CPSC 453a – Machine Learning for Biology – Fall 2016

Based on lectures by Prof. Smita Krishnaswamy

James Diao

Table of Contents

Lecture 1: Intro to Single Cell Data

1. Importance of proteins: receptors, antibodies, hormones, enzymes, structural components, transport/storage, transcription factors, signal transduction, etc.
2. Cell types are not precisely defined, even within single systems.
3. Bulk data gives you the average of many cell responses together.
4. Single-cell data allows you to see the actual distribution.
5. **Single-Cell Proteomics: Mass Cytometry (CyTOF)**
    a. CyTOF: Flow cytometry by time-of-flight of heavy metals in mass spectrometry.
    b. Steps: experiment, preserve cellular state, permeabilize cells, label biomarkers with metal isotopes, cell by cell: vaporize biological material and run through MS machine.
    c. Advantages: targeted (by antibodies) and non-redundant. Avoids autofluorescence and spectral spillover from fluorescent tags (big problem with many tags).
    d. Each cell has a vector of intensity readouts for each isotope mass (length = number of isotopes).
6. Single-Cell RNA-Seq
    a. Droplet-based technologies:
    b. Steps:
        i. Both methods: capture single-cells and barcoded beads in droplets, cell-lysis + hybridization of cellular RNA to oligo-T on the beads.
        ii. Then, either cleave off primers and reverse transcribe in droplets, before breaking them (Klein) or break the droplets first, then reverse transcribe in bulk, before cleaving from bead (Macasko).
        iii. PCR amplification and sequencing.
    c. Problem: differences in cell lysis (and other issues) create differences in RNA counts between cells.
        i. The noise in number of transcripts is overdispersed Poisson (sample variance > expected (sample mean))
        ii. The negative binomial can account for this: has more parameters that allow variance > mean.
    d. Drop-out: when scRNA-seq has inefficient transcript capture (<5%), leading to lots of gene columns with 0s.
    e. Advantages: high-dimensional, high-throughput, high resolution, heterogeneous. System-level view.
7. Topics of interest in computational biology
    a. Dimensionality reduction, progressions (pathways, differentiation, apoptosis), dependencies and relationships, complex systems and cellular logic.

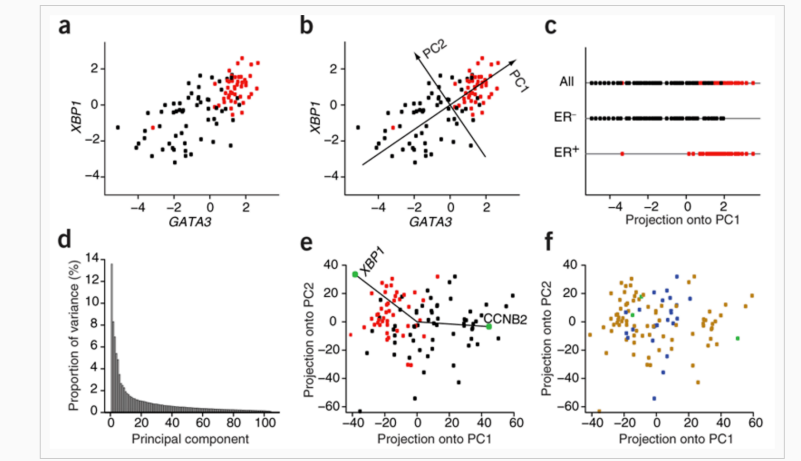
Lecture 2: Dimensionality Reduction

1. Data is often a matrix: observations x variables (cells x genes)
2. Principle Components Analysis (PCA)
3. Multidimensional Scaling (MDS)
4. Related Spectral Methods: take the eigenvectors of some matrix
    a. Eigenvector: $Mx = \lambda x$
        i. Direction along which a matrix is invariant.
        ii. Vectors in that direction are only stretched.
5. Linear methods: can we re-express the data matrix in a different basis?
    a. Take a few linear combinations of observed variables to capture max info.
    b. Maximizing variation means minimizing redundancy = minimizing covariance.
    c. Linear transformation: rotations and stretching.
    d. Covariance and covariance matrix:

$$\begin{aligned}\operatorname{cov}(X,Y) &= \operatorname{E}[(X - \operatorname{E}[X])\,(Y - \operatorname{E}[Y])] \\ &= \operatorname{E}[XY - X\operatorname{E}[Y] - \operatorname{E}[X]Y + \operatorname{E}[X]\operatorname{E}[Y]] \\ &= \operatorname{E}[XY] - \operatorname{E}[X]\operatorname{E}[Y] - \operatorname{E}[X]\operatorname{E}[Y] + \operatorname{E}[X]\operatorname{E}[Y] \\ &= \operatorname{E}[XY] - \operatorname{E}[X]\operatorname{E}[Y].\end{aligned}$$

$$\Sigma = \begin{bmatrix} \operatorname{E}[(X_1 - \mu_1)(X_1 - \mu_1)] & \operatorname{E}[(X_1 - \mu_1)(X_2 - \mu_2)] & \cdots & \operatorname{E}[(X_1 - \mu_1)(X_n - \mu_n)] \\ \operatorname{E}[(X_2 - \mu_2)(X_1 - \mu_1)] & \operatorname{E}[(X_2 - \mu_2)(X_2 - \mu_2)] & \cdots & \operatorname{E}[(X_2 - \mu_2)(X_n - \mu_n)] \\ \vdots & \vdots & \ddots & \vdots \\ \operatorname{E}[(X_n - \mu_n)(X_1 - \mu_1)] & \operatorname{E}[(X_n - \mu_n)(X_2 - \mu_2)] & \cdots & \operatorname{E}[(X_n - \mu_n)(X_n - \mu_n)] \end{bmatrix}.$$

    e. The covariance matrix is an operator that maps a linear combination (c) onto a vector of covariances with the original variables.
    f. PCA creates 2 conditions: (a) the new dimensions are in decreasing order by captured variance – var(PC1) > var(PC2) > … var(PCn), and (b) the new dimensions are uncorrelated (orthogonal).
    g. The first k principle components are the k largest eigenvectors, or when scaled, the eigenvectors with the largest eigenvalues (equal to variance captured).
    h. Example figures:



6. Multidimensional Scaling (MDS)
    a. Same as PCA, but performed on distance matrix, so that distances are preserved.
    b. Different ways of measuring distance:
        i. Euclidean, 1 – cosine(angle), Hamming distance.
        ii. Distance functions must satisfy: nonnegativity, symmetry, and triangle inequality. Only Euclidean in the above is an actual distance function.
    c. Centering matrix: subtracts the mean of the components of the vector from every component. Equal to $I_n - 1/n\ [\mathbf{1}\mathbf{1}^T]$
    d. Double centering: subtract the row mean, subtract the col mean, add the overall matrix mean, and multiply by -1/2. Now, the colmeans = rowmeans = 0. The origin is at the geometric center of the cloud of N points.
    e. Apply double-centering to the distance matrix before finding the largest eigenvalues + eigenvectors.

Lecture 3: tSNE and MATLAB

1. Structure of biological data
    a. Nonlinear dependencies are abundant in biology.
    b. Cells exist on a phenotypic manifold
        i. Gene coordination
        ii. Progression due to differentiation and state change
2. Kernel function: nonnegative, symmetric, integrates to 1.
    a. Easy ones: student T, Gaussian exp(-1/2*dist^2)
    b. Adaptive kernel: adjusts parameters to fit perplexity.
        i. Perplexity is the effective number of neighbors, defined as 2^H(p).
        ii. Ex: you adjust the std in the Gaussian distribution.
3. Kullback-Liebler (KL) Divergence:
    $$D_{\mathrm{KL}}(P\|Q) = \sum_i P(i)\,\log\frac{P(i)}{Q(i)}.$$
    a.
    b. Divergence is an asymmetric distance
    c. Minimizes "distance" between probability distributions
    d. Used to match probability distributions of high-dim & low-dim neighbors.
    e. Measures info gained when updating from 1 distribution to another.
4. tSNE: stochastic neighbor embedding.
    a. Based on the student-t distribution: heavier tails alleviate the crowding problem and some optimization problems.
    b. Map high-dim space to low-dim space while preserving local/global shape.
    c. tSNE preserves similarity, defined through a kernel function.
    d. Stochastic neighbors: P($\text{cell}_i \rightarrow \text{cell}_j$) = normalized kernel_{ij}: dividing by sum(kernel_{ik})
    e. Steps: (1) find the high-dim neighbor probs (P) and low-dim neighbor probs (Q), (2) define the cost function as the sum of KL(Pi||Qi) for all n points. (3) minimize this by gradient descent.

High dimensional Neighbor probabilities

$$p_{j|i} = \frac{\exp\left(-\|x_i - x_j\|^2/2\sigma_i^2\right)}{\sum_{k\neq i}\exp\left(-\|x_i - x_k\|^2/2\sigma_i^2\right)},$$

Low Dimensional Neighbor probabilities

$$q_{j|i} = \frac{\exp\left(-\|y_i - y_j\|^2\right)}{\sum_{k\neq i}\exp\left(-\|y_i - y_k\|^2\right)}.$$

KL Divergence of Cost function

$$C = \sum_i KL(P_i||Q_i) = \sum_i\sum_j p_{j|i}\log\frac{p_{j|i}}{q_{j|i}},$$

Gradient to Perform Descent

$$\frac{\delta C}{\delta y_i} = 2\sum_j (p_{j|i} - q_{j|i} + p_{i|j} - q_{i|j})(y_i - y_j).$$

Iterative updates

$$\mathcal{Y}^{(t)} = \mathcal{Y}^{(t-1)} + \eta\frac{\delta C}{\delta \mathcal{Y}} + \alpha(t)\left(\mathcal{Y}^{(t-1)} - \mathcal{Y}^{(t-2)}\right),$$

f. Improvements:
   i. Symmetricized cost function: single KL divergence instead of multiple for each point.(double summation: for each possible pair of points)
   ii. Student-t distribution: repulses dissimilar points more (because the heavy tails can actually reach them)

g. Algorithm:

Algorithm 1: Simple version of t-Distributed Stochastic Neighbor Embedding.

Data: data set $\mathcal{X} = \{x_1, x_2, ..., x_n\}$,
cost function parameters: perplexity $Perp$,
optimization parameters: number of iterations T, learning rate η, momentum $\alpha(t)$.
Result: low-dimensional data representation $\mathcal{Y}^{(T)} = \{y_1, y_2, ..., y_n\}$.
begin
  compute pairwise affinities $p_{j|i}$ with perplexity $Perp$ (using Equation 1)
  set $p_{ij} = \frac{p_{j|i} + p_{i|j}}{2n}$
  sample initial solution $\mathcal{Y}^{(0)} = \{y_1, y_2, ..., y_n\}$ from $\mathcal{N}(0, 10^{-4}I)$
  for t=1 **to** T **do**
    compute low-dimensional affinities q_{ij} (using Equation 4)
    compute gradient $\frac{\delta C}{\delta \mathcal{Y}}$ (using Equation 5)
    set $\mathcal{Y}^{(t)} = \mathcal{Y}^{(t-1)} + \eta \frac{\delta C}{\delta \mathcal{Y}} + \alpha(t)\left(\mathcal{Y}^{(t-1)} - \mathcal{Y}^{(t-2)}\right)$
  end
end

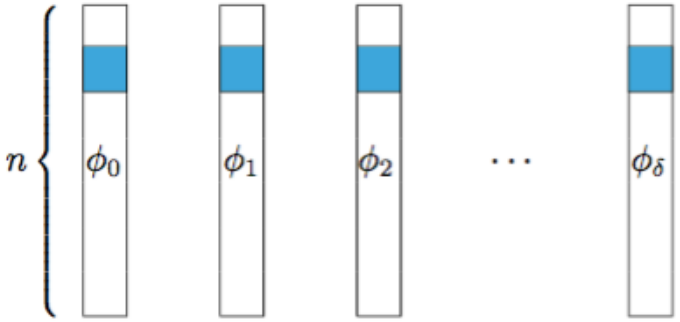
Lecture 4: Spectral Methods for Dimensionality Reduction

1. Detecting linear structure
    a. See if PCA captures most of the variance; if there's an eisengap.
2. Principle Components Analysis (PCA)
    a. Center the inputs on the origin, compute covariance matrix, project into a subspace, and maximize projected variance.
    b. Eigenvectors: principal axes of the maximum variance subspace.
    c. Eigenvalues: variance of inputs along principle axes
    d. Estimated dimensionality: number of significant eigenvalues
    e. Strengths: no tuning parameters, non-iterative, no local optima
    f. Weaknesses: limited to linear projections.
3. Multidimensional Scaling (MDS): Linear method
    a. Non-metric: NMDS: nonlinear, by preserving rank order of distances.
    b. This allows nonlinear embeddings, but has iterative optimization / local minima.
4. Isomap?
    a. Preserves geodesic distances along a submanifold. Uses geodesic distances.
    b. Build connected adjacency graph using kNN, inputs within radius r, prior knowledge, etc. Weight edges by local distances and compute the shortest path through the graph.
    c. Use MDS on geodesic distances
    d. Advantages: no local minima, non-iterative, nonparametric.
    e. Weaknesses: sensitive to shortcuts, no out-of-sample extension. Expensive to compute all shortest paths for large datasets.
5. Landmark isomap
    a. Identify subset as landmarks, estimate geodesics to/from landmarks, apply MDS to landmark distances. Reduces computation.
6. Spectral methods
    a. Derive sparse graph from kNN, derive matrix from graph weights, derive embedding from eigenvectors.

Lecture 5: Diffusion-Based Manifold Learning (Guy Wolf)

1. Manifold learning
    a. Finds locally low-dim geometries in high-dim space.
2. Diffusion maps
    a. The spectral embedding is the *ith* row of the unnormalized eigenvector matrix.
    b. The first eigenvalue is always equal to 1.
    A always row-sums to 1, so A [c c c…] = c*1 => returns norm of eigenvectors.
    c. **They are all positive.** Positive semidefinite matrix => symmetric and often appear from dot products.

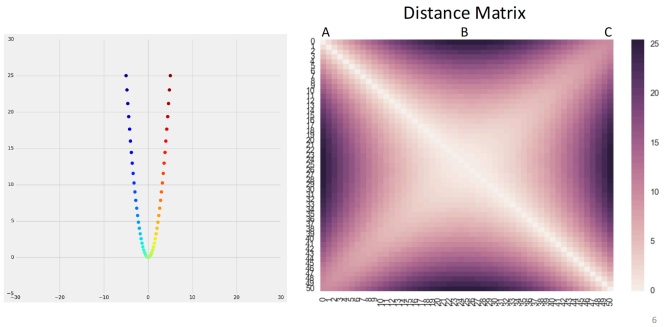
$$1 = \boxed{\lambda_0} \geq \boxed{\lambda_1} \geq \boxed{\lambda_2} \geq \cdots \geq \boxed{\lambda_\delta} > 0$$



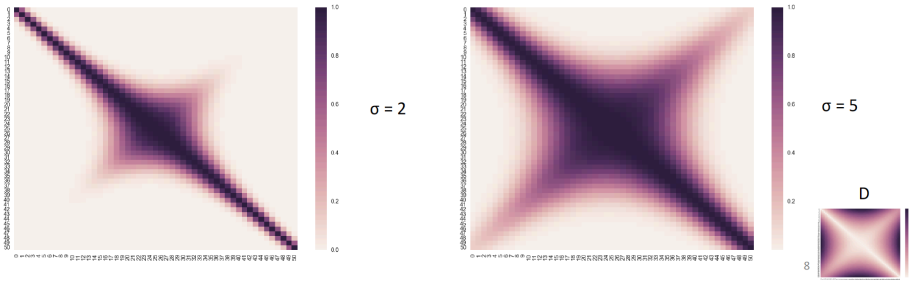
$$x \mapsto \Phi(x) \triangleq [\lambda_0\phi_0(x)\,,\ \lambda_1\phi_1(x)\,,\ \lambda_2\phi_2(x)\,,\ \ldots\,,\ \lambda_\delta\phi_\delta(x)]^T$$

Lecture 6: Eigenvectors, Progression, and Dimensionality Estimation

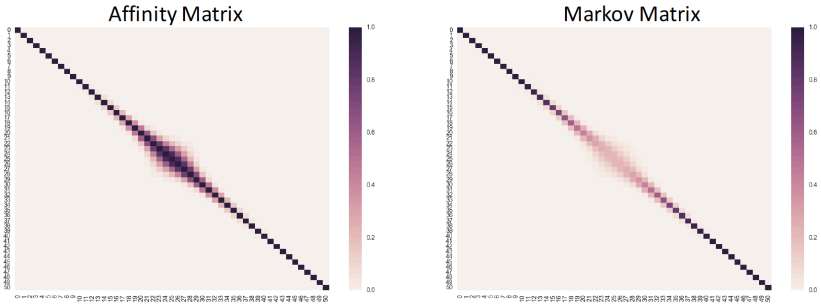
1. The similarity matrix puts some distance metric through a kernel.
2. Since each entry is the probability of moving from i→j in 1 step, powers of the matrix give the probabilities of walking/diffusing through the map.
3. This is NOT necessary. This just allows you to spread out the distance matrix.
4. Diffusion Map Example:
    a. Take the data and calculate distance matrix



    b. Calculate the affinities using a kernel

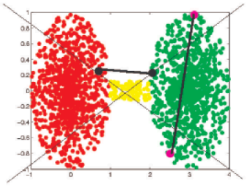


    c. Normalize these into transition probabilities

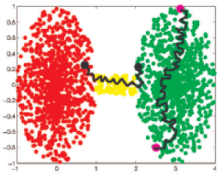


5. Steady state of diffusion: what happens if you keep going?
6. Diffusion Map: right eigenvectors of the Markov-adjusted affinity matrix
    a. Like PCA, eigenvectors explain the extent of variation covered by that diffusion component.
    b. Left and right eigenvalues are the same.
    c. The first left eigenvector with eigenvalue 1 is the steady state distribution.
    d. This provides a density estimate of the data (more dense = more likely to stay there long-term)

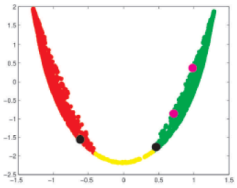
7. Diffusion distances
   - Diffusion distance $D_t^2(\boldsymbol{x}_i, \boldsymbol{x}_j) = ||\boldsymbol{P}_{i:}^t - \boldsymbol{P}_{j:}^t||_{\boldsymbol{D}^{-1}}^2 = ||\boldsymbol{\Phi}_t(x_i) - \boldsymbol{\Phi}_t(x_j)||^2 = \sum_{m \geq 1} {\lambda_m}^{2t}(\phi_m(i) - \phi_m(j))^2$.



(a) Euclidean distance



(b) Random walk



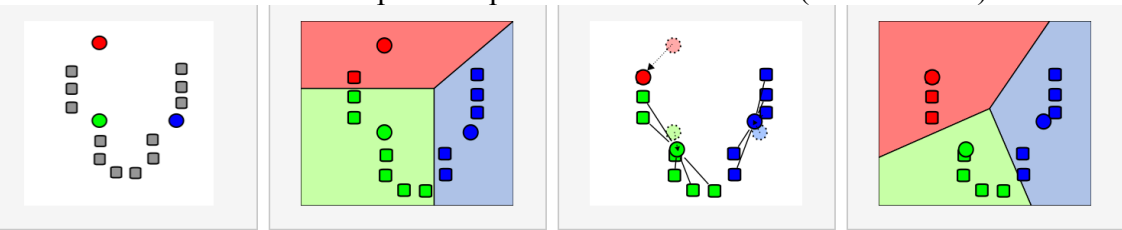
(c) Embedded dataset

Lecture 8: Clustering: K-Means and Spectral Methods

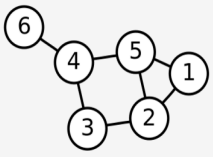
1. Clustering: partitioning dataset OR state space; each data point is in exactly 1 cluster.
2. Pick the partition by minimizing a cost function:
    a. Closeness of members within the group
    b. Distance between groups
    c. Ratio of the 2
    d. Minimum cut of a NN-graph
3. Modularity: actual edges / expected edges
4. K-means: picks K circular clusters
    a. Minimizes within-group sum of squared distances

$$\arg\min_{\mathbf{S}} \sum_{i=1}^{k} \sum_{\mathbf{x} \in S_i} \|\mathbf{x} - \boldsymbol{\mu}_i\|^2$$

    b. Initialize K means randomly
        i. Either randomly assign points to clusters (starts with same mean for each cluster (all at the middle)
        ii. randomly pick k observations as the means (spreads means out)
    c. Iterative refinement that alternates between:
        i. Assignment of points to means
        ii. Recomputation of means based on the points
        iii. Convergence:
        iv. ^Example of expectation maximization (local minima)

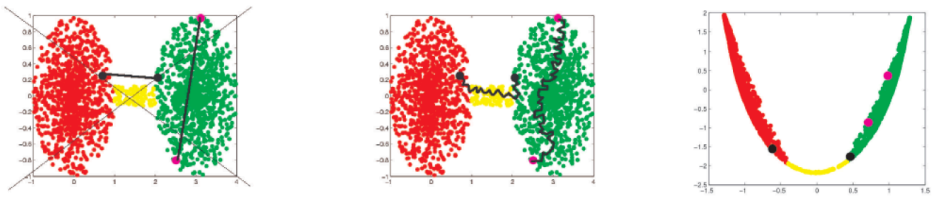


    d. Limitations:
        i. Assumes a shape for the cluster
        ii. Assumes the number of clutsers
        iii. Finds a local minima
    e. Balancing fit versus higher K (overfitting)
        i. Akaike information criterion (AIC = 2k – 2ln(L))
        ii. Bayesian information criterion (BIC = -2 * ln(L) + k*ln(n))
        iii. Where L is the likelihood function
5. Spectral clustering: K means on reduced dimensions
    a. W = similarity/adjacency matrix
    b. D = degree matrix
    c. Graph Laplacian: L = D – W

Labeled graph	Degree matrix	Adjacency matrix	Laplacian matrix
	$\begin{pmatrix} 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$	$\begin{pmatrix} 2 & -1 & 0 & 0 & -1 & 0 \\ -1 & 3 & -1 & 0 & -1 & 0 \\ 0 & -1 & 2 & -1 & 0 & 0 \\ 0 & 0 & -1 & 3 & -1 & -1 \\ -1 & -1 & 0 & -1 & 3 & 0 \\ 0 & 0 & 0 & -1 & 0 & 1 \end{pmatrix}$

d. Properties of L: symmetric, positive semidefinite, double-centered (0 is an eigenvalue with any connected component of the graph). Smallest non-zero value is the spectral gap. 2nd smallest value is Fiedler value: how connected the graph is.
e. Steps:
   i. compute W and L
   ii. normalize Laplacian (symmetric or random walk normalizations)
   $$L_{\text{sym}} := D^{-1/2}LD^{-1/2} = I - D^{-1/2}WD^{-1/2}$$
   $$L_{\text{rw}} := D^{-1}L = I - D^{-1}W.$$
   iii. compute first k eigenvalues of L
   iv. project onto that space
   v. perform K-means clustering.

- Diffusion distance $D_t^2(\boldsymbol{x}_i, \boldsymbol{x}_j) = ||\boldsymbol{P}_{i:}^t - \boldsymbol{P}_{j:}^t||_{\boldsymbol{D}^{-1}}^2 = ||\boldsymbol{\Phi}_t(x_i) - \boldsymbol{\Phi}_t(x_j)||^2 = \sum_{m\geq 1} {\lambda_m}^{2t}(\phi_m(i) - \phi_m(j))^2$.



(a) Euclidean distance (b) Random walk (c) Embedded dataset

f. Splitting the eigenvectors is ideal because they go between clusters. (because that captures the largest differences).
g. Generalized: taking K-means exactly computes a multi-way cut of K eigenvectors.

6. RatioCuts/Normalized Cuts
   a. Takes a small value of clusters are reasonably sized and don't cut too many vertices.
7. Other techniques:
   a. Recursive spectral clustering
   b. Hierarchical clustering
   c. Biclustering: clusters both cells and genes (diagonal squares on the data matrix)

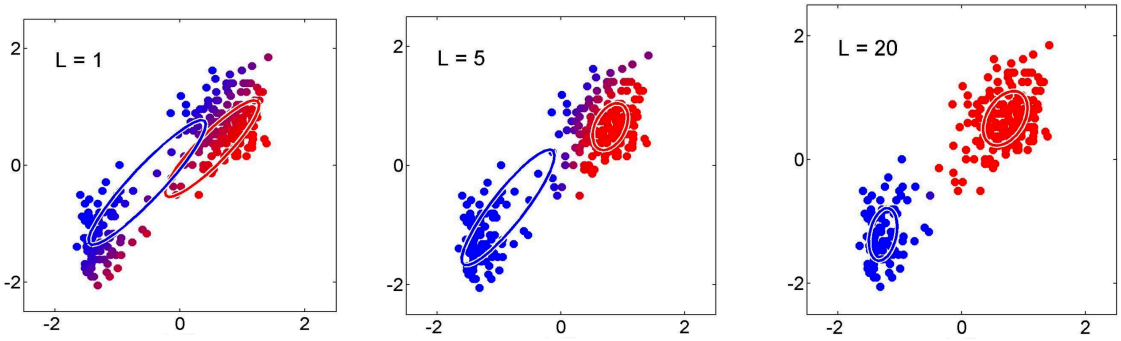
Lecture 9: Gaussian Mixture Models, Linkage, and Modularity

1. Gaussian Mixture Models
    a. Generates datapoints as the linear combination of Gaussian mixtures
    b. Pick parameters (mean and covariance) that maximize the (log-)likelihood of a point being generated by the model.

$$\mathcal{N}(\mathrm{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{d/2}\,|\boldsymbol{\Sigma}|^{1/2}} \exp\left\{-\frac{1}{2}(\mathrm{x}-\boldsymbol{\mu})^{\top}\boldsymbol{\Sigma}^{-1}(\mathrm{x}-\boldsymbol{\mu})\right\}$$

mean covariance

    c. After maximum likelihood for 1 Gaussian: the best parameter mean is the sample mean, and the best parameter covariance is the sample covariance.
    d. Data generation by linear superposition of Gaussians:
    (Each Gaussian times π_k "responsibilities". Normalized sum to 1: interpretable as prior probabilities).
    e. Sample from Gaussians by: (1) sample a component with prob π_k, then (2) sample that Gaussian.
    f. No closed form solution for log-likelihood; find by iterative refinement.
2. Expectation maximization
    a. Initialize solution (means and covariances)
        i. E-step: use current parameters to compute likely responsibilities.
        ii. M-step: use responsibilities (mixing weights) to compute best parameters.
        iii. Convergence: When log-likelihood increase < n (often n = 0).



3. Cluster assignments
    a. Place each point in its most likely component. This gives a hard assignment.
4. Agglomerative / Hierarchical Clustering
    a. Clumping data into bigger and bigger groups.
    b. Naturally leads to hierarchy: each phase/step is a level.
        i. Minimum, or single-linkage: uses min distance b/tw the points in 2 different clusters.
            1. Start with every point in its own cluster
            2. Merge 2 closest clusters (min).
            In this case, compute min(pairwise distances).
            3. Stop when everything is in 1 cluster.
            4. Result: dendrogram.
        ii. Maximum, or complete-linkage: same, but uses max distance.

5. Linkage clustering (graphs)
    a. Weight edges by distance or kNN connected/not
    b. Graphs can be clustered without space being clustered
6. Graph partitioning objectives
    a. Minimum cut (need to know sizes of groups)
    b. RatioCut, Normalized Cut
7. Modularity:
    a. Q = # expected edges – # real edges within group
    b. A_{ij} = actual edges between groups i and j
    c. P_{ij} = prob of edge between I and j
    d. How do you form an expectation of an $edge_{ij}$?
        i. Generate random graphs with same degrees
        ii. Degree(i) * Degree(j) / 2m where m = total number of edges.
    e. Modularity is the sum of $B_{ij} = A_{ij} - P_{ij}$ for all i and j
    f. B_{ij} is the modularity matrix.

$$Q = \frac{1}{2m} \sum_{vw} \left[A_{vw} - \frac{k_v k_w}{2m} \right] \frac{s_v s_w + 1}{2}$$

Modularity measure:

$$Q = \frac{1}{2m} \sum_{i,j} \left[A_{ij} - \frac{k_i k_j}{2m} \right] \delta(c_i, c_j),$$

A is the affinity matrix

$k_i = \sum_j A_{ij}$ K is the degree of the vertex

$m = \frac{1}{2} \sum_{ij} A_{ij}$ M is the sum of all degrees, i.e. volume of graph

$\delta(u, v)$ is 1 if $u = v$ and 0 This is a function that returns one if the two vertices are in the same community

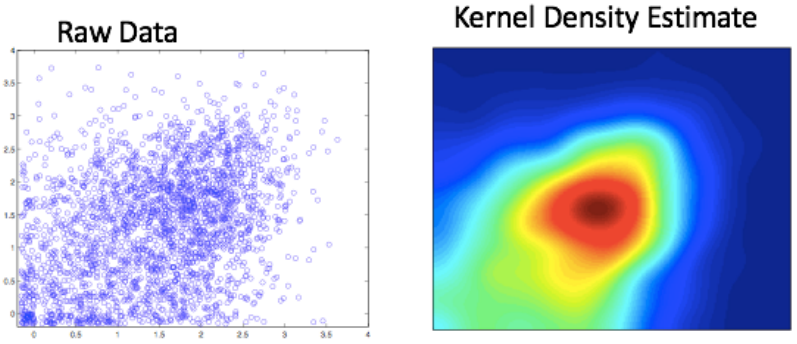
    g. Edge indicator: A_{vw} = 1 if edge, 0 if not.
    h. Membership indicators: s_v and s_w are 1 if in the group and -1 if not in the group. You ONLY look at connections in the same group.
    i. M is the total number of edges. 2m is the sum of all degrees for all vertices.
8. Take the eigenvalues of the modularity matrix and set s = 1 (put into the group) when the largest eigenvector gives best split into 2 communities.

9. Modularity measures how densely edges within a cluster are connected compared towhat is expected at random. In other words, how connected things are within clusters, and how disconnected between clusters.
10. Tries for different Ks naturally. You can calculate different modularities for different Ks- highest modularity for the best K.
    a. At first, all points in their own clusters.

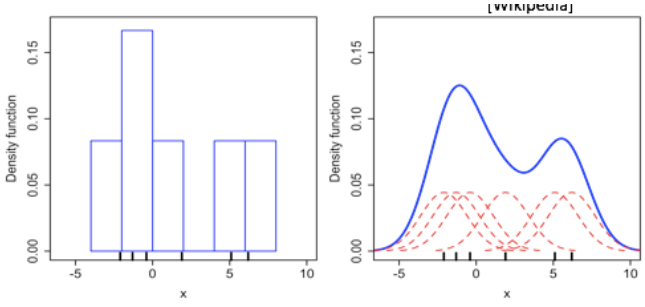
   b. Affinity can also be whether you're connected or not (graph theoretic)
   c. Affinity matrix:
   d. Start with 100 points.
   e. Phase 1 of Louvain method: stop at 20 clusters => 80 empty clusters.
   f. Phase 2: Sums the affinities within a cluster, and takes differences outside of clusters. Collapse the clusters.
11. Gaussian Mixture Models: picks K Gaussian clusters
12. Linkage: hierarchical link to nearest neighbors
13. Community detection: look for natural disconnections of a kNN graph.
14. K-means: randomly initialize, and assign points to the closest mean. Then calculate the new set-means based on the means of the points assigned to that set.

Lecture 12: Kernel Density Estimation, Constrained Programming

1. Kernel Density Estimation



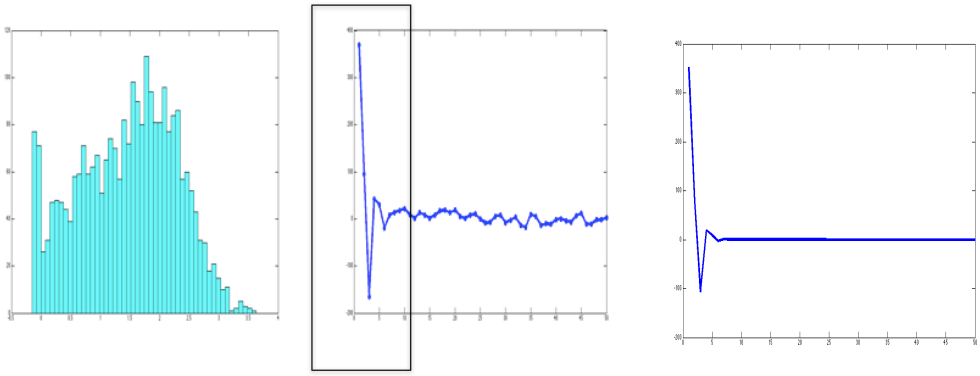
   a. Properties
      i. Approximates true probability density function.
      ii. Real valued, non-negative, integrates to 1
      iii. Nonparametric, no assumptions on shape
   b. To estimate density
      i. Learn how dependencies under different conditions.
      ii. Smooth approximation to density- interpolates gaps.
      iii. Visualize relationships
      iv. Measure strength of relationships/associations
   c. Disadvantages: noisy in sparse regions, needs proper bandwidth
2. Histograms
   a. Algorithm for Data X = (x_1, …, x_N)
      i. 10 bins with width w=max(X)/10;
      ii. For i=1:N
      iii. val = ceil(x_i/w)
      iv. hist(val) = hist(val+1);
      v. end
   b. Width of histogram controls smoothness (missing info vs. noisy)
   c. Disadvantages: has gaps and jumps; discontinuous
3. KDE



   a. Bandwidth = width of scaled kernel (std or smth else)
      i. Higher oversmooths, lower undersmooths
   b. Estimating bandwidth

i. Mean Integrated Square Error
   1. MISE = integral((KDE – actual)^2 dx): both distributions!
   2. Approximate the actual distribution with a Gaussian and minimize MISE. This gives bandwidth h = (4*sigma^5 / 3n)$^{1/5}$
ii. Gives the speed of an equi-binned histogram with smoothness of KDE

4. Heat-equation based density estimation
   a. Steps
      i. Start with histogram
      ii. Transfer to frequency domain via DCT.
      iii. Evolve/smooth using the heat equation, mimics heat spread.
      iv. Invert DCT
   b. At time 0, peaky Gaussian. Later times = flatter Gaussians.
   c. Heat equation: f' = a*f'' . Heat diffuses over larger areas with time. Spreads from peaks to uniform equilibrium.
5. Discrete Cosine Transform (DT) represents data as sum of cosines.



   a. Multiply each coefficient of each cosine

$$f(x) = \sum_{m=0}^{\infty} a_m \cos(m\pi x) \exp\left(\frac{-m^{\alpha}\pi^2 t}{2}\right)$$

m^{th} coefficient of DCT of the histogram

Multiplied by decaying exponential in t

      i. This basically tosses out the lower values.
6. Complexity Analysis
   a. Histogram: O(n)
   b. O(M log M) to compute DCT
   c. O(M) to smooth DCT
   d. O(M log M) to invert DCT
   e. Overall: O(N+M log M)
7. Scaling up to high dimensions
   a. Hard to scale up regular density estimation
   b. Easier with kNN.
8. ND Density Estimation on kNN
   a. P(t at point) = prob mass * 1/volume = k/N * 1/πR^2 (or other).
   b. k is a parameter you optimize.

Lecture 13: Entropy, Mutual Info, and MI Estimation

1. Why is information theory useful?
    a. Similarity of classes to each other (divergence)
    b. Intrinsic dimension of data (entropy)
    c. Best possible error rate (divergence, MI)
    d. Dependency of variables (MI)
    e. Anomalies in the data (entropy)
    f. Relevant features for classification (MI)
2. Definitions:

    Entropy

$$H(\mathbf{X}) = -\int f_X(x)\log\left(f_X(X)\right)dx,$$
$$H(\mathbf{X}) = -\sum_x p_x \log\left(p_X\right).$$

    Mutual Information

$$I(\mathbf{X};\mathbf{Y}) = \int f_{XY}(x,y)\log\left(\frac{f_{XY}(x,y)}{f_X(x)f_Y(y)}\right)$$

    a. –log is replaceable with other functions g(.)
    b. Renyi entropy has $g(t) = t^a$ for $a > 0$. Must decay to zero sub-linearly.
3. Properties of entropy
    a. H(X,Y) <= H(X) + H(Y): triangle inequality for independent X and Y.
    b. Discrete case: entropy is maximized when X is uniform. H(X) > 0.
    c. Continuous case: entropy is maximized when X is uniform (for a given variance)
    d. Independent of location of data (imagine splitting a Gaussian into 2 halves)
4. Local intrinsic dimension
    a. LID can be estimated by looking at entropy in the neighborhood of a point.
    b. Advantages: this returns an identifiable number for local values.
    c. Disadv: computationally intensive, based on random initializations.
5. Conditional Entropy
    - Conditional Shannon Entropy:

$$H(X|Y) = -\sum_{x\in X} p_x \sum_{y\in Y} p_{x|y}\log(p_{x|y})$$
$$= -\sum_{x\in X}\sum_{y\in Y} p_{xy}\log(p_{x|y})$$

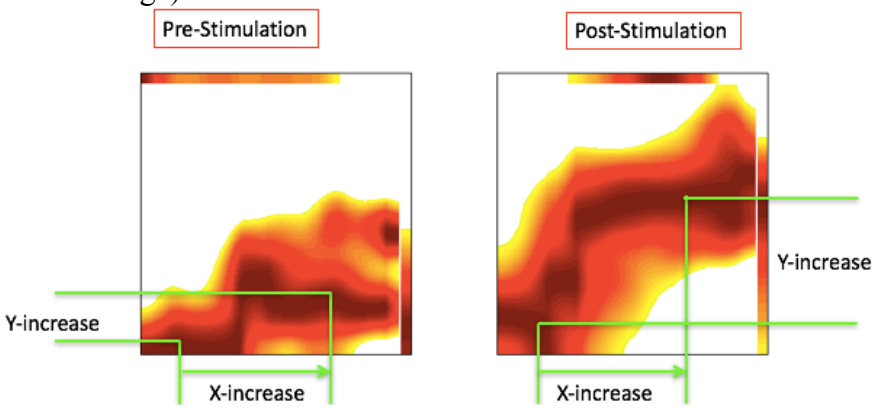
    - Conditional Differential Entropy:
      $H(X|Y) = \int f_{XY}(x,y)\log(f_{X|Y}(x|y))dxdy$
    - The amount of uncertainty left in X if Y is known
    - Properties
        1. $H(X|Y) = 0$ iff X is a deterministic function of Y
        2. $H(X|Y) = H(X)$ iff X and Y are independent
        3. $H(X|Y) \leq H(X)$ (knowing Y can only reduce the uncertainty about X)
        4. $H(X|Y) = H(X,Y) - H(X)$

6. Mutual Information
   - Shannon Mutual Information
     - Discrete: $I(X;Y) = -\sum_{y\in Y}\sum_{x\in X} p_{xy}\log\left(\frac{p_x p_y}{p_{xy}}\right)$
     - Continuous: $I(X;Y) = -\int f_{XY}(x,y)\log\left(\frac{f_X(x)f_Y(y)}{f_{XY}(x,y)}\right)dxdy$
7. Describes the amount of information that X gives about Y and vice versa.
8. Properties:
   a. Symmetric
   b. I(X; Y) = H(X) – H(X|Y) = H(Y) – H(Y|X)
      The amount of reduction in uncertainty in X by knowing Y, and vice versa.
   c. I(X; Y) >= 0 (equality iff X and Y are independent)
9. Feature selection: take the feature variables Xs with the highest MI with Y (most info).
10. KL Divergence
    - Kullback-Leibler (KL) Divergence
      - Discrete: $D_{KL}(f_1||f_2) = \sum_{x\in X} f_1(x)\log\left(\frac{f_1(x)}{f_2(x)}\right)$
      - Continuous: $D_{KL}(f_1||f_2) = \int f_1(x)\log\left(\frac{f_1(x)}{f_2(x)}\right)dx$

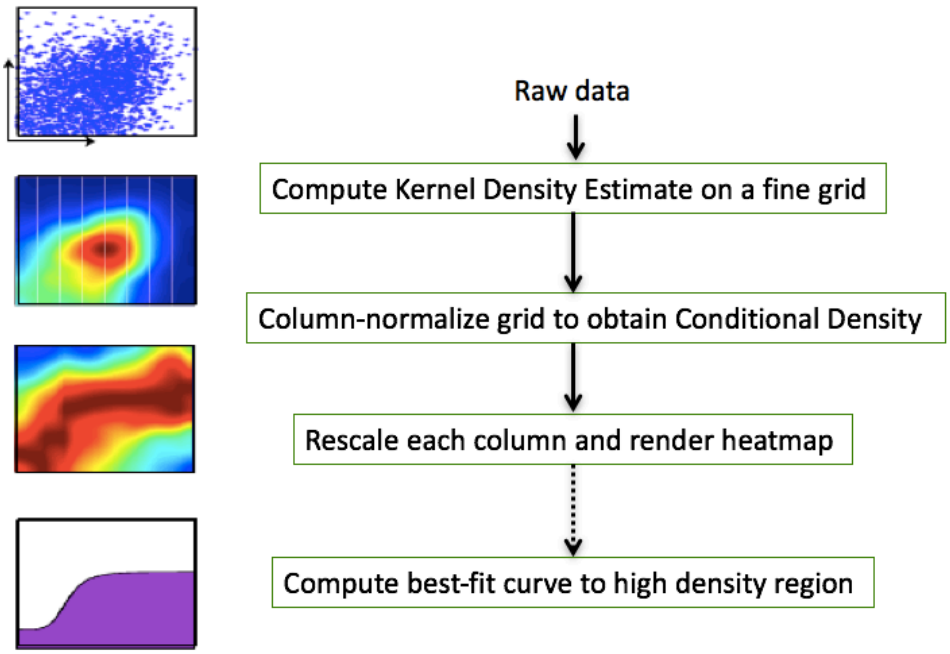
    1. $D_{KL}(f_1||f_2) \geq 0$, equality iff $f_1 = f_2$ a.e.
    2. KL divergence is not symmetric and therefore not a true distance
    3. Divergence is the most general information measure
       - $I(X;Y) = D_{KL}\left(f_{XY}(X,Y)||f_X(X)f_Y(Y)\right)$
       - $H(X) = \log N - D_{KL}(f_X(X)||f_U(U))$ where f_U is the pmf of a uniform random variable

Lecture 14: DREMI and Gene/Protein Regulation

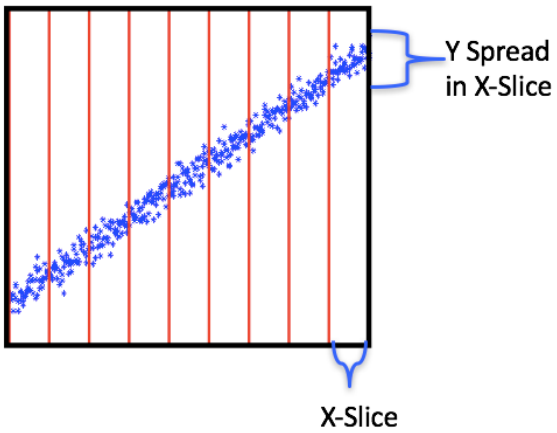
1. DREVI: conditional-**D**ensity **RE**scaled **Vi**sual
    a. Captures behavior across full range, and of small populations
    b. Eliminates conditionally sparse regions to obtain sharper signal
    c. Filters out low values: v < fraction of peak density.
2. DREVI reveals changes in signal transfer relationship
    a. Conditioning on X: all column sums now equal 1.
    b. Look for Δy given Δx, and a difference between two conditions (this suggests some change)



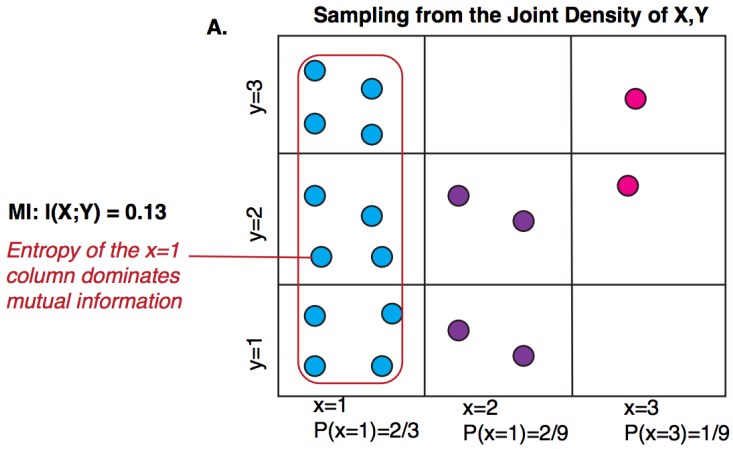
3. Fit a curve to the dense region: linear, (multi) sigmoidal, etc.



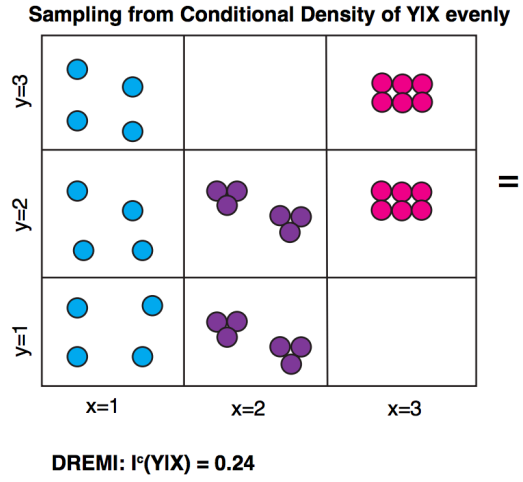
4. Finding joint probability distributions (density estimation)
    a. Diffusion-based KDE uses DCT. Hall estimate uses rule of thumb for std Gaussian.



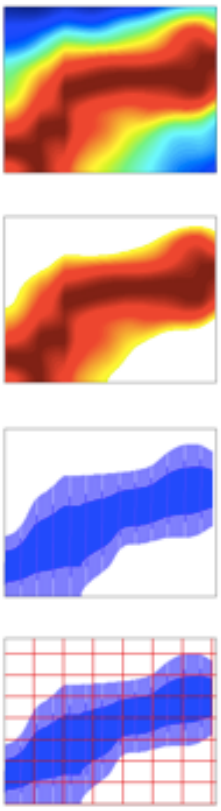
5. At its core mutual information determines how the spread of a variable Y decreases as you condition on another variables X. But mutual information works when there is an even sampling along the dynamic range.
6. Problem
    a. MI is biased by sampling and noise (denoise with high-pass filter)
    b. Most cells exist in a narrow band (conditional density fixes this)
    c. Score is not provided for a full relationship (conditional density fixes this)

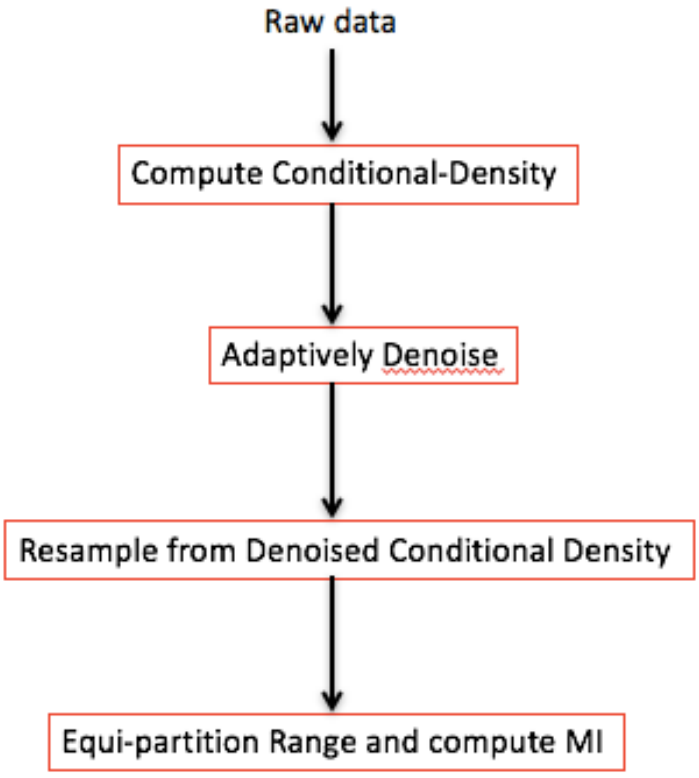


7. Solve by taking the DENOISED, CONDITIONAL density
    a. Ic(X,Y) = Hc (Y|X) –Hc ((Y|X)|X)



8. DREMI





9. Why is DREMI good
    a. Good at adapting to noisy and non-linear relationships.
    b. Good at ignoring mild correlations from cell size / background noise.
    c. Scalable to large datasets