Um Modelo Oculto de Markov para encontrar promotores em seqüências de DNA

Nalvo Franco de Almeida Jr.
nalvo@dct.ufms.br

João Carlos Setubal
setubal@dcc.unicamp.br

Relatório Técnico IC–98-37

Outubro de 1998

Um Modelo Oculto de Markov para encontrar promotores em seqüências de DNA*

Nalvo Franco de Almeida Jr. †
nalvo@dct.ufms.br

João Carlos Setubal ‡
setubal@dcc.unicamp.br

Sumário

Apresentamos um Modelo Oculto de Markov (HMM) para encontrar sítios de ligação, como promotores, numa seqüência de DNA. Esta abordagem permite que o espaço entre as seqüências consenso seja de tamanho variável. O modelo foi construído usando 150 promotores conhecidos do genoma de *Escherichia coli* e usa o algoritmo de Expectation-Maximization (EM) para reestimar parâmetros. Para testar o modelo, usamos 30 regiões de *E.coli*, cada uma delas contendo um promotor. Cortes aleatórios dessas regiões produziram 20 conjuntos de 30 seqüências. O modelo foi capaz de determinar corretamente ou quase corretamente (dentro de 6 bp) 78% das seqüências consenso de um conjunto, na média. O programa está disponível via WWW e pode ser utilizado como uma ferramenta de auxílio na busca de um promotor numa seqüência de DNA procariótico.

Abstract

We present a Hidden Markov Model (HMM) to find binding sites, like promoters, in a DNA sequence. This approach allows variable-length spacers between the consensus sequences. The model was built using 150 known promoters of the *Escherichia coli* genome and uses the Expectation-Maximization (EM) algorithm to reestimate parameters. In order to test the model, we used 30 regions of *E.coli*, each one known to contain a promoter. By cutting randomly these regions, we produced 20 sets of 30 sequences. The model was able to determine the correct or nearly correct (within 6 bp) 78% of the consensus sequences of a set, on average. The program is available through the WWW and can be useful as a tool to find a promoter in any procaryotic DNA sequence.

1 Introdução

Podemos definir *Biologia Molecular Computacional* como o estudo e a aplicação de modelos e técnicas computacionais aos problemas de biologia molecular. Uma das sub-áreas mais

*Parte deste trabalho foi desenvolvido pelo primeiro autor no Department of Computer Science & Engineering, University of Washington, Seattle, USA, com bolsa CAPES/Fulbright(0627/96). Os autores são membros do Projeto PRONEX 107/97 (MCT/FINEP).

†Departamento de Computação e Estatística, UFMS, CP 549, Campo Grande, MS, 79070-900.

‡Instituto de Computação, Universidade Estadual de Campinas, CP 6176, Campinas, SP, 13081-970.

importantes consiste no desenvolvimento de ferramentas para a análise de biosseqüências (DNA ou proteína). Em particular, estamos interessados na procura de trechos especiais, chamados *promotores*, em seqüências de DNA. Estamos supondo que o leitor esteja familiarizado com alguns conceitos básicos de Biologia Molecular Computacional, bem como de Biologia Molecular. Tais conceitos básicos podem ser vistos em [6] e [10].

Um *promotor* é uma seqüência de nucleotídeos que sinaliza a proximidade do início de um gene. É o ponto onde a enzima de transcrição do DNA (RNA Polimerase II) se liga fortemente, dando início à transcrição. Por isso é considerado um *sítio de ligação*. Em contraste com regiões expressas de seqüências de DNA, cuja função se torna aparente somente quando são traduzidas para proteínas ou para outras seqüências de DNA, a função de um promotor é dada diretamente pela sua seqüência de nucleotídeos. Além disso, as seqüências que definem os vários promotores de um organismo apresentam variação relativamente pequena entre si, mesmo entre organismos diferentes mas pertencentes a um mesmo grupo. Quando esse fenômeno ocorre dizemos que as seqüências se *conservam*.

Neste trabalho estaremos lidando com promotores de procariotos. Basicamente, um promotor procariótico tem três principais componentes: a *seqüência -35*, a *seqüência -10*, e o espaço entre elas. Essas seqüências são também chamadas de *seqüências consenso*, e têm 6 bases cada. Esses números (-10 e -35) estão relacionados à distância de cada trecho (em número de bases) ao ponto inicial da transcrição, e foram convencionados dessa forma. A principal característica do promotor consiste no fato de que cada posição de ambos os trechos obedece a algum tipo de conservação. Além disso, o tamanho (número de bases) do espaço (que é variável) é relevante, enquanto que suas bases parecem não ter qualquer conservação [6].

Várias tentativas de se encontrar sítios com algum tipo de conservação aparecem na literatura. Em [11], por exemplo, um algoritmo guloso para encontrar alinhamentos entre tais segmentos conservados é apresentado. Essa abordagem encontra somente segmentos de tamanho fixo.

Apresentamos um Modelo Oculto de Markov (HMM – Hidden Markov Model) [8, 9] e um algoritmo de Expectation-Maximization (EM) [8, 9] para encontrar um promotor numa seqüência de DNA procariótico. Um HMM descreve uma série de observações através de um processo estocástico oculto. Basicamente, nosso modelo tem a seqüência de DNA como uma série de símbolos observados e as posições do promotor como a componente oculta do modelo. O algoritmo EM é uma maneira iterativa de melhorar o modelo, no sentido de se obter resultados cuja probabilidade é maior do que a obtida no passo anterior, até que se consiga um máximo local.

O uso de HMM na biologia molecular computacional é bem difundido. Krogh *et al.* [5] usam HMM na construção de alinhamentos múltiplos de seqüências. Henderson *et al.* [3] usam HMM para identificar introns, exons e sítios de junção. Stormo e Cardon [1] encontram sítios de ligação com um HMM e um algoritmo EM.

A principal diferença entre nosso modelo e aquele descrito em [1] está no fato de que Stormo e Cardon usam o algoritmo EM numa forma bem mais complexa, enquanto que o nosso usa

um caso particular deste algoritmo, aplicado em um HMM muito simples, como proposto em [8, 9].

Na seção 2 mostramos uma descrição mais detalhada de um HMM e do algoritmo EM. A seção 3 mostra nosso modelo. Resultados empíricos são mostrados na seção 4. Finalmente, na seção 5, fazemos alguns comentários.

2 Preliminares

Nesta seção fazemos uma breve introdução ao HMM e ao algoritmo EM. Para uma introdução mais detalhada, veja [8, 9].

Nosso objetivo é modelar um processo no qual alguns detalhes não são conhecidos, ou estão *ocultos*. Um bom exemplo dado por Rabiner [8] dá uma visão geral de HMM e como este modelo se encaixa neste tipo de situação. Suponha que você esteja numa sala com uma cortina através da qual você não pode ver o que está acontecendo. No outro lado está uma pessoa que tem 3 moedas e que fará uma série de lançamentos dessas moedas. A pessoa não lhe dirá qual moeda está sendo lançada, somente o resultado de cada lançamento. Dada a seqüência *observada* de caras e coroas, o principal problema é construir um modelo que descubra qual a seqüência *oculta* de moedas usadas. Obviamente esta seqüência oculta dependerá do vício de cada uma das moedas e também da probabilidade de a pessoa trocar de moeda, após cada lançamento.

No nosso problema, a parte observada do modelo é a seqüência de DNA, enquanto que a parte oculta é a posição de cada um dos componentes do promotor na seqüência.

Agora vamos descrever formalmente os elementos de um HMM.

Estados – Os estados são a parte escondida de um HMM. Seja $\mathcal{S} = \{S_0, \ldots, S_{n-1}\}$ o conjunto de n estados. Denotamos o estado no instante t como s_t.

Símbolos de observação – É a parte do modelo que realmente pode ser observada. Nós denotamos o conjunto de símbolos como $\mathcal{V} = \{V_1, \ldots, V_m\}$. Isto significa que a cada instante t, estes são os m símbolos possiveis de serem observados em qualquer estado do modelo.

Distribuição de probabilidade das transições de estados – A distribuição de probabilidade das transições $A = \{a_{ij}\}$, dada por

$$a_{ij} = \Pr[s_{t+1} = S_j \mid s_t = S_i], \ \ 0 \leq i, j \leq n-1,$$

é a probabilidade de que o modelo possa estar no estado S_j no instante $t+1$, dado que o modelo esteve em S_i no instante t. Note que $\sum_{j=0}^{n-1} a_{ij} = 1, \ \ 0 \leq i \leq n-1$.

Distribuição de probabilidade dos símbolo de observação – A distribuição de probabilidade $B = \{b_{v_k,j}\}$, dada por

$$b_{v_k,j} = \Pr[V_k \text{ at time } t | s_t = S_j], \quad 0 \leq j \leq n-1, \ \ 1 \leq k \leq m,$$

é a probabilidade de que o modelo observe o símbolo V_k no instante t, dado o estado S_j no instante t.

Distribuição de probabilidade dos estados iniciais $C = \{c_j\}$.
$c_j = \Pr[s_1 = S_j],\ 0 \leq j \leq n-1$, é a probabilidade de que o modelo alcance o estado S_j no instante $t = 1$.

Dados valores para n, m, A, B e C, o HMM pode ser usado para gerar a seqüência observada $O = o_1 o_2 \ldots o_{\bar{t}}$, onde cada $o_t \in \mathcal{V}$, $t = 1 \ldots \bar{t}$ de acordo com o seguinte algoritmo.

1. escolha, de acordo com $c(i)$, um estado inicial S_i, isto é, $s_1 = S_i$
2. $t = 1$
3. escolha $o_t = V_k$, de acordo com $b_{v_k,i}$
4. se $t = \bar{t}$ então pare
5. vá para o estado S_j, de acordo com a_{ij}
6. $t = t + 1$
7. vá para o passo 3

O algoritmo acima pode ser visto tanto como um gerador de observações, quanto como um modelo para reconhecer a seqüência observada. A notação usual para um HMM Λ é $\Lambda = (A, B, C)$.

Dado um modelo Λ e uma seqüência de observação $O = o_1 \ldots o_{\bar{t}}$, uma boa pergunta é: qual a probabilidade de O, dado Λ? Calcular $\Pr[O|\Lambda]$ é como enumerar todos os possíveis caminhos através do modelo, começando em s_1 e terminando no estado $s_{\bar{t}}$, ou seja,

$$\Pr[O|\Lambda] = \sum_{\substack{\text{para qualquer} \\ s_1 \ldots s_{\bar{t}}}} c_{s_1} \cdot b_{o_1,s_1} \cdot a_{s_1 s_2} \cdot b_{o_2,s_2} \cdots a_{s_{\bar{t}-1} s_{\bar{t}}} \cdot b_{o_{\bar{t}},s_{\bar{t}}}$$

Este cálculo requer tempo $O(\bar{t} \cdot n^{\bar{t}})$, o que obviamente não é viável. Existe um algoritmo, chamado *Forward-Backward* [8, 9], que responde a questão acima em tempo $O(n^2 \cdot \bar{t})$. Vamos agora mostrar como ele funciona.

Considere $\alpha_t(i) = \Pr[o_1 \ldots o_t, s_t = S_i|\Lambda]$. Esta é a probabilidade de se ter a seqüência parcial de observação $o_1 \cdots o_{\bar{t}}$ e o estado s_i no instante t, dado o modelo Λ. Então,

$$\begin{aligned} \alpha_1(i) &= c_i b_{o_1,i}, \quad 0 \leq i \leq n-1, \quad \text{e} \\ \alpha_{t+1}(j) &= \left(\sum_{i=0}^{n-1} \alpha_t(i) a_{ij} \right) \cdot b_{o_{t+1},j}, \quad 1 \leq t \leq \bar{t} - 1, \quad 0 \leq j \leq n-1. \end{aligned}$$

Ao final, temos $\Pr[O|\Lambda] = \sum_{i=0}^{n-1} \alpha_{\bar{t}}(i)$.

Da mesma maneira, podemos ter o oposto, da seguinte forma. Considere a variável

$$\beta_t(j) = \Pr[o_{t+1} \ldots o_t | s_t = S_i, \Lambda].$$

Esta é a probabilidade da seqüência parcial de observaçao de $t+1$ até o fim, dado o estado S_i no instante t e o modelo Λ. Novamente, podemos calcular todos os $\beta_t(j)$ em tempo

$O(n^2 \cdot \bar{t})$ indutivamente, como segue:

$$\begin{aligned} \beta_{\bar{t}}(j) &= 1, \quad 0 \le j \le n-1, \\ \beta_t(i) &= \sum_{j=0}^{n-1} a_{ij} b_{o_{t+1},j} \beta_{t+1}(j), \quad t = \bar{t}, \ldots, 1, \quad 0 \le i \le n-1. \end{aligned}$$

As variáveis α e β são úteis para resolver alguns problemas relacionados ao HMM, como veremos mais adiante.

A segunda importante pergunta a respeito de um HMM está relacionada à escolha do melhor caminho envolvendo os estados do modelo. Em outras palavras, como escolher a seqüência oculta $R = s_1 \ldots s_{\bar{t}}$, que melhor explica a observação, dados a seqüência O de símbolos observados e o modelo Λ.

Existe um algoritmo de programação dinâmica, chamado *algoritmo de Viterbi* [8, 9], que resolve este problema eficientemente. Para tanto, precisamos definir a variável

$$\delta_t(i) = \max_{s_1 \ldots s_{t-1}} \Pr[s_1 \ldots s_t = S_i, o_1 \ldots o_t | \Lambda].$$

$\delta_t(i)$ é o melhor score (mais alta probabilidade) alcançado por um único caminho, no instante t, que leva em conta os primeiros t símbolos de observação e que termina no estado S_i. Por indução,

$$\begin{aligned} \delta_1(i) &= c_i b_{o_1,i}, \quad 0 \le i \le n-1, \text{ and} \\ \delta_t(j) &= \max_{0 \le i \le n-1} \{\delta_{t-1}(i) a_{ij}\} \cdot b_{o_t,j}, \quad 2 \le t \le \bar{t}, \quad 0 \le j \le n-1. \end{aligned}$$

Com o objetivo de recuperar a seqüência de estados, é suficiente mantermos o argumento que maximizou $\delta_t(j)$.

Algoritmo de Expectation-Maximization (EM)

O mais difícil problema envolvendo HMMs é o de ajustar os parâmetros do modelo (A, B e C), no sentido de maximizar a probabilidade da seqüência de observação, dado o modelo. Não existe um método para tal tarefa. No entanto, podemos encontrar um máximo local, começando com um modelo $\Lambda = (A, B, C)$. Tal procedimento é chamado *método de Baum-Welch* [8], também conhecido como *algoritmo de expectation-maximization*.

Considere agora algumas variáveis necessárias para a descrição do algoritmo EM.

$\gamma_t(i) = \Pr[s_t = S_i \mid O, \Lambda]$ é a probabilidade de o modelo estar no estado S_i no instante t, dados O e Λ. Usando as variáveis forward-backward,

$$\gamma_t(i) = \frac{\alpha_t(i) \cdot \beta_t(i)}{\Pr[O|\Lambda]}, \quad 0 \le i \le n-1, \quad 1 \le t \le \bar{t}.$$

$\epsilon_t(i,j)$, dada por

$$\epsilon_t(i,j) = \Pr[s_t = S_i, s_{t+1} = S_j \mid O, \Lambda],$$

é a probabilidade de o modelo estar no estado S_i no instante t e no estado S_j no instante $t+1$, dados O e Λ. Esta probabilidade pode ser dada por

$$\epsilon_t(i,j) = \frac{\alpha_t(i) a_{ij} b_{o_{t+1},j} \beta_{t+1}(j)}{\Pr[O|\Lambda]}, \quad 0 \leq i,j \leq n-1, \quad 1 \leq t \leq \bar{t}-1.$$

Então, o número esperado de transições a partir do estado S_i é dado por $\sum_{t=1}^{\bar{t}-1} \gamma_t(i)$, enquanto que o número esperado de transições a do estado S_i para o estado S_j é dado por $\sum_{t=1}^{\bar{t}-1} \epsilon_t(i,j)$, $0 \leq i,j \leq n-1$.

Desta forma, podemos reestimar C, A e B, criando $\overline{C}$, $\overline{A}$ e $\overline{B}$, respectivamente, como segue:

$$\overline{c}_i = \text{ número de vezes no estado } S_i \text{ no instante } 1 = \gamma_1(i), \quad 0 \leq i \leq n-1,$$

$$\overline{a}_{ij} = \frac{\text{\# esperado de transições } S_i \to S_j}{\text{\# esperado de transições de } S_i} = \frac{\sum_{t=1}^{\bar{t}-1} \epsilon_t(i,j)}{\sum_{t=1}^{\bar{t}-1} \gamma_t(i)}, \quad 0 \leq i,j \leq n-1,$$

e

$$\overline{b}_{v_k,j} = \frac{\text{\# esperado de vezes em } S_j \text{ e o símbolo } V_k}{\text{\# esperado de vezes em } S_j} = \frac{\sum_{\substack{t=1 \\ o_t = V_k}}^{\bar{t}} \gamma_t(j)}{\sum_{t=1}^{\bar{t}} \gamma_t(j)}, \quad 0 \leq j \leq n-1.$$

Se o modelo atual é $\Lambda = (A,B,C)$ e usarmos o algoritmo EM para computar um novo modelo $\overline{\Lambda}$, então $\Pr[O|\overline{\Lambda}] \geq \Pr[O|\Lambda]$. O resultado final desse procedimento de reestimação é chamado *uma estimativa de maior verossimilhança* (*a maximum likelihood estimate*) do HMM.

3 Nosso Modelo

Nosso problema é encontrar um promotor numa seqüência de DNA procariótico. Lembre-se, da seção 1, que um promotor desse tipo tem 3 componentes principais. O primeiro é a seqüência -35, que tem 6 bases; o segundo consiste de um espaço de tamanho variável; e o terceiro é a seqüência -10, que também tem 6 bases. Ambas seqüências -35 e -10 são conservadas. Como dissemos antes, cada base de nossa seqüência de DNA será um símbolo observado do modelo. Então, no nosso modelo, $m = 4$ e $\mathcal{V} = \{\texttt{A},\texttt{C},\texttt{G},\texttt{T}\}$.

Precisamos de um estado para cada uma das posições das duas seqüências conservadas; um estado para o espaço; e mais dois estados, um para cada lado externo do promotor (-35 mais espaço mais -10). Daí, $S = \{S_0, \ldots, S_{14}\}$, onde cada S_i representa uma das seguintes situações:

S_0: a parte anterior à seqüência -35;
S_i, $1 \leq i \leq 6$: i-ésima posição da seqüência -35;
S_7: o espaço;
S_i, $8 \leq i \leq 13$: $(i-7)$-ésima posição da seqüência -10; e
S_{14}: a parte posterior à seqüência -10.

A figura 1 mostra um diagrama onde cada estado S_i é representado por um vértice com rótulo i, e cada probabilidade de transição é mostrada com uma aresta. Somente as probabilidades não nulas são mostradas. Entrando no estado S_0 e S_1, estão as probabilidades iniciais c_0 and c_1.

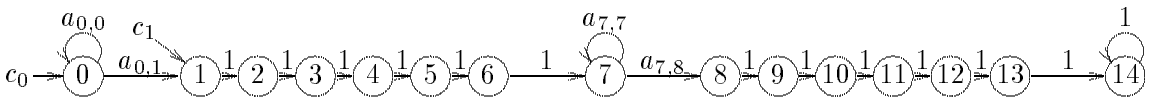

Figura 1: Nosso modelo HMM.

As probabilidades $B = \{b_{v_k,j}\}$, $0 \leq j \leq 14$, $1 \leq k \leq 4$, são mostradas abaixo, onde $b_{v_k,j}$ está na posição (V_k, j) da tabela.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
A	0.25	0.17	0.05	0.15	0.56	0.17	0.43	0.25	0.12	0.73	0.23	0.45	0.37	0.15	0.25
C	0.25	0.08	0.08	0.11	0.16	0.53	0.16	0.25	0.06	0.05	0.14	0.15	0.17	0.04	0.25
G	0.25	0.04	0.03	0.57	0.11	0.10	0.16	0.25	0.09	0.07	0.13	0.15	0.16	0.05	0.25
T	0.25	0.71	0.84	0.17	0.17	0.20	0.25	0.25	0.73	0.15	0.50	0.25	0.30	0.76	0.25

Agora, nós descreveremos como obtivemos os valores iniciais para os parâmetros. Todos os $b_{v_k,j}$, $0 \leq j \leq 14$, $1 \leq k \leq 4$ são baseados nas características encontradas nos 150 promotores de *Escherichia coli* em [4, 12]. Veja a seção 4 para mais detalhes. Os consensos encontrados entre as seqüências mostram uma boa conservação entre elas e são similares aqueles reportados em [6], que são $T_{82}T_{84}G_{78}A_{65}C_{54}A_{45}$ e $T_{80}A_{95}T_{45}A_{60}A_{50}T_{96}$ (onde o índice denota o percentual de ocorrência da base encontrada com mais freqüência), para as seqüências -35 e -10, respectivamente. O genoma de *E.coli* tem praticamente freqüências iguais para as 4 bases [2], assim decidimos atribuir o valor 0.25 a todas as posições externas às seqüências -35 e -10, incluindo o espaço.

O tamanho médio do espaço encontrado entre os 150 promotores analisados foi de aproximadamente 17 (que também é similar aos números mostrados em [6]). Por isso fizemos $a_{7,7} = 16/17$. Também por isso, o tamanho esperado de todo o promotor (incluindo o espaço) foi inicialmente estimado em $6 + 17 + 6 = 29$. Daí, uma boa estimativa para c_1 foi $\frac{1}{t-29}$. Note que $c_0 = 1 - c_1$.

Finalmente, um bom valor inicial para $a_{0,1}$ é dado por

$$\frac{1}{\frac{\bar{t}}{2} - 14.5}.$$

Mais detalhes com respeito a essas escolhas podem ser vistos na seção 4.

É importante notar que os únicos componentes das probabilidades A que necessitam de alguma reestimação são $a_{0,1}$ e $a_{7,8}$ (ou equivalentemente $a_{0,0}$ e $a_{7,7}$).

4 Implementação e Resultados

4.1 Dados para Teste

Com o objetivo de testar o desempenho do nosso modelo, decidimos selecionar ao acaso 180 promotores apresentados em `http://susi.bio.uni-giessen.de/ecdc.html` [4, 12]. Neste site há mais de 500 promotores de *E.coli*.

Dentre esses 180 promotores, escolhemos também ao acaso 150 para construirmos o modelo. As 30 seqüências restantes foram usadas para construir seqüências de teste como segue. Como tinhamos somente os segmentos envolvendo as duas seqüências -35 e -10, mais o espaço dos promotores, nós os procuramos, usando o programa FASTA [7], no sentido de obtermos seqüências reais do genoma da *E.coli* contendo promotores. Daí geramos aleatoriamente um par de inteiros entre 5 e 25 para cada uma das 30 seqüências de teste, para determinar o tamanho dos seus prefixos e sufixos.

4.2 Parâmetros

A partir das 150 seqüências usadas na construção do modelo, obtivemos os valores para as probabilidades B, mostradas na seção 3. As colunas 0, 7 e 14 da tabela da seção 3 tem 0.25 para cada símbolo em $\mathcal{V}$ porque é esta a freqüência desses símbolos no genoma da *E.coli*.

Uma primeira estimativa para $a_{7,8}$ foi baseada na análise dos 150 espaços. Em cerca de 78% dos casos, o espaço tem tamanho entre 16 e 18. A média e a moda são 17, o desvio padrão é menor que 2. Dados estes valores, decidimos fazer $a_{7,8} = 1/17$.

Testamos os valores 1/16 e 1/18 como alternativas para $a_{7,8}$ e, respectivamente, trocando 14.5 para 14 e 15 na fórmula de $a_{0,1}$ acima, na seção 3. Testamos cada combinação destes valores em 20 conjuntos de 30 seqüências (espaços de tamanho aleatório com as mesmas seqüências -35 e -10) e não houve diferenças relevantes entre os resultados.

Pelo fato dos tamanhos dos prefixos e sufixos terem sido gerados aleatoriamente, torna-se difícil estimar um valor inicial para $a_{0,1}$. Por outro lado, como o valor esperado do tamanho total do promotor é 29 e o tamanho total da seqüência é $\bar{t}$, decidimos estimar

$$a_{0,1} = \frac{1}{\frac{\bar{t}}{2} - 14.5}.$$

Em outras palavras, estamos esperando ter um promotor no meio da seqüência.

4.3 Resultados

Considerando que o modelo foi construído levando-se em conta promotores conhecidos e considerando também o fato de que o tamanho da parte externa dos promotores não é conhecido, algumas vezes nosso modelo encontrou o promotor correto antes do último passo do algoritmo EM e depois encontrou um promotor errado, obviamente com probabilidade mais alta, no último passo.

Além disso, nossa implementação é tal que o algoritmo EM é executado enquanto o mesmo promotor encontrado não se repete por mais do que 20 passos, mesmo quando a probabilidade aumenta. Isto é, uma vez que o algoritmo EM não muda o promotor encontrado por 20 passos, ele pára.

Como o algoritmo EM quase sempre converge antes de 10 mudanças (promotores diferentes), e a probabilidade tende a aumentar muito pouco a cada passo, decidimos considerar os últimos 3 promotores encontrados, quando eles obviamente existirem. Em outras palavras, o programa mostra os 3 últimos promotores encontrados, ao invés de uma única solução.

Abaixo, mostramos um conjunto de 30 seqüências de teste. Em letras maiúsculas podemos ver as posições preditas pelo nosso modelo, considerando a melhor solução encontrada (entre as últimas 3), enquanto que os promotores reais estão sublinhados.

1. gtttcTTTTCAcctttcctcctgttTATTCTtattaccccgtg
2. ggcgcaacaagctggTTGCAAaacgattgatggatacggcatgtTGTTGTggcaaggggctgaaca
3. tgacggaaataaaagtaTTGAGAttttgttcttaatcaaTATGTTatttaccgtgacg
4. tattcggcggatacctgacgaATGACGcgggcgatagcgaggatTATCCTgcagcgtc
5. aatatataaaaaagcccTTGCTTtcTAACGTgaaagtggtttaggttaaaagacatc
6. gcaaagtaggacgtaacgatccTTGCCCgtgcggttctggtAAAAAAtacaagcagtgccatggccg
7. gccgtggttcaTTGCAAttgttagcgattgaagAATAATccccacttcgt
8. cctgctaactttgcgtcgATGACCacgagaaTAGATTgtgaccgctttttctaccct
9. tttttcttctttTTGTCTgctatcagcgtagttagccctctggTATGATgagtccaactttgttttgc tgtgtt
10. catactgaaaTTGAATttttttcactcactatttTATTTTtaaaaaacaaca
11. acgtgTTTACGtggcgttttgcttTTATATctgtaatcttaatgccgcgct
12. gatttgcaTTGATTtacgtcatcattgtgaatTAATATgcaaataaagtgagtgaatattctct
13. taaataatcttgttgtgattacttTTGAAAatTAGAGTgagtgcacaacattccgggtgtgtgg
14. gaaatgcgtttctcacTTGCCCgacatatgcgTAAAATgagcggcagat
15. aatcagattcgcttATGACGgcgatgaaGAAATTgcgatgaaatgtgaggtgaatcag
16. taaacatctataaataaagATAACAaTAGAATattaagccaacaaataaactgaaaaagt
17. atatgTTGGCGcggtatcgacgaatttgcTATATTtgcgcccctgacaacagg
18. gattttagacgacgatcgttttcatTTGAAAggaaacgcaaaaagaaaaTAATCTttttgcctctattta ttatttaag
19. aaacgaaatccatgtgtgaagTTGATCacaaattTAAACActggtagggtaaaaaggtcattaactgcc

20. gcgaaacatgaatTTGATAaaacgcgggagTATGATgatat
21. tgagaaaaaccgTTGACGaaggtcgaggcaatccgTAATATtcgcctcgttcccaacggaac
22. gcgaggcctgtgTTTCAAtctttaaatcagTAAACTtcatacgcttgac
23. caggacggagataaagtcttaccgcTTGAGCgcaaagagctccttggccAATTATtactcgacgag
24. ttctgcgcccaacattgttgggaTTGATGggccgTAGATAtcggcgtccagaatgccgactttcgc
25. gctttatTTGCCAtttctccTGAATTacgaaaacatttgcaacactcg
26. taatcaTTTTCAatatcatttaattaacTATAATgaacca
27. atgtccagatgtaTTGACGtccattaacacaatgTTTACTctggtgcctgacatttcaccgac
28. ATGGCAtaaccgaatcgtatgaaacgcgacccaatttcacaaTAAAATgtaaaaa
29. tattaaataTTGACTttttcaccgatgcgtcaagaaaagcggctgAAATTTttacgatcgggt
30. acttttttTTTGCAgtttatggtctattgcaTAGACTgagggggcagcacg

4.4 Análise

Consideramos um segmento (um das seqüências consenso) corretamente predito se ele ocorre numa distância máxima de 6 bases da sua posição real. De acordo com esse critério, obtivemos, após gerarmos 20 conjuntos de 30 seqüências, e após estimarmos todos os parâmetros iniciais como descrito anteriormente, uma média de 78% de identificações corretas. As 30 seqüências acima mostram um caso extremo, onde obtivemos 50 identificações, ou 83.3%.

Muitos dos erros aparecem porque é difícil prever a exata posição do início do promotor na seqüência. Em outras palavras, é difícil estimar $a_{0,1}$. Por outro lado, por termos alguns espaços cujos tamanhos ficaram fora do desvio padrão, é difícil também estimar o valor esperado para $a_{7,8}$. Veja o exemplo

```
tttttcttctttTTGTCTgctatcagcgtagttagccctctggTATGATgagtccaactttgttttgctgtgtt
```

onde o espaço tem tamanho 28. Este tipo de situação mostra quão fraco é o HMM na modelagem da duração do tempo em que o modelo permanece num estado, como reportado em [8, 9].

5 Conclusão

Apresentamos um HMM muito simples mais o uso do algoritmo EM para encontrar um promotor numa seqüência de DNA procariótico e o implementamos. Nosso modelo é comparável com aquele apresentado em [1] no sentido de que ambos os modelos usam o algoritmo EM. A diferença está no fato de que o nosso modelo é muito mais simples, e usa um caso particular do algoritmo de EM, aplicado sobre um HMM bastante simples como proposto em [8, 9].

O desempenho do nosso modelo foi de aproximadamente 78%, em termos do número de segmentos conservados encontrados corretamente. O desempenho de Cardon e Stormo [1] é testado pelo uso de um sistema de contagem completamente diferente. Eles sabem onde

se encontra o ponto exato de início da transcrição. Um promotor é considerado por eles corretamente encontrado se a distância entre o ponto de início da transcrição e a última base da seqüência -10 encontrada está entre 3 e 11. Não sabemos onde se encontra o início da transcrição. Por outro lado sabemos exatamente onde as seqüências -35 e -10 estão posicionadas nas seqüências de teste.

Acreditamos que nosso programa pode ser útil para encontrar, ou pelo menos para dar uma boa pista sobre a posição correta do promotor numa seqüência, especialmente se o usuário estiver certo sobre a existência de um promotor na seqüência de entrada.

O programa está disponível em `http://neper.dcc.unicamp.br:8368/promoter.html`. Lá o usuário pode fornecer como entrada uma seqüência de DNA de até 500 bases e o programa devolverá o mais provável promotor (com mais alto score), de acordo com nosso modelo, para ambas as fitas de DNA. O código pode ser obtido diretamente dos autores.

Agradecimentos

Somos muito gratos a Martin Tompa (CSE - University of Washington), pela ajuda e pelo incentivo no desenvolvimento deste trabalho. Gostaríamos também de agradecer a Lin Tzy Li por tornar o programa disponível na rede.

Referências

[1] L. Cardon and G. Stormo. Expectation maximization algorithm for identifying protein-binding sites with variable lengths from unaligned DNA fragments. *J. Mol. Biol.*, 223:159–170, 1992.

[2] P. Green. Genome sequence analysis. Lecture Notes - University of Washington, October 1996. (web site: http://www.genome.washington.edu/MBT599C/).

[3] J. Henderson, S. Salzberg, and K. Fasman. Finding genes in DNA with a hidden Markov model. *JCB*, 4(2):127–142, 1997.

[4] M. Kroeger and R. Wahl. Compilation of DNA sequences of escherichia coli k12; description of the interactive databases ecd and ecdc (update 1996). *Nucleic Acids Research*, 25:39–42, 1997.

[5] A. Krogh, M. Brown, I. Mian, K. Sjölander, and D. Haussler. Hidden markov models in computational biology. *J. Mol. Biol.*, 235:1501–1531, 1994.

[6] B. Lewin. *Genes VI.* Oxford: Oxford University Press, 1994.

[7] D.J. Lipman and W.R. Pearson. Rapid and sensitive protein similarity search. *Science*, 227:1435–1441, 1985.

[8] L.R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. In *Proceedings of the IEEE*, volume 77, pages 257–286, February 1989.

[9] L.R. Rabiner and B.H. Juang. An introduction to hidden Markov models. *IEEE ASSP Magazine*, pages 4–16, 1986.

[10] J. C. Setubal and J. Meidanis. *Introduction to computational molecular biology.* PWS Publishing Co., 1997.

[11] G. Stormo and G. Hartzell III. Identifying protein-binding sites from unaligned DNA fragments. In *Proc. Natl. Acad. Sci. USA*, volume 86, pages 1183–1187, February 1989.

[12] R. Wahl and M. Kroeger. Ecdc - a totally integrated and interactively usable genetic map of Escherichia coli k12. *Microbiological REsearch (Jena)*, 150:7–61, 1995.