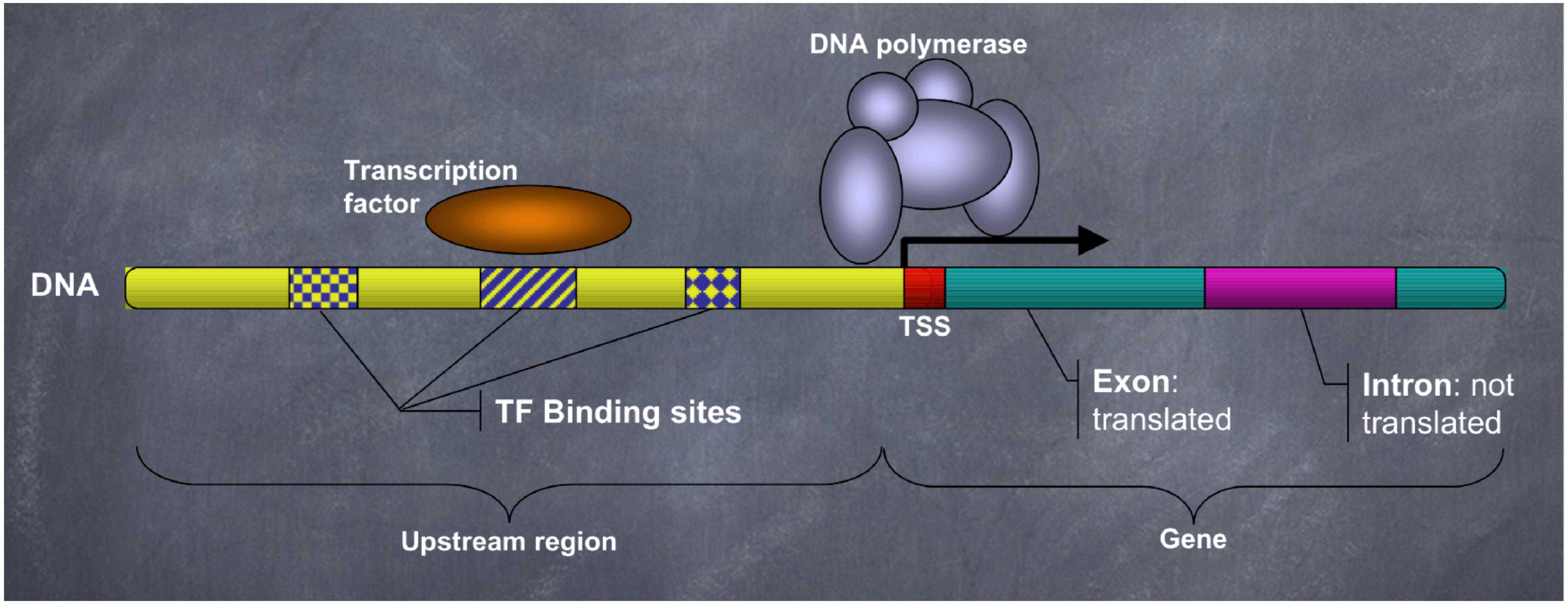
Motif Finding & Gibbs Sampling

02-251

Slides by Carl Kingsford

DNA → mRNA → Protein



- Finding transcription factor binding sites can tell us about the cell's regulatory network.

$\underbrace{\text{RNA}}_{\text{b/c it makes RNA}}$ $\underbrace{\text{Polymer}}_{\text{into a polymer}}\underbrace{\text{ase}}_{\text{is an enzyme}}$

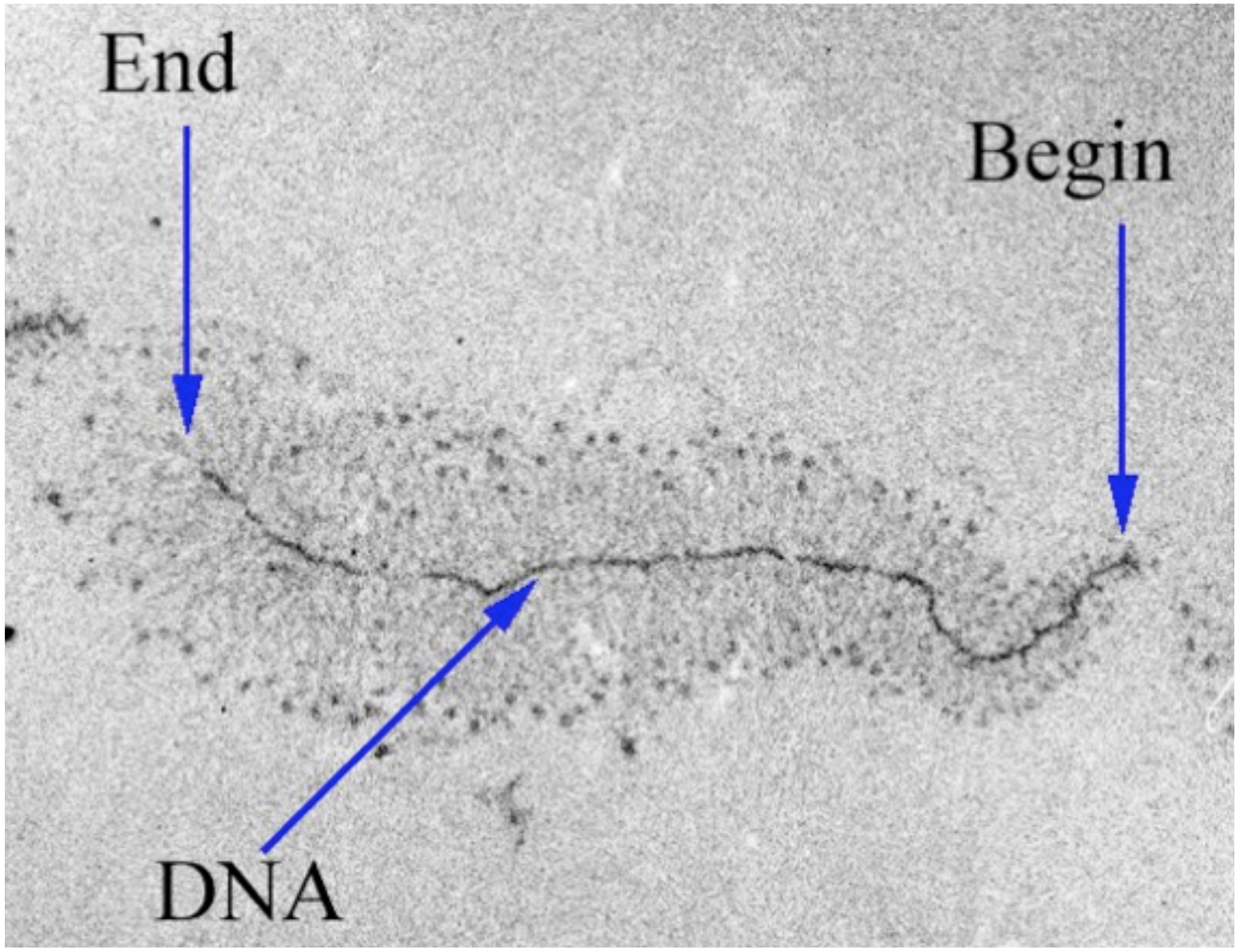


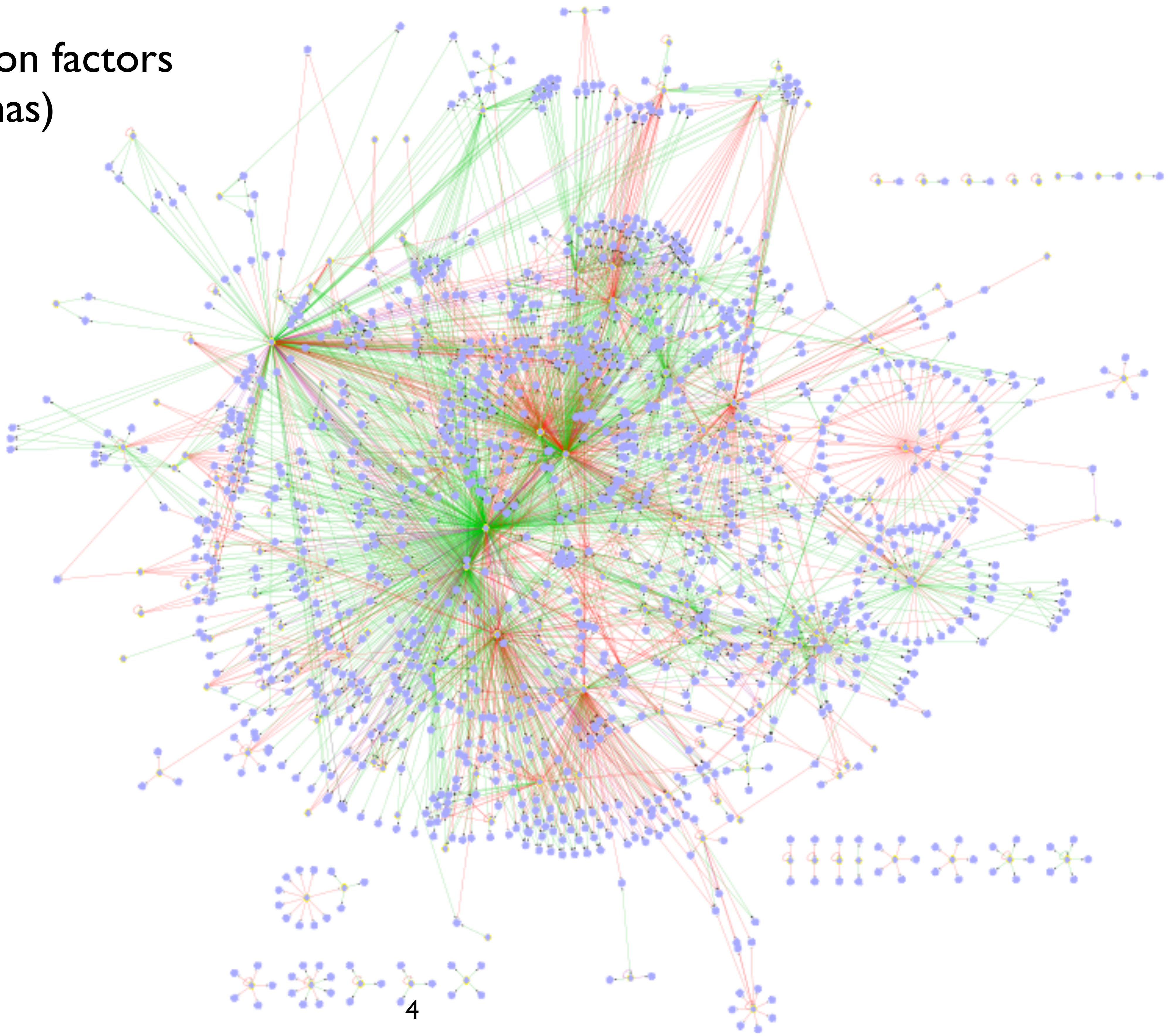
Image of transcription occurring.
Each "hair" is a piece of RNA that RNA pol is growing off of the DNA.

Transcription Network (E. coli)

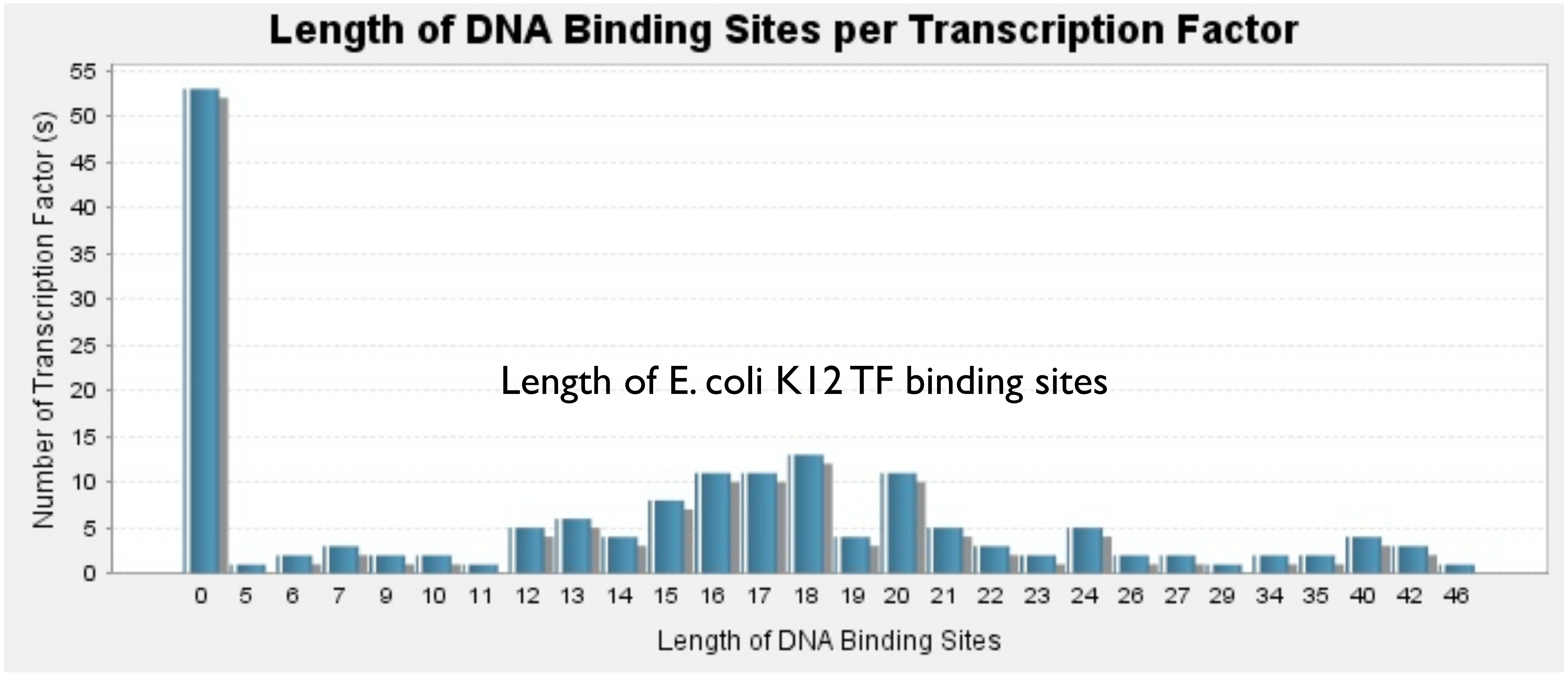
169 transcription factors
(excluding sigmas)

3322 edges

1753 activation,
1369 repression,
185 both,
3 unknown



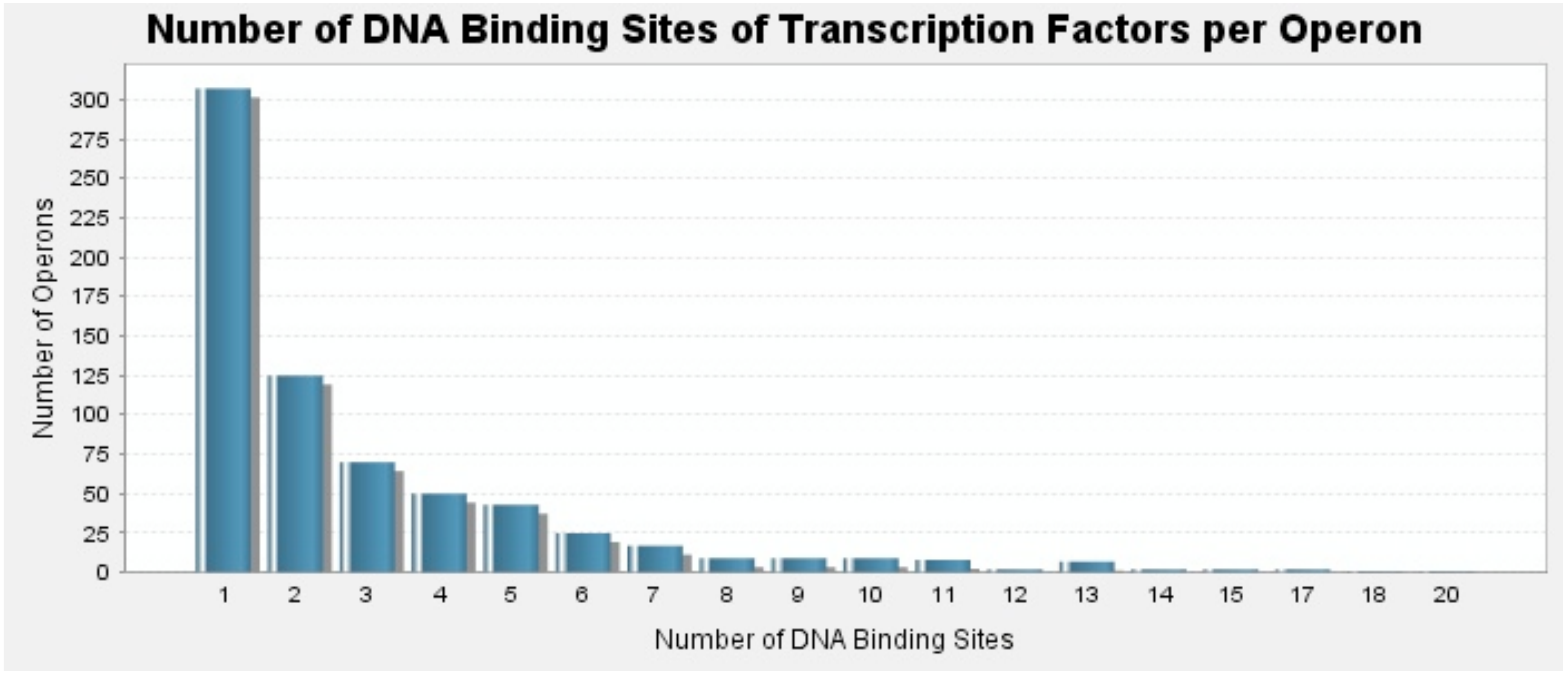
Transcription Factor Binding Sites



Length of E. coli K12 TF binding sites

RegulonDB (Feb 27, 2010)

Transcription Factor Binding Sites



RegulonDB (Feb 27, 2010)

Motif Finding

Transcription factor

```
1. ttgccacaaaataatccgccttcgcaaattgaccTACCTCAATAGCGGTAgaaaaacgcaccactgcctgacag
2. gtaagtacctgaaagttacggtctgcgaacgctattccacTGCTCCTTTATAGGTAcaacagtatagtctgatgga
3. ccacacggcaaataaggagTAACTCTTTCCGGGTAtgggtatacttcagccaatagccgagaatactgccattccag
4. ccatacccggaaagagttactccttatttgccgtgtggttagtcgcttTACATCGGTAAGGGTAgggattttacagca
5. aaactattaagattttatgcagatgggtattaaggaGTATTCCCCATGGGTAacatattaatggctctta
6. ttacagtctgttatgtggtggctgttaaTTATCCTAAAGGGGTAtcttaggaatttactt
```

Given p sequences, find the most mutually similar length-k subsequences, one from each sequence:

$$\operatorname*{argmin}_{s_1,\ldots,s_p} \sum_{i<j} \mathrm{dist}(s_i, s_j)$$

$\mathrm{dist}(s_i,s_j)$ = Hamming distance between s_i and s_j.

Hundreds of papers, many formulations (Tompa05)

Motif-finding by Gibbs Sampling

Problem. Given *p* strings and a length *k*, find the most "mutually similar" length-k substring from each string.

"Gibbs sampling" is the basis behind a general class of algorithms that is a type of local search.

It doesn't guarantee good performance, but often works well in practice.

Assumes:

1. we know the length *k* of the motif we are looking for.
2. each input sequence contains exactly 1 real instance of the motif.
3. No gaps

Gibbs Sampling: Profiles

If we knew the starting point of the motif in each sequence, we could construct a Sequence Profile (PSSM) for the motif:

x_1

1. ttgccacaaaataatccgccttcgcaaattgacc**TACCTCAATAGCGGTA**gaaaaacgcaccactgcctgacag

x_2

2. gtaagtacctgaaagttacggtctgcgaacgctattccac**TGCTCCTTTATAGGTA**caacagtatagtctga

x_3

3. ccacacggcaaataaggag**TAACTCTTTCCGGGTA**tgggtatacttcagccaatagccgagaatactgccatt

x_4

4. ccatacccggaaagagttactccttatttgccgtgtggttagtcgctt**TACATCGGTAAGGGTA**gggatttt

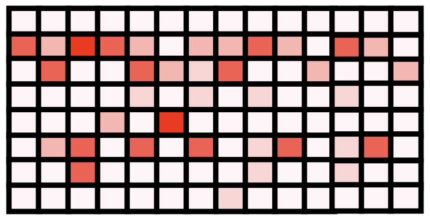
x_5

5. aaactattaagatttttatgcagatgggtattaagga**GTATTCCCCATGGGTA**acatattaatggctctta

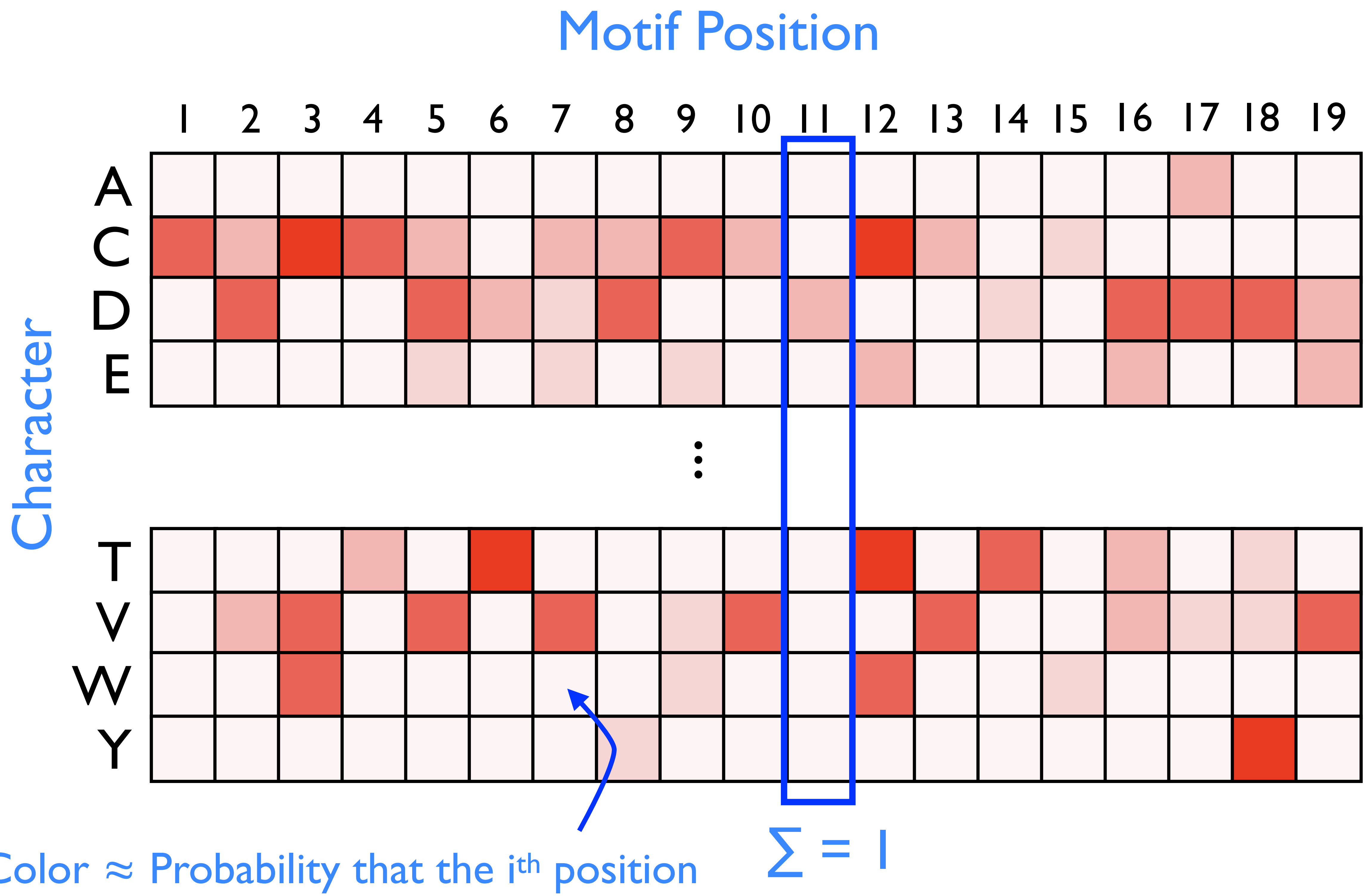
x_6

6. ttacagtctgttatgtggtggctgttaa**TTATCCTAAAGGGGTA**tcttaggaatttactt

TACCTCAATAGCGGTA
TGCTCCTTTATAGGTA
TAACTCTTTCCGGGTA
TACATCGGTAAGGGTA
GTATTCCCCATGGGTA
TTATCCTAAAGGGGTA



Sequence Profiles (PSSM)

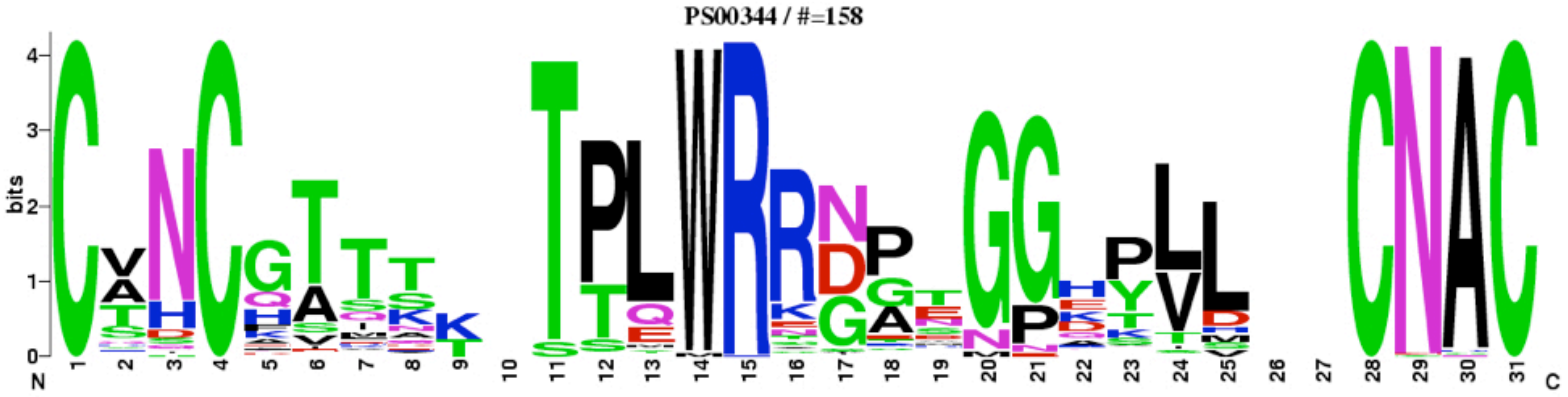


Color ≈ Probability that the i^{th} position has the given amino acid = $e_i(x)$.

Sequence Logos

Height of letter ≈ fraction of time that letter is observed at that position.

(Height of all the letters in a column ≈ to how conserved the column is)



Motif Position

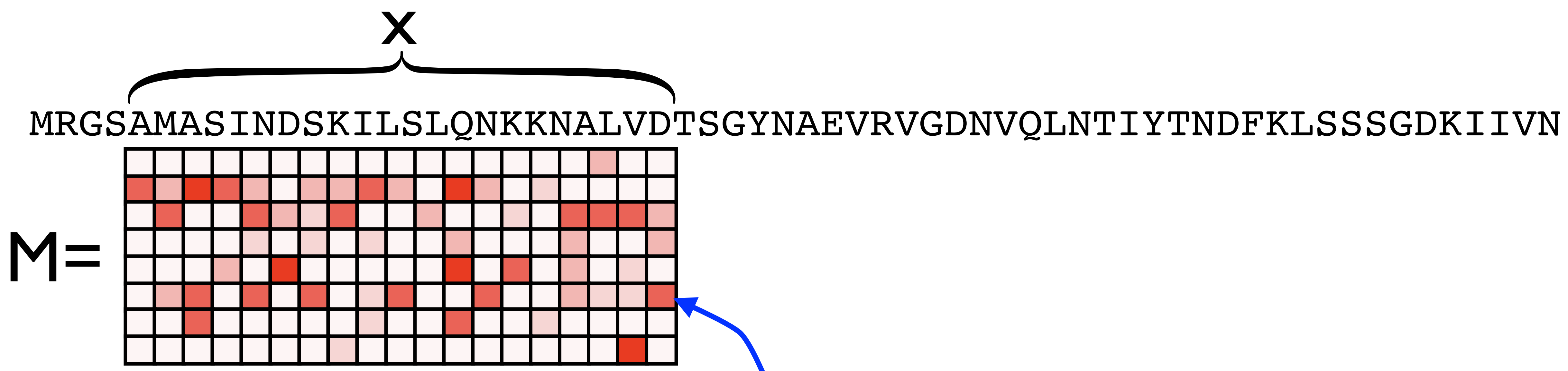
Schneider, Stephens, 1990

Gibbs Sampling, Version 1: Pseudocode

```
Set ($x_1$, $x_2$, ..., $x_p$) to random positions in each input string.

repeat until the answer ($x_1$, $x_2$, ..., $x_p$) doesn't change:
    for i = 1 ... p:
        Build a profile M using sequences at ($x_1$, $x_2$, ..., $x_p$)
        except $x_i$
        Set $x_i$ to where the profile M matches best in string i.
```

Scoring a Subsequence x


x
MRGSAMASINDSKILSLQNKKNALVDTSGYNAEVRVGDNVQLNTIYTNDFKLSSSGDKIIVN
M=

Color ≈ Probability that the ith position has the given amino acid = $e_i(x)$.

$$\text{Score}(x) = \Pr(x \mid M) = \prod_{i=1}^{L} e_i(x_i)$$

Score of a string according to profile M = Product of the probabilities you would observe the given letters.

Background Frequencies

Interested in how different this motif position is from we expect by chance.

Correct for "expect by chance" by dividing by the probability of observing x in a random string:

$$\text{ScoreCorrected}(x) = \frac{\Pr(x \mid M)}{\Pr(x \mid \text{background})} = \prod_{i=1}^{L} \frac{e_i(x_i)}{b(x_i)}$$

$b(x_i)$:= probability of observing character x_i at random.
Usually computed as (# x_i in entire string) / (length of string)

Often, to avoid multiplying lots of terms, we take the log and then sum:

$$\text{ScoreCorrectedLog}(x) = \log \prod_{i=1}^{L} \frac{e_i(x_i)}{b(x_i)} = \sum_{i=1}^{L} \log\left(\frac{e_i(x_i)}{b(x_i)}\right)$$

```
def gibbs(Seqs, k):
    """Seqs is a list of strings. Find the best motif."""

    # start with random indices
    I = [random.randint(0, len(x) - k) for x in Seqs]

    LastI = None
    while I != LastI:      # repeat until nothing changes
        LastI = list(I)

        # iterate through every string
        for i in xrange(len(Seqs)):
            # compute the profile for the sequences except i
            P = profile_for([
                    x[j : j + k] for q, (x, j) in enumerate(zip(Seqs, I))
                        if q != i
                  ])

            # find the place the profile matches best
            best = None
            for j in xrange(len(Seqs[i]) - k + 1):
                score = profile_score(P, Seqs[i][j : j + k])
                if score > best or best is None:
                    best = score
                    bestpos = j
            # update the ith position with the best
            I[i] = bestpos

    return I, [x[j : j + k] for x, j in zip(Seqs, I)]
```

Gibbs Example

```
gibbs(["thequickdog", "browndog", "dogwood"], k=3)
1: [8, 1, 2] ['dog', 'row', 'gwo']
2: [8, 5, 0] ['dog', 'dog', 'dog']
F: [8, 5, 0] ['dog', 'dog', 'dog']
```

random starting positions

Small bias toward “o” in the middle is correct.

```
gibbs(["thequickdog", "browndog", "dogwood"], k=3)
1: [4, 3, 1] ['uic', 'wnd', 'ogw']
2: [6, 2, 4] ['ckd', 'own', 'ood']
3: [8, 5, 0] ['dog', 'dog', 'dog']
F: [8, 5, 0] ['dog', 'dog', 'dog']
```

```
gibbs(["thequickdog", "browndog", "dogwood"], k=3)
1: [2, 0, 1] ['equ', 'bro', 'ogw']
2: [7, 4, 2] ['kdo', 'ndo', 'gwo']
F: [7, 4, 2] ['kdo', 'ndo', 'gwo']
```

Might not find the optimal.

Another Example

```
gibbs(["aaa123", "678aaa45", "9a7aaab", "32aa19a8aaa"], 3)
1: [0, 5, 0, 2] ['aaa', 'a45', '9a7', 'aa1']
2: [1, 3, 3, 8] ['aa1', 'aaa', 'aaa', 'aaa']
3: [0, 3, 3, 8] ['aaa', 'aaa', 'aaa', 'aaa']
F: [0, 3, 3, 8] ['aaa', 'aaa', 'aaa', 'aaa']
```

Bias toward “a” in the profile quickly leads to finding the implanted “aaa”

Can be multiple optimal answers

```
gibbs(["aaabbb", "bbbaaabb", 'babaaab', 'ababacaaabac', 'abbbababaaabbbaba'], 3)
1: [1, 4, 0, 4, 11] ['aab', 'aab', 'bab', 'aca', 'bbb']
2: [1, 4, 4, 7, 9]  ['aab', 'aab', 'aab', 'aab', 'aab']
F: [1, 4, 4, 7, 9]  ['aab', 'aab', 'aab', 'aab', 'aab']
gibbs(["aaabbb", "bbbaaabb", 'babaaab', 'ababacaaabac', 'abbbababaaabbbaba'], 3)
1: [0, 3, 3, 3, 8] ['aaa', 'aaa', 'aaa', 'bac', 'aaa']
2: [0, 3, 3, 6, 8] ['aaa', 'aaa', 'aaa', 'aaa', 'aaa']
F: [0, 3, 3, 6, 8] ['aaa', 'aaa', 'aaa', 'aaa', 'aaa']
```

Randomness: Gibbs Sampling

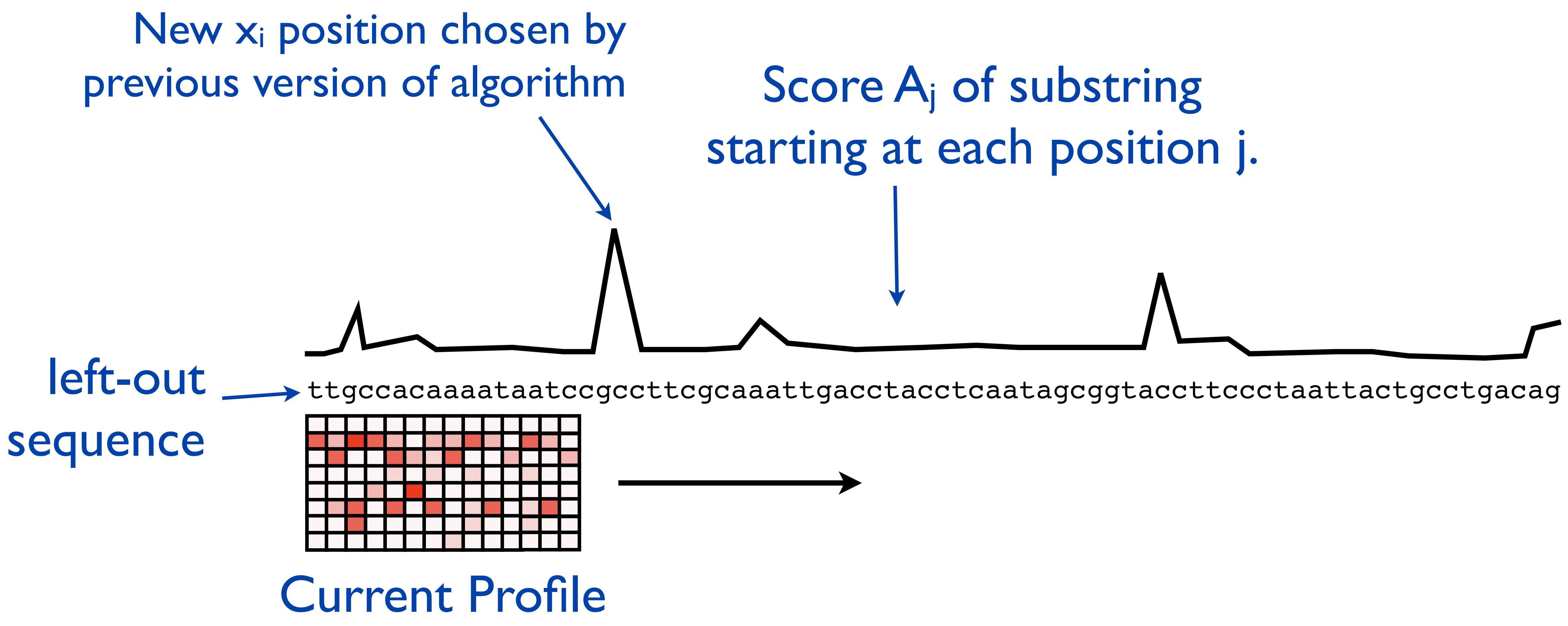
- Run the Gibbs sampling multiple times to make it more likely you find the global optimal.
- Can increase the use of randomness to further avoid getting stuck in local optima by choosing new x_i randomly.

```
Set ($x_1, x_2, ..., x_p$) to random positions in each input string.

repeat until the best ($x_1, x_2, ..., x_p$) doesn't change too often
    for i = 1 ... p:
        Build a profile M using sequences at ($x_1, x_2, ..., x_p$) except $x_i$
        Choose $x_i$ according to the profile probability distribution of M
        in string i.
```

Lawrence CE et al. (1993)

Profile Probability Distribution


New x_i position chosen by previous version of algorithm
Score A_j of substring starting at each position j.
left-out sequence
ttgccacaaaataatccgccttcgcaaattgacctacctcaatagcggtaccttccctaattactgcctgacag
Current Profile

Instead of choosing the position with the best match, choose a position randomly such that:

$$\text{Probability of choosing position } j = \frac{A_j}{\sum_i A_i}$$

(Lawrence, et al., *Science*, 1994)

Gibbs Sampling - Recap

- "Motif finding" is the problem of finding a set of common substrings within a set of strings.

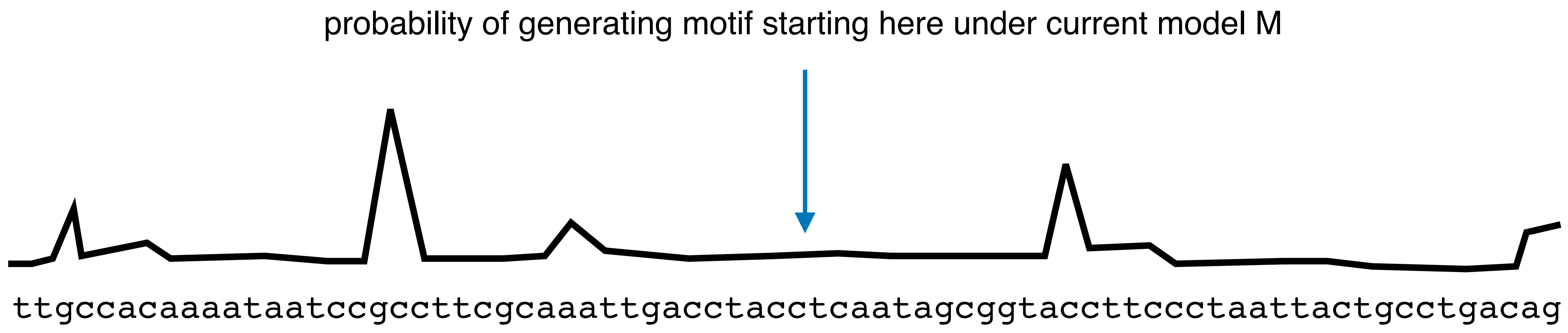
- Useful for finding transcription factor binding sites.

- **Gibbs sampling:** repeatedly leave one sequence out and optimize the motif location in the left-out sequence.
- Doesn't guarantee finding a good solution, but often works.

Expectation Maximization (for motif finding)

A Problem with Gibbs

- Gibbs maintains a single current guess (x_1, x_2, ..., x_p) about where the motif instances are
- This entire distribution is used only to sample the next x_i:



- Instead, "Expectation Maximization" (EM) uses the entire distribution to update the PSSM model M.

EM: maximization of the expected likelihood

Goal: Find $x_1, x_2, \ldots, x_p$ and M to maximize the likelihood:

Likelihood →

$$\Pr(x_1, \ldots, x_p \mid M) = \prod_{j=1}^{p} \prod_{i=1}^{k} e_i(x_{ji})$$

The probability of x being observed, given the model M.

Challenge is that both x and M are unknown

If M was fixed,
easy to find the best x:

$$\max_x f(x) = \Pr(x \mid M)$$

Scan each sequence, picking the k-mer x that maximizes f(x)

If the x's were fixed,
easy to find the best M:

$$\max_M g(M) = \Pr(x \mid M)$$

Build M from the given x's.

Concept: Alternate between these two views

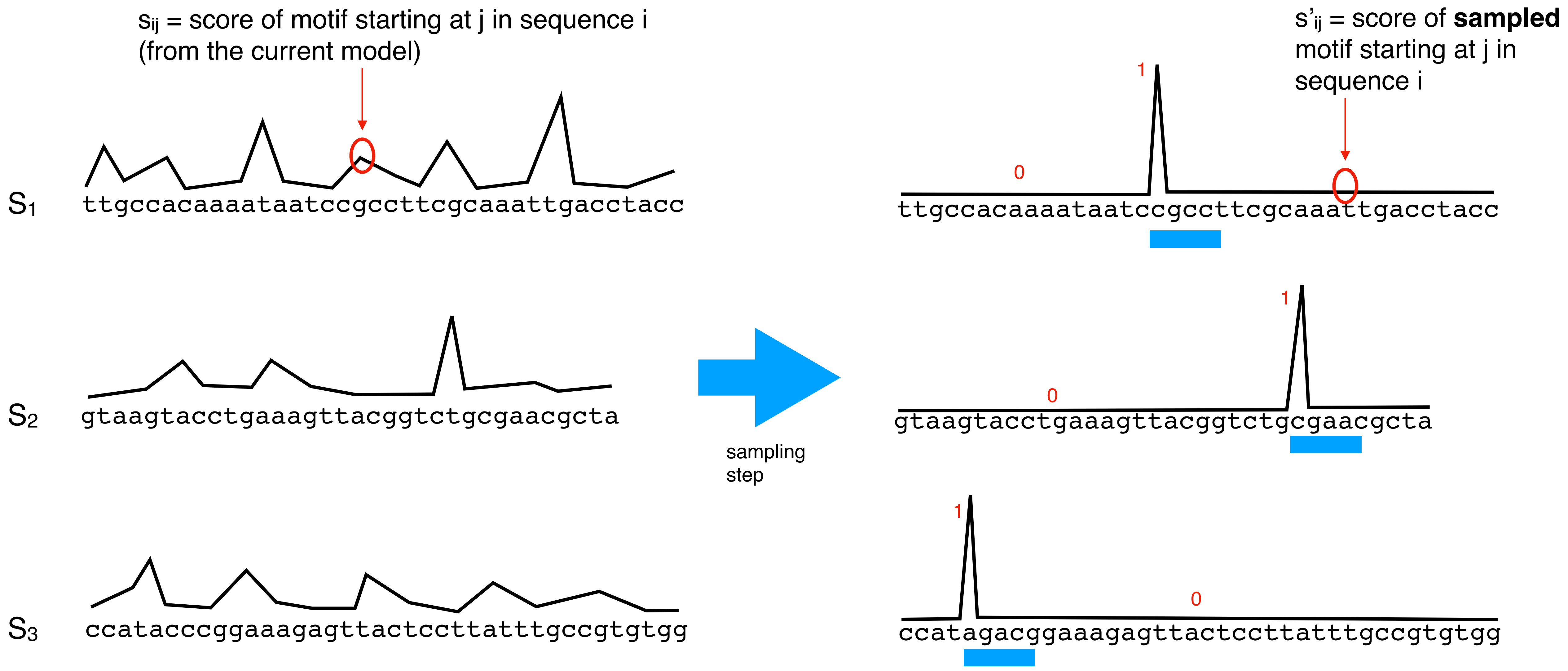
- Gibbs Sampling (version 1): alternate between solving these two maximization problems

- Gibbs Sampling (version 2): alternate between maximizing the model (g(M)) and sampling from the current model:

$$\begin{tikzcd} \text{sample from } f(x) = \Pr(x \mid M) \arrow[d, bend left] \\ \max_{M} g(M) = \Pr(x \mid M) \arrow[u, bend left] \end{tikzcd}$$

- EM: Repeatedly maximize the current value of g(M) expected over random choices of x.

Another View of Gibbs Sampling



s_ij = score of motif starting at j in sequence i (from the current model)
S1 ttgccacaaaataatccgccttcgcaaattgacctacc
S2 gtaagtacctgaaagttacggtctgcgaacgcta
S3 ccatacccggaaagagttactccttatttgccgtgtgg
sampling step
s'_ij = score of **sampled** motif starting at j in sequence i
1
0
ttgccacaaaataatccgccttcgcaaattgacctacc
gtaagtacctgaaagttacggtctgcgaacgcta
ccatagacggaaagagttactccttatttgccgtgtgg

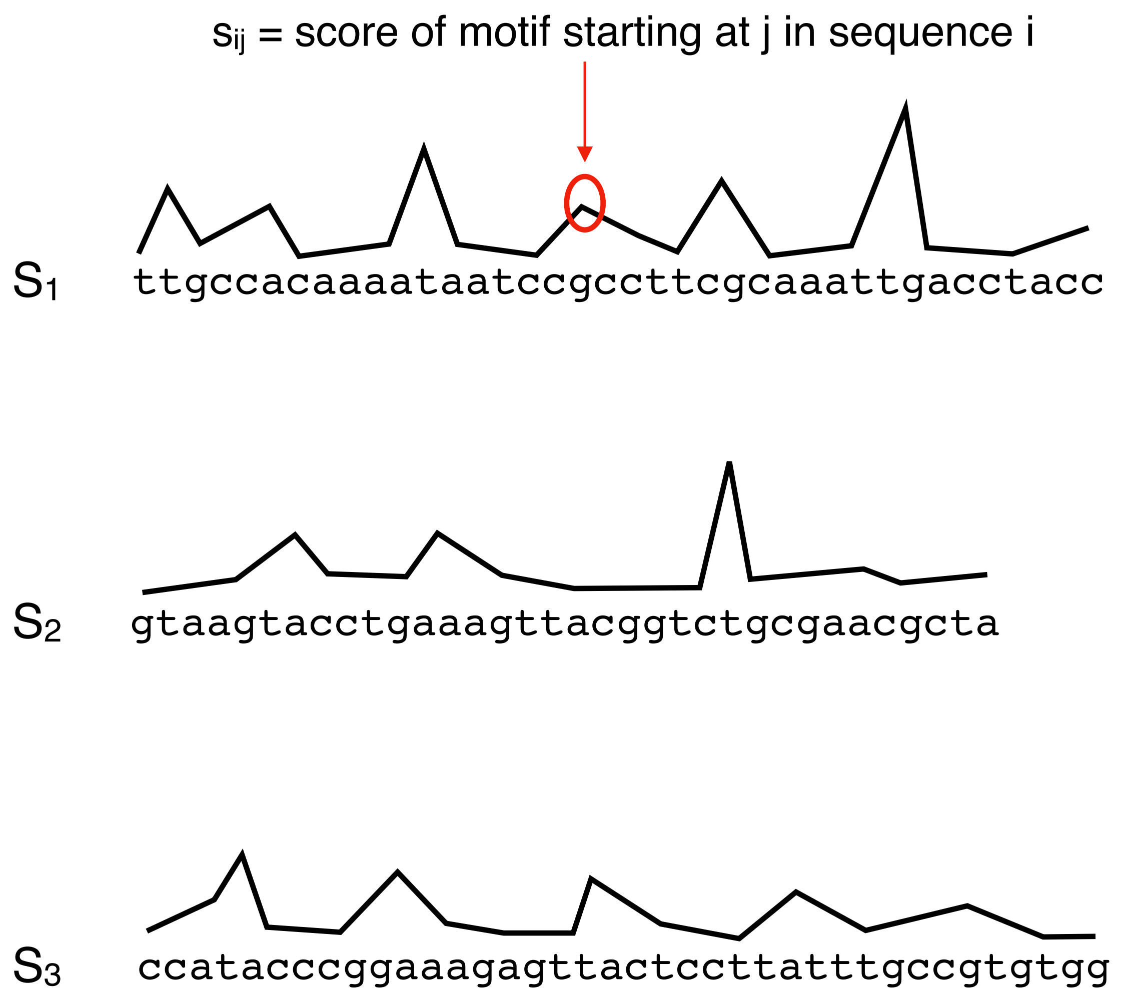
- First column of new matrix (model) computed using:

$$M[c,0] = \frac{\sum \{s'_{ij} \mid S_i[j] = c\}}{\sum s'_{ij}}$$

numerator = # of 🟦 starting with c

denominator = # of sequences

EM for motif finding: skip the sampling step


s_{ij} = score of motif starting at j in sequence i
S_1 ttgccacaaaataatccgccttcgcaaattgacctacc
S_2 gtaagtacctgaaagttacggtctgcgaacgcta
S_3 ccatacccggaaagagttactccttatttgccgtgtgg

First column of new matrix:

$$M[c,0] = \frac{\sum \{s_{ij} \mid S_i[j] = c\}}{\sum s_{ij}}$$

mth column of new matrix:

$$M[c,m] = \frac{\sum \{s_{ij} \mid S_i[j+m] = c\}}{\sum s_{ij}}$$

- Doesn’t “commit” to a sampled choice of motif instances
- Instead uses each possible sequence weighted by score

EM Summary Part 1

- EM: Compute a series of models $M_1 \rightarrow M_2 \rightarrow M_3 \dots$ using the equations on the previous slide. Stop when M doesn't change much.

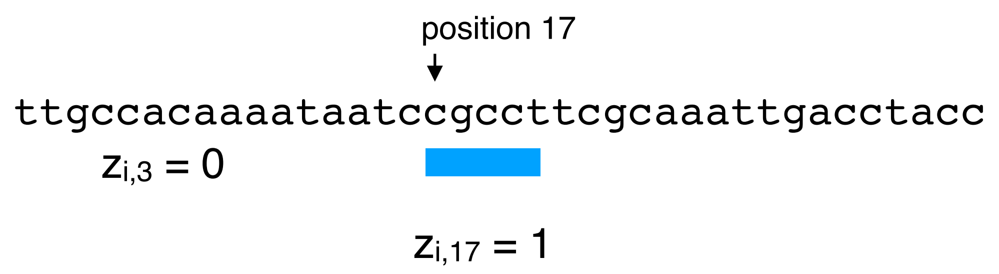
- Gibbs (version 1) → Gibbs (version 2) → EM uses more and more of the computed distribution

- EM maximizes the expected value of the likelihood over random choices of the "hidden" variables, which are the locations of the motifs
  - It's not clear yet that that is what the EM algorithm we presented is doing
  - We'll see more intuition about this soon.

Hidden Variables View of EM

- Hidden variables z_{ij} tell us where the motifs are

$$z_{ij} = \begin{cases} 1 & \text{motif starts at position } j \text{ in sequence } i \\ 0 & \text{otherwise} \end{cases}$$



- Now, want to find M to maximize:

Likelihood $\longrightarrow$ $$\Pr(\text{data} \mid M) = \sum_{\text{choices of } z} \Pr(\text{data}, z \mid M)$$ by definition of joint probability

Instead Maximize Expected log likelihood

$$\text{Likelihood} \longrightarrow \quad \Pr(\text{data} \mid M) = \sum_z \Pr(\text{data}, z \mid M) \qquad \text{by definition of joint probability}$$

- Expected log likelihood:

$$\underbrace{\mathbb{E}_z \log \Pr(\text{data} \mid M_{\text{new}})}_{\uparrow\; Q(M_{\text{new}} \mid M_{\text{old}})} = \underbrace{\sum_z \Pr(z \mid x, M_{\text{old}}) \overbrace{\log \Pr(\text{data}, z \mid M_{\text{new}})}^{\text{quality of new model}}}_{\text{in expectation over choices of hidden variables (drawn according to the old model)}}$$

- EM now iteratively computes:

$$\max_{M_{\text{new}}} Q(M_{\text{new}} \mid M_{\text{old}})$$