Bioinformatics: Finding Regulatory Motifs

Swakkhar Shatabda

Department of Computer Science and Engineering
BRAC University

Book Reference

Chapter 2, Bioinformatics Algorithms: An Active Learning Approach - I

Hidden Message Once Again!

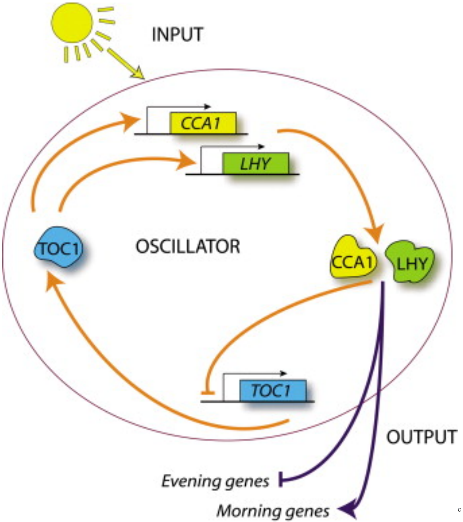
- Gene regulation is the process used to control the timing, location and amount in which genes are expressed.
- The process can be complicated and is carried out by a variety of mechanisms, including through regulatory proteins and chemical modification of DNA. Gene regulation is key to the ability of an organism to respond to environmental changes.
- A **transcription factor** regulates a gene by binding to a specific short DNA interval called a **regulatory motif**, or **transcription factor binding site**.
- Transcription factors bind to either **enhancer** or **promoter** regions of DNA adjacent to the genes that they regulate based on recognizing specific DNA motifs.

Regulatory Motifs

- Motifs are short recurring patterns.
- Transcription factors often regulate a group of genes that are involved in similar cellular processes.
- Thus, genes that contain the same motif in their upstream regions are likely to be related in their functions.
- In fact, many regulatory motifs are identified by analyzing the regions upstream of genes known to have similar functions.
- The life of a bioinformatician would be easy if regulatory motifs were completely conserved, but the reality is more complex, as regulatory motifs may vary at some positions, **denegerate**.

Circadian Clock

- Plant cell keeps track of day and night independently of other cells, and that just three plant genes, called LHY, CCA1, and TOC1, are the clock's master timekeepers.
- TOC1 promotes the expression of LHY and CCA1, whereas LHY and CCA1 repress the expression of TOC1, resulting in a negative feedback loop.
- CCA1 binds to AAAAAATCT in the upstream region of many genes regulated by CCA1



The evening element

- In 2000, Steve Kay used DNA arrays to determine which genes in the plant *Arabidopsis thaliana* are activated at different times of the day.
- He then extracted the upstream regions of nearly 500 genes that exhibited circadian behavior and looked for frequently appearing patterns in their upstream regions.
- If you concatenated these upstream regions into a single string, you would find that AAAATATCT is a surprisingly frequent word, appearing 46 times.
- After he mutated the evening element in the upstream region of one gene, the gene no longer exhibited circadian behavior.
- Not all motifs are as conserved as the evening element.

Immunity genes in a Fly

- If we infect a fly with a bacterium, the fly will switch on its immunity genes to fight the infection.
- The genes with elevated expression levels after the infection are likely to be immunity genes.
- These genes have 12-mers similar to TCGGGGATTTCC in upstream regions, binding site of a transcription factor called NF-κB that activates various immunity genes in flies.

1	T	C	G	G	G	G	g	T	T	T	t	t
2	c	C	G	G	t	G	A	c	T	T	a	C
3	a	C	G	G	G	G	A	T	T	T	t	C
4	T	t	G	G	G	G	A	c	T	T	t	t
5	a	a	G	G	G	G	A	c	T	T	C	C
6	T	t	G	G	G	G	A	c	T	T	C	C
7	T	C	G	G	G	G	A	T	T	c	a	t
8	T	C	G	G	G	G	A	T	T	c	C	t
9	T	a	G	G	G	G	A	a	c	T	a	C
10	T	C	G	G	G	t	A	T	a	a	C	C

Another string finding problem

```
1   atgaccgggatactgataaaaaaaagggggggggcgtacacattagataaacgtatgaagtacgttagactcggcgccgccg
2   acccctattttttgagcagatttagtgacctggaaaaaaaatttgagtacaaaacttttccgaataaaaaaaaaggggggga
3   tgagtatccctgggatgacttaaaaaaaagggggggtgctctcccgatttttgaatatgtaggatcattcgccagggtccga
4   gctgagaattggatgaaaaaaaagggggggtccacgcaatcgcgaaccaacgcggacccaaaggcaagaccgataaaggaga
5   tcccttttgcggtaatgtgccgggaggctggttacgtagggaagccctaacggacttaataaaaaaaagggggggcttatag
6   gtcaatcatgttcttgtgaatggatttaaaaaaaaggggggggaccgcttggcgcacccaaattcagtgtgggcgagcgcaa
7   cggttttggcccttgttagaggcccccgtaaaaaaaagggggggcaattatgagagagctaatctatcgcgtgcgtgttcat
8   aacttgagttaaaaaaaagggggggctggggcacatacaagaggagtcttccttatcagttaatgctgtatgacactatgta
9   ttggcccattggctaaaagcccaacttgacaaatggaagatagaatccttgcataaaaaaaagggggggaccgaaagggaag
10  ctggtgagcaacgacagattcttacgtgcattagctcgcttccggggatctaatagcacgaagcttaaaaaaaaggggggga
```

Frequent Words Problem?

```
 1   atgaccgggatactgatAAAAAAAAGGGGGGGggcgtacacattagataaacgtatgaagtacgttagactcggcgccgccg
 2   acccctattttttgagcagatttagtgacctggaaaaaaaatttgagtacaaaacttttccgaataAAAAAAAAGGGGGGGa
 3   tgagtatccctgggatgacttAAAAAAAAGGGGGGGtgctctcccgatttttgaatatgtaggatcattcgccagggtccga
 4   gctgagaattggatgAAAAAAAAGGGGGGGtccacgcaatcgcgaaccaacgcggacccaaaggcaagaccgataaaggaga
 5   tcccttttgcggtaatgtgccgggaggctggttacgtagggaagccctaacggacttaatAAAAAAAAGGGGGGGcttatag
 6   gtcaatcatgttcttgtgaatggatttAAAAAAAAGGGGGGGgaccgcttggcgcacccaaattcagtgtgggcgagcgcaa
 7   cggttttggcccttgttagaggcccccgtAAAAAAAAGGGGGGGcaattatgagagagctaatctatcgcgtgcgtgttcat
 8   aacttgagttAAAAAAAAGGGGGGGctggggcacatacaagaggagtcttccttatcagttaatgctgtatgacactatgta
 9   ttggcccattggctaaaagcccaacttgacaaatggaagatagaatccttgcatAAAAAAAAGGGGGGGaccgaaaggaag
10   ctggtgagcaacgacagattcttacgtgcattagctcgcttccggggatctaatagcacgaagcttAAAAAAAAGGGGGGGa
```

Frequent Words with Mismatches Problem?

1 atgaccgggatactgat**AgAAgAAAGGttGGG**ggcgtacacattagataaacgtatgaagtacgttagactcggcgccgccg

2 acccctattttttgagcagatttagtgacctggaaaaaaatttgagtacaaaacttttccgaata**cAAtAAAAcGGcGGG**a

3 tgagtatccctgggatgactt**AAAAtAAtGGaGtGG**tgctctcccgatttttgaatatgtaggatcattcgccagggtccga

4 gctgagaattggatg**cAAAAAAAGGGattG**tccacgcaatcgcgaaccaacgcggacccaaaggcaagaccgataaaggaga

5 tcccttttgcggtaatgtgccgggaggctggttacgtagggaagccctaacggacttaat**AtAAtAAAGGaaGGG**cttatag

6 gtcaatcatgttcttgtgaatggattt**AAcAAtAAGGGctGG**gaccgcttggcgcacccaaattcagtgtgggcgagcgcaa

7 cggttttggcccttgttagaggcccccgt**AtAAAcAAGGaGGGc**caattatgagagagctaatctatcgcgtgcgtgttcat

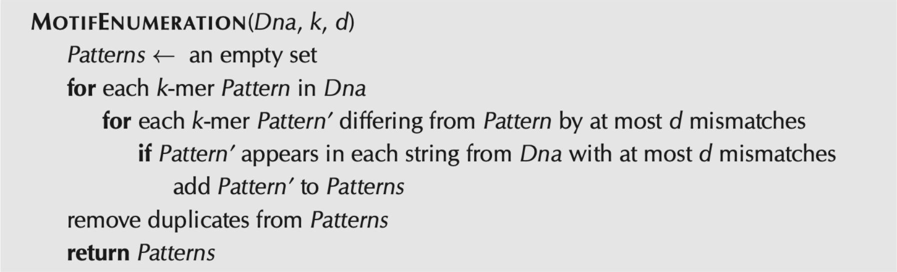
8 aacttgagtt**AAAAAAtAGGGaGcc**ctggggcacatacaagaggagtcttccttatcagttaatgctgtatgacactatgta

9 ttggcccattggctaaaagcccaacttgacaaatggaagatagaatccttgcat**ActAAAAAGGaGcGG**accgaaagggaag

10 ctggtgagcaacgacagattcttacgtgcattagctcgcttccggggatctaatagcacgaagctt**ActAAAAAGGaGcGG**a

- Concatenating all the sequences into a single string is inadequate because it does not correctly model the biological problem of motif finding.
- A DnaA box is a pattern that clumps, or appears frequently, within a relatively short interval of the genome.
- In contrast, a regulatory motif is a pattern that appears at least once (with variation) in each of many different regions that are scattered throughout the genome.

A brute force algorithm



```
MotifEnumeration(Dna, $k$, $d$)
    Patterns ← an empty set
    for each $k$-mer Pattern in Dna
        for each $k$-mer Pattern′ differing from Pattern by at most $d$ mismatches
            if Pattern′ appears in each string from Dna with at most $d$ mismatches
                add Pattern′ to Patterns
    remove duplicates from Patterns
    return Patterns
```

- Each string length = n, number of strings = t, what will be the run time?

A better algorithm?

$$d(\mathit{Pattern}, \mathit{Text}) = \min_{\text{all } k\text{-mers } \mathit{Pattern}' \text{ in } \mathit{Text}} \textsc{HammingDistance}(\mathit{Pattern}, \mathit{Pattern}')\,.$$

$$d(\mathbf{GATTCTCA}, \texttt{gcaaa}\mathbf{GACGCTGA}\texttt{ccaa}) = \mathbf{3}\,.$$

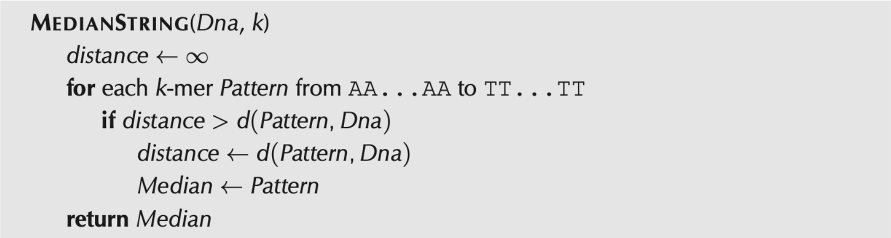
$$d(\mathit{Pattern}, \mathit{Dna}) = \sum_{i=1}^{t} d(\mathit{Pattern}, \mathit{Dna}_i).$$

For example, for the strings *Dna* shown below, $d(\mathbf{AAA}, \mathit{Dna}) = \mathbf{1} + \mathbf{1} + \mathbf{2} + \mathbf{0} + \mathbf{1} = \mathbf{5}$.

Dna	ttacctt**AAC**	**1**
	g**ATA**tctgtc	**1**
	ACGgcgttcg	**2**
	ccct**AAA**gag	**0**
	cgtc**AGA**ggt	**1**

Inspiring Excellence

A better algorithm?



```
MEDIANSTRING(Dna, k)
    distance ← ∞
    for each k-mer Pattern from AA...AA to TT...TT
        if distance > d(Pattern, Dna)
            distance ← d(Pattern, Dna)
            Median ← Pattern
    return Median
```

- Runtime? Comaparison to the brute force?

Profile Matrix

```
 1   T C G G G G g T T T t t
 2   c C G G t G A c T T a C
 3   a C G G G G A T T T t C
 4   T t G G G G A c T T t t
 5   a a G G G G A c T T C C
 6   T t G G G G A c T T C C
 7   T C G G G G A T T c a t
 8   T C G G G G A T T c C t
 9   T a G G G G A a c T a C
10   T C G G G t A T a a C C
```

$$
\mathit{Profile}\quad
\begin{array}{lcccccccccccc}
\mathtt{A}: & .2 & .2 & .0 & .0 & .0 & .0 & .9 & .1 & .1 & .1 & .3 & .0 \\
\mathtt{C}: & .1 & .6 & .0 & .0 & .0 & .0 & .0 & .4 & .1 & .2 & .4 & .6 \\
\mathtt{G}: & .0 & .0 & 1 & 1 & .9 & .9 & .1 & .0 & .0 & .0 & .0 & .0 \\
\mathtt{T}: & .7 & .2 & .0 & .0 & .1 & .1 & .0 & .5 & .8 & .7 & .3 & .4
\end{array}
$$

$$
\Pr(\mathtt{ACGGGGATTACC}|\mathit{Profile}) = .2 \cdot .6 \cdot 1 \cdot 1 \cdot .9 \cdot .9 \cdot .9 \cdot .5 \cdot .8 \cdot .1 \cdot .4 \cdot .6 = 0.000839808
$$

$$
\Pr(\mathtt{TCGGGGATTTCC}|\mathit{Profile}) = 0.7 \cdot 0.6 \cdot 1.0 \cdot 1.0 \cdot 0.9 \cdot 0.9 \cdot 0.9 \cdot 0.5 \cdot 0.8 \cdot 0.7 \cdot 0.4 \cdot 0.6
$$

$$
= 0.0205753\,,
$$

A scoring function for the motifs

Motifs

T	C	G	G	G	G	g	T	T	T	t	t
c	C	G	G	t	G	A	c	T	T	a	C
a	C	G	G	G	G	A	T	T	T	t	C
T	t	G	G	G	G	A	c	T	T	t	t
a	a	G	G	G	G	A	c	T	T	C	C
T	t	G	G	G	G	A	c	T	T	C	C
T	C	G	G	G	G	A	T	T	c	a	t
T	C	G	G	G	G	A	T	T	c	C	t
T	a	G	G	G	G	A	a	c	T	a	C
T	C	G	G	G	t	A	T	a	a	C	C

Score(*Motifs*) $3 + 4 + 0 + 0 + 1 + 1 + 1 + 5 + 2 + 3 + 6 + 4 = 30$

Count(*Motifs*)

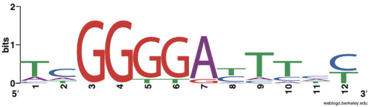
A:	2	2	0	0	0	0	9	1	1	1	3	0
C:	1	6	0	0	0	0	0	4	1	2	4	6
G:	0	0	10	10	9	9	1	0	0	0	0	0
T:	7	2	0	0	1	1	0	5	8	7	3	4

Profile(*Motifs*)

A:	.2	.2	0	0	0	0	.9	.1	.1	.1	.3	0
C:	.1	.6	0	0	0	0	0	.4	.1	.2	.4	.6
G:	0	0	1	1	.9	.9	.1	0	0	0	0	0
T:	.7	.2	0	0	.1	.1	0	.5	.8	.7	.3	.4

Consensus(*Motifs*)

T	C	G	G	G	G	A	T	T	T	C	C



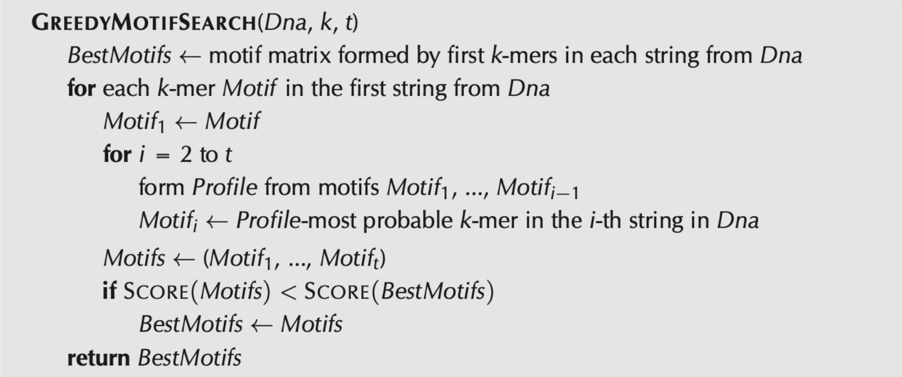
Entropy

- Entropy is a measure of the uncertainty of a probability distribution $(p_1, \cdots, p_N)$, and is defined as:

$$H(p_1, \cdots, p_N) = -\sum_{i=1}^{N} p_i \cdot log_2(p_i)$$

- The entropy of the probability distribution (0.2, 0.6, 0.0, 0.2) corresponding to the second column is $-(0.2 log_2 0.2 + 0.6 log_2 0.6 + 0.0 log_2 0.0 + 0.2 log_2 0.2) \approx 1.371$
- The entropy of the more conserved final column (0.0, 0.6, 0.0, 0.4) is $-(0.0 log_2 0.0 + 0.6 log_2 0.6 + 0.0 log_2 0.0 + 0.4 log_2 0.4) \approx 0.971$
- The entropy of the very conserved 5th column (0.0, 0.0, 0.9, 0.1) is $-(0.0 log_2 0.0 + 0.0 log_2 0.0 + 0.9 log_2 0.9 + 0.1 log_2 0.1) \approx 0.467$.
- The entropy of a motif matrix is defined as the sum of the entropies of its columns.

A Greedy Algorithm

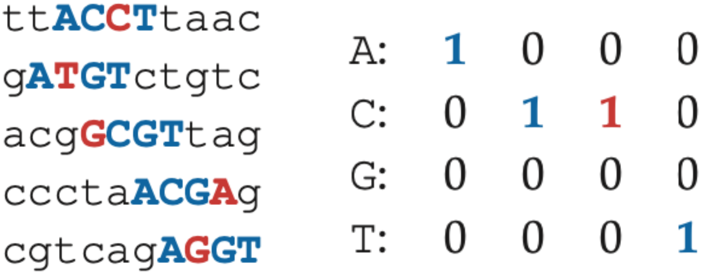


```
GREEDYMOTIFSEARCH(Dna, k, t)
    BestMotifs ← motif matrix formed by first k-mers in each string from Dna
    for each k-mer Motif in the first string from Dna
        $Motif_1$ ← Motif
        for i = 2 to t
            form Profile from motifs $Motif_1$, ..., $Motif_{i-1}$
            $Motif_i$ ← Profile-most probable k-mer in the i-th string in Dna
        Motifs ← ($Motif_1$, ..., $Motif_t$)
        if SCORE(Motifs) < SCORE(BestMotifs)
            BestMotifs ← Motifs
    return BestMotifs
```

- How good is this algorithm? It fails too.

Why greedy algorithm fails?

- Find the (4,1)-motif **ACGT** implanted in the following strings Dna.

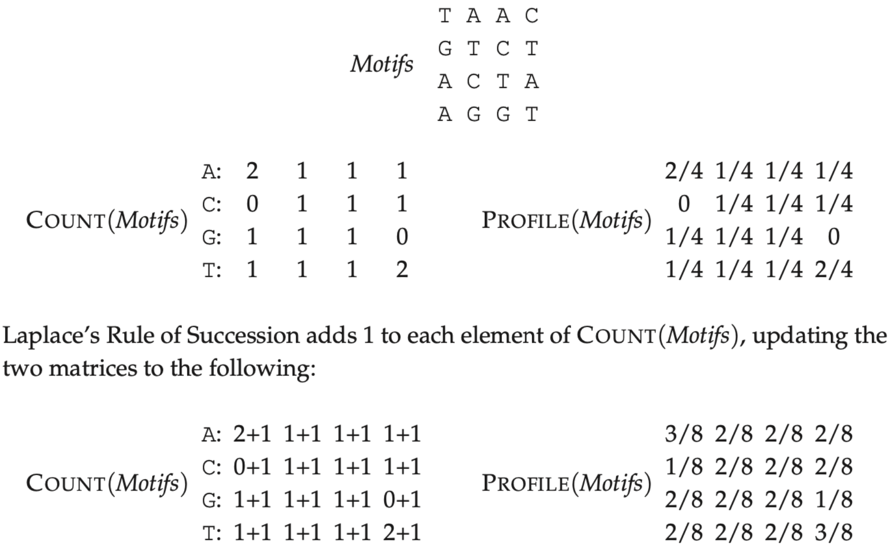


```
ttACCTtaac
gATGTctgtc
acgGCGTtag
ccctaACGAg
cgtcagAGGT
```

A:	1	0	0	0
C:	0	1	1	0
G:	0	0	0	0
T:	0	0	0	1

- Laplacian Correction Needed.

Laplacian Correction



$$\textit{Motifs}\quad \begin{array}{cccc} \mathtt{T} & \mathtt{A} & \mathtt{A} & \mathtt{C} \\ \mathtt{G} & \mathtt{T} & \mathtt{C} & \mathtt{T} \\ \mathtt{A} & \mathtt{C} & \mathtt{T} & \mathtt{A} \\ \mathtt{A} & \mathtt{G} & \mathtt{G} & \mathtt{T} \end{array}$$

$$\textsc{Count}(\textit{Motifs})\quad \begin{array}{lcccc} \mathtt{A}: & 2 & 1 & 1 & 1 \\ \mathtt{C}: & 0 & 1 & 1 & 1 \\ \mathtt{G}: & 1 & 1 & 1 & 0 \\ \mathtt{T}: & 1 & 1 & 1 & 2 \end{array} \qquad \textsc{Profile}(\textit{Motifs})\quad \begin{array}{cccc} 2/4 & 1/4 & 1/4 & 1/4 \\ 0 & 1/4 & 1/4 & 1/4 \\ 1/4 & 1/4 & 1/4 & 0 \\ 1/4 & 1/4 & 1/4 & 2/4 \end{array}$$

Laplace's Rule of Succession adds 1 to each element of $\textsc{Count}(\textit{Motifs})$, updating the two matrices to the following:

$$\textsc{Count}(\textit{Motifs})\quad \begin{array}{lcccc} \mathtt{A}: & 2{+}1 & 1{+}1 & 1{+}1 & 1{+}1 \\ \mathtt{C}: & 0{+}1 & 1{+}1 & 1{+}1 & 1{+}1 \\ \mathtt{G}: & 1{+}1 & 1{+}1 & 1{+}1 & 0{+}1 \\ \mathtt{T}: & 1{+}1 & 1{+}1 & 1{+}1 & 2{+}1 \end{array} \qquad \textsc{Profile}(\textit{Motifs})\quad \begin{array}{cccc} 3/8 & 2/8 & 2/8 & 2/8 \\ 1/8 & 2/8 & 2/8 & 2/8 \\ 2/8 & 2/8 & 2/8 & 1/8 \\ 2/8 & 2/8 & 2/8 & 3/8 \end{array}$$

Greedy Algorithm: Iteration 1

Motifs **ACCT**

$$\textsc{Count}(\mathit{Motifs}) \begin{array}{lcccc} \texttt{A:} & 1{+}1 & 0{+}1 & 0{+}1 & 0{+}1 \\ \texttt{C:} & 0{+}1 & 1{+}1 & 1{+}1 & 0{+}1 \\ \texttt{G:} & 0{+}1 & 0{+}1 & 0{+}1 & 0{+}1 \\ \texttt{T:} & 0{+}1 & 0{+}1 & 0{+}1 & 1{+}1 \end{array} \qquad \textsc{Profile}(\mathit{Motifs}) \begin{array}{cccc} 2/5 & 1/5 & 1/5 & 1/5 \\ 1/5 & 2/5 & 2/5 & 1/5 \\ 1/5 & 1/5 & 1/5 & 1/5 \\ 1/5 & 1/5 & 1/5 & 2/5 \end{array}$$

We use this profile matrix to compute the probabilities of all 4-mers in the second string from *Dna*:

g**ATG**	**ATGT**	**TGT**c	**GT**ct	**T**ctg	ctgt	tgtc
$1/5^4$	$4/5^4$	$1/5^4$	$4/5^4$	$2/5^4$	$2/5^4$	$1/5^4$

Greedy Algorithm: Iteration 2

Motifs ACCT
ATGT

$$\textsc{Count}(\textit{Motifs}) \quad \begin{array}{lcccc} \texttt{A:} & 2{+}1 & 0{+}1 & 0{+}1 & 0{+}1 \\ \texttt{C:} & 0{+}1 & 1{+}1 & 1{+}1 & 0{+}1 \\ \texttt{G:} & 0{+}1 & 0{+}1 & 1{+}1 & 0{+}1 \\ \texttt{T:} & 0{+}1 & 1{+}1 & 0{+}1 & 2{+}1 \end{array} \qquad \textsc{Profile}(\textit{Motifs}) \quad \begin{array}{cccc} 3/6 & 1/6 & 1/6 & 1/6 \\ 1/6 & 2/6 & 2/6 & 1/6 \\ 1/6 & 1/6 & 2/6 & 1/6 \\ 1/6 & 2/6 & 1/6 & 3/6 \end{array}$$

We use this profile matrix to compute the probabilities of all 4-mers in the third string from *Dna*:

acgG	cgGC	gGCG	GCGT	CGTt	GTta	Ttag
$12/6^4$	$2/6^4$	$2/6^4$	$12/6^4$	$3/6^4$	$2/6^4$	$2/6^4$

Greedy Algorithm: Iteration 3

$$Motifs \quad \begin{array}{l} \texttt{ACCT} \\ \texttt{ATGT} \\ \texttt{acgG} \end{array}$$

$$\text{COUNT}(Motifs) \quad \begin{array}{lcccc} \texttt{A:} & 3+1 & 0+1 & 0+1 & 1+1 \\ \texttt{C:} & 0+1 & 2+1 & 1+1 & 0+1 \\ \texttt{G:} & 0+1 & 0+1 & 2+1 & 1+1 \\ \texttt{T:} & 0+1 & 1+1 & 0+1 & 2+1 \end{array} \qquad \text{PROFILE}(Motifs) \quad \begin{array}{cccc} 4/7 & 1/7 & 1/7 & 1/7 \\ 1/7 & 3/7 & 2/7 & 1/7 \\ 1/7 & 1/7 & 3/7 & 2/7 \\ 1/7 & 2/7 & 1/7 & 3/7 \end{array}$$

We use this profile matrix to compute probabilities of all 4-mers in the fourth string from *Dna*:

ccct	ccta	ctaA	taAC	aACG	ACGA	CGAg
$18/7^4$	$3/7^4$	$2/7^4$	$1/7^4$	$16/7^4$	$36/7^4$	$2/7^4$

Greedy Algorithm: Iteration 4

Motifs

ACCT
ATGT
acgG
ACGA

COUNT(*Motifs*)

A:	4+1	0+1	0+1	0+1
C:	0+1	3+1	1+1	0+1
G:	0+1	0+1	3+1	1+1
T:	0+1	1+1	0+1	2+1

PROFILE(*Motifs*)

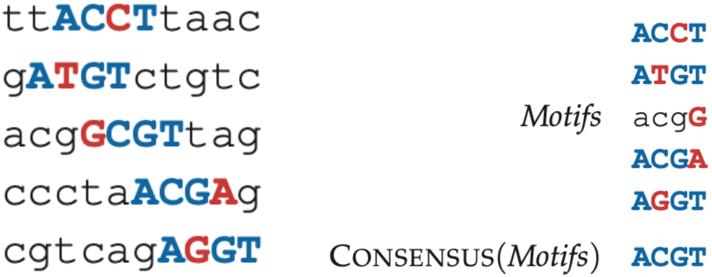
5/8	1/8	1/8	2/8
1/8	4/8	2/8	1/8
1/8	1/8	4/8	2/8
1/8	2/8	1/8	3/8

We now use this profile to compute the probabilities of all 4-mers in the fifth string in *Dna*:

cgtc	gtca	tcag	cagA	agAG	gAGG	AGGT
$1/8^4$	$8/8^4$	$8/8^4$	$8/8^4$	$10/8^4$	$8/8^4$	$60/8^4$

Greedy Algorithm: Consensus

- Find the (4,1)-motif **ACGT** implanted in the following strings Dna.



ttACCTtaac

gATGTctgtc

acgGCGTtag

ccctaACGAg

cgtcagAGGT

Motifs

ACCT

ATGT

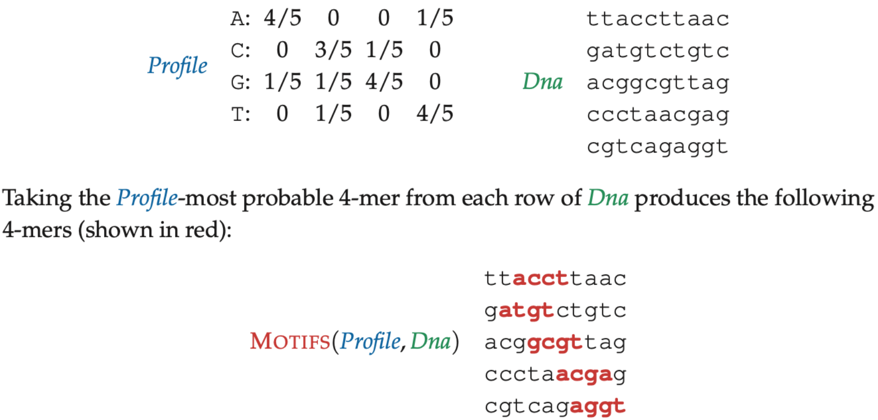
acgG

ACGA

AGGT

CONSENSUS(*Motifs*) ACGT

Further Improvement!



Profile

A:	4/5	0	0	1/5
C:	0	3/5	1/5	0
G:	1/5	1/5	4/5	0
T:	0	1/5	0	4/5

Dna

```
ttaccttaac
gatgtctgtc
acggcgttag
ccctaacgag
cgtcagaggt
```

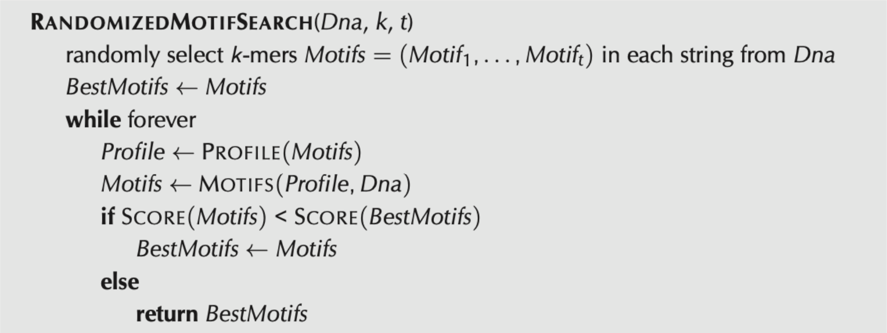
Taking the *Profile*-most probable 4-mer from each row of *Dna* produces the following 4-mers (shown in red):

MOTIFS(*Profile*, *Dna*)

tt**acct**taac
g**atgt**ctgtc
acg**gcgt**tag
cccta**acga**g
cgtcag**aggt**

- Why would we do this? Because our hope is that MOTIFS(PROFILE(Motifs), Dna) has a better score than the original collection of k-mers Motifs. We can then form the profile matrix of these k-mers, PROFILE(MOTIFS(PROFILE(Motifs), Dna)), and continue...

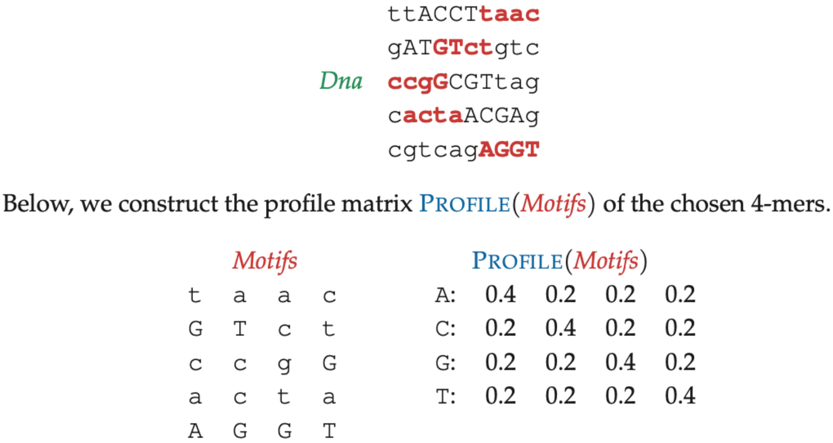
A Monte Carlo Algorithm



```
RANDOMIZEDMOTIFSEARCH(Dna, k, t)
    randomly select k-mers Motifs = ($Motif_1, \ldots, Motif_t$) in each string from Dna
    BestMotifs ← Motifs
    while forever
        Profile ← PROFILE(Motifs)
        Motifs ← MOTIFS(Profile, Dna)
        if SCORE(Motifs) < SCORE(BestMotifs)
            BestMotifs ← Motifs
        else
            return BestMotifs
```

- Since a single run of RANDOMIZEDMOTIFSEARCH may generate a rather poor set of motifs, bioinformaticians usually run this algorithm thousands of times. On each run, they begin from a new randomly selected set of k-mers, selecting the best set of k-mers found in all these runs.

A Monte Carlo Algorithm



Dna

ttACCT**taac**
gAT**GTct**gtc
ccgGCGTtag
c**acta**ACGAg
cgtcag**AGGT**

Below, we construct the profile matrix PROFILE(*Motifs*) of the chosen 4-mers.

Motifs

t	a	a	c
G	T	c	t
c	c	g	G
a	c	t	a
A	G	G	T

PROFILE(*Motifs*)

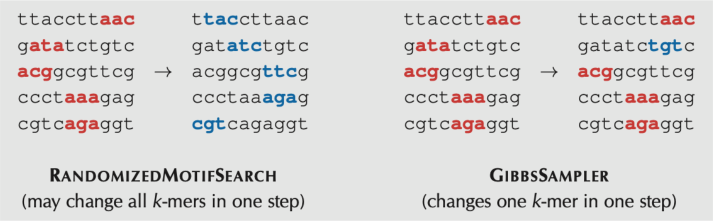
A:	0.4	0.2	0.2	0.2
C:	0.2	0.4	0.2	0.2
G:	0.2	0.2	0.4	0.2
T:	0.2	0.2	0.2	0.4

A Monte Carlo Algorithm

ttAC	tACC	ACCT	CCTt	CTta	Ttaa	taac
.0016	.0016	.0128	.0064	.0016	.0016	.0016
gATG	ATGT	TGTc	GTct	Tctg	ctgt	tgtc
.0016	.0128	.0016	.0032	.0032	.0032	.0016
ccgG	cgGC	gGCG	GCGT	CGTt	GTta	Ttag
.0064	.0036	.0016	.0128	.0032	.0016	.0016
cact	acta	ctaA	taAC	aACG	ACGA	CGAg
.0032	.0064	.0016	.0016	.0032	.0128	.0016
cgtc	gtca	tcag	cagA	agAG	gAGG	AGGT
.0016	.0016	.0016	.0032	.0032	.0032	.0128

Gibbs Sampling

- RANDOMIZEDMOTIFSEACH may change all *t* strings in Motifs in a single iteration.
- This strategy may prove reckless, since some correct motifs (captured in Motifs) may potentially be discarded at the next iteration. GIBBSSAMPLER is a more cautious iterative algorithm that discards a single *k*-mer from the current set of motifs at each iteration and decides to either keep it or replace it with a new one.

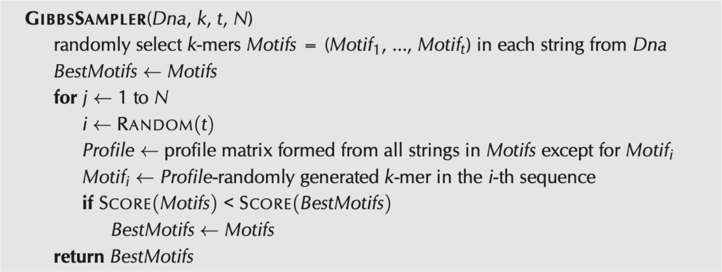


ttacctt**aac**		t**tac**cttaac	ttacctt**aac**		ttacctt**aac**
gatatctgtc		gat**atc**tgtc	**gat**atctgtc		gatatc**tgt**c
acggcgttcg	→	acggcg**ttc**g	**acg**gcgttcg	→	**acg**gcgttcg
ccct**aaa**gag		ccctaa**aga**g	ccct**aaa**gag		ccct**aaa**gag
cgtc**aga**ggt		**cgt**cagaggt	cgtc**aga**ggt		cgtc**aga**ggt

RANDOMIZEDMOTIFSEARCH
(may change all *k*-mers in one step)

GIBBSSAMPLER
(changes one *k*-mer in one step)

Gibbs Sampling



```
GIBBSSAMPLER($Dna$, $k$, $t$, $N$)
    randomly select $k$-mers $Motifs = (Motif_1, ..., Motif_t)$ in each string from $Dna$
    $BestMotifs \leftarrow Motifs$
    for $j \leftarrow 1$ to $N$
        $i \leftarrow$ RANDOM($t$)
        $Profile \leftarrow$ profile matrix formed from all strings in $Motifs$ except for $Motif_i$
        $Motif_i \leftarrow Profile$-randomly generated $k$-mer in the $i$-th sequence
        if SCORE($Motifs$) < SCORE($BestMotifs$)
            $BestMotifs \leftarrow Motifs$
    return $BestMotifs$
```

- What is the value of k?
- What if the nucleotide distribution is skewed?

Gibbs Sampling: Iteration 1

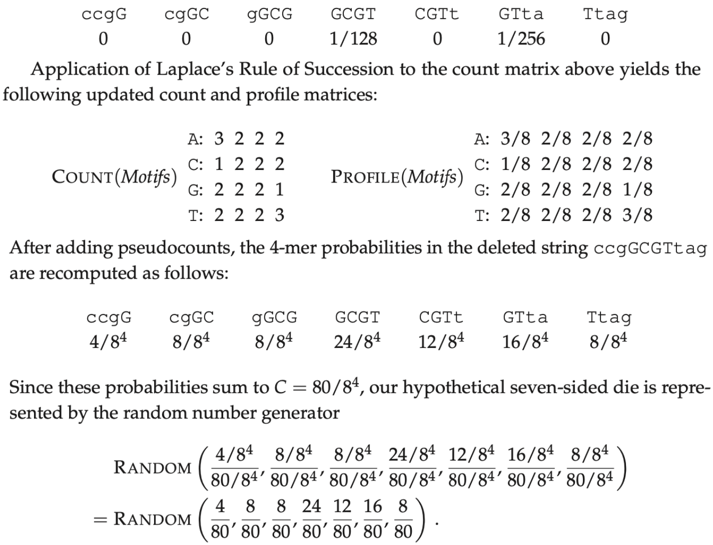
	ttACCT**taac**		ttACCT**taac**
	gAT**GTct**gtc		gAT**GTct**gtc
Dna	**ccgG**CGTtag	⟶	----------
	c**acta**ACGAg		c**acta**ACGAg
	cgtcag**AGGT**		cgtcag**AGGT**

This results in the following motif, count, and profile matrices.

$$\textit{Motifs} \quad \begin{array}{cccc} \mathtt{t} & \mathtt{a} & \mathtt{a} & \mathtt{c} \\ \mathtt{G} & \mathtt{T} & \mathtt{c} & \mathtt{t} \\ \mathtt{a} & \mathtt{c} & \mathtt{t} & \mathtt{a} \\ \mathtt{A} & \mathtt{G} & \mathtt{G} & \mathtt{T} \end{array}$$

$$\text{COUNT}(\textit{Motifs}) \quad \begin{array}{lcccc} \mathtt{A}: & 2 & 1 & 1 & 1 \\ \mathtt{C}: & 0 & 1 & 1 & 1 \\ \mathtt{G}: & 1 & 1 & 1 & 0 \\ \mathtt{T}: & 1 & 1 & 1 & 2 \end{array} \qquad \text{PROFILE}(\textit{Motifs}) \quad \begin{array}{lcccc} \mathtt{A}: & 2/4 & 1/4 & 1/4 & 1/4 \\ \mathtt{C}: & 0 & 1/4 & 1/4 & 1/4 \\ \mathtt{G}: & 1/4 & 1/4 & 1/4 & 0 \\ \mathtt{T}: & 1/4 & 1/4 & 1/4 & 2/4 \end{array}$$

Gibbs Sampling: Iteration 2



ccgG	cgGC	gGCG	GCGT	CGTt	GTta	Ttag
0	0	0	1/128	0	1/256	0

Application of Laplace's Rule of Succession to the count matrix above yields the following updated count and profile matrices:

$$\text{COUNT}(\textit{Motifs}) \begin{array}{lllll} \texttt{A:} & 3 & 2 & 2 & 2 \\ \texttt{C:} & 1 & 2 & 2 & 2 \\ \texttt{G:} & 2 & 2 & 2 & 1 \\ \texttt{T:} & 2 & 2 & 2 & 3 \end{array} \qquad \text{PROFILE}(\textit{Motifs}) \begin{array}{lllll} \texttt{A:} & 3/8 & 2/8 & 2/8 & 2/8 \\ \texttt{C:} & 1/8 & 2/8 & 2/8 & 2/8 \\ \texttt{G:} & 2/8 & 2/8 & 2/8 & 1/8 \\ \texttt{T:} & 2/8 & 2/8 & 2/8 & 3/8 \end{array}$$

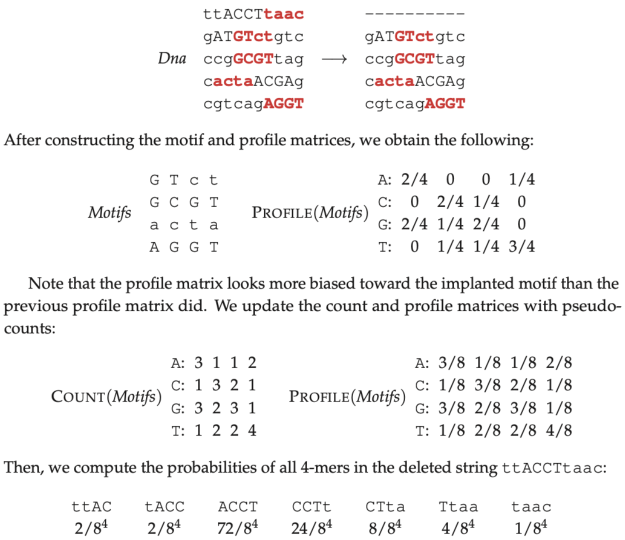
After adding pseudocounts, the 4-mer probabilities in the deleted string `ccgGCGTtag` are recomputed as follows:

ccgG	cgGC	gGCG	GCGT	CGTt	GTta	Ttag
$4/8^4$	$8/8^4$	$8/8^4$	$24/8^4$	$12/8^4$	$16/8^4$	$8/8^4$

Since these probabilities sum to $C = 80/8^4$, our hypothetical seven-sided die is represented by the random number generator

$$\text{RANDOM}\left(\frac{4/8^4}{80/8^4}, \frac{8/8^4}{80/8^4}, \frac{8/8^4}{80/8^4}, \frac{24/8^4}{80/8^4}, \frac{12/8^4}{80/8^4}, \frac{16/8^4}{80/8^4}, \frac{8/8^4}{80/8^4}\right)$$
$$= \text{RANDOM}\left(\frac{4}{80}, \frac{8}{80}, \frac{8}{80}, \frac{24}{80}, \frac{12}{80}, \frac{16}{80}, \frac{8}{80}\right).$$

Gibbs Sampling: Iteration 3



$$\textit{Dna}\quad \begin{array}{l} \texttt{ttACCT}\mathbf{\texttt{taac}} \\ \texttt{gAT}\mathbf{\texttt{GTct}}\texttt{gtc} \\ \texttt{ccg}\mathbf{\texttt{GCGT}}\texttt{tag} \\ \texttt{c}\mathbf{\texttt{acta}}\texttt{ACGAg} \\ \texttt{cgtcag}\mathbf{\texttt{AGGT}} \end{array} \longrightarrow \begin{array}{l} \texttt{----------} \\ \texttt{gAT}\mathbf{\texttt{GTct}}\texttt{gtc} \\ \texttt{ccg}\mathbf{\texttt{GCGT}}\texttt{tag} \\ \texttt{c}\mathbf{\texttt{acta}}\texttt{ACGAg} \\ \texttt{cgtcag}\mathbf{\texttt{AGGT}} \end{array}$$

After constructing the motif and profile matrices, we obtain the following:

$$\textit{Motifs}\quad \begin{array}{cccc} \texttt{G} & \texttt{T} & \texttt{c} & \texttt{t} \\ \texttt{G} & \texttt{C} & \texttt{G} & \texttt{T} \\ \texttt{a} & \texttt{c} & \texttt{t} & \texttt{a} \\ \texttt{A} & \texttt{G} & \texttt{G} & \texttt{T} \end{array} \qquad \textsc{Profile}(\textit{Motifs}) \begin{array}{lcccc} \texttt{A:} & 2/4 & 0 & 0 & 1/4 \\ \texttt{C:} & 0 & 2/4 & 1/4 & 0 \\ \texttt{G:} & 2/4 & 1/4 & 2/4 & 0 \\ \texttt{T:} & 0 & 1/4 & 1/4 & 3/4 \end{array}$$

Note that the profile matrix looks more biased toward the implanted motif than the previous profile matrix did. We update the count and profile matrices with pseudocounts:

$$\textsc{Count}(\textit{Motifs}) \begin{array}{lcccc} \texttt{A:} & 3 & 1 & 1 & 2 \\ \texttt{C:} & 1 & 3 & 2 & 1 \\ \texttt{G:} & 3 & 2 & 3 & 1 \\ \texttt{T:} & 1 & 2 & 2 & 4 \end{array} \qquad \textsc{Profile}(\textit{Motifs}) \begin{array}{lcccc} \texttt{A:} & 3/8 & 1/8 & 1/8 & 2/8 \\ \texttt{C:} & 1/8 & 3/8 & 2/8 & 1/8 \\ \texttt{G:} & 3/8 & 2/8 & 3/8 & 1/8 \\ \texttt{T:} & 1/8 & 2/8 & 2/8 & 4/8 \end{array}$$

Then, we compute the probabilities of all 4-mers in the deleted string `ttACCTtaac`:

ttAC	tACC	ACCT	CCTt	CTta	Ttaa	taac
$2/8^4$	$2/8^4$	$72/8^4$	$24/8^4$	$8/8^4$	$4/8^4$	$1/8^4$

Gibbs Sampling: Iteration 4

	tt**ACCT**taac		tt**ACCT**taac
	gAT**GTct**gtc		gAT**GTct**gtc
Dna	ccg**GCGT**tag	⟶	ccg**GCGT**tag
	c**acta**ACGAg		----------
	cgtcag**AGGT**		cgtcag**AGGT**

We further add pseudocounts and construct the resulting count and profile matrices:

$$\textit{Motifs} \quad \begin{array}{cccc} \mathtt{A} & \mathtt{C} & \mathtt{C} & \mathtt{T} \\ \mathtt{G} & \mathtt{T} & \mathtt{c} & \mathtt{t} \\ \mathtt{G} & \mathtt{C} & \mathtt{G} & \mathtt{T} \\ \mathtt{A} & \mathtt{G} & \mathtt{G} & \mathtt{T} \end{array}$$

$$\textsc{Count}(\textit{Motifs}) \quad \begin{array}{lcccc} \mathtt{A}: & 3 & 1 & 1 & 1 \\ \mathtt{C}: & 1 & 3 & 3 & 1 \\ \mathtt{G}: & 3 & 2 & 3 & 1 \\ \mathtt{T}: & 1 & 2 & 1 & 5 \end{array} \qquad \textsc{Profile}(\textit{Motifs}) \quad \begin{array}{lcccc} \mathtt{A}: & 3/8 & 1/8 & 1/8 & 1/8 \\ \mathtt{C}: & 1/8 & 3/8 & 3/8 & 1/8 \\ \mathtt{G}: & 3/8 & 2/8 & 3/8 & 1/8 \\ \mathtt{T}: & 1/8 & 2/8 & 1/8 & 5/8 \end{array}$$

We now compute the probabilities of all 4-mers in the deleted string cactaACGAg:

cact	acta	ctaA	taAC	aACG	ACGA	CGAg
$15/8^4$	$9/8^4$	$2/8^4$	$1/8^4$	$9/8^4$	$27/8^4$	$2/8^4$