Recitation 4-9-14

EF Lectures #14 & 15
Protein Interactions & Gene Networks
Announcements

• Problem Set 4 due next Thursday (April 17)
• Project write-up due Tuesday, April 22
Outline

• Experimenta methods to detect protein interactions
  – Affinity Purification
  – Tandem Affinity Purification (TAP)
  – Mass Spectrometry
  – Yeast two-hybrid
• Bayesian Networks
• Clustering methods
  – Hierarchical clustering
  – K-means clustering
• Linea Regressio Mutual information
Affinity Purification

- To detect interaction partners of a protein of interest (bait), the bait is tagged by introducing protein-tag DNA construct into cells. Once the construct is expressed and incorporated into cellular complexes, the tag is used to pull down other interacting proteins, by Mass Spectrometry.

- Can do this for every protein to analyze proteins on a proteome-wide scale.

- Fairly high (~30% for 2002 yeast genome-wide study) False Positive Rate with single-affinity purification, but also some False N known interactors & comple
Tandem Affinity Purification (TAP)

-To cut down on false positives, two affinity purification steps

- However, fewer false positives likely means more false negatives


Courtesy of Macmillan Publishers Limited. Used with permission.

Nature Reviews Molecular Cell Biology 4, 74-80
Mass Spectrometry (MS)

- Analytical technique that produces spectra of the masses of atoms or molecules that comprise a sample

- Works by ionizing chemical compounds to generate charged molecules & measuring the mass-to-charge (m/z) ratio

http://dmd.aspetjournals.org/content/35/8/1408/F2.large.jpg

Courtesy of The American Society for Pharmacology and Experimental Therapeutics. Used with permission.


http://jp.physoc.org/content/563/1/11/F1.large.jpg

© Physiological Society Publications. All rights reserved. This content is excluded from our Creative Commons license. For more information, see http://ocw.mit.edu/help/faq-fair-use.

Yeast Two-Hybrid (Y2H)

- Used to detect interactors with your bait protein of interest
- Introduce two plasmids into yeast cells:
  - 1. DNA-binding domain (DBD) - Bait fusion (in this case, **Snf1** is bait)
  - 2. Prey - Activator domain (AD) fusion (in this case **Snf4** is prey)
    - AD needed to recruit PolII for transcription of reporter gene
    - Often will screen a library of potential prey molecules
- This example uses the well characterized Gal4 transcription activator protein in yeast.
- **lacZ** transcription can be detected by colorimetric inspection (can use other reporter genes such as metabolic enzyme (His production) and growth on minimal media lacking His)
- Y2H will miss interactions for prey proteins that are not soluble and/or don’t localize to the nucleus
- But can detect more transient interactions that may not be captured by affinity purification

http://mmbr.asm.org/content/76/2/331/F7.large.jpg
Bayesian Networks

• If we have 3 binary variables A, B, C that we can observe, how many variables do we need to fully specify joint probability $P(A=a, B=b, C=c)$ in the following situations:
  – A, B, C are all independent of each other?
    • $P(A=a, B=b, C=c) = P_A(a) P_B(b) P_C(c)$ 3 parameters (more generally, $n$ for $n$ binary variables since 1 probability (prob. of ON) needed for each)
  – Cannot assume any independencies?
    • Need all possible combinations of A, B, C = $2^3-1$ ($= 2^n-1$ for $n$ binary variables since there are $2^n$ combinations, but last one is determined since all probabilities must sum to 1)
    – The Bayesian network tells us about independencies between variables, and allows us to factor the joint probability accordingly

• A Bayesian networks is a way of representing a set of random variables and their conditional dependencies. Consists of:
  1. Directed (acyclic) graph over the variables
  2. Associated probability distributions:
    • Prior probabilities of all root nodes and
    • Conditional probabilities of all child nodes given their parents
Bayesian Networks

• The directed graph consists of:
  – Nodes = random variables (events)
  – Edges indicate dependencies between variables

• Then the distribution of a random variable depends only on its parent nodes:

A Bayesian network with 4 nodes

- C has parent B, B has child C
- A has parents B and D

Parent and child nodes: if there is directed edge starting from i and ending at j, then i is a parent of j and j is a child of i

Root nodes = nodes with no parents (no incoming edges) – here B and D
Leaf nodes = nodes with no children – here A and C

Need the following probabilities to fully specify the model:
- Prior probabilities of all root nodes = P(B) and P(D)
- Conditional prob. of all child nodes given parents = P(C|B) and P(A|B,D)
Independencies in Bayesian Networks

There are 3 types of connections that can occur between a random variable $B$ and its immediate neighbors $A$ and $C$:

- **Linear**

  Factor $P(A,B,C)$ according to the independencies indicated in this graph:

  $$P(A,B,C) = P(A)P(B|A)P(C|B)$$

  Are $A$ and $C$ independent if $B$ is **unknown**?
  
  **No** – if $B$ is unknown, then knowing $A$ tells us something about $C$ (through unknown $B$)

  Are $A$ and $C$ independent if $B$ is **known**?
  
  **Yes** – if $B$ is known, there is no further information in $A$ about $C$
Independencies in Bayesian Networks

There are 3 types of connections that can occur between a random variable $B$ and its immediate neighbors $A$ and $C$:

**Diverging**

Factor $P(A,B,C)$ according to the independencies indicated in this graph:

$$P(A,B,C) = P(B)P(A|B)P(C|B)$$

Are $A$ and $C$ independent if $B$ is **unknown**?

**No** – if $B$ is unknown, then knowing $A$ tells us something about $C$ (through unknown $B$)

Are $A$ and $C$ independent if $B$ is **known**?

**Yes** – if $B$ is known, there is no further information that $A$ can tell us about $C$
Independencies in Bayesian Networks

There are 3 types of connections that can occur between a random variable $B$ and its immediate neighbors $A$ and $C$:

**Converging**

Factor $P(A,B,C)$ according to the independencies indicated in this graph:

$$P(A,B,C) = P(A)P(C)P(B | A,C)$$

Are $A$ and $C$ independent if $B$ is unknown? Yes – if $B$ is unknown, then knowing $A$ tells us nothing about $C$

Are $A$ and $C$ independent if $B$ is known? No – if $B$ is known, it tells us something about both $A$ and $C$, so $A$ and $C$ are no longer independent
Independencies in Bayesian Networks

Given this graph structure, A and C are marginally independent (e.g. independent when B is marginalized out):

Need to show that $P(A, C') = P(A)P(C')$

$$P(A, C') = \sum_B P(A, B, C') \quad \text{(marginalize out B)}$$

$$= \sum_B P(A)P(C')P(B|A, C')$$

$$= P(A)P(C') \sum_B P(B|A, C')$$

$$= P(A)P(C') \checkmark$$
Bayesian Networks

let this be a Bayesian Network over 5 binary random variables with the following distributions:

\[
P(Ro = 1) = 0.1 \quad P(Pa = 1) = 0.3
\]

\[
P(Ti = 1|Ro, Pa) = \begin{bmatrix} 0.1 & 0.3 \\ 0.5 & 0.9 \end{bmatrix}
\]

\[
P(Co = 1|Ro) = \begin{bmatrix} 0.1 & 0.5 \end{bmatrix}
\]

\[
P(EF = 1|Ti, Co) = \begin{bmatrix} 0.1 & 0.8 \\ 0.4 & 0.9 \end{bmatrix}
\]
Bayesian Networks

What is the probability that your experiment will fail given that there is a new rotation student, but there was no party last night? What is $P(EF = 1 | Ro = 1, Pa = 0)$?

$$P(EF = 1 | Ro = 1, Pa = 0) = \sum_{Ti,Co} P(EF = 1, Ti, Co | Ro = 1, Pa = 0)$$

From graph structure:

$$= \sum_{Ti,Co} P(EF = 1 | Ti, Co) P(Ti | Ro = 1, Pa = 0) P(Co | Ro = 1)$$
Bayesian Networks

\begin{align*}
P(Ro = 1) &= 0.1 \\
P(Pa = 1) &= 0.3 \\
P(Ti = 1 | Ro, Pa) &= \begin{bmatrix} 0.1 & 0.3 \\ 0.5 & 0.9 \end{bmatrix} \\
P(Co = 1 | Ro) &= \begin{bmatrix} 0.1 & 0.5 \end{bmatrix} \\
P(EF = 1 | Ti, Co) &= \begin{bmatrix} 0.1 & 1 \\ 0.4 & 0.9 \end{bmatrix}
\end{align*}

From graph structure:

\[
= \sum_{Ti, Co} P(EF = 1 | Ti, Co) P(Ti | Ro = 1, Pa = 0) P(Co | Ro = 1)
\]

= \[1-P(Ti=1 | Ro=1, Pa=0)\] + \[1-P(Co=1 | Ro=1)\]

4 possible combinations of \{Ti, Co\} to sum over.

For Ti = 0, Co = 0: \(P(EF=1 | Ti=0, Co=0)=0.1\), \(P(Ti=0 | Ro=1, Pa=0)=0.7\), \(P(Co=0 | Ro=1)=0.5\)

For Ti = 0, Co = 1: \(P(EF=1 | Ti=0, Co=1)=0.8\), \(P(Ti=0 | Ro=1, Pa=0)=0.7\), \(P(Co=1 | Ro=1)=0.5\)

Likewise for Ti = 1, Co = 0 and Ti = 1, Co = 1

\[
= (0.1)(0.7)(0.5) + (0.8)(0.7)(0.5) + (0.4)(0.3)(0.5) + (0.9)(0.3)(0.5) = 0.51
\]
Learning Bayesian Networks: parameters for given network

• Given a network structure (vertices and edges) and observations, we can learn the most likely conditional probabilities (e.g. we know a signaling pathway from previous experiments, but would like to determine its probabilities in response to a new stress condition)
  – This is an inference task, in contrast to the previous predictive task. **Maximum Likelihood (ML) estimation – based on observed counts**
  – find parameters (conditional probs.) that maximize the likelihood of the data:
    \[
    \theta_{ML} = \arg \max_{\theta} P(Data|\theta)
    \]
  – example - given structure and observed counts below for binary vars A and B, estimate P(A) and P(B|A):
    \[
    P(A=1) = (4+22)/(15+3+4+22) = 26/44 \approx 0.59
    \]
    \[
    P(B=1|A=0) = 3/(3+15) \approx 0.167
    \]
    \[
    P(B=1|A=1) = 22/(22+4) \approx 0.846
    \]

• **Maximum a posteriori (MAP)**
  – incorporate prior knowledge \(P(\theta)\) about how params are distributed
    \[
    \hat{\theta}_{ML} = \arg \max_{\theta} P(\theta|data) = \arg \max_{\theta} \frac{P(data|\theta)P(\theta)}{P(Data)}
    \]
  – Observed counts plus pseudocounts corresponding to prior
Learning Bayesian Networks: network structure

• There are way too many possible structures for an exhaustive approach (e.g. trying every possible structure and calculating the likelihood of the data given that structure)

• **Common greedy approach (what Pebl does in Pset 4):**
  – start with a random network
  – make a small perturbation (e.g. adding or removing an edge) and rescore network
  – if network scores higher, accept (otherwise reject change)
  – repeat from many starting points, pick best one

• **Simulated Annealing approach:**
  – similar to above, but accept lower scoring network with some probability proportional to difference in scores and temperature
  – accept with higher probability initially, then “lower” temp gradually
Hierarchical Clustering

• Useful when trying to find structure (e.g. clusters of genes upregulated in response to a stress) in your data

• Algorithm:
  – initialize every point to be its own cluster
  – until only 1 cluster left:
    • calculate distance between each cluster and all other clusters: $O(N^2)$ for each connection -> $O(N^3)$ overall
    • find the two closest clusters, merge them into one cluster

• Can use various distance/similarity metrics (e.g. Euclidean distance, correlation, etc.)
Hierarchical Clustering

- Let the following be 7 points in a 2-dimensional dataset – we want to do agglomerative hierarchical clustering on these points, using Euclidean distance as distance metric
Hierarchical Clustering

- **(initialization)** – initialize each point to be its own cluster

Build dendogram as we go to keep track of clusters – initially all nodes of dendogram are unconnected, connect them as we merge points into clusters
Hierarchical Clustering

- (repeat until only 1 cluster left) – calculate distances between each pair of clusters, merge the two closest into single cluster

Closest are points 2 and 3 – merge these into a single cluster which we’ll call Update dendogram:
Hierarchical Clustering

• **(repeat until only 1 cluster left)** – calculate distances between each pair of clusters, merge the two closest into single cluster
  – how do we do this for clusters with more than 1 point?

Let cluster A contain the set of points i and cluster B contains the set of points j, then the distance between A and B is:

Option (1): **Single or complete linkage**
- Calculate all distances $d_{ij}$ between points i in A and all points j in other cluster B, and consider $\text{dist}(A,B) = \min(d_{ij})$ for single linkage, $\text{dist}(A,B) = \max(d_{ij})$ for complete linkage

Option (2): **Centroid linkage**
- For clusters A and B, compute the “centroid” or geometric center of the points in the cluster $A_C$ and $B_C$, and $\text{dist}(A,B) = \text{dist}(A_C, B_C)$
Hierarchical Clustering

• (repeat until only 1 cluster left) – calculate distances between each pair of clusters, merge the two closest into single cluster
  – use centroid linkage

Closest clusters are points 4 and 6 – merge these into a single cluster B
Update dendogram:
Hierarchical Clustering

• (repeat until only 1 cluster left) – calculate distances between each pair of clusters, merge the two closest into single cluster
  – use centroid linkage

Closest clusters are B and 5 – merge these into a single cluster B
Update dendogram:
Hierarchical Clustering

• (repeat until only 1 cluster left) – calculate distances between each pair of clusters, merge the two closest into single cluster
  – use centroid linkage

Closest clusters are A and 1 – merge these into a single cluster
Update dendogram:
Hierarchical Clustering

• (repeat until only 1 cluster left) – calculate distances between each pair of clusters, merge the two closest into single cluster
  – use centroid linkage

Closest clusters are B and 7 – merge these into a single cluster B
Update dendogram:
Hierarchical Clustering

• (repeat until only 1 cluster left) – calculate distances between each pair of clusters, merge the two closest into single cluster
  – use centroid linkage

Only two clusters left, merge them.
Update dendogram:

Only one cluster remaining, so we’re done!
Hierarchical Clustering

- Can always cluster data, get a dendrogram and discover some “structure” in your data, but interpreting or assigning meaning to clusters is much more difficult
  - clusters may not corresponding to anything biologically meaningful

- In contrast to agglomerative (“bottom-up”) clustering shown thus far, there is also divisive hierarchical clustering (top-down):
  - start with everything in one cluster, then cut the cluster into 2, then cut those clusters, etc., until you have the desired number of clusters
K-means clustering

• Goal: Find a set of $k$ clusters that minimizes the distances of each point in the cluster to the cluster’s mean

• You must a priori select $k$, the number of clusters to return

• Algorithm:
  – For all points $X_i$:
    • Assign $X_i$ to the cluster with the closest mean
  – Recalculate the mean of each cluster based on previous iteration’s assignments

Repeat until convergence (no assignments change)
K-means clustering example ($k=4$)

http://shabal.in/visuals/kmeans/2.html
K-means clustering

- Deterministic given:
  1. Choice of $k$
  2. The $k$ starting points for the clusters

- For 2: Generally want to run many times with different starting points to obtain most robust partition

- For 1:
  - Try many different $k$s (below and above what you think it might be)
  - Intuitively, you should see large decreases in the intra-cluster distance when uncovering true underlying clusters & smaller decreases when overfitting

[Graph showing the decrease in within-cluster distance with increasing $k$]

Big steps occur when we are dividing data into natural clusters
Smaller steps occur when we are overclustering
K-means clustering

• Deterministic given:
  – 1. Choice of $k$
  – 2. The $k$ starting points for the clusters

• For 2: Generally want to run many times with different starting points to obtain most robust partition

• For 1:
  – Try many different $k$s (below and above what you think it might be)
  – Intuitively, you should see large decreases in the intra-cluster distance when uncovering true underlying clusters & smaller decreases when overfitting
  – Decision can be made automatically through frameworks such as Bayesian Information Criterion (BIC – penalizes addition of more free parameters; accepts model (i.e., $k$) that optimizes a tradeoff between increased likelihood of data from more clusters and increased number of free parameters)
Variations on K-means clustering

• Fuzzy k-means:
  – Rather than hard assignments (assigning each point to strictly 1 cluster), give soft assignments \( u_{i,j} (\mu_{i,j}) \) for all points \( 1 \leq i \leq N \), clusters \( 1 \leq j \leq K \)
    - Constraint is \( \sum_{j=1}^{K} \mu_{i,j} = 1 \)
    - Consider these soft assignments when recalculating the cluster means:
      \[
      \hat{Y}_j = \frac{\sum_{i=1}^{N} \mu_{i,j} X_i}{\sum_{i=1}^{N} \mu_{i,j}}
      \]

• k-medioids: restrict ourselves to the actual data points
  – Rather than the mean (which likely doesn’t correspond exactly to any data point), have the cluster center be the data point closest to the mean
Regression-based modeling

- Relevant if you assume a number of variables (e.g. transcription factors) have independent linear effects

\[ Y_g = \sum_{t \in T_g} \beta_{t,g} X_t + \varepsilon \]

- \( \beta_{t,g} > 0 \): Transcription factor t positively regulates gene g
- \( \beta_{t,g} < 0 \): Transcription factor t negatively regulates gene g

- Often, we only want to consider TFs with a large impact on gene expression definitely above noise, so we set a minimum threshold for \( \beta \) or maximum number of nonzero \( \beta \) (other shrinkage methods possible)
Nonlinear effects on gene expression

• **Mutual information** between pairs of gene expression measurements can detect complex, nonlinear regulatory relationships
  – Feed forward loops
  – Cooperativity (multiple subunits to dimerize or multimerize before functional activity)