

Gibbs Sampling in Motif Finding

Gibbs Sampling (1)

- Given:
 - x^1, \dots, x^N ,
 - motif length K ,
 - background B ,
 - Find:
 - Model M
 - Locations a_1, \dots, a_N in x^1, \dots, x^N
- Maximizing log-odds likelihood ratio:

$$\sum_{i=1}^N \sum_{k=1}^K \log \frac{M(k, x_{a_i+k}^i)}{B(x_{a_i+k}^i)}$$

Gibbs Sampling (2)

- AlignACE: first statistical motif finder
- BioProspector: improved version of AlignACE

Algorithm (sketch):

- Initialization:
 - Select random locations in sequences x^1, \dots, x^N
 - Compute an initial model M from these locations
- Sampling Iterations:
 - Remove one sequence x^i
 - Recalculate model
 - Pick a new location of motif in x^i according to probability the location is a motif occurrence

Gibbs Sampling (3)

Initialization:

- Select random locations a_1, \dots, a_N in x^1, \dots, x^N
- For these locations, compute M :

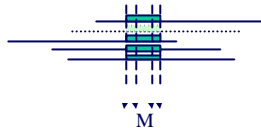
$$M_{kj} = \frac{1}{N} \sum_{i=1}^N (x_{a_i+k}^i = j)$$

- That is, M_{kj} is the number of occurrences of letter j in motif position k , over the total

Gibbs Sampling (4)

Predictive Update:

- Select a sequence $x = x^i$
- Remove x^i , recompute model:



$$M_{kj} = \frac{1}{(N-1)+B} (b_j + \sum_{s=1, s \neq i}^N (x_{a_s+k}^s = j))$$

where β_j are pseudocounts to avoid 0s,
and $B = \sum_j \beta_j$

Gibbs Sampling (5)

Sampling:

For every K -long word $x_{i-1}, \dots, x_{i+K-1}$ in x :

$$Q_i = \text{Prob}[\text{word} \mid \text{motif}] = M(1, x) \times \dots \times M(K, x_{i+K-1})$$

$$P_i = \text{Prob}[\text{word} \mid \text{background}] = B(x_1) \times \dots \times B(x_{i+K-1})$$

Let

$$A_j = \frac{Q_j / P_j}{\sum_{j=1}^{|\Sigma|-K+1} Q_j / P_j}$$



Sample a random new position a_i according to the probabilities $A_1, \dots, A_{|\Sigma|-K+1}$.

Gibbs Sampling (6)

Running Gibbs Sampling:

1. Initialize
2. Run until convergence
3. Repeat 1,2 several times, report common motifs

Advantages / Disadvantages

- Very similar to EM

Advantages:

- Easier to implement
- Less dependent on initial parameters
- More versatile, easier to enhance with heuristics

Disadvantages:

- More dependent on all sequences to exhibit the motif
- Less systematic search of initial parameter space

Repeats, and a Better Background Model

- Repeat DNA can be confused as motif
 - Especially low-complexity CACACA... AAAAA, etc.

Solution:

more elaborate background model

0th order: $B = \{ p_A, p_C, p_G, p_T \}$

1st order: $B = \{ P(A|A), P(A|C), \dots, P(T|T) \}$

...

Kth order: $B = \{ P(X | b_1 \dots b_K) : X, b_i \in \{A, C, G, T\} \}$

Has been applied to EM and Gibbs (up to 3rd order)

Example Application: Motifs in Yeast

Group:

Tavazoie et al. 1999, G. Church's lab, Harvard

Data:

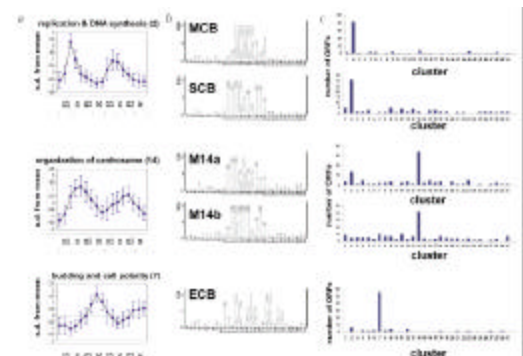
- Microarrays on 6,220 mRNAs from yeast Affymetrix chips (Cho et al.)
- 15 time points across two cell cycles

Processing of Data

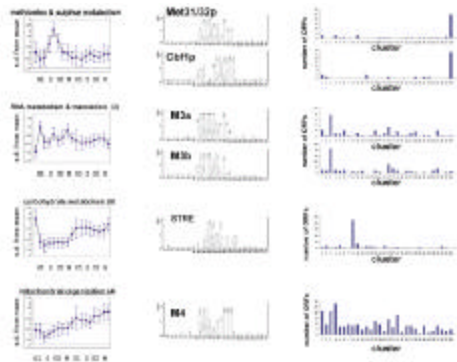
1. Selection of 3,000 genes

Genes with most variable expression were selected
2. Clustering according to common expression
 - K-means clustering
 - 30 clusters, 50-190 genes/cluster
 - Clusters correlate well with known function
3. AlignACE motif finding
 - 600-long upstream regions
 - 50 regions/trial

Motifs in Periodic Clusters



Motifs in Non-periodic Clusters



Phylogenetic Footprinting (Slides by Martin Tompa)

Phylogenetic Footprinting

(Tagle *et al.* 1988)

Functional sequences evolve slower than nonfunctional ones

- Consider a set of *orthologous* sequences from different species
- Identify unusually well conserved regions

Substring Parsimony Problem

Given:

- phylogenetic tree T ,
- set of orthologous sequences at leaves of T ,
- length k of motif
- threshold d

Problem:

- Find each set S of k -mers, one k -mer from each leaf, such that the "parsimony" score of S in T is at most d .

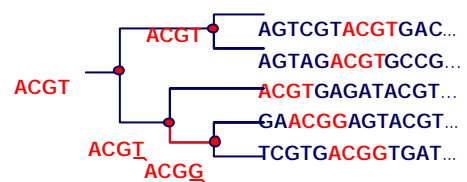
This problem is *NP*-hard.

Small Example



Size of motif sought: $k = 4$

Solution



Parsimony score: 1 mutation

[illegible]

4^k entries

ACGG: 2
ACGT: 1

ACGG: 1
ACGT: 0

ACGG: 0
ACGT: 2

ACGG: 0
ACGT: 1

ACGG: 0
ACGT: 2

ACGG: 0
ACGT: 1

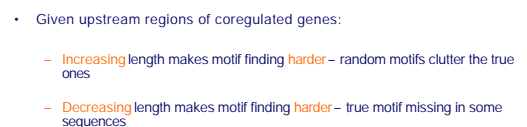
AGTCGTACGTG

ACGGGACGTGC

CGTGAGATAC

GAACGGAGTAC

CGTGACGGTG



A (k,d) -motif is a k-long motif with d random differences per copy

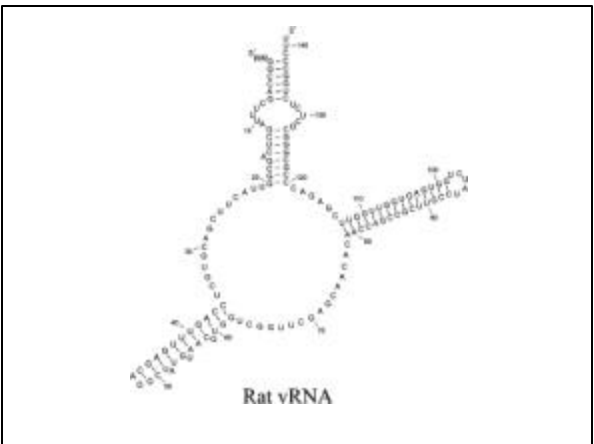
Motif Challenge problem:

Find a (15,4) motif in N sequences of length L

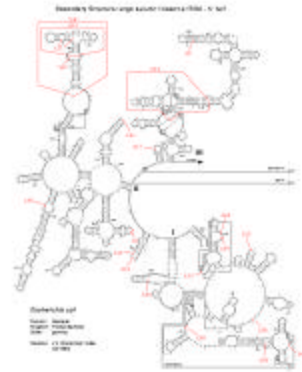
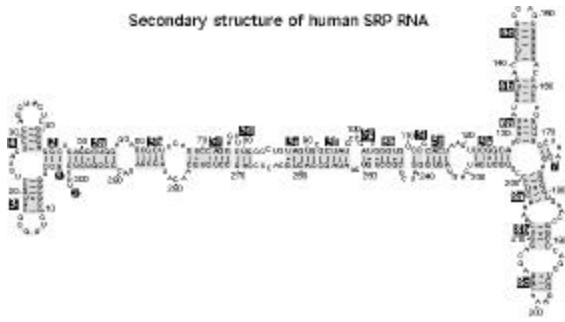
CONSENSUS, MEME, AlignACE, & most other programs fail for $N = 20$, $L = 1000$

RNA Secondary Structure

Figure 6. Pathway for spliceosome assembly. **5'** binds to the 5' splice site and **112** to the branch site. A preformed **114-125-126** complex joins this assembly to form a complete spliceosome.
 (After Dwyer "Proteomics" 1995, April 33-38, pp. 363)



Secondary structure of human SRP RNA



Modeling RNA Secondary Structure: Context-Free Grammars

A Context Free Grammar

$S \rightarrow AB$
 $A \rightarrow aAc \mid a$
 $B \rightarrow bBd \mid b$

Nonterminals: S, A, B
 Terminals: a, b, c, d

Derivation:

$S \rightarrow AB \rightarrow aAcB \rightarrow \dots \rightarrow aaaaccbB \rightarrow aaaaccbBd \rightarrow \dots \rightarrow aaaaccbBBBBBBBB$

Produces all strings $a^i c^j b^i d^j$, for $i, j \geq 0$

Example: modeling a stem loop

$S \rightarrow aW_1u$
 $W_1 \rightarrow cW_2g$
 $W_2 \rightarrow gW_3c$
 $W_3 \rightarrow gLc$
 $L \rightarrow agucg$

$ACGG$ AG
 $UGCC$ U
 CG

What if the stem loop can have other letters in place of the ones shown?

Example: modeling a stem loop

$S \rightarrow aW_1u \mid gW_1u$
 $W_1 \rightarrow cW_2g$
 $W_2 \rightarrow gW_3c \mid gW_3u$
 $W_3 \rightarrow gLc \mid aLu$
 $L \rightarrow agucg \mid agccg \mid cugugc$

$ACGG$ AG
 $UGCC$ U
 CG

More general: Any 4-long stem, 3-5-long loop:

$S \rightarrow aW_1u \mid gW_1u \mid cW_1g \mid uW_1g \mid uW_1a$
 $W_1 \rightarrow aW_2u \mid gW_2u \mid cW_2g \mid uW_2g \mid uW_2a$
 $W_2 \rightarrow aW_3u \mid gW_3u \mid cW_3g \mid uW_3g \mid uW_3a$
 $W_3 \rightarrow aL_u \mid gL_u \mid cL_g \mid uL_g \mid uL_a$

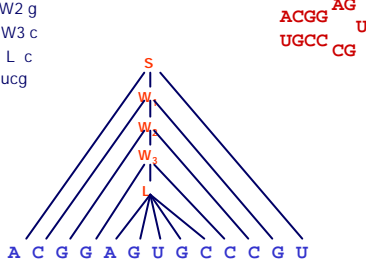
$GCGA$ AG
 $UGCU$ C
 CG

$L \rightarrow aL_1 \mid cL_1 \mid gL_1 \mid uL_1$
 $L_1 \rightarrow aL_2 \mid cL_2 \mid gL_2 \mid uL_2$
 $L_2 \rightarrow a \mid c \mid g \mid u \mid aa \mid \dots \mid uu \mid aaa \mid \dots \mid uuu$

$GCGA$ CUG
 $UGUU$ U
 CG

A parse tree: alignment of CFG to sequence

- $S \rightarrow a W^1 u$
- $W^1 \rightarrow c W^2 g$
- $W^2 \rightarrow g W^3 c$
- $W^3 \rightarrow g L c$
- $L \rightarrow agucg$



Alignment scores for parses!

We can define each rule $X \rightarrow s$, where s is a string, to have a score.

Example:

- $W \rightarrow a W^1 u$: 3 (forms 3 hydrogen bonds)
- $W \rightarrow g W^1 c$: 2 (forms 2 hydrogen bonds)
- $W \rightarrow g W^1 u$: 1 (forms 1 hydrogen bond)
- $W \rightarrow x W^1 z$: -1, when (x, z) is not an a/u, g/c, g/u pair

Questions:

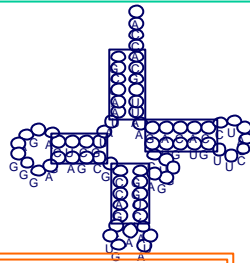
- How do we best align a CFG to a sequence? (DP)
- How do we set the parameters? (Stochastic CFGs)

The Nussinov Algorithm

Let's forget CFGs for a moment

Problem:

Find the RNA structure with the maximum (weighted) number of nested pairings



The Nussinov Algorithm

Given sequence $X = x_1 \dots x_N$,

Define DP matrix:

$F(i, j)$ = maximum number of bonds if $x_i \dots x_j$ folds optimally

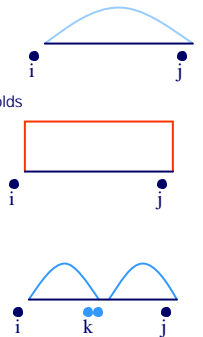
Two cases, if $i < j$:

1. x_i is paired with x_j

$$F(i, j) = s(x_i, x_j) + F(i+1, j-1)$$

- x_i is not paired with x_j

$$F(i, j) = \max\{k: i \leq k < j\} \quad F(i, k) + F(k+1, j)$$



The Nussinov Algorithm

Initialization:

$$F(i, i-1) = 0; \quad \text{for } i = 2 \text{ to } N$$

$$F(i, i) = 0; \quad \text{for } i = 1 \text{ to } N$$

Iteration:

For $l = 2$ to N :

For $i = 1$ to $N - l$:

$$j = i + l - 1$$

$$F(i, j) = \max \begin{cases} F(i+1, j-1) + s(x_i, x_j) \\ \max\{i \leq k < j\} \quad F(i, k) + F(k+1, j) \end{cases}$$

Termination:

Best structure is given by $F(1, N)$

(Need to trace back; refer to the Durbin book)

The Nussinov Algorithm and CFGs

Define the following grammar, with scores:

$$S \rightarrow a S u : 3 \quad | \quad u S a : 3$$

$$g S c : 2 \quad | \quad c S g : 2$$

$$g S u : 1 \quad | \quad u S g : 1$$

$$S S : 0 \quad |$$

$$a S : 0 \quad | \quad c S : 0 \quad | \quad g S : 0 \quad | \quad u S : 0 \quad | \quad \epsilon : 0$$

Note: ϵ is the "string"

Then, the Nussinov algorithm finds the optimal parse of a string with this grammar

The Nussinov Algorithm

Initialization:

$F(i, i-1) = 0;$ for $i = 2$ to N
 $F(i, i) = 0;$ for $i = 1$ to N $S \rightarrow a \mid c \mid g \mid u$

Iteration:

For $i = 2$ to N :
 For $i = 1$ to $N - i$:
 $j = i + i - 1$
 $F(i, j) = \max \begin{cases} F(i+1, j-1) + S(x_i, x_j) & S \rightarrow a S u \mid \dots \\ \max\{i \leq k < j\} & F(i, k) + F(k+1, j) \\ & S \rightarrow S S \end{cases}$

Termination:

Best structure is given by $F(1, N)$

Stochastic Context Free Grammars

In an analogy to HMMs, we can assign probabilities to transitions:

Given grammar

$$X_1 \rightarrow S_{11} \mid \dots \mid S_{1n}$$

...

$$X_m \rightarrow S_{m1} \mid \dots \mid S_{mn}$$

Can assign probability to each rule, s.t.

$$P(X_i \rightarrow S_{i1}) + \dots + P(X_i \rightarrow S_{in}) = 1$$

Example

$$S \rightarrow a S b : \frac{1}{2}$$

$$a : \frac{1}{4}$$

$$b : \frac{1}{4}$$

Probability distribution over all strings x :

$$x = a^n b^{n+1},$$

$$\text{then } P(x) = 2^{-n} \times \frac{1}{4} = 2^{-(n+2)}$$

$$x = a^{n+1} b^n,$$

same

$$\text{Otherwise: } P(x) = 0$$