Motif Discovery

Pål Sætrom

Topics

- Introduction – What controls co-expressed genes?
- Motifs
  - Prosite
  - Position Weight Matrix
- Motif-discovery – exact methods
- Motif-discovery – randomized methods

Which genes are affected by our inner clock?

1. Measure gene expression every 4 hours for 2 days
2. Identify cycling genes (cosine pattern, T = 20-28h)
3. What controls (evening) cycling genes’ expression?

Microarrays measure global gene expression

1. Isolate RNA
2. Reverse-transcribe to cDNA
3. Label cDNA
4. Hybridize
5. Scan
6. Analyze

Microarrays measure global gene expression (2)

- Array of probes measure gene RNA levels
- “All” genes in one sample
- Microarrays identify co-expressed genes

Transcription regulation

- Proteins (transcription factors - TFs) bind short DNA segments
  - Initiate transcription (core promoters)
  - Activate transcription (enhancers)
  - Repress transcription (silencers)
- Location of sequence elements varies
  - Core promoter close to transcription start site
  - Enhancers and silencers many kb away
- Problem: Identify DNA elements causing co-expression
Motif finding problem

- Find motif common to genes with common expression patterns

**Input:** Set of genes, G  
**Output:** Motif common to all genes in G (and significant)

*Easy?*

```
<table>
<thead>
<tr>
<th>Protein</th>
<th>Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>A45_SSV1/4-25</td>
<td>Clr..CggiFnkrrevveHllvg.H</td>
</tr>
<tr>
<td>ABC3_LAGLA/285-305</td>
<td>Cfs..CaekVaeflqenpHvnl..H</td>
</tr>
<tr>
<td>ABRU_DROME/546-567</td>
<td>Cpk..CgkiYrsahtlrtHledk.H</td>
</tr>
<tr>
<td>ACE1_TRIRE/402-424</td>
<td>CrepgCtkeFkrpcdltkHekt..H</td>
</tr>
<tr>
<td>ACE2_SCHPO/445-467</td>
<td>ClyngCnkrIarkynvesHiqt..H</td>
</tr>
<tr>
<td>ACE2_SCHPO/475-495</td>
<td>Cdl..CkagFvrhhdlkrHlri..H</td>
</tr>
<tr>
<td>ACE2_YEAST/605-627</td>
<td>ClypnCnkvFkrrynirsHiqt..H</td>
</tr>
<tr>
<td>ACE2_YEAST/635-657</td>
<td>CdfpgCtkaFvrnhdlirHkis..H</td>
</tr>
<tr>
<td>ADNP2_HUMAN/772-793</td>
<td>Clf..CpctFhdikglseHsrnr.H</td>
</tr>
<tr>
<td>ADNP2_HUMAN/877-899</td>
<td>Cpf..CfgpFvtteayelHlkerhH</td>
</tr>
<tr>
<td>ADNP2_MOUSE/802-823</td>
<td>Clf..CpctFhdvrglveHsrtk.H</td>
</tr>
<tr>
<td>ADNP2_MOUSE/907-929</td>
<td>Cpf..ClstFmtadayelHlkerhH</td>
</tr>
<tr>
<td>ADNP_HUMAN/514-535</td>
<td>Cpy..CrstFndvekmaaHmrmv.H</td>
</tr>
<tr>
<td>ADNP_MOUSE/233-254</td>
<td>Cpy..CrstFndvekmaaHmrmv.H</td>
</tr>
</tbody>
</table>
```

```
Zinc finger domains bind DNA and RNA

Formal description of domain?

Prosite patterns define protein motifs

```
C-\(x(2,4)-C-x(3)-[\text{LIVMFYWMC}]-x(8)-H-x(3,5)-H\)
```

- Regular expressions
  - Concatenations: -
  - Alternatives: [ ]
  - Wildcards: x(l, h)
- Model the dominant positions in the domain
- **Limitations?**

Identifying motifs by counting \(l\)-mers

1. Create index of all \(l\)-mers
2. Motifs are \(l\)-mers that occur in all (most) sequences
3. Compare actual occurrences with reference (negative control) set

Binding sites for Myc are ambiguous

- Consensus string too strict (3 of 7)
  - CACGTG
- Pattern too loose
  - C[AG][TG][TAC]G
  - CGGGAG, CGGCCG
- **Solution?**
Profiles are weighted sequence patterns

- Position weight matrix (PWM)
- Position-specific scoring matrix (PSSM)
- Position-specific weight matrix (PSWM)

Different alignments give different PWMs

<table>
<thead>
<tr>
<th>Last left by 1</th>
<th>Last right by 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>A 1 0 0 0 1 0 0</td>
<td>A 2 0 1 0 1 0 0</td>
</tr>
<tr>
<td>C 3 6 1 5 0 1 0</td>
<td>C 2 6 1 5 1 0 0</td>
</tr>
<tr>
<td>G 3 0 1 2 4 1 7 5</td>
<td>G 3 1 1 1 4 1 6 6</td>
</tr>
<tr>
<td>T 0 0 0 0 0 1 4 0 1</td>
<td>T 0 0 0 0 0 4 1 1</td>
</tr>
</tbody>
</table>

Best PWM?

Length and set of positions define PWM

- Set of \( t \) DNA sequences with \( n \) nucleotides each
- PWM length \( l \)
- PWM given by \( s = (s_1, s_2, \ldots, s_t), 1 \leq s_i \leq n - l + 1 \)

PWM scores define "best" PWM

- Consensus score:
  \[
  \text{Score}(s) = \sum_{j=1}^{l} M_j(s_j)
  \]
  - \( M_j(s_j) \) is max in column \( j \) in PWM
- Entropy:
  \[
  \text{Entropy}(s) = - \sum_{j=1}^{l} \frac{M_j(s_j)}{l} \log \frac{M_j(s_j)}{l}
  \]
  - \( M_j(s_j) \) is count at element \((i,j)\) in PWM

Using consensus score to find best PWM

<table>
<thead>
<tr>
<th>Last left by 1</th>
<th>Last right by 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>A 1 1 5 0 0 1 0 0</td>
<td>A 2 0 1 0 1 0 0</td>
</tr>
<tr>
<td>C 3 6 1 5 0 1 0 1</td>
<td>C 2 6 1 5 1 0 0 0</td>
</tr>
<tr>
<td>G 3 0 1 2 4 1 7 5</td>
<td>G 3 1 1 1 4 1 6 6</td>
</tr>
<tr>
<td>T 0 0 0 0 0 1 4 0 1</td>
<td>T 0 0 0 0 0 4 1 1</td>
</tr>
</tbody>
</table>

Motif finding problem (feasible definition)

- Given set of DNA sequences, find \( l \)-length PWM that maximizes consensus score

Input: A \( t \times n \) matrix of DNA, and the length \( l \)
Output: An array of \( t \) starting positions \( s = (s_1, s_2, \ldots, s_t) \) maximizing \( \text{Score}(s) \)

Easy?
Consensus score = counting mismatches with consensus string

- Max consensus score: \( l \)
- Min consensus score: \( l/4 \)

\[
\text{Score}(s) = \sum_{i=1}^{l} \left( M(s,i) - l + d_w(s,i) - l - \frac{1}{4} d_w(s,i) \right)
\]

- \( w \) is consensus string of PWM
- \( d_w(s,i) \) is Hamming distance between \( w \) and \( s \)

Solving motif finding = finding string with minimum total Hamming distance.

Median string problem

- Given set of DNA sequences, find \( l \)-mer median string

Input: A \( t \times n \) matrix of DNA, and the length \( l \)
Output: A string \( v \) of \( l \) nucleotides (\( l \)-mer) with minimal total Hamming distance \( d_H(v,s) \) of all possible \( l \)-mers

Easier?

Naïve motif finding and median string solutions

- Motif finding
  - Find optimal starting positions \( s = (s_1, s_2, ..., s_t) \)
  - Consider all \((n - l + 1)^t \) starting positions

- Median string
  - Find optimal \( l \)-mer
  - Consider all \( 4^l \) \( l \)-mers (DNA)

Brute force motif finding

\( \text{BruteForceMotifSearch}(t, n, l) \):
1. bestScore = 0
2. for each \((s_1, s_2, ..., s_t) \) from \((1, ..., 1) \) to \((n - l + 1, ..., n - l + 1) \):
   1. if Score\((s)\) > bestScore:
      1. bestScore = Score\((s)\)
      2. bestMotif = \((s_1, s_2, ..., s_t)\)
3. return bestMotif

\[ O((n - l + 1)^t) \times O(l) = O(ln^t) \]

Brute force median string

\( \text{BruteForceMedianString}(t, n, l) \):
1. bestWord = AA...AA
2. bestDistance = \( \infty \)
3. for each \( l \)-mer \( word \) from A...A to T...T:
   1. if \( d_H(word,s) < bestDistance \):
      1. bestScore = Score\((s)\)
      2. bestWord = \( word \)
4. return bestWord

\[ O(4^l) \times O(nl) = O(nl4^l) \]

Using tree to implement for loop in brute force motif finding

- Store starting positions in leaves
  - \((n - l + 1)^t \) starting positions
  - \( "l-mer" \) in alphabet of \((n - l + 1) \) symbols

- Traverse tree (ignoring internal nodes)

\[
\text{NextLeaf}(a, L, k):
1. for \( i \) in range\((L, 1, -1)\):
   1. if \( a < k \):
      1. \( a = a + 1 \)
      2. return \( a \)
2. \( a = 1 \)
2. return \( a \)
Branch and bound motif finding

BranchAndBoundMotifSearch(t, n, l):
1. bestMotif = s
2. bestScore = 0
3. for each l-mer word from A…A to T…T:
   1. if \( d_s(word, s) \) < bestDistance:
      1. bestScore = Score(s)
      2. bestWord = word
   4. return bestWord
O(4^l) * O(nt) = O(nt4^l)

Conservative bound on median string

- Internal nodes represents word prefixes
  - Skip subtrees where prefix has higher distance than current best
  - Might be some extension with 0 distance

5.1.1: \( \text{optimisticDistance} = d_s(word, s) \)
Identifying motifs by counting l-mers

1. Create index of all l-mers
2. Motifs are l-mers that occur in all (most) sequences
3. (Compare actual occurrences with reference (negative control) set)

Ex: aaaaa - aaaaaa
aaaaac
aaaaag
aaaaat

Solve median string problem?

Ex:

Alignment

Profile

Median

A greedy approach to motif discovery

1. Select random starting positions
2. Create profile, P
3. Use P to find best-scoring locations
4. Update P

A greedy algorithm for motif discovery

GreedyProfileMotifSearch(t, n, l):
1. Randomly select starting positions s = (s₁, s₂, ..., sₜ)
2. Form profile P from s
3. bestScore = 0
4. while Score(s) > bestScore:
   1. bestScore = Score(s)
   2. for i in range(t):
      1. Find the highest scoring l-mer a from ith sequence
      2. sᵢ = starting position of a
5. return s

Characteristics? n vs. t?

Median may not be in any input string

Alignment

Profile

Median

A greedy algorithm for motif discovery

GreedyProfileMotifSearch(t, n, l):
1. Randomly select starting positions s = (s₁, s₂, ..., sₜ)
2. Form profile P from s
3. bestScore = 0
4. while Score(s) > bestScore:
   1. bestScore = Score(s)
   2. for i in range(t):
      1. Find the highest scoring l-mer a from ith sequence
      2. sᵢ = starting position of a
5. return s

Characteristics? n vs. t?

Optimization in energy landscapes

- Gradient descent will find local minima
  - GreedyProfileMotifSearch:
    1. Find the highest scoring l-mer a from ith sequence
    2. sᵢ = starting position of a
- How to handle local minima?
Gibbs sampler for motif discovery

- **GibbsProfileMotifSearch**
  1. Randomly select starting position \( s = (i_1, i_2, ..., i_l) \)
  2. Randomly choose sequence \( l-mer \)
  3. Form profile \( P \) from \( s \)
  4. Calculate probability \( P \) of \( l-mer \) starting at position \( i_l \) from \( P \)
  5. Randomly choose new starting position \( n \) based on \( P_{n}\) : 
  6. Go to 2

- **GreedyProfileMotifSearch**
  1. Randomly select starting position \( s = (i_1, i_2, ..., i_l) \)
  2. Form profile \( P \) from \( s \)
  3. Set \( s \) with probability \( \exp(-\Delta E/kT) \)
  4. Go to 2

**Energy landscapes and Gibbs sampling**

- From statistical mechanics
  - State, \( S \)
  - Probability of finding a system in state \( S \), \( P(S) \)
- Simulating system of discrete states
  - Allow suboptimal state \( S \) with probability \( \sim P(S) \)
  - Focus on neighboring states to narrow search

**Computing probability of l-mer from \( P \)**

\[
P(w|P) = \prod_{i=1}^{l} P_w(i) = \frac{1}{7^l}\begin{cases} 
1 & \text{if} \, w \in \{ACACGTGG, ACACGGAGT, TACACGCTGG\}, \\
0 & \text{otherwise}
\end{cases}
\]

- \( P(ACACGTGG, P) = 1/7^l \times 1/7^6/7^1 ) \times 1/7^1 \times 1/7^5/7^1 \times 1/7^6/7 \)
- \( P(ACACGGAGT, P) = 1 ) \times 1/7^l \times 1/7^1 \times 1/7^1 ) \times 1/7^1 \times 1/7 \)
- \( P(TACACGCTGG, P) = 0 \)
- \( P(TACACGCTGG, P) = 0 \) (unacceptable)
Alternative target functions

- **Entropy**
  \[ \text{Entropy}(s) = -\sum_{i=1}^{T} p_i \log p_i \]

- **Relative entropy**
  \[ \text{RelativeEntropy}(s) = -\sum_{i=1}^{T} \frac{p_i \log \frac{p_i}{p}}{p} \]

\[ e^{-\Delta E/kT} \]

1-mer counting revisited

- Consensus likely infrequent
  - GCACGTGG (1 of 7)
  - CACGTG (3 of 7)

- ...but consensus with mismatches likely frequent
  - xCxCxCGxG (5 of 7)
  - xxCAxGxGG (5 of 7)
  - xxCACGxGG (5 of 7)

- Index random (k of l)-mers
- Remember frequent positions (in DNA)
- Use most frequent positions

A greedy approach to motif discovery

**GreedyProfileMotifSearch** \( (t, n, l) \):
1. Randomly select starting positions \( s = (s_1, s_2, \ldots, s_t) \)
2. Form profile \( P \) from \( s \)
3. bestScore = 0
4. while Score(\( s \)) > bestScore:
   1. bestScore = Score(\( s \))
   2. for \( i \) in range(\( t \)):
      1. Find the highest scoring \( l \)-mer \( a \) from \( i \)th sequence
      2. \( s_i = \) starting position of \( a \)
5. return \( s \)

Characteristics? \( n \) vs. \( t \)?

Greedy (gradient descent) optimization

- Optimization in energy landscapes
- Gradient descent will find local minima
  - GreedyProfileMotifSearch:
    1. Find the highest scoring \( l \)-mer \( a \) from \( i \)th sequence
    2. \( s_i = \) starting position of \( a \)
- Need means to escape
- How?

Allow moves “uphill”

- From statistical mechanics
  - State, \( S \)
  - Probability of finding a system in state \( S \), \( P(S) \)
- Simulating system of discrete states
  - Allow suboptimal state \( S \) with probability \( P(S) \)
  - Focus on neighboring states to narrow search

Gibbs sampling (in general)

1. Choose random starting state \( S \)
2. Choose random neighbor state \( S' \)
3. if \( P(S') \leq P(S) \):
   1. \( S = S' \)
4. else
   1. \( \Delta E = P(S') - P(S) \)
   2. Set \( S = S' \) with probability \( e^{-\Delta E/kT} \)
5. Go to 2
Energy landscapes and Gibbs sampling

Gibbs sampler for motif discovery

Computing probability of l-mer from P

Alternative target functions

l-mer counting revisited