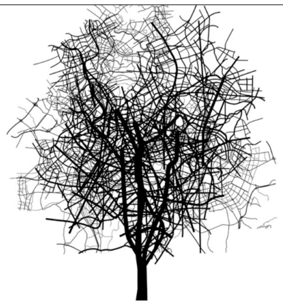


Pattern discovery in biological data

Omar Wagih | BCB410H | 23-11-11

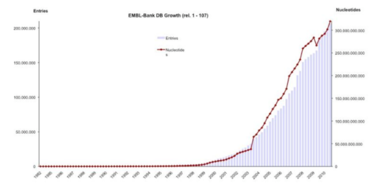
Brief outline

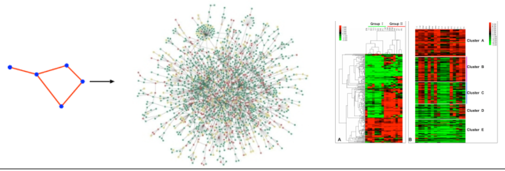
- Introduction to pattern discovery
  - Definition
  - Importance of patterns
- DNA Sequence pattern discovery
  - Alignment based methods
  - Non-alignment based methods
- Pattern discovery in High-throughput screening data
  - Hierarchical clustering

Definition

- "**pattern matching** is the act of checking some sequence of tokens for the presence of the constituents of some pattern"
- Not to be confused with **pattern recognition**, where the match usually has to be exact

Introduction





Importance of patterns in biological data

- Sequences
  - Identification of functionally important repeats
    - Trinucleotide repeat diseases: fragile-X mental retardation, Huntington's disease, myotonic dystrophy etc…
    - Gene regulation via interaction with TFs & alter structure of chromatin or act as protein binding sites
- High-throughput screening data
  - Similar profiles across screens
    - E.g. in protein interaction networks, usually resembles physical or functional relations
    - Used to define functional modules or genes with unknown functions

Many types of pattern recognition

- We will focus on two types
  - k-nucleotide repeats (Fully alignment based)
  - k-tuple matching (Non-alignment based)

Sequence pattern discovery: Alignment based

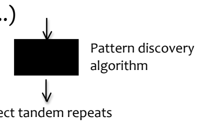
- **Benson and Waterman, 1995:** A method for fast database search for all k-nucleotide repeats
- **Kannan and Myers, 1996:** algorithm for finding the two non-overlapping substrings of a given string of length, which have the highest-scoring alignment between them

1. A method for fast database search for all k-nucleotide repeats

1. Scan sequence for **suspicious** patterns. For all suspicious patterns detected do the following:
    1. Compute a similarity score for the pattern and the sequence where it was found
    2. If similarity score exceeds some threshold:
        1. Align pattern and sequence
        2. Determine a consensus pattern and re-compute alignment with consensus sequence
        3. Report sequence identification and alignment

Process

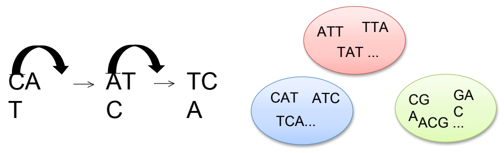
- Some database of sequences
- Parameters (e.g. k, score threshold, matrices for similarity scores...)



Two things to consider (sometimes counterpoise each other):
1. Efficiency
2. Sensitivity

1.Finding suspicious patterns

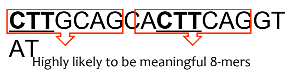
- **E.g.** for p=3, there are 4^3 there are 64 possible 3 letter words
- Can reduce this number by grouping cyclic rotations



- **Reduced to 24 district classes of patterns**
- Feasible to search database with just 24 district classes

1.Finding suspicious patterns

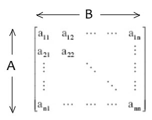
- **Great! What about for larger ps?**
    - For p = 8, the number of distinct classes = **8230**
    - For p = 15, the number of distinct classes = **71,582,716!**
- Impossible to search that many across database
- Not all classes represent biologically meaningful repeats
- To trim the impractical number of classes for larger ps (e.g. 8):
    - Identify patterns for smaller p (e.g. 3)



2.Calculating similarity scores

- **Recall:** Need to determine if the pattern belongs to a tandem repeat through calculation of a similarity score of pattern and local region of the sequence

Consider two sequences of **n** bases $A_i = a_1a_2...a_n$ and $B_j = b_1b_2...b_m$ we can calculate a similarity score by filling out the the matrix:



$$\begin{bmatrix} a_{11} & a_{12} & \cdots & \cdots & a_{1n} \\ a_{21} & a_{22} & & & \vdots \\ \vdots & & \ddots & & \vdots \\ \vdots & & & \ddots & \vdots \\ a_{n1} & \cdots & \cdots & \cdots & a_{nn} \end{bmatrix}$$

Using...

$$S(i, j) = \max \begin{Bmatrix} S(i-1, j-1) + \mu(i, j) \\ S(i-1, j) + \delta \\ S(i, j-1) + \delta \\ 0 \end{Bmatrix}$$

$$S(i, 0) = 0, \; S(0, j) = 0$$

$\mu(i, j)$	Binary value: match or mismatch (e.g 2 & -1)
δ	Value given to an indel of a base (e.g. -2)

2.Calculating similarity scores

- We have 2 sequences A and B where **A** is the database sequence and **B** is a sequence of a repeated pattern (p), **k times**
- We don't know before hand, how many repeats k we're going to find of our pattern of length p
- To avoid missing a long string of repeats, k should be large
- If k is large, it will be very computing similarity scores will be computationally consuming
- **What do we do?**

Wraparound technique: Using <u>one</u> single instance of the pattern, the similarity score can be calculated by wrapping around at the end of each p

For each row, compute 2 passes:

First pass: $S(i, 1) = 0$

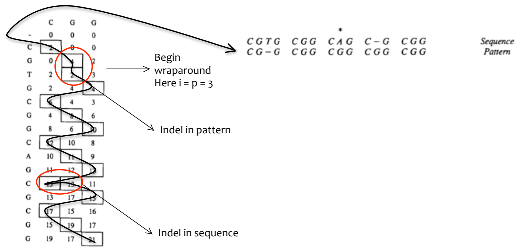
Second pass: $S(i, 1) = \max \left\{ \begin{array}{l} S(i-1, p) + \mu(i, j) \\ S(i-1, 1) + \delta \\ S(i, p) + \delta \\ 0 \end{array} \right\}$

We then maximize the score matrix and compare to some threshold parameter: If similar enough, do alignment

3.Alignment

"Is a representation of two sequences indicating which bases are match, substituted, inserted and deleted"

For an optimal alignment: start at the maximum value and trace back to determine where each value came from



4.Consensus patterns

- Many suspicious patterns may pass the alignment score threshold. But are they the best?
- Need to find some consensus sequence which defines the alignment more uniquely

4.Consensus patterns

Example: given a suspicious pattern **P = ACGTT**, its alignment produces:

```
 **       *             *   *
ACGAA ACGGTA -CGTT ACGT- AGGTA A
ACGTT ACG-TT ACGTT ACGTT ACGTT A
```

By selecting the majority base for each position: we get a consensus pattern: **P_c=ACGT<u>A</u>** which produces the alignment:

```
 *               *       *
ACGAA ACGGTA -CGTT ACGT- AGGTA A
ACGTA ACG-TA ACGTA ACGTA ACGTA A
```

Which contains **two more matches, one less substitution** and **two less indels** than the previous alignment

2. Sequence pattern discovery: non-alignment based

- Alignment based pattern matching is computationally intensive due to **full scale matrix operations**
- k-tuple matching: **Benson, 1999**
  - No need for full scale matrix operations
  - No previous knowledge of the pattern required (pattern size/number of copies)
  - No restrictions on the size of the pattern

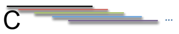
How does it work?

k-tuple matching

- Search for matching nucleotides (k-tuples) separated by a common distance *d*
- k-tuples is a window of k consecutive characters from nucleotide sequence

Example: size of window=6

CATGGCTCATTG
C



k-tuple matching: identification of tandem repeats candidates

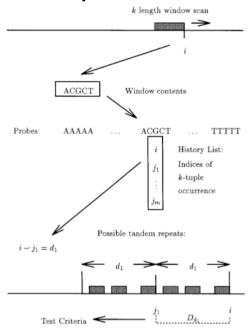
- Let S be some nucleotide sequence.
- Let k (window size) be some small integer (e.g. 5)
- Let P be the set of all possible k-mers (4^k) called *probes*
- Let H_p be the list of positions containing the position *i* where each probe occurs in the sequence

1. Slide the window of length **k** (5 in this case) across the sequence
2. Identify the probe and store its start position ***i*** in $\boldsymbol{H_p}$

3. When each ***i*** for probe **p** is added to $\mathbf{H_p}$, check $\mathbf{H_p}$ for **ALL** previous occurrences of the same probe
    - Let one earlier occurrence of probe ***p*** be at position ***j***
    - Distance ***d = i-j*** becomes a possible pattern size for a tandem repeat

4. The distance between i and j is stored in a list of distances D_d: to keep track of k-tuple matches at the same distance

k-tuple matching: identification of tandem repeats candidates



Selection of statistically significant predicted tandem repeats

- Statistical criteria based on runs of heads in Bernoulli sequences (number of matches detected in the k-tuples within distance ***d***)

Example (window size=5):

CATGGTCATTGTGAACAT
GG

Runs of heads:
HHHHHFHHHHHFFFFH
HHHH

- Selection criteria is based on four distributions, which depend on
    1. Pattern length (***d***)
    2. Matching probability ($\boldsymbol{p_M}$)
    3. Indel probability ($\boldsymbol{p_I}$)
    4. Tuple size (***k***)

Selection criteria

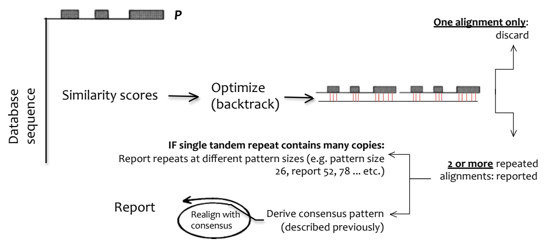
- Four distributions:
    1. Sum of heads distributions: indicates how many matches are required
    2. Random walk distribution: describes how distances between matches vary due to indels
    3. Apparent size distribution: distinguish between tandem repeats and non-tandem direct repeats
    4. Waiting time distribution: Used to pick tuple sizes (longer tuples, less matches, less running time)

Cutoff for each distribution is calculated using a formulae or simulation

- Explanation of each distribution is pretty dense so will not go into detail
- Algorithm will proceed with predicted tandem repeats that pass the selection criterion of all four distributions

Alignment

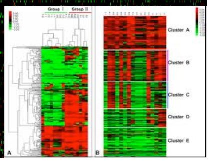
- Given a pattern ***P*** of length ***d*** passes the selection criteria it is aligned using **wraparound** dynamic programming (described earlier)



P
Database sequence
Similarity scores
Optimize (backtrack)
One alignment only: discard
2 or more repeated alignments: reported
IF single tandem repeat contains many copies:
Report repeats at different pattern sizes (e.g. pattern size 26, report 52, 78 … etc.)
Derive consensus pattern (described previously)
Realign with consensus
Report

Pattern discovery in high-throughput screens data

- Difficult to make any inferences from the raw data
- Want to find patterns among the profiles of proteins/genes
- Can group the similar profiles for genes/proteins using hierarchical clustering



Pattern discovery in high-throughput screens data

1. Start by assigning each item to a cluster, so that if you have N items. Let the distances (similarities) between the clusters the same as the distances (similarities) between the items they contain.
2. Find the closest (most similar) pair of clusters and merge them into a single cluster, so that now you have one cluster less.
3. Compute distances (similarities) between the new cluster and each of the old clusters.
4. Repeat steps 2 and 3 until all items are clustered into a single cluster of size N.

Distance metric: Distances between two points

- Most common:
  - Euclidean distance $\|a-b\|_2 = \sqrt{\sum_i (a_i - b_i)^2}$
  - squared Euclidean distance $\|a-b\|_2^2 = \sum_i (a_i - b_i)^2$
  - Manhattan distance $\|a-b\|_1 = \sum_i |a_i - b_i|$

Linkage criteria: distance between clusters as a function of the pairwise distances

- Complete-linkage

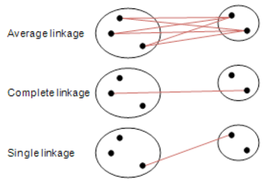
$$\max\{\, d(a,b) : a \in A,\, b \in B \,\}.$$

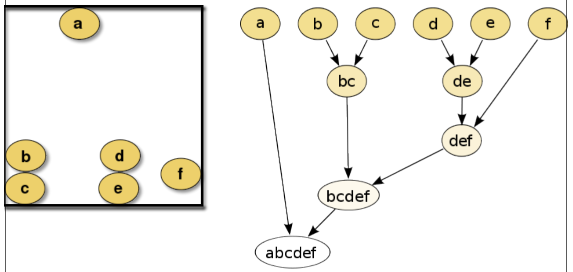
- Single-linkage

$$\min\{\, d(a,b) : a \in A,\, b \in B \,\}.$$

- Average-linkage

$$\frac{1}{|A||B|}\sum_{a\in A}\sum_{b\in B} d(a,b).$$

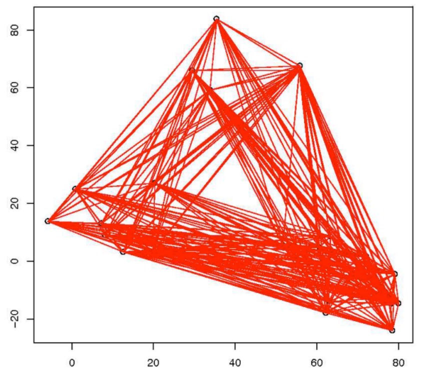




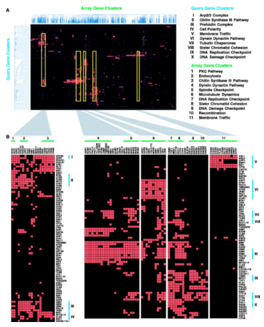
Note: Each pairwise distances between clusters are stored in a distance matrix used to generate the dendrogram

Agglomerative *vs* divisive hierarchical clustering

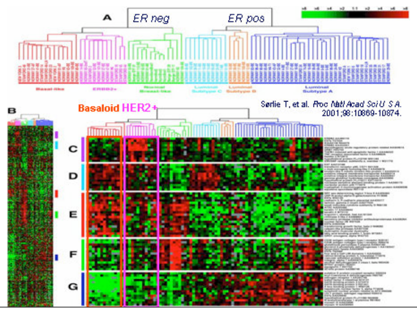
- Agglomerative merges clusters, bottom up
- Divisive does the reverse by starting with all objects in one cluster and subdividing them into smaller pieces



Examples of pattern discovery: **genetic interaction screens**



Examples of pattern discovery: **gene expression profiling**



Thank you!