

Neuronal Classification from Network Connectivity

Rebecca E. Goldin (Mathematical Sciences, George Mason University),
Paul M. Salomonsky (Naval Surface Warfare Center, Dahlgren),
Carey E. Priebe (Applied Mathematics and Statistics, Johns Hopkins University),
David J. Marchette (Naval Surface Warfare Center, Dahlgren),
Giorgio A. Ascoli (Krasnow Institute, George Mason University)

The Big Questions

- Given a large amount of data on neuronal connectivity, can we use statistical properties of these data to determine neuronal classes? (Everyone agrees they exist)
- How do any of these findings drive experimental efforts to collect this data: for example, can we estimate how many or what percentage of neuronal connections in a rat hippocampus are required in order to make an estimate about the entire hippocampus?

The numbers are huge: 10^{11} neurons in the human brain; with a conservative estimate on connectivity (1%), this amounts to 10^{20} connections

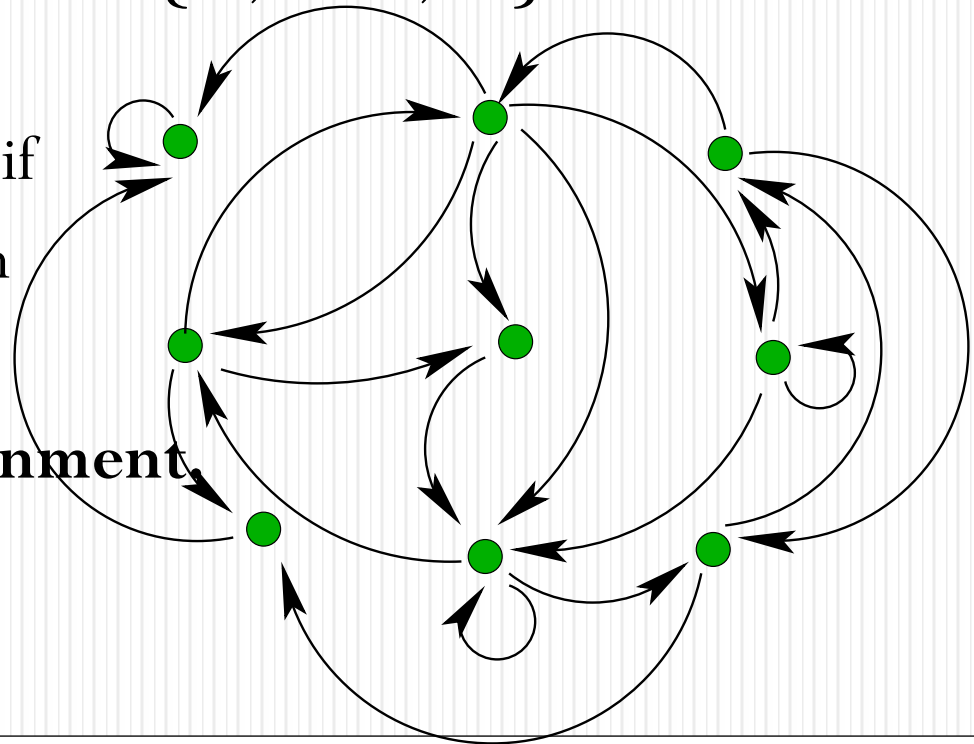
Premise: Neurons are connected according to a directed graph

- A directed graph has *vertices* and *directed edges* representing neurons and axonal-dendritic communication.
- Definition: A **brain** is a directed graph (V, E) and a map

$$\tau : V \longrightarrow \{1, \dots, k\}$$

We say that (V, E, τ) is **equivalent** to (V, E, τ') if there exists a permutation σ such that $\sigma\tau' = \tau$.

We call τ a **block assignment**.



Some Mathematical Assumptions

















- An axon either connects, or does not connect, to a dendrite. We do not keep track of number of connections, or any properties of synapses between them
- Neurons can be classified as a first approximation according to their connectivity behavior, rather than other properties (such as morphology).
- Classification forms a partition of the neurons; each neuron is in exactly one neuronal class

If the data were in...

What would we do with it?

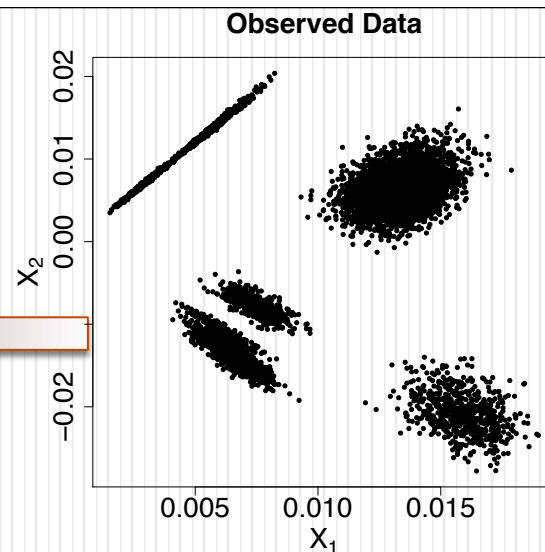
Put the data into a large adjacency matrix. Each neuron corresponds to one row and one column. A 0 or a 1 is placed in the ij^{th} entry, according to whether the axon in the i^{th} row connects to the dendrite in the j^{th} column.

$$\begin{pmatrix} 0 & 0 & \mathbf{1} & 0 & 0 & \dots & \dots & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \dots & \dots & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \dots & \dots & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 \\ \mathbf{1} & 0 & 0 & 0 & 0 & \dots & \dots & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{1} & \mathbf{1} \\ 0 & 0 & 0 & 0 & 0 & \dots & \dots & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \mathbf{1} & 0 & & & & & & & & & & \\ 0 & 0 & 0 & 0 & 0 & & & & & & & & & & \\ \mathbf{1} & 0 & 0 & 0 & 0 & & & & & & & & & & \\ 0 & 0 & 0 & 0 & 0 & & & & & & & & & & \\ \vdots & \vdots & \vdots & \vdots & \vdots & & & & & & & & & & \\ \vdots & \vdots & \vdots & \vdots & \vdots & & & & & & & & & & \\ 0 & 0 & 0 & 0 & 0 & & & & & & & & & & \\ 0 & 0 & 0 & 0 & 0 & & & & & & & & & & \\ 0 & 0 & \mathbf{1} & 0 & 0 & & & & & & & & & & \\ 0 & 0 & 0 & 0 & 0 & & & & & & & & & & \end{pmatrix}$$

P								
	0.02	0.02	0.006666667	0.00	0.02	0.04	0.04	0.02
	0.02	0.00	0.006666667	0.02	0.00	0.00	0.00	0.00
	0.02	0.00	0.006666667	0.00	0.00	0.00	0.00	0.00
	0.02	0.00	0.006666667	0.02	0.00	0.00	0.00	0.00
	0.02	0.02	0.006666667	0.00	0.02	0.00	0.00	0.00
	0.00	0.00	0.00	0.00	0.00	0.04	0.04	0.02
	0.04	0.00	0.013333333	0.04	0.00	0.02	0.02	0.01
	0.00	0.00	0.00	0.00	0.00	0.02	0.02	0.01

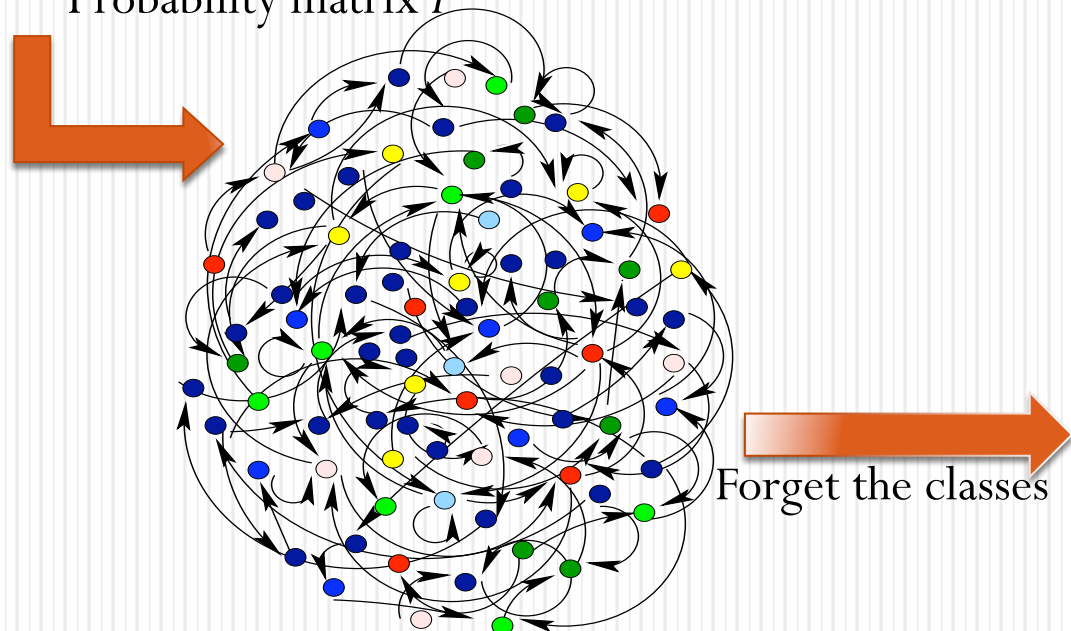
Probability matrix P

Cluster analysis using BIC



Projections of first coordinates of data

SVD top dimension



A "hippocampus" with classes sampled from P

Forget the classes

0	0	1	0	0	0	0	0	1	0	0	0	0
0	0	0	0	0	0	0	1	0	0	0	0	0
0	0	0	0	0	0	0	1	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	1	1	
0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	0										
0	0	0	0	0										
1	0	0	0	0										
0	0	0	0	0										
...										
...										
0	0	0	0	0										
0	0	0	0	0										
0	0	1	0	0										
0	0	0	0	0										

Approximately
1.6% nonzero

Random Dot Product Model

We consider simple random graphs with directed edges: $G=(V, E, X, Y)$, where we assume:

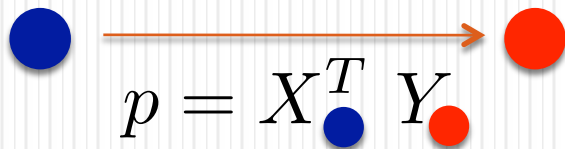
- $|V| = n, E \subset V \times V - \{(v, v)\}$
- $X_u, Y_u \in \Delta^d$ a d simplex. Could choose these in a ball if we want.
- The vertices and edges are observed perfectly
- The attributes (X, Y) are latent (unobserved)
- The edge probabilities are given by

$$P[u \rightarrow v \in E] = X_u^T Y_v$$

Constrained RDPG Model

Constrain the model to have k distinct pairs of vectors, one for each neural group. In other words,

$$\tau(u) = \tau(v) \text{ implies } X_u = X_v \text{ and } Y_u = Y_v,$$


$$p = X^T Y$$

To fit the model,

- Determine the number k of classes
- Partition the neurons according to class membership (up to permutation)
- Assign to each group an in- and out- vector X and Y , respectively.

Analyze the data









Use *singular value decomposition* to identify the most important dimensions of the data.

Use *cluster analysis* to identify the classes of neurons according to these most important dimensions.

















Simulation Design

Producing simulated data

















Neuronal types (secret)

	15768 CA1 Pyramidal Cells: Principal output neurons of the hippocampus. Excitatory. Constitute one of the most studied and best characterized neuron types in the brain.
	4000 CA1 Oriens/Lacunosum-Moleculare Cells: local inhibitory neurons. Dendrites are in the oriens layer and their axons start in the oriens and go up to Lacunosum-Moleculare .
	1000 CA1 Basket Cells: local peri-somatic inhibitory interneurons. Axons target pyramidal and basket cells. Their dendrites span all layers of CA1.
	3000 CA1 Perforant Pathway-Associated Cells: local inhibitory interneurons with dendrites and axons confined to the Lacunosum-Moleculare layer.
	2000 CA1 Oriens Cells: Local inhibitory interneurons. Dendrites and axons confined to the oriens layer.
	2500 Entorhinal Cortex Layer 5 Pyramidal Cells: play the role of deep layer 'input' neurons. They are excitatory and have dendrites and axons through the deep and superficial layers of the entorhinal cortex.
	2500 Entorhinal Cortex Layer 3 Pyramidal Cells: One of the superficial excitatory layer 'output' neurons. Dendrites through the deep and superficial layers of the EC. Axons starting in layer 3, projecting to CA1LM.
	2000 Entorhinal Cortex GABAergic Cells: Inhibitory local interneurons of the EC, with axons and dendrites through the deep and superficial layers of the entorhinal cortex.

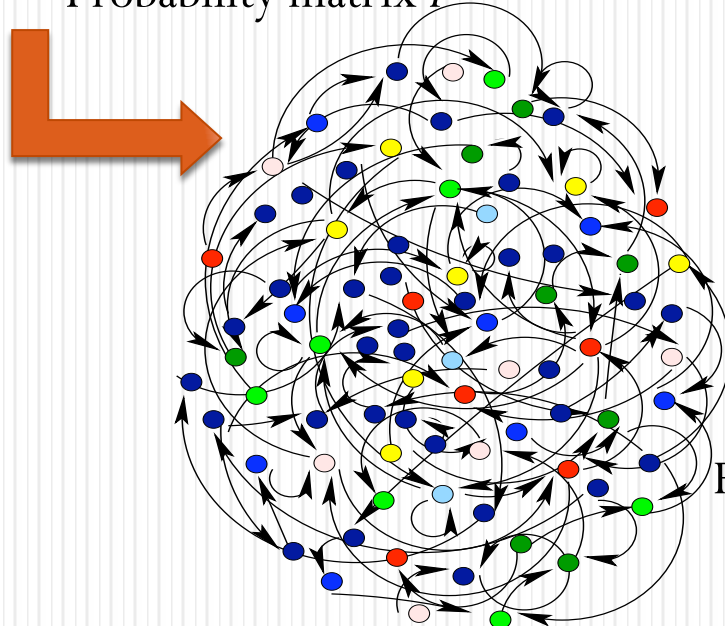
Generating matrix (secret)

<i>P</i>								
	0.02	0.02	$\frac{0.00666}{6667}$	0.00	0.02	0.04	0.04	0.02
	0.02	0.00	$\frac{0.00666}{6667}$	0.02	0.00	0.00	0.00	0.00
	0.02	0.00	$\frac{0.00666}{6667}$	0.00	0.00	0.00	0.00	0.00
	0.02	0.00	$\frac{0.00666}{6667}$	0.02	0.00	0.00	0.00	0.00
	0.02	0.02	$\frac{0.00666}{6667}$	0.00	0.02	0.00	0.00	0.00
	0.00	0.00	0.00	0.00	0.00	0.04	0.04	0.02
	0.04	0.00	$\frac{0.01333}{33333}$	0.04	0.00	0.02	0.02	0.01
	0.00	0.00	0.00	0.00	0.00	0.02	0.02	0.01

$$v = (15768, 4000, 1000, 3000, 2000, 2500, 2500, 2000)$$

P								
	0.02	0.02	$\frac{0.00666}{6667}$	0.00	0.02	0.04	0.04	0.02
	0.02	0.00	$\frac{0.00666}{6667}$	0.02	0.00	0.00	0.00	0.00
	0.02	0.00	$\frac{0.00666}{6667}$	0.00	0.00	0.00	0.00	0.00
	0.02	0.00	$\frac{0.00666}{6667}$	0.02	0.00	0.00	0.00	0.00
	0.02	0.02	$\frac{0.00666}{6667}$	0.00	0.02	0.00	0.00	0.00
	0.00	0.00	0.00	0.00	0.00	0.04	0.04	0.02
	0.04	0.00	$\frac{0.01333}{33333}$	0.04	0.00	0.02	0.02	0.01
	0.00	0.00	0.00	0.00	0.00	0.02	0.02	0.01

Probability matrix P



A “hippocampus” with classes sampled from P

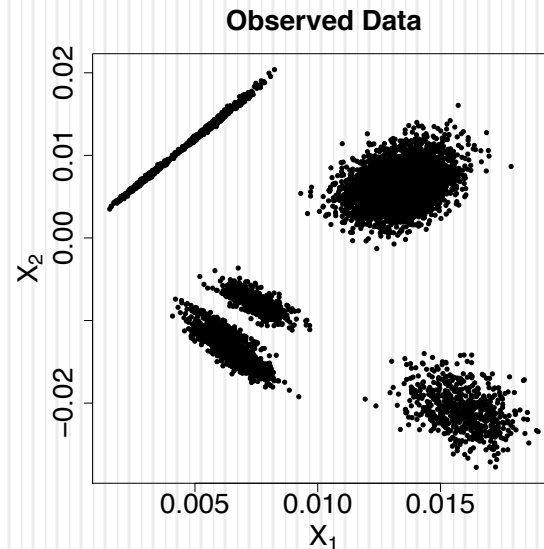
Generate Graphs

- For various sizes n of the number of neurons, generate graphs according to a specified proportion of vertices in each class.
- From such a graph, simply write down the adjacency matrix.

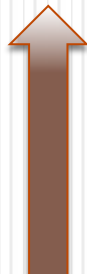
Forget the classes

$$\begin{pmatrix} 0 & 0 & 1 & 0 & 0 & \dots & \dots & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \dots & \dots & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \dots & \dots & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & \dots & \dots & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & \dots & \dots & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & & & & & & & & & & \\ 0 & 0 & 0 & 0 & 0 & & & & & & & & & & \\ 1 & 0 & 0 & 0 & 0 & & & & & & & & & & \\ 0 & 0 & 0 & 0 & 0 & & & & & & & & & & \\ \vdots & \vdots & \vdots & \vdots & \vdots & & & & & & & & & & \\ \vdots & \vdots & \vdots & \vdots & \vdots & & & & & & & & & & \\ 0 & 0 & 0 & 0 & 0 & & & & & & & & & & \\ 0 & 0 & 0 & 0 & 0 & & & & & & & & & & \\ 0 & 0 & 1 & 0 & 0 & & & & & & & & & & \\ 0 & 0 & 0 & 0 & 0 & & & & & & & & & & \end{pmatrix}$$

Approximately
1.6% nonzero



Projections of first coordinates of data



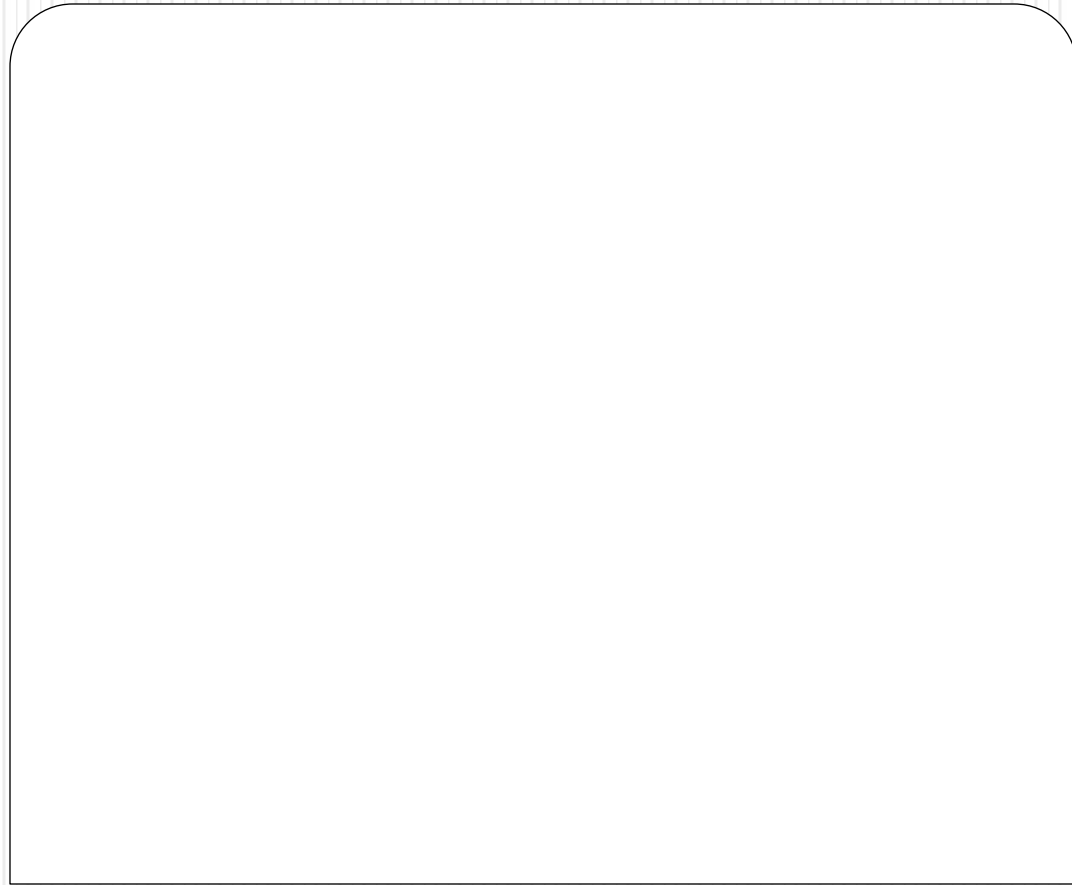
SVD top dimension

$$\begin{pmatrix} 0 & 0 & \mathbf{1} & 0 & 0 & \dots & \dots & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \dots & \dots & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \dots & \dots & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 \\ \mathbf{1} & 0 & 0 & 0 & 0 & \dots & \dots & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{1} & \mathbf{1} \\ 0 & 0 & 0 & 0 & 0 & \dots & \dots & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \mathbf{1} & 0 & & & & & & & & & & \\ 0 & 0 & 0 & 0 & 0 & & & & & & & & & & \\ \mathbf{1} & 0 & 0 & 0 & 0 & & & & & & & & & & \\ 0 & 0 & 0 & 0 & 0 & & & & & & & & & & \\ \vdots & \vdots & \vdots & \vdots & \vdots & & & & & & & & & & \\ \vdots & \vdots & \vdots & \vdots & \vdots & & & & & & & & & & \\ 0 & 0 & 0 & 0 & 0 & & & & & & & & & & \\ 0 & 0 & 0 & 0 & 0 & & & & & & & & & & \\ 0 & 0 & \mathbf{1} & 0 & 0 & & & & & & & & & & \\ 0 & 0 & 0 & 0 & 0 & & & & & & & & & & \end{pmatrix}$$

Approximately
1.6% nonzero

Singular Value Decomposition

We take the adjacency matrix A and we change the diagonal slightly. Use Singular Value Decomposition to break this matrix into a unitary times a diagonal times a unitary matrix.



Singular Value Decomposition in 2 Dimensions

Thanks to Wikimedia!

The adjacency matrix is approximately equal to the product:

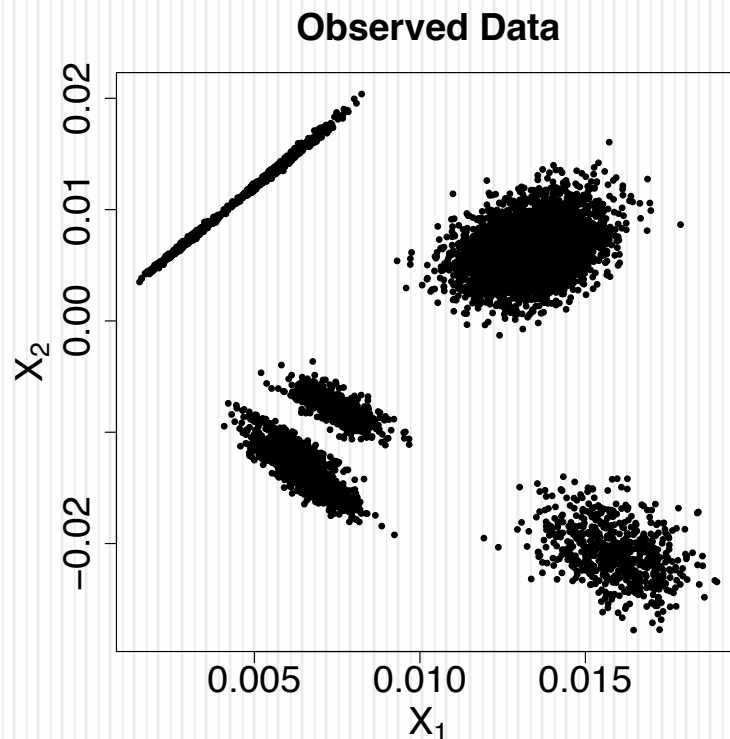
$$\begin{pmatrix}
 u_{11} & u_{12} & u_{13} & u_{14} \\
 u_{21} & u_{22} & u_{23} & u_{24} \\
 \vdots & \vdots & \vdots & \vdots \\
 \vdots & \vdots & \vdots & \vdots \\
 u_{n-1,1} & u_{n-1,2} & u_{n-1,3} & u_{n-1,4} \\
 u_{n1} & u_{n2} & u_{n3} & u_{n4}
 \end{pmatrix}
 \begin{pmatrix}
 s_1 & 0 & 0 & 0 \\
 0 & s_2 & 0 & 0 \\
 0 & 0 & s_3 & 0 \\
 0 & 0 & 0 & s_4
 \end{pmatrix}
 \begin{pmatrix}
 v_{11} & v_{12} & \cdots & v_{1,n-1} & v_{1n} \\
 v_{21} & v_{22} & \cdots & v_{2,n-1} & v_{2n} \\
 v_{31} & v_{32} & \cdots & v_{3,n-1} & v_{3n} \\
 v_{41} & v_{42} & \cdots & v_{4,n-1} & v_{4n}
 \end{pmatrix}$$

Singular Value Decomposition

Break a matrix down into its most important components. The size of the matrix in the middle is a parameter; we did this for varying size of the matrix in the middle. The larger the dimension, the better an approximation we get.

On the embeddings (SVD)

Before clustering, it may “look” like obvious clusters



- As the dimension of data recorded increases, the closer we get to A .
- As the dimension of data goes up, the clustering we do subsequently gets more time consuming.
- For every fixed dimension, the time to carry out the SVD increases approximately linearly with $\ln(n)$.

Cluster Analysis: What to cluster?

$$\begin{pmatrix}
 u_{11} & u_{12} & u_{13} & u_{14} \\
 u_{21} & u_{22} & u_{23} & u_{24} \\
 \vdots & \vdots & \vdots & \vdots \\
 u_{n-1,1} & u_{n-1,2} & u_{n-1,3} & u_{n-1,4} \\
 u_{n1} & u_{n2} & u_{n3} & u_{n4}
 \end{pmatrix}
 \begin{pmatrix}
 s_1 & 0 & 0 & 0 \\
 0 & s_2 & 0 & 0 \\
 0 & 0 & s_3 & 0 \\
 0 & 0 & 0 & s_4
 \end{pmatrix}
 \begin{pmatrix}
 v_{11} & v_{12} & \cdots & v_{1,n-1} & v_{1n} \\
 v_{21} & v_{22} & \cdots & v_{2,n-1} & v_{2n} \\
 v_{31} & v_{32} & \cdots & v_{3,n-1} & v_{3n} \\
 v_{41} & v_{42} & \cdots & v_{4,n-1} & v_{4n}
 \end{pmatrix}$$

- The colors corresponded to specific dimensions of the data. The number of rows correspond to the number of data points.
- Think of the (length 4) rows of the left-most matrix, and (length 4) columns of the right-most matrix, as vectors. The row $(u_{i1} \ u_{i2} \ u_{i3} \ u_{i4})$ records the vectors connecting to the i^{th} neuron while the column $(v_{1i} \ v_{2i} \ v_{3i} \ v_{4i})$ records those vertices to which the i^{th} neuron connects. There are n vectors total, where n is the number of vertices in the graph we began with.
- Apply cluster analysis to these n (length 8) vectors.

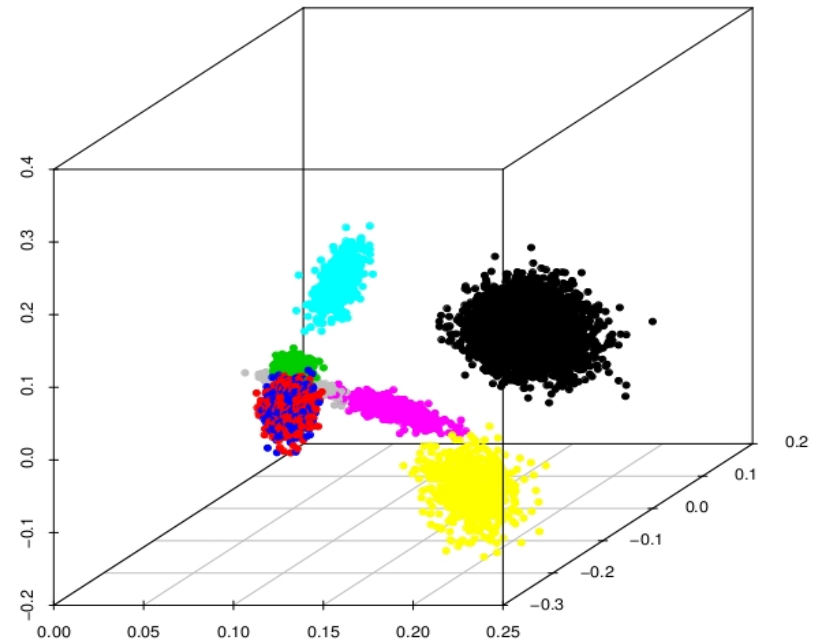
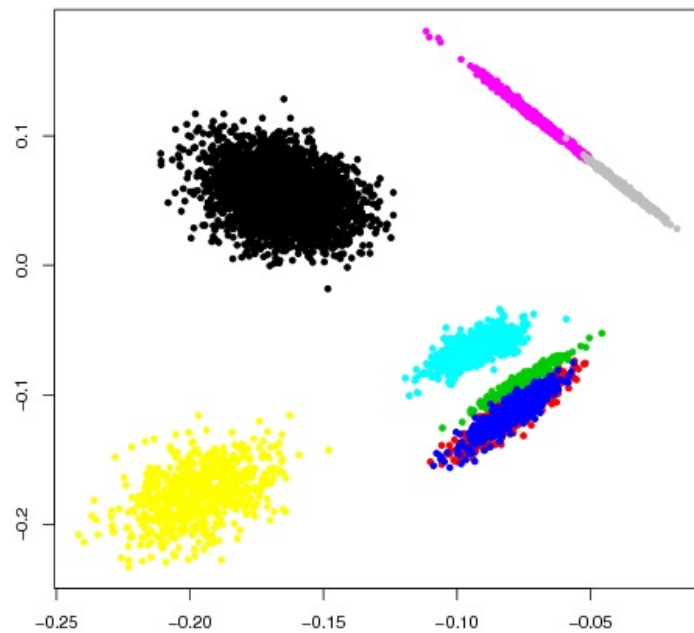
Estimating the dimension

One way to estimate the ideal dimension (the number of diagonal entries in our estimate) is to consider the singular values, and looking for the point at which the difference between successive singular values is small (< 0.1).

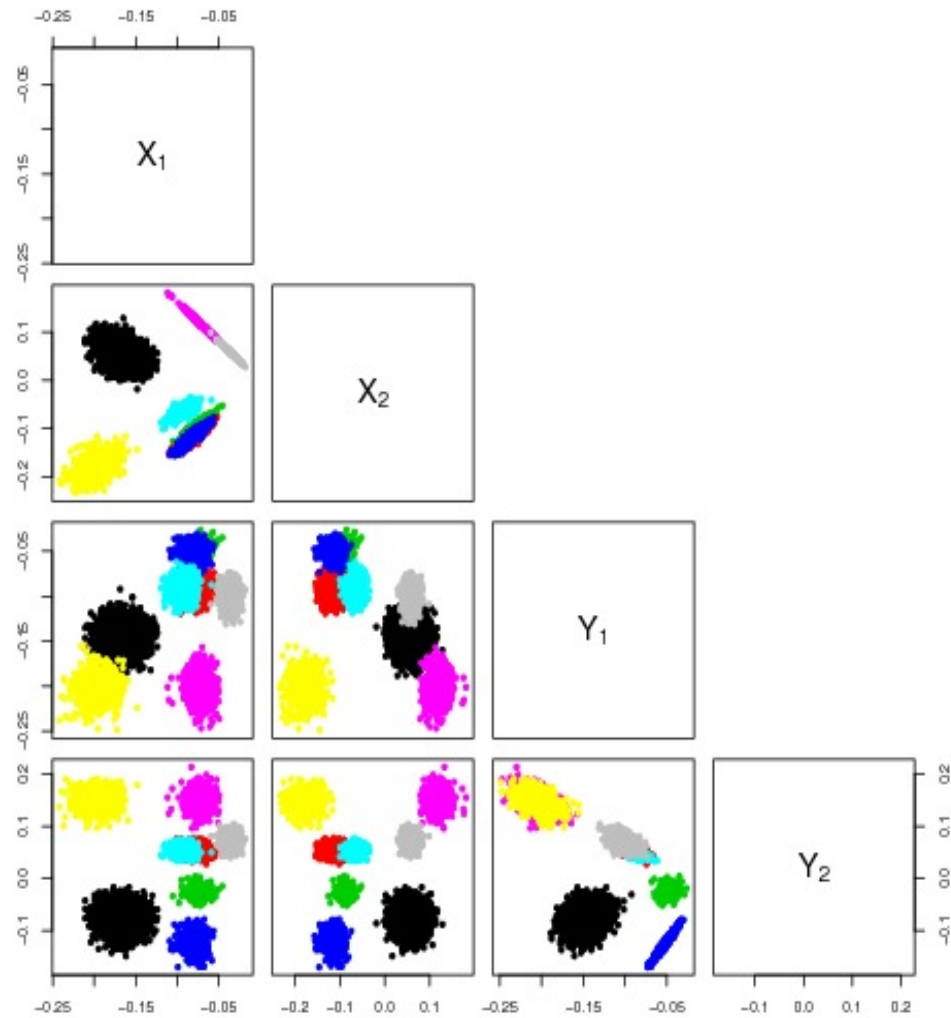
From this, we decide to pick $d = 4$ throughout, although for n greater than 100,000, the data suggests that $d=5$ might be useful.

Embeddings ($n=8192$)

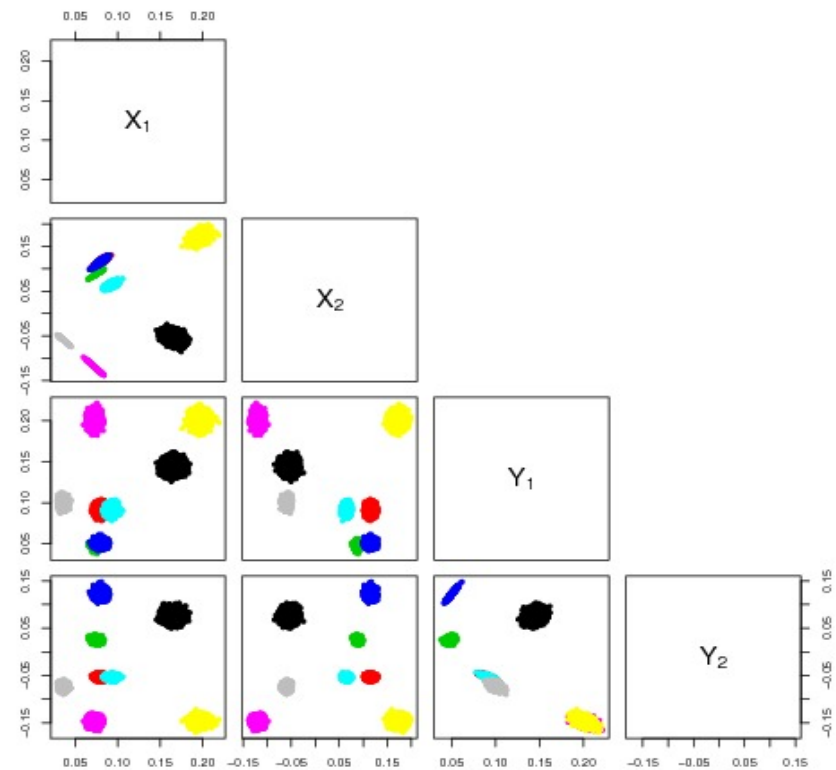
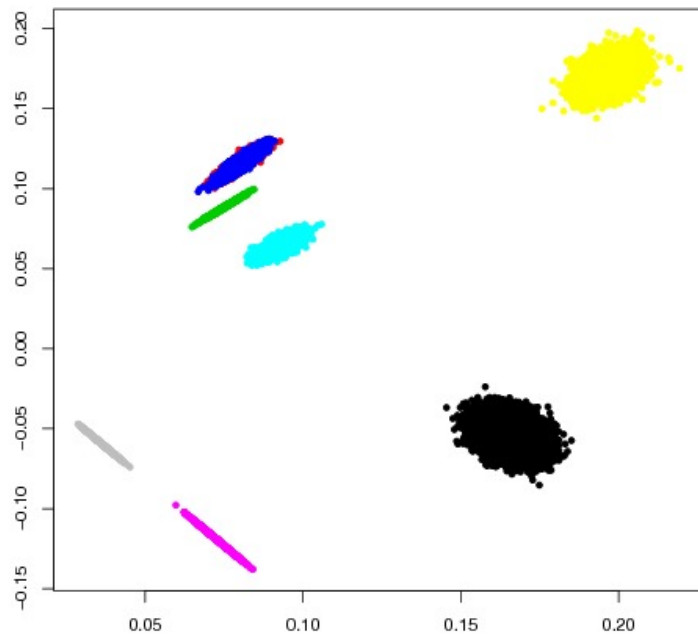
The colors illustrate the clusters in order to visualize the embeddings more clearly.



















Embeddings ($n=8192$)



Embeddings: $n=65536$

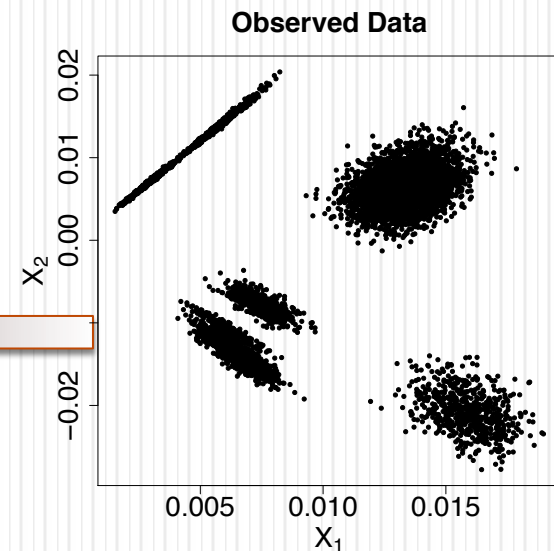


On Cluster Analysis

P								
	0.02	0.02	$\frac{0.00666}{6667}$	0.00	0.02	0.04	0.04	0.02
	0.02	0.00	$\frac{0.00666}{6667}$	0.02	0.00	0.00	0.00	0.00
	0.02	0.00	$\frac{0.00666}{6667}$	0.00	0.00	0.00	0.00	0.00
	0.02	0.00	$\frac{0.00666}{6667}$	0.02	0.00	0.00	0.00	0.00
	0.02	0.02	$\frac{0.00666}{6667}$	0.00	0.02	0.00	0.00	0.00
	0.00	0.00	0.00	0.00	0.00	0.04	0.04	0.02
	0.04	0.00	$\frac{0.01333}{33333}$	0.04	0.00	0.02	0.02	0.01
	0.00	0.00	0.00	0.00	0.00	0.02	0.02	0.01

Probability matrix P

Cluster analysis using BIC



Projections of first coordinates of data

Estimating the Block Structure

- Using SVD on the adjacency matrix, we choose a number d of dimensions to keep track of. After this choice, X and Y are embedded in \mathbb{R}^d .
- The product XY^T is a best rank- d approximation of the adjacency matrix A .
- Now we can cluster the length- $2d$ vectors of $[X \mid Y]$.
- We clustered using both k -means and also model-based clustering.
- We select the best number of classes k using the Bayesian Information Criterion (BIC).
- X, Y are only defined up to rotation. We apply rotation invariant clustering to them.
- The k latent vectors are the centers of the k clusters.

Cluster analysis

- Cluster analysis is highly dependent on the model
- The goal of cluster analysis is to try to separate the data into groups; the model should penalize the computer for making too many clusters, as well as penalize the computer for putting distant vectors into the same group.
- There are several ways to determine the penalty (BIC, AIC)
- There are several ways to determine closeness (spherical, elliptical).
- Model-based clustering is computationally expensive but it can handle elliptical data, whereas k means clustering is cheaper but only handles spherical clusters.

Clustering: k-means

- The idea is to cluster based on Euclidean distance.
- Works best when the clusters are spherical
- Given $x_1, \dots, x_n \in \mathbb{R}^d$, initialize with $C_1, \dots, C_k \in \mathbb{R}^d$
- Find the closest center: let $\tau_i = \arg \min_{1 \leq j \leq k} d(C_j, x_i)$,

And also

$$C_j = \frac{\sum_i x_i I\{\tau_i = j\}}{\sum_i I\{\tau_i = j\}}$$

- Return $\{\tau_j\}$ and $\{C_j\}$
- Repeat until convergence.
- Finds the closest center and recompute the mean.
- Repeat several times with different random starts.
- Assumes round and well-separated groups.

Model-based clustering

- Given $x_1, \dots, x_n \in \mathbb{R}^d$ the model is

$$f(x) = \sum_{j=1}^k f_j = \sum_{j=1}^k \pi_j \phi(x; \mu_j, \Sigma_j)$$

- We initialize the constants π_j, μ_j, Σ_j and then recursively define new constants based on the initial choice, repeating until convergence.

$$\tau_{ij} = \frac{f_j(x_i)}{\sum_{j=1}^k f_j(x_i)} \quad \mu_j = \sum_{i=1}^n \tau_{ij} x_i$$

$$\pi_j = \sum_{i=1}^n \tau_{ij} \quad \Sigma_j = \sum_{i=1}^n \tau_{ij} (x_i - \mu_j)(x_i - \mu_j)^T$$

- Like assuming that each point x_i is in the j th group a little bit, described by the probability τ_{ij}
- Handles elliptical clusters

Limits on class size

Bayesian Information
Criterion

=

Penalize the creation of too
many classes

- Define the log likelihood, given a model $f(x; \theta)$, as:
$$\ell(\theta) = \sum \ln(f(x_i; \theta)).$$
- BIC penalizes extra terms.
- If p_k is the number of parameters in a k cluster model on n points,

$$\hat{k} = \arg \min_k \{ -2\ell(\theta_k) + p_k \ln(n) \}.$$

where θ_k is the maximum likelihood estimate for the true values of the parameters given the data of x_1, \dots, x_n and how they split into groups.

Comparing the original block assignment to an estimated one

- Let $\tau : V \rightarrow \{1, \dots, k\}$ be a block assignment, and suppose that $\hat{\tau}$ is obtained by embedding using random dot product graphs, and then clustering using k -means. Then under some reasonable conditions, almost always
$$\min_{\pi \in S_k} |\{u \in V : \tau(u) \neq \pi(\hat{\tau}(u))\}| \leq C \ln(n)$$
(Sussman, Tang, Fishkind, Priebe).
 - This means that for all but a small number of points, we obtain the correct clustering. It guarantees that the estimated number of clusters is correct, i.e. $\hat{k} = k$ as long as we insist that there are an order of magnitude of $\ln(n)$ points in each group.

Experimental Design: the Setup

Assume that $k=8$, that the proportion of neurons of each type is given by a vector v (of length 8) and that the probability that a neuron of any one type will connect to any other type is given by a probability matrix P .

- We generate graphs of vertices, where the size of the vertex set varies from about 8,000 to 256,000. We use $n=2^{13}, 2^{14}, \dots, 2^{18}$.
- Fit the RPDG model to the graph
- Cluster using k -means or model based clustering
- Evaluate performance.

After clustering....

$$\begin{pmatrix}
 u_{11} & u_{12} & u_{13} & u_{14} \\
 u_{21} & u_{22} & u_{23} & u_{24} \\
 \vdots & \vdots & \vdots & \vdots \\
 \vdots & \vdots & \vdots & \vdots \\
 u_{n-1,1} & u_{n-1,2} & u_{n-1,3} & u_{n-1,4} \\
 u_{n1} & u_{n2} & u_{n3} & u_{n4}
 \end{pmatrix}
 \begin{pmatrix}
 s_1 & 0 & 0 & 0 \\
 0 & s_2 & 0 & 0 \\
 0 & 0 & s_3 & 0 \\
 0 & 0 & 0 & s_4
 \end{pmatrix}
 \begin{pmatrix}
 v_{11} & v_{12} & \cdots & v_{1,n-1} & v_{1n} \\
 v_{21} & v_{22} & \cdots & v_{2,n-1} & v_{2n} \\
 v_{31} & v_{32} & \cdots & v_{3,n-1} & v_{3n} \\
 v_{41} & v_{42} & \cdots & v_{4,n-1} & v_{4n}
 \end{pmatrix}$$

- We estimate the probability matrix P by the matrix \tilde{P} whose ab^{th} entry is given by

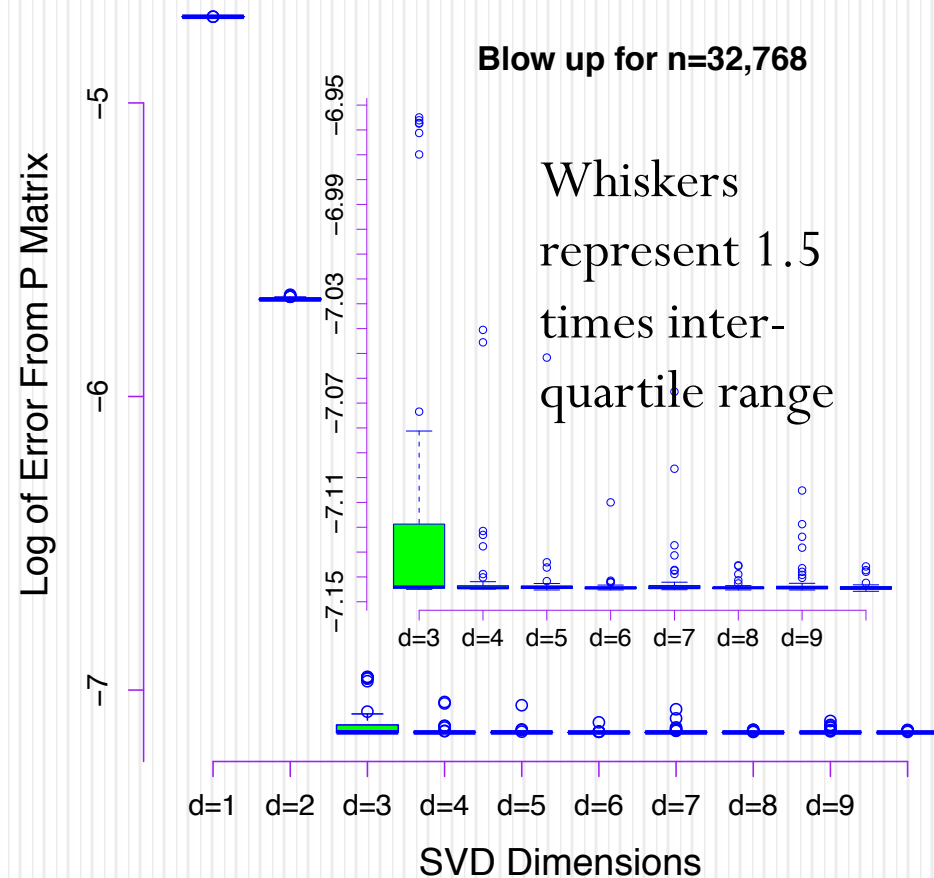
$$p_{ab} = \bar{u}_{a1}s_1\bar{v}_{1b} + \bar{u}_{a2}s_2\bar{v}_{2b} + \bar{u}_{a3}s_3\bar{v}_{3b} + \bar{u}_{a4}s_4\bar{v}_{4b}$$

STEP 1: Unprincipled Correction: We correct the clustering by 10 percent of the minimal distance among rows and columns in P .

Results (Correction)

n=32,768

The probabilities differed from true probability (in Euclidean norm, square root of sum of squares of differences in matrices) on the order of 10^{-7} for $d > 2$. Classes were assigned correctly 100 percent of the time for $d > 1$. For smaller n and/or for $d=1$, class assignment was not perfect.



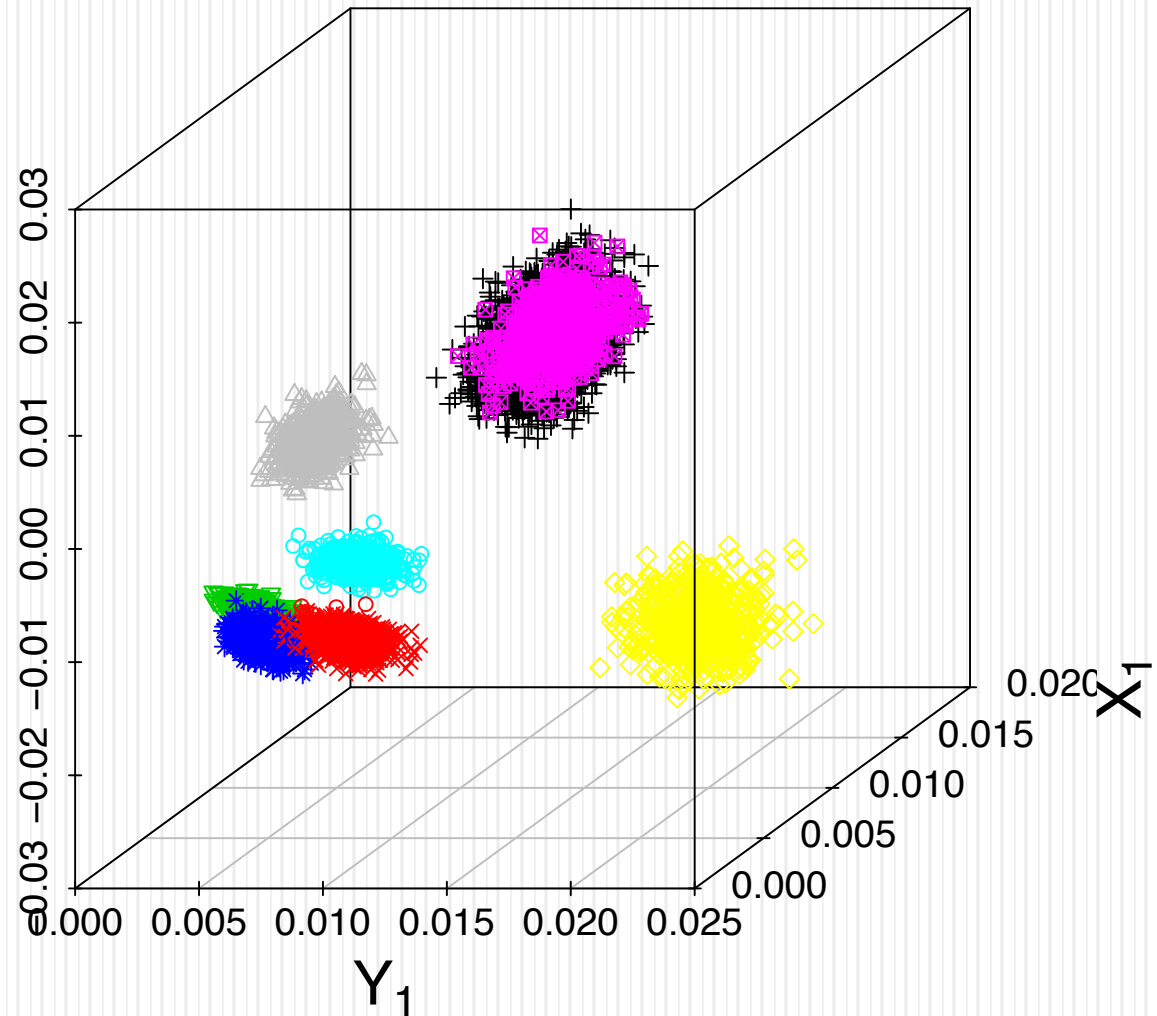
Error in probability table found compared to true probability

Results

We ran 50 simulations (adjacency matrices) with different random seeds for 32,678 vertices; this results in 500 experiments as we change the SVD analysis ($d=1, \dots, 10$.)

We analyzed the data for different values of d . After cluster analysis, the correct number of classes was found 100 percent of the cases for $d > 1$.

Estimated and True Classes



8 true classes (by color) and 8 estimated class (by shape)

Principled Fits (No ad-hoc correction of class assignment)

Model	n	k->	7	8	9	10	11	>=12
MB	8192			18	82			
KM				5	12	14	19	6
MB	16384			46	54			
KM			1	9	25	30	26	9
MB	32768			63	37			
KM			1	6	31	36	16	10
MB	65536			94	6			
KM			1	8	19	36	22	14
MB	131072			100				
KM			3	8	29	19	26	15
MB	262144			100				
KM			3	13	16	2719	27	14

Principled Fits (No ad-hoc correction of class assignment)

Model	n	k->	7	8	9	10	11	>=12
MB	8192			18	82			
KM				5	12	14	19	6
MB	16384			46	54			
KM			1	9	25	30	26	9
MB	32768			63	37			
KM			1	6	31	36	16	10
MB	65536			94	6			
KM			1	8	19	36	22	14
MB	131072			100				
KM			3	8	29	19	26	15
MB	262144			100				
KM			3	13	16	2719	27	14

What about the class assignment, rather than the number of classes

- When $\hat{k} = 8$ the class assignment is almost perfect. There were very few (under .1%) vertices placed in the wrong class, and that happened in fewer than 5% of the experiments.
- When $\hat{k} = 9$ there was almost always two groups that split fairly evenly in the estimate.

Robustness: Work in Progress

- Vary the probability matrix
- Vary the vector of proportions of true class membership
- Introduce noise: after generating a graph, randomly delete and/or add some edges.
- Introduce noise: before generating a graph, randomly change some of the assignments of vertices to groups.

The Forest Rather Than the Trees

- ✧ Robustness tests need to be implemented.
- ✧ Spatial position (or other properties) of the neurons could be considered to affect connection probabilities.
- ✧ In the simulations, connections are formed by sampling a binomial distribution for each probability (Erdős-Renyí graphs). Other distributions may be more appropriate to describe brain networks.
- ✧ Dependence among connections could be considered.

Conclusions

- ✧ Large quantities of connectivity data can be analyzed to obtain a meaningful interpretation of neuronal class. Suitable data include dense electron microscopy reconstructions (identified synapses) and light microscopy (potential synapses).
- ✧ This work may direct data collection methods, indicating whether experimental data from a large number of preparations, each with small samples, or a large sample of from fewer preparation would yield better statistical results.
- ✧ Determination of neuronal classes has the prospect of becoming rigorous, verifiable and reproducible.

References

Schwarz, G. Estimating the Dimension of a Model, *Annals of Statistics*, 6:461-464, 1978.

Lloyd, S. P. Least Squares Quantization in PCM. *IEEE Trans. on Information Theory* 28(2):129-137, 1982.

Scheinerman E. and Tucker, K. Modeling graphs using dot product representations. *Computational Statistics*, 25:1–16, 2010.

Support: NIH R01 NS39600 & ONR MURI 141010198 to GAA
Partial Support, Office of Naval Research In-House Laboratory
Independent Research Program, to DM