k}(x_i) = \left[\frac{1}{k-1} \sum_{j=1}^{k-1} \log \frac{T_k(x_i)}{T_j(x_i)} \right]^{-1}$$

The average of these local estimates over the N points of the data set gives a k-neighborhood overall estimate $\widehat{d_k}$. A final estimate $\hat{d}$ is obtained by averaging the preceding k-neighborhood estimates over several values of k (e.g., $k = 6, \ldots 12$, per example of MATLAB DR Toolbox).

The motivation for this approach is presented in detail in (8). In essence, we are assuming that our D-dimensional data is obtained by a smooth mapping from a space of lower dimension d (perhaps with modest distortion due to noise, etc.). In this scheme, our specific data set can be seen ultimately as sampled from some unknown density f on $\mathbb{R}^d$. The idea is now to (1) fix a point x and assume that $f(x)$ is approximately constant over a small sphere $S_x(R)$ of radius R about x and (2) to regard the count of neighbors falling within the described neighborhood as a Poisson process. Let $n(t,x)$, $(0 \le t \le R)$, indicate the inhomogeneous process which counts observations falling within a distance t of x over N 'trials' corresponding to the number of points in the data set. Applying the Poisson approximation for this binomial process, we can express the rate $\lambda(t)$ of the process $n(t)$ (given x) as $\lambda(t) = f(x)V(d)(d)t^{d-1}$, where $V(d)$ is the volume of the unit sphere in $\mathbb{R}^d$. Setting $\theta = \log f(x)$, expressing the log-likelihood $L(d,\theta)$, and solving in the standard way gives the following MLE for d:

$$\hat{d}_R(x) = \left[\frac{1}{N(R,x)} \sum_{j=1}^{N(R,x)} \log \frac{R}{T_j(x)} \right]^{-1}$$

The initially presented local estimate is then derived (with respect to point x_i) from the above one by fixing the number of neighbors k, rather than the radius of the sphere R.

Past work has yielded good results with neighborhood resolution parameter settings in the range of NN = 8 to NN = 20, which are along the lines of the values applied to example data sets in the original Laplacian Eigenmaps paper (2). For the simpler synthetic data sets used in algorithm validation, values in this range were applied, while fixing the kernel width parameter at one. With the larger experimental data set used for the main integrated system testing, a more heuristic approach was taken. The idea was to sample a grid of reasonable settings for the neighborhood resolution and kernel bandwidth parameters. It is known that gene expression data sets possess intrinsic cluster structure deriving from the correlated expression of co-regulated genes. The idea was then to select parameters that maximized cluster structure in the mapped data spaces, which eigenmap techniques should potentially enhance. Clustering was done by an implementation of the k-means algorithm. Cluster structure was then assessed using the silhouette coefficient, a frequently applied measure of intra-cluster cohesion and inter-cluster separation. Further details are provided in the results section.

SOFTWARE IMPLEMENTATION

The main software development accomplishments for this project include:

- Implementation and focused validation of a C++ implementation of Laplacian Eigenmaps.
- Implementation of basic (distance matrix threshold-based) network reconstruction.
- Implementation of code to analyze network models and validated their components against databases of experimentally validated biological knowledge.

The key challenges faced were (1) selecting and integrating appropriate public domain software to efficiently solve structured (sparse, symmetric) eigenvalue problems and (2) organizing data structures and operations to conserve memory and support scalability.

Linear Algebra Libraries

A longer-term objective is to integrate elements of code developed for this project into the National Cancer Institute's Cancer Bioinformatics Grid, which aims to provide an open-source platform for computational cancer biology. This motivated the selection of public domain libraries for all software development undertaken in this project. The C++ standard libraries, together with widely used Boost C++ packages have been more than adequate for basic program development. A somewhat more challenging choice came with linear algebra libraries. There seems to be a division in which most higher-level numerical code development and prototyping is done using environments like MATLAB or Python/SciPy, while code optimized for speed and resource management utilizes basic Fortran libraries like BLAS and LAPACK. As a result, there appear to be relatively fewer well-developed, public domain C/C++ libraries with broad numerical support. The Intel Math Kernel Library provides much of the latter, but remains a commercial product. Several C++ libraries for linear algebra were identified and tested, but all were found to be limited in various ways. Some, like TNT and LAPACK++, did not seem to be very widely used or actively maintained. Others like uBLAS, Eigen and Armadillo++ were more current, and even offered convenient high-level programming interfaces, but lacked adequate support for sparse matrices and associated eigenvalue problems.

Emerging technology platforms are certain to generate much larger data sets than the ones under immediate consideration in this project. Since scalability to accommodate these data sets is a major objective of this work, it seemed important to build upon tools that exploited sparsity and structure in basic underlying numerical problems. To achieve this, the ARPACK (Fortran 77) package for large-scale sparse eigenvalue problems was ultimately applied. ARPACK remains the leading software for these sorts of problems, and is often utilized at the base of commercial packages like MATLAB. For symmetric matrices, ARPACK applies a variant of the Lanczos algorithm, and is highly stable and resource efficient. In particular, for sufficiently sparse matrices, the applied algorithm is linear over each iteration and requires $O(kn)$ memory (where k is the number of desired eigenvalues and n is the order of the problem matrix) (14).

Interface Development and Resource Management

Although ARPACK provided precisely the functionality required to implement Laplacian Eigenmaps efficiently, its integration with C++ code was somewhat challenging. Some helpful guidance was offered by some publicly available code (15). Still, the latter only provided a relatively thin, low-level wrapper to basic ARPACK routines, with a host of detailed and sometimes obscure parameters exposed. To provide a more convenient programming interface, a higher-level, object-oriented wrapper was implemented. As the detailed implementation requirements for Laplacian Eigenmaps were better understood, this grew to include basic linear algebra operations over resource-efficient, compressed matrix representations. A particularly attractive feature of ARPACK made all of this possible. In particular, most ARPACK eigensolver routines do not require explicit passage of matrices and other basic problem elements. Instead, a so-called 'reverse communication interface' is utilized, where routines simply require a function providing the action of the matrix on an arbitrary vector. Matrices can be stored in any suitable format (or not stored at all), as long as the matrix vector product is properly provided.

For this project, two compact sparse matrix representations were developed. For sparse matrices with dynamically varying content, an adjacency list-based representation was provided. This is used in the early steps of the Laplacian Eigenmaps algorithm, when data point neighborhood-based weight matrices are constructed. For fixed-content sparse matrices, an even more compact array-based representation is implemented. This utilizes a relatively standard compressed row format, where one array records all non-zero elements in row-wise sequence, a second array indicates their column indices, and finally a third array records the index (in the above arrays) of the first non-zero element in a given matrix row. Matrix vector products over both representations are provided. But, the array-based format is ultimately used to store the Laplacian matrix in the Laplacian Eigenmaps implementation. For modest problem sizes, there is likely little difference. But the array-based format is a bit more compact, not requiring storage for pointers, as in the linked-list-based adjacency list representation. For larger problem sizes, the repeatedly applied matrix vector product (computed during eigendecomposition) is likely to be (observably) faster with the array-based representation, since more of the contiguously stored matrix entries will fit in any given cache level, reducing memory traffic.

In addition to the described use of resource-efficient data structures, care was also taken in the implementation to perform operations 'in place' when possible, without gratuitously allocating new memory for intermediate operations if this could be avoided. Finally, to facilitate future algorithm implementations, some convenient refinements to the programming interface were added. For example, C++'s operator overloading features were applied to allow use of natural syntax for basic (compressed) matrix and matrix vector operations. Accordingly, one can naturally set elements of a matrix `A` using `A(i,j)`, or write `A*x` to multiply by a vector. C++ templates and general function overloading ensure that the correct operation is transparently performed over a range of suitable

types and data structures. Underneath, low-level, pointer-based data structures are accessed and manipulated for efficiency, with destructor methods carefully implemented to de-allocate memory. To provide a sense for the interface, a developed wrapper method for solving a sparse, symmetric eigenvalue problem is contrasted below with the original C-level interface to the ARPACK (Fortran) routine:

```
template<typename T>
void sparseSymEigSolve(const CompressedMatrix<T>& M,
    const Matrix<T>& evecs, const NumVector<T>& evals);

extern "C" void dsaupd_(int *ido, char *bmat, int *n, char
*which, int *nev, double *tol, double *resid, int *ncv, double
*v, int *ldv, int *iparam, int *ipntr, double *workd, double
*workl, int *lworkl, int *info);
```

Implementation Validation

The developed Laplacian Eigenmaps implementation was validated by thorough comparison with established MATLAB methods. First, the sparse, symmetric eigensolver method presented above was compared with the MATLAB method 'eigs', which also applies the indicated ARPACK routine. Over a variety of randomly generated test matrices, results aligned up to occasional (global) differences in sign in computed eigenvectors. After this, the overall Laplacian Eigenmaps implementation was tested against an implementation provided by the MATLAB-based Dimensionality Reduction Toolbox, which has been extensively used by our research group and others (11). The toolbox provides code for generating synthetic data sets containing points sampled from relatively simple nonlinear manifolds in three-dimensional space ('Swiss Roll', 'Broken Swiss Roll', 'Twinpeaks', and 'Helix'). These are often applied in the research literature for initial assessment of nonlinear dimensionality reduction techniques. Data sets with up to 10000 points were generated in each case, and mappings produced by the developed Laplacian Eigenmaps implementation matched those produced by the MATLAB one up to 4 to 5 significant digits. The relatively small variation might derive from differences between detailed eigensolver parameters applied by the MATLAB eigs method and the corresponding developed method. These may be further investigated, though the results seem to adequately validate the developed implementation. Conveniently, the same general approach can be used to validate future implementations of related dimensionality reduction techniques (e.g., Diffusion Maps), which are also provided by the MATLAB DR Toolbox.

A note on performance – although the initial testing focused on correctness, it was noted that the solution of the basic eigenvalue problem was comparable with the corresponding MATLAB eigs method. The overall Laplacian Eigenmaps implementation was slower – most notably with larger data sets (> 4000 points). On investigation, this seems to derive largely from the MATLAB implementation's use of a heuristic, approximate nearest neighbor search method (as compared to the brute-force direct search currently implemented). Integration of comparable approaches in extensions of the project could readily close this performance gap, though a current priority remains understanding the nature and fidelity of network structure recovery.

In addition to the Laplacian Eigenmaps implementation, several other software components were developed for this project. These include elements of the parameter tuning approach, such as the maximum likelihood dimensionality estimator, and the k-means clustering algorithm. In addition, codes for deriving the network model (by distance ranking and thresholding) were implemented in C++, together with a set of Python scripts for validation of gene expression network models against a biological (gene) database.

RESULTS

Overview

After careful validation of the Laplacian Eigenmaps (LE) implementation, the main work of the project involved detailed consideration of network reconstructions derived using LE-processed data. This integrated system testing proceeded in two stages. First, reconstruction of networks derived from simulation of relatively simple artificial gene networks was assessed. The results here can be scored objectively according to recovery of model-prescribed edges. Second, results over a well-studied experimental gene expression data set were analyzed. Here, the focus was on resolution of a biologically plausible first-neighbor network module around the well-known oncogene *MYC*. Since many genes belonging to the latter group are known through extensive experimental work, the results can once again be scored with relative objectivity (for what is still a real-world experimental data set). For perspective, the results obtained were compared with those published for the leading ARACNE network reconstruction algorithm (1,9). In addition to the main comparison with this leading network reconstruction technique, some additional comparisons were made with respect to more basic network reconstructions (derived from the same *MYC*-focused data set). These included one obtained using a 'vanilla version' of the reconstruction workflow (which forgoes dimensionality reduction), as well as one applying the standard principal component analysis (PCA) technique for linear dimensionality reduction. Taken together, the results provide an initial assessment of where nonlinear dimensionality reduction approaches may have particular value in accurately recovering complex biological network structures.

Synthetic Network Reconstruction

Ordinary differential equation (ODE)-based models have been used to simulate the kinetics of relatively simple genetic networks structured by a specified topology, i.e. – organization of activating and inhibiting interactions. Synthetic data is often generated by simulating a series of 'null-mutant' experiments, where individual genes are deleted one at a time, and steady-state data is gathered. This is intended to mimic a 'gene knock-out' experiment, in which genetic techniques are used to inactivate a particular gene, and its function is imputed through analysis of the resulting perturbations. For this work, a set of models provided by a leading research group (16) was used as a starting point for generating synthetic data using the COPASI biochemical network simulator (17). In particular, 4 data sets were generated using the above-described null-mutant simulation approach. Each modeled a 100 gene by 200 interaction network. The specific input data to the analysis workflow was a 100 x 101 expression data matrix. Gene expression vectors are organized along rows, while columns capture simulated steady-state expression values from the original (unperturbed) model, as well as from successive inactivation of each of the 100 genes. A representative network topology is presented on the following page.

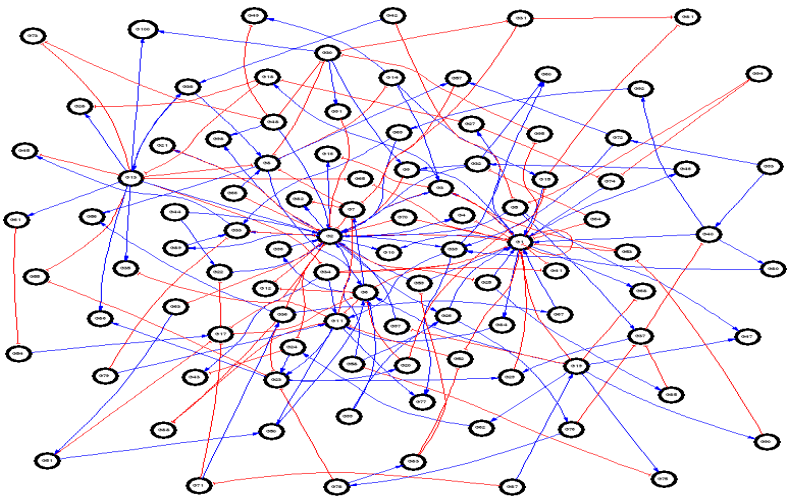

Figure 2: A representative simulated network topology. Positive or activating interactions are in blue, while negative or inhibitory interactions are in red. A roughly scale-free topology is evident, with most nodes participating in only a few interactions, while a few hubs drive many genes.

The objective with the simulated data was to recover the positive interactions. The average accuracy values (true positives over all positives) over 4 data sets are indicated in the table below.

Original Data	PCA-Mapped	LE-Mapped
0.46	0.47	0.44

While the results are far from spectacular, with some consideration, they are not surprising. The network reconstruction approach applied is relatively direct, being based on a global distance threshold. Such methods will invariably apply many extra edges in simple prescribed networks such as the ones considered. This is because associations among co-regulated genes (which are often activated in near lockstep) will appear stronger (e.g, more correlated) than the ones with their regulators. Indeed, consideration of the Laplacian Eigenmaps-derived results, in particular, shows many edges added between tightly co-regulated genes. Although this clearly diminishes the global accuracy, it is still a broad validation of the data-organizing feature of the eigenmap technique, as closely related (e.g., co-activated) gene expression profiles are 'squeezed together' in the mapped space. More successful methods for global network reconstruction invariably apply more complex strategies. One approach is to admit a relatively large pool of candidate edges, and then apply mathematical criteria and/or prior knowledge to prune edges less likely to derive from direct interactions. Other techniques aim to fit the parameters of mechanistic (ODE) models thought to underlie the data, though this is currently tractable only with the simplest gene networks, typically in microbial organisms.

Recovery of a MYC Oncogene First Neighbor Network

The preceding discussion motivates consideration of a related but somewhat different problem: local network structure recovery. With larger, more complex gene networks (such as those operating in mammalian cells), accurate recovery of global network structure is somewhat unrealistic, given the relatively limited data available. In this setting, it is still quite valuable to consider recovery of interactions around a known network 'hub', or gene participating in many regulatory interactions. For this study, we considered the well-known oncogene *MYC*. *MYC* is the most frequently mutated

or otherwise deregulated gene in human cancers, largely due to its role as a 'regulator of regulators'. Hundreds of direct *MYC* targets have been identified, though far less is known about the much smaller number of interactions present in any given cellular context. Basic goals for computational techniques in this setting are to (1) suggest which targets are engaged with *MYC* in a given cell type or disease context and (2) to prioritize the most promising targets for more labor-intensive laboratory investigation. Relative to the simulated or comparably simple microbial gene networks, there are many additional challenges, largely deriving from the far more complex interaction patterns, and much larger, noisier data sets. At the same time, there is a relatively lower accuracy threshold relative to still limited validation data. Three to four-fold enrichment around a known hub gene (with respect to experimentally validated targets) can be a solid, useful computational result – at least a starting point for more detailed gene selection and investigation.

To assess the value of the nonlinear dimensionality reduction-based approach, we applied the described workflow to recover a local, first-neighbor network around the *MYC* oncogene. Results are derived from a 12,600 x 336 expression data set described in publications detailing the ARACNE network reconstruction method (1,9). They are additionally checked against an established research database of experimentally validated *MYC* direct targets described in the same publications, and accessible at http://www.myccancergene.org/. For a point of comparison, we begin by summarizing the published results derived from the ARACNE method. The latter recovers 56 direct targets around MYC, 24 of which are matched in the described target database (validated fraction = 42.9%).

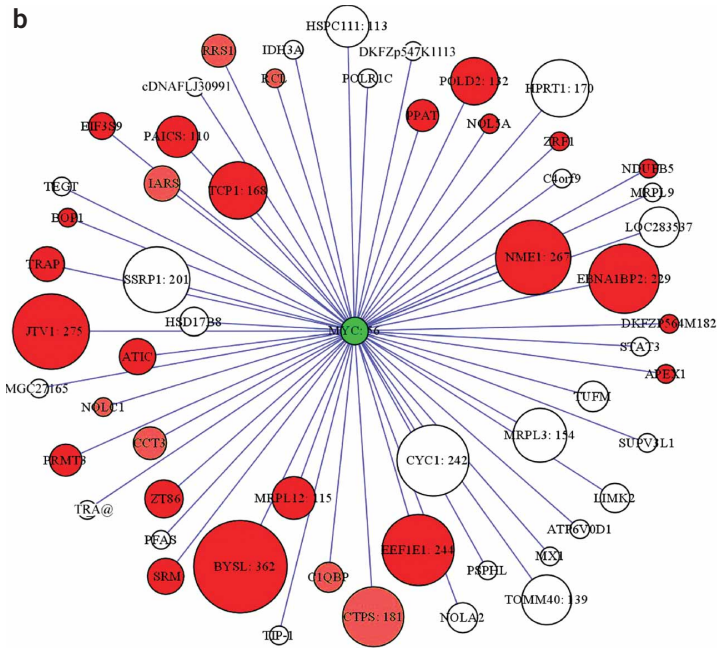

Figure 3: 56 member MYC direct-target network recovered by ARACNE algorithm, reproduced from (1). 24 matched an experimental database, and 5 others were biochemically validated. (Combined set is shown in red.)

While the fraction of validated genes may not seem very high, it represents a substantial enrichment over the approximately 10% fraction expected by chance. This is substantial in this challenging setting for a purely computational method. The results inspire some confidence that a biologically meaningful interaction module has been recovered. This sense was further strengthened by the biochemical validation of 5 novel candidates selected from the identified set of 56 direct targets (1).

A comparable *MYC* direct target network was derived using the Laplacian Eigenmaps-based workflow described in this report. This network produced 61 direct MYC interactions, of which 23 matched the experimental database, yielding a validated fraction of 37.7%. While this is less than the 42.9% fraction derived using the ARACNE method, it still represents a substantial enrichment suggestive of a biologically meaningful result. In addition, the extracted network was obtained by a relatively simple thresholding to construct edges corresponding to the smallest 1% of pairwise distances (between gene expression vectors in the mapped data space). The leading ARACNE method also applies a thresholding strategy, but it (1) uses a somewhat more sophisticated mutual information-based interaction measure and (2) prunes edges using the data processing inequality, an information theoretic criterion (1,9). It seems plausible that integration of alternative distance measures and edge-filtering strategies could improve the results obtained.

It is additionally noteworthy that there was relatively little overlap between the first neighbor networks derived using the ARACNE method and the LE-based method – just 5 genes overall, and 3 genes among the group matching the target database. This does not discredit the LE-based method, as the ARACNE developers note that their algorithm does not aim to recover all possible *MYC* interactions, but rather, a set substantially enriched for direct targets (1). As such, the LE-based method, with its still relatively strong enrichment for validated genes, could be drawing together new and relatively independent biological information. Further work, both computational, and ultimately experimental, is clearly needed to confirm this.

With this summary of the results, we now present some additional details relating to the derivation of the described LE-based network model. The first step in the process was estimating the intrinsic dimensionality of the explicitly 336-dimensional data set. While 14 principal components were sufficient to capture 95% of the data variance, a maximum likelihood estimate (8) suggested a target dimensionality of 28. To err on the side of preserving potentially essential information, this larger value was selected and fixed. The next step was to select the neighborhood resolution (NN) and kernel bandwidth (σ) parameters. Based on past experience with comparable expression data sets, a small candidate parameter grid was constructed, with NN = 8, 12, 16, 20 and σ = ½, 1. Laplacian Eigenmaps was run with each of the 8 parameter combinations. K-means clustering was then run with the mapped data, with the aim of selecting parameters yielding the best cluster structure. In each case, the cluster number was set to 15 and 30, reflecting approximate upper and lower bounds for the expected number of clusters (once again, reflecting past experience with gene expression data sets). Cluster structure was assessed using the maximum value of the average silhouette coefficient measure over 10 runs of the k-means algorithm. The latter metric ranges between -1 and 1, with values closer to 1 indicating more cohesive and well-separated clusters. The associated cluster quality measures varied over a relatively small range (0.37 to 0.44). The maximum value was obtained with the parameter selection NN = 16, σ = ½, and this value was selected to generate the presented results. In particular, a pairwise distance matrix was constructed using the eigenmap data associated with the above parameter selection. A network adjacency matrix was then derived by selecting the smallest 1% of observed pairwise distances. From this larger network, the described 61-member direct target network for *MYC* was extracted.

The 1% pairwise-distance threshold was selected to approximate a relatively stringent p-value of 0.01, and also to obtain a first neighbor network around *MYC* of approximately the same size the one derived using the ARACNE method. An exact p-value was not determined, due to the absence of a reasonable model for the process generating the expression data. For comparison, it is worth considering the size and quality (database validated fraction) of *MYC* first neighbor networks derived from alternative (reasonably stringent) distance thresholds. This information is presented in the figure below. As can be noted, the quality of the first neighbor network gradually declines as the distance threshold is increased.

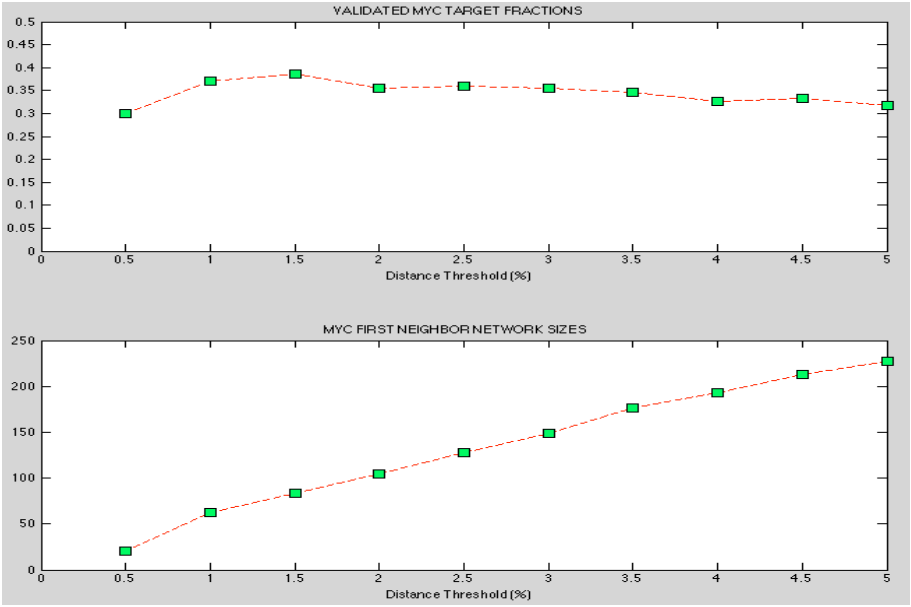

Figure 4: Size and direct target database-validated fractions for *MYC* first neighbor networks as a function of distance threshold.

Beyond these main results, a few additional comparisons were made. In particular, distance thresholded networks were derived using the original expression data and PCA-reduced data (top 14 principal components, capturing 95% of the total data variance). These approaches did not yield strong results. In particular, the following table captures the distance thresholds required to recover just a few (<5) edges around the MYC gene with the original and PCA-mapped data:

LE-Mapped	PCA-Mapped	Original Data
1%	> 40%	> 50%

Unreasonably high thresholds are clearly required to recover any structure around a major network hub. By comparison, the Laplacian Eigenmaps procedure appears to produce more meaningful distances over local network neighborhoods. Ongoing work to further validate and perhaps improve these results is detailed in the final section of the report.

DISCUSSION

There are two ongoing avenues for improving and further validating the nonlinear dimensionality reduction-based network reconstruction approach presented in this report. The first is to try and improve the recovery of local network structure by incorporating additional measures of gene association – including perhaps the mutual information measure applied by the ARACNE method. There is considerable flexibility in the derivation of the kernel matrix used in Laplacian Eigenmaps and related methods. Kernels derived from different association measures can even be combined. There may be value in this sort of 'information fusion' approach, especially given the way that the ARACNE and LE-based methods seemed to recover somewhat distinct structures, each with biological plausibility. Another approach for strengthening the results might involve network edge filtering, with edges ideally being pruned in a more adaptive manner, based on local network structure. If measures like correlation are applied, partial correlation calculations can potentially be used to isolate more direct gene-target interactions.

A second general avenue is to use additional computational techniques to prioritize genes in a constructed local network for further investigation. Candidate gene sets can be 'scored' for biological coherence using statistical analysis of the functional annotations associated with most genes (18). This can additionally allow further validation of network models, by enabling analysis of local structure around other known hubs. The ultimate aim is to identify potentially novel interactions for experimental investigation. In the context of cancer-related genes like *MYC*, these interactions could yield new insights into tumor progression, and in particular, its critically important variation across patients. Further development is clearly required to build a computational tool capable of focusing experimental work in this manner. The work undertaken in this project suggests some promising directions for ongoing exploration.

REFERENCES

1. K. Basso, A. Margolin, G. Stolovitzky, U. Klein, R. Dalla-Favera, A. Califano *Reverse engineering of regulatory networks in human B cells* Nature Genetics 37, 382 - 390 (2005)
2. M. Belkin and P. Niyogi, *Laplacian Eigenmaps for dimensionality reduction and data representation, Neural Computation.* 15 (2004), No. 6, 1373-1396
3. M. Ehler, V. Rajapakse, B. Zeeberg, B. Brooks, J. Brown, W. Czaja, and R. F. Bonner, *Analysis of temporal-spatial co-variation within gene expression microarray data in an organogenesis model.* 6th International Symposium on Bioinformatics Research and Applications (ISBRA'10), Lecture Notes in Bioinformatics, Springer Verlag, 2010
4. J. Chen, H. Fang, Y. Saad *Fast Approximate kNN Graph Construction for High Dimensional Data via Recursive Lanczos Bisection* Journal of Machine Learning Research 10 (2009) 1989-2012
5. RR Coifman, S Lafon, A Lee, M Maggioni, B Nadler, FJ Warner, and SW Zucker.*Geometric diffusions as a tool for harmonic analysis and structure definition of data. part i: Diffusion maps* , Proc. of Nat. Acad. Sci., 102:7426--7431, May 2005.
6. Kaskie S, Nikkila J, Oja M, Venna J, Törönen P, Castrén E. *Trustworthiness and metrics in visualizing similarity of gene expression.* BMC Bioinformatics. 2003 Oct 13;4:48.
7. R. Karbauskaite, O. Kurasova, G. Dzemyda *Selection of the number of neighbors of each data point for the locally linear embedding algorithm.* Information Technology and Control, 2007, Vol. 36, No. 4
8. E. Levina and P.J. Bickel. *Maximum likelihood estimation of intrinsic dimension.* In Advances in Neural Information Processing Systems, volume 17, Cambridge, MA, USA, 2004. The MIT Press.
9. A. A. Margolin, I Nemenman, K. Basso, C. Wiggins, G. Stolovitzky, R Dalla Favera, A. Califano, *ARACNE: an algorithm for the reconstruction of gene regulatory networks in a mammalian cellular context.* BMC Bioinformatics. 2006 Mar 20;7 Suppl 1:S7.
10. van der Maaten, Postma, van den Herik, *Dimensionality Reduction: A Comparative Review.* Tilburg Centre for Creative Computing Technical Report 2009-005
11. http://homepage.tudelft.nl/19j49/Matlab_Toolbox_for_Dimensionality_Reduction.html
12. http://igraph.sourceforge.net/
13. J. R. White, M. Pop *Statistical methods for detecting differentially abundant taxa in metagenomic samples.* University of Maryland AMSC 664 Advanced Scientific Computing Project Report, 2008.
14. http://www.caam.rice.edu/software/ARPACK/UG/ug.html
15. http://www-heller.harvard.edu/people/shaw/programs/lapack.html
16. http://www.comp-sys-bio.org/AGN/data.html
17. http://www.copasi.org/
18. http://discover.nci.nih.gov/gominer/