Assessing a Nonlinear Dimensionality Reduction-Based Approach to Biological Network Reconstruction. Interim Project Status Report

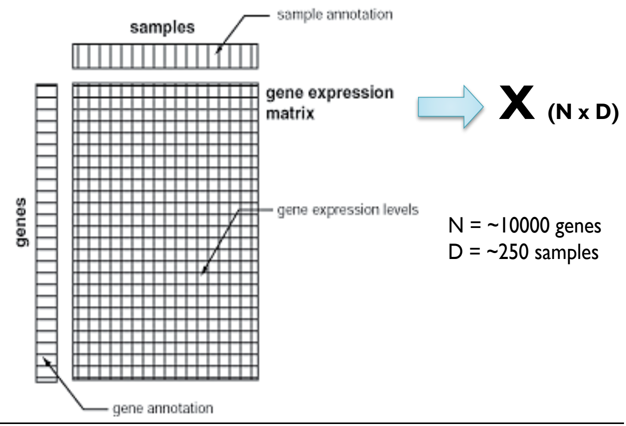
Vinodh N. Rajapakse – vinodh@math.umd.edu
PhD Advisor: Professor Wojciech Czaja – wojtek@math.umd.edu

Overview

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 is to build a basic, integrated data analysis pipeline for deriving gene association network models from gene expression data. Dimensionality reduction techniques will be applied to map the input data in ways that aim to capture intrinsic structure. After this step, standard network reconstruction and analysis techniques will be applied. Algorithm implementations will be individually validated using well-characterized data sets and established software. Following this foundational work, the impact of dimensionality reduction on the overall network reconstruction will be systematically assessed using additional validation and testing data sets. This report will detail progress in the areas described above, together with an overview of upcoming work to complete the remaining project objectives. Some accessible extensions of the basic framework will also be considered.

Detailed Approach



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 statistical significance-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 will be used at least initially; 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, $\mathbf{D}$ 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 $\mathbf{W}$). $\mathbf{L} = \mathbf{D} - \mathbf{W}$ 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 *i*th 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 $\mathbf{W^*}$ 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 $\mathbf{W^*}$ 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) is being investigated, and can likely be developed and applied (13).

Additional Details and Possible Extensions

Eigenvalue problem formulation: Note that we have:

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

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 k, 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 an established, data-driven maximum likelihood technique will be used to estimate the intrinsic data dimensionality (8). Past work with gene expression data sets of the size considered in this project has yielded good results with neighborhood resolution parameter settings in range of k = 10 to k = 20, which are along the lines of the values applied to example data sets in the original Laplacian Eigenmaps paper (2). For the first phase of the project, values in this range will be used and assessed. Following successful validation and testing of the core algorithm and basic framework, more refined approaches may be implemented and assessed (7). The kernel width parameter is typically 1, and other parameters are tuned from this starting point. The research literature provides relatively less guidance in this area. If time permits, some data-sensitive heuristics being considered by our research group for selecting the σ parameter can be applied. Generally speaking, parameter tuning is essential to proper application of Laplacian Eigenmaps and related nonlinear dimensionality reduction techniques, with estimation of the target dimensionality and neighborhood resolution being especially important.

Likely Extensions: With successful implementation, validation, and testing of the basic Laplacian Eigenmaps-based network reconstruction approach outlined above, some extensions are likely in the upcoming phase of the project. In particular, approximate nearest neighbor selection algorithms could be incorporated and assessed. These will facilitate handling of much larger data sets by expediting the construction of the initial nearest neighbor-restricted weight matrix utilized by the Laplacian Eigenmaps method (and several related nonlinear dimensionality reduction methods). One possible approach with an available library is described in the reference (4). In addition, another dimensionality reduction technique could be implemented and assessed in context of network reconstruction. The Diffusion Maps technique, which can also be reduced to an eigenvalue problem, would be a strong candidate (5).

Accomplished Objectives

The objectives for the first phase of this project have been met. These include:

- Implementation and focused validation of a C++ implementation of Laplacian Eigenmaps.
- Implementation of basic (distance matrix threshold-based) network reconstruction.

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 will be investigated over the inter-term period, 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 final 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 the second phase of the project should close this performance gap.

<u>Next Steps</u>

With validation of the basic Laplacian Eigenmaps algorithm implementation completed, the next steps will involve detailed validation of derived network representations. In particular, these will be assessed by comparison with published results obtained using the leading ARACNE algorithm, which has been featured in several strong publications (1, 9). In particular, two levels of integrated system testing will be performed. First, the reconstruction results obtained using data derived from simulation of a realistic artificial gene network will be assessed (9). The results here can be precisely and objectively scored according to recovery of known edges. Second, results over a well-studied biological data set focusing on the cancer gene *myc* will be considered. Here, proper resolution of a coherent network module around *myc* will be assessed, along with its elements. Since many genes belonging to the latter group are known through extensive past experimental work, the results can be once again scored with relative objectivity (for what is still a real-world experimental data set). In particular, results can be compared with those published for the leading ARACNE network reconstruction algorithm (1,9).

In addition to comparisons with the above leading network reconstruction technique, we will build and compare networks derived using a 'vanilla version' of the reconstruction workflow (which forgoes dimensionality reduction), as well as one which incorporates dimensionality reduction based on the more basic, linear Principal Components Analysis technique. The overall results should provide an initial assessment of whether nonlinear dimensionality reduction approaches have particular value in accurately recovering complex biological network structures.

<u>Remaining Schedule and Milestones</u>

- Phase II: Integrated Testing of Network Reconstruction + Possible Extensions
    - Target Date: end of March 2011
    - Milestones:
        - Integrated testing of network reconstruction, method comparisons
        - Approximate Nearest Neighbor Selection Algorithms
        - Possible: Enhanced tuning of Laplacian Eigenmaps algorithm parameters
        - Possible: Implementation and integrated testing of alternative dimensionality reduction techniques (Diffusion Maps)

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