JOURNAL OF COMPUTATIONAL BIOLOGY
Volume 3, Number 2, 1996
Mary Ann Liebert, Inc.
Pp. 223–234

Recognition of Genes in Human DNA Sequences

M.S. GELFAND[1], L.I. PODOLSKY[2], T.V. ASTAKHOVA,[2] and M.A. ROYTBERG[2]

ABSTRACT

A new approach to computer-assisted gene recognition in higher eukaryote DNA is suggested. It allows one to use not only linear functions for scoring structures, but all functions satisfying natural monotonicity conditions. The algorithm constructs the set of structures guaranteed to contain an optimal structure for every function. So, it uncouples the time-consuming step of generation of this set from the fast step of structure scoring, thus making it simple to experiment with different functions. One particular scoring function, taking into account only codon usage and positional nucleotide frequencies of the splicing sites, has been implemented in the Genome Recognition and Exon Assembly Tool program, and has been tested on an independent sample of human genes, yielding 88% sensitivity and 79% specificity.

Key words: exon–intron structure, gene recognition, exons, multicriterial optimization, Pareto set.

1. INTRODUCTION

RECOGNITION OF PROTEIN CODING REGIONS is an important step in computer-assisted analysis of newly sequenced DNA. It is well known that protein-coding regions have statistical properties different from those of noncoding regions. Thus it is possible to consider some function measuring these differences and to evaluate coding potential of open reading frames or sliding windows. This approach has been named "search by content" (Staden 1984a), and several dozen different coding potentials have been suggested up to date (Fickett and Tung, 1992; Gelfand, 1995).

The other possibility is to predict functional sites that serve as boundaries of protein-coding regions, that is, sites of translation initiation and, in the case of higher eukaryotes, splicing sites. Most "search by signal" algorithms are reviewed in Gelfand (1995).

Unfortunately, the specificity and sensitivity of algorithms for prediction of splicing sites are insufficient for reliable prediction of exons. On the other hand, the relative shortness of exons in human genes makes it useless to compute coding potential of open reading frames, whereas the application of sliding window technique is seriously complicated by statistical noise. Moreover, search by content cannot exactly map exon boundaries.

These problems are, to some extent, overcome by the combined approach suggested in Gelfand (1990) and Fields and Soderlund (1990) and further developed in Uberbacher and Mural (1991), Guigo *et al.*

[1]Institute of Protein Research and [2]Institute of Mathematical Problems of Biology, Russian Academy of Sciences, Pushchino, Moscow Region, 142292, Russia.

(1992), Gelfand and Roytberg (1993), Snyder and Stormo (1993, 1995), Milanesi *et al.* (1993), Dong and Searls (1994), Xu *et al.* (1994), Gelfand *et al.* (1995). The basic idea is as follows. First, splicing sites are predicted using a very low recognition threshold so as not to lose any true site. Then all possible combinations of the candidate sites, each corresponding to some exon–intron structure, are considered. Each structure is characterized by scores of the corresponding sites and coding potential of the constituent exons. Some function of these parameters is used to compute the overall structure score.

However, direct implementation of the described approach leads to two major problems. First, in realistic situations the complete search over the set of all possible structures is computationally unfeasible. Second, it is difficult to combine in a reasonable way several numerical parameters of diverse nature. These problems were addressed by empirical techniques based on filtering of candidate exons (Guigo *et al.,* 1992; Milanesi *et al.,* 1993), the theory of formal languages (Dong and Searls, 1994), and combinations of dynamic programming and neural networks (Snyder and Stormo, 1993, 1995; Xu *et al.,* 1994).

A somewhat more general approach has been suggested in Gelfand and Roytberg (1993). It can be applied to a wide class of scoring functions (the only natural requirement is monotonicity on each parameter). The basic idea is as follows. First, we use the so-called "vector dynamic programming" to construct a set of structures guaranteed to contain the best structure for any scoring function from our class. This set is much smaller than the set of all structures. Then the structures within this set are ordered by decrease of some particular scoring function and one or several leading structures are output as the prediction. The main advantage of the method is that we uncouple the time-consuming step of structure generation and the fast step of structure scoring and ordering, thus making it possible to experiment with different scoring functions and parameters and to apply various pattern recognition and neural network techniques.

Here we report the results of prediction for one particular scoring function. Testing of the algorithm on an independent set of human genes produced results comparable to those demonstrated by other programs. We considered also a set of sequence fragments not containing protein-coding regions and demonstrated that the method is sufficiently specific to recognize this situation.

2. METHODS

2.1. Vector dynamic programming

We start with exons, each of which is characterized by m parameters $p_1, \dots, p_m$. These parameters can score splicing sites, coding potential, exon length, etc.

Parameters of a structure (chain of exons) consisting of N exons are determined by component-wise addition, that is, if $p_j(i)$ is the jth parameter of the ith exon ($i = 1, \dots, N, j = 1, \dots, m$), then the jth structure parameter is $q_j = \sum_{i=1}^{N} p_J(i)$. The structure quality is defined as a function $R(q_1, \dots, q_m)$ *monotonically increasing* on each variable. The monotonicity condition is a natural one and all recently applied scoring functions satisfy it, whereas the condition of increasing is a technical one and is introduced purely for clarity of the exposition.

We say that a structure s *dominates* over a structure t (denoted $s \succ t$), if $q_j(s) \geq q_j(t)$ for all j and $q_j(s) > q_j(t)$ for at least one $j, j = 1, \dots, m$. Clearly, if $s \succ t$, then for any quality function $R(s) > R(t)$.

Our aim is to construct the *Pareto-optimal set* of structures P that contains all structures not dominated by any other structure and only such structures. More formally,

- for any $t \notin P$ there exists $s \in P$ such that $s \succ t$;
- if $s, s' \in P$, then neither $s \succ s'$ nor $s' \succ s$.

It is simple to see that for any quality function R the Pareto set P is guaranteed to contain a structure maximizing R. Moreover, for any structure $s \in P$ there exists a quality function reaching maximum on s. Thus the Pareto set contains all necessary structures and no "unnecessary" ones.

To describe the vector dynamic programming algorithm building the Pareto set, we extend the definition of domination to incomplete structures. Denote the length of a structure s by $L(s)$, and denote the donor site position of its last exon by $E(s)$. We say that $s \succ t$ if in addition to the standard conditions the following constraints are satisfied:

```
main begin
  $CurrentStructures := \emptyset$;
  for $x := 1$ to $SequenceLength$ do begin
    if $x$ is at donor site or stop codon then begin
      for $e \in NewExons(x)$ do begin
        for $s \in NewStructures(e)$ do begin
          for $t \in CurrentStructures$ do begin
            if $s \prec t$ then
              go to $DoneStructure$; /* If $s$ is dominated, forget it */
            if $s \succ t$ then
              $CurrentStructures := CurrentStructures \setminus \{t\}$; /* If $t$ is dominated, delete it */
          end;
          $CurrentStructures := CurrentStructures \cup \{s\}$;
        end;
        $DoneStructure$:;
      end;
    end;
  end;
  $P := \{\, t \in CurrentStructures \mid t$ is complete $\}$;
  output $P$;
end.
function $NewStructures(e)$ begin
  /* Add e to all suitable structures from $CurrentStructures$ */
  if left boundary of $e$ is at start codon then
    $NewStructures := \{e\}$;
  if left boundary of $e$ is at acceptor site then begin
    $NewStructures := \emptyset$;
    for $s \in CurrentStructures$ consistent with $e$ do
      $NewStructures := NewStructures \cup \{s \odot e\}$;
  end;
end.
function $NewExons(x)$ begin
  $NewExons := \{$all exons with right boundary $x\}$;
end.
```

FIG. 1. Generation of the Pareto set.

- $L(s) = L(t) \bmod 3$;
- $E(s) \leq E(t)$.

Complete and incomplete structures are incomparable.

The simplest version of the algorithm is shown in Figure 1.

In the beginning the current set of structures *CurrentStructures* is empty. The current position x moves along the sequence. If a donor site or a stop codon is encountered, two steps are done. At the first step we generate all exons ending at x and add each of them to all suitable structures from *CurrentStructures,* obtaining the set of candidate structures *NewStructures.* At the second step we update *CurrentStructures.*

At the first step we consider all exons with the right boundary x. If an exon begins at a start codon, it initializes a new structure. (It should be noted that our use of the term *exon* is slightly different from the standard biological usage, since we consider only translated exons or translated parts of exons.) If an exon begins at an acceptor site (note that a pair of an acceptor site and a donor site can generate up to three candidate exons, since the reading frame is taken into account), it extends suitable structures from *CurrentStructures,* and newly generated structures are placed into *NewStructures.* The requirements are that the exon does not overlap with exons of the structure to be extended, the reading frame is consistent, and restrictions on the minimum intron length are satisfied.

At the second step each structure s from *NewStructures* is compared with all structures from *CurrentStructures* (here we omit technical tricks allowing us to skip some comparisons). If s is dominated by some structure t from *CurrentStructures,* then the former is not considered further. Similarly, if s dominates over an old structure t, the latter is deleted from *CurrentStructures.* Finally, if *CurrentStructures* does not contain any structure dominating s, then s is added to *CurrentStructures.*

At the final stage we extract from *CurrentStructures* all complete structures and obtain the desired Pareto set. Indeed, let s and t be structures and let e be an exon. Denote concatenation of exons by $\odot$. It is simple to see that if $s \succ t$ and both structures $s \odot e$ and $t \odot e$ exist, then $s \odot e \succ t \odot e$. On the other hand, the generalized domination conditions guarantee that for an exon e generated at position x and structures s and t $(s \succ t)$ existence of $t \odot e$ implies existence of $s \odot e$. Thus ignoring dominated incomplete structures does not lead to loss of structures from the Pareto set, and we can sharply decrease the number of considered structures.

As mentioned above, this is the simplest variant of the algorithm. Let us briefly describe the most important of the technical tricks allowing us to decrease the size of *CurrentStructures* and the number of structure comparisons to be performed.

To decrease the number of structures in *CurrentStructures* we can redefine domination for incomplete structures, making it position-dependent. Indeed, if the current position x is so far from both $E(s)$ and $E(t)$ that each exon ending at x can be added either to both s and t, or to none of them, then we need not require $E(s) \leq E(t)$.

Similarily, to decrease the number of structures in *NewStructures,* we can support for each structure $s \in$ *CurrentStructures* some flags allowing us to avoid consideration of $s \odot e$ if it is immediately dominated by $t \odot e$ for some $t \in$ *CurrentStructures*.

2.2 *Implementation*

The algorithm was implemented using the simplest site scoring and coding potential functions.

Splicing sites were scored by the discrimination energy function (Berg and von Hippel, 1987; Gelfand, 1989). Denote the count of nucleotide b at site position k in the learning sample by $N(b,k)(k = 1, \ldots, K)$, and let $N^*(k)$ be the count of the consensus nucleotide, so that $N^*(k) = \max_b N(b,k)$. The score of a site $b_1 \ldots b_K$ is defined as

$$S(b_1 \ldots b_K) = \sum_{k=1}^{K} \log[(N(b_k, k) + 0.5]/[N^*(k) + 0.5)]$$

It is a negative parameter reaching the maximum zero value on the consensus sequence. Denote the average scores of acceptor and donor sites in the site learning sample by μ_A and μ_D, and denote the standard deviations by σ_A and σ_D, respectively.

Coding potential of exons was calculated using the simplest variant of the codon usage analysis related to the one of Staden (1984b). Denote the frequency of the codon abc in the learning sample by $F(abc)$. The codon weight is defined by

$$W(abc) = 100[\log F(abc) - \log F_{\min}]/(\log F_{\max} - \log F_{\min})$$

where $F_{\max}$ and $F_{\min}$ denote the frequencies of the most frequent and the most rare codon in the learning set, respectively.

A fragment $a_1b_1c_1 \ldots a_Kb_Kc_K$ that codes for K amino acids has the coding potential

$$C(a_1b_1c_1 \ldots a_Kb_kc_K) = \sum_{k=1}^{K} W(a_kb_kc_k)$$

It is a positive parameter whose value is large if the fragment consists of preferred codons. It coincides up to a linear transformation with the log-likelihood of the fragment to be generated if the codon probabilities equal the observed codon frequencies. We denote the average codon weight in the learning sample by μ_C, and denote the standard deviation by σ_C.

TABLE 1. RESULTS OF PREDICTION FOR THE TEST SET OF COMPLETE HUMAN GENES[a]

NUM	ID	SeqL	PrL	NE	Rank	Score	L.Sc	L.L	Caa	Ov.	Un.L	NE
1	2	3	4	5	6	7	8	9	10	11	12	13
1	HUMAT1A	1829	359	1	-	-	5.18	330	330	0	29	1
2	HUMCD43	3050	400	1	1	11.54						
3	HUMSPRPC	2651	89	1	4	8.12	8.97	102	89	13	0	2
4	HUMHIAPPA	7160	89	2	>500	-	5.50	48	0	48	89	2
5	HUMREELAS	2309	117	2	4	5.13	5.80	46	26	20	91	2
6	HUMTCRBRA	736	118	2	1	6.56						
7	HUMTRHYAL	9591	1898	2	1	8.75						
8	HUMDEF5A	2800	94	2	1	7.53						
9	HUMELAFIN	1878	117	2	4	5.13	5.80	46	26	20	91	2
10	HUMGOS24B	3135	326	2	2	11.19	11.29	332	326	6	0	2
11	HUMNTRI	3710	94	2	440	6.71	7.74	157	94	63	0	5
12	HUMNTRIII	3710	94	2	101	6.70	7.51	146	94	52	0	5
13	HUMGLPEX	4452	201	2	-	-	10.64	145	145	0	46	2
14	HUMTNP1	1448	55	2	25	3.04	4.49	58	12	46	43	2
15	HUMETMAGA	3343	307	2	463	7.62	10.06	385	307	78	0	4
16	HUMMRP14A	4439	114	2	2	9.74	9.74	110	110	0	4	2
17	HUMCNP	1699	126	2	1	8.93						
18	HUMCRPGA	2480	224	2	1	9.68						
19	HUMCYCAA	3088	105	2	1	8.03						
20	HUMGOS19B	4788	93	3	49	7.89	8.59	271	77	194	16	6
21	HUMCACY	3671	88	2	-	-	7.45	49	46	3	43	2
22	HUMPROT1B	1306	50	2	-	-	4.62	79	48	31	2	3
23	HUMV2R	2282	371	3	4	12.83	12.91	381	362	19	9	4
24	HUMBHSD	9404	375	3	335	10.23	11.67	372	325	47	50	5
25	HUMBNPA	1922	134	3	2	8.40	8.99	191	134	57	0	3
26	HUMTHY1A	2886	160	3	-	-	11.33	287	160	227	0	5
27	HUMPRPH1	4946	196	3	>500	-	10.93	135	123	12	73	3
28	HUMHST	4000	206	3	1	11.06						
29	HUMI309	3709	96	3	60	5.60	5.90	156	25	161	71	6
30	HUMPGAMMG	3771	253	3	1	13.50						
31	HUMPNMTA	3174	282	3	1	13.68						
32	HUMPPPA	2775	95	3	139	8.25	9.14	219	71	148	68	5
33	HUMSAA	3460	122	3	1	6.58						
34	HUMCRYGBC	4500	175	3	3	9.72	9.96	183	175	8	0	4
35	HUMMCHEMP	2776	99	3	1	8.72						
36	HUMTNFBA	2140	205	3	1	9.34						
37	HUMBQ1A	1114	142	3	1	8.86						
38	HUMPF4VLA	1468	104	3	1	9.91						
39	HUMGOS19A	4102	92	3	7	8.06	8.77	167	92	75	0	4
40	HUMMIF	2167	115	3	1	9.11						
41	HUMHSD3BA	8000	372	3	421	9.07	10.69	419	372	47	49	5

continued

TABLE 1. (Continued)

42	HUMCBRG	3326	277	3	5	11.13	11.36	216	211	5	66	6
43	HUMMGPA	7734	103	4	499	3.36	5.05	183	86	97	17	6
44	HUMHAP	3046	318	4	1	9.93						
45	HUMHLL4G	4428	135	4	48	10.40	11.57	167	132	35	3	3
46	HUMHMG2A	4341	209	4	3	9.14	9.39	221	209	12	0	4
47	HUMHMGIY	6000	107	4	-	-	10.78	419	62	357	45	5
48	HUMIBP3	10884	291	4	1	10.85						
49	HUMIL5A	3241	134	4	164	3.30	4.18	127	91	36	73	6
50	HUMUBILP	3583	157	4	186	10.72	12.03	239	157	82	29	3
51	HUMOP18A	6000	148	4	-	-	8.02	150	142	8	6	4
52	HUMLUCT	3296	99	4	1	12.89						
53	HUMLYTOXBB	4800	243	4	-	-	10.87	284	197	87	46	6
54	HUMMK	4638	144	4	-	-	11.72	376	110	266	34	6
55	HUMGAD45A	5378	134	4	308	9.17	10.39	177	126	51	8	4
56	HUMDZA2G	10000	294	4	-	-	13.53	289	289	0	6	5
57	HUMSFTPLA	4732	248	4	64	8.94	9.99	200	170	30	78	3
58	HUMTBGA	5769	415	4	18	10.89	11.45	330	310	20	115	4
59	HUMAPEXN	3730	318	4	1	10.01						
60	HUMIGFBP1A	6128	259	4	-	-	7.49	224	142	82	117	4
61	HUMIL4A	9900	152	4	-	-	9.12	151	120	31	32	4
62	HUMIL8A	5191	99	4	1	6.46						
63	HUMPPCI	6000	406	4	2	16.48	16.53	425	406	19	0	5
64	HUMFABP	5204	132	4	3	3.96	4.22	153	116	37	16	4
65	HUMPALD	7616	147	4	1	7.87						
66	HUMCOX5B	2593	129	4	-	-	7.65	231	129	102	0	4
67	HUMPSAP	4000	248	4	462	9.53	11.19	309	217	92	245	6
68	HUMTNFX	3103	233	4	2	14.82	15.01	217	217	0	16	3
69	HUMIGFBP1	6480	259	4	1	8.59						
70	HUMANT2X	3982	298	4	2	10.97	11.10	424	298	126	0	5
71	HUMIL5	3230	134	4	231	3.27	4.47	146	104	42	30	5
72	HUMCAPG	3734	255	5	17	9.62	10.28	276	255	21	0	6
73	HUMGFP40H	4379	144	5	41	6.71	10.53	162	118	44	26	6
74	HUMTRPY1B	2609	255	5	1	11.86						

continued

Now let $A(s)$ and $D(s)$ be the sums of scores of the acceptor and donor sites, respectively, forming a structure s, and let $C(s)$ be the total coding potential of the exons. Denote the structure length (in codons) by $L(s)$ and the number of exons by $N(s)$. We use the following structure quality function:

$$R(s) = \frac{A(s)/[N(s)-1] - \mu_A}{\sigma_A} + \frac{D(s)/[N(s)-1] - \mu_D}{\sigma_D} + \frac{C(s) - \mu_C L(s)}{\sigma_C L(s)^{1/2}}$$

The first two terms are the average scores of acceptor and donor sites in the structure, measured in the standard deviation units. The last term expresses the coding potential in the standard deviation units; $L(s)^{1/2}$ in the denominator accounts for the fact that $C(s)$ is the sum of $L(s)$ individual codon weights. The function R increases on A, D, and C and decreases on N and L (formally, for compatibility with the previous section, we can say that R is a function of $1/N$ and $1/L$ increasing on all its variables).

After construction of the Pareto set the structures are ordered by decrease of R and several best structures (or just one leader structure) are output as the prediction. In the tested cases the number of structures in P was within the range of low thousands, and thus a simple ordering procedure was sufficient.

We also considered μ_C as an adjustable parameter. We analyzed the learning set using two additional values of μ_C (data are not shown). It turned out that the best recognition was achieved when μ_C was slightly lower than the observed mean. That means that we favor longer structures as compared to the

TABLE 1. (Continued)

75	HUMSPROZ	3011	246	5	11	11.93	12.42	248	199	49	47	4
76	HUMPRS17A	4029	135	5	5	7.07	7.11	153	109	44	26	6
77	HUMREGB	3651	166	5	1	10.26						
78	HUMEMBPA	3608	221	5	-	-	9.84	239	148	91	73	2
79	HUMMIS	3100	560	5	1	15.39						
80	HUMCSPB	3393	247	5	-	-	12.48	290	200	90	47	5
81	HUMPLPSPC	3409	197	5	16	11.21	11.72	327	197	130	0	5
82	HUMOPS	6953	348	5	2	15.80	15.80	411	348	63	0	5
83	HUMFCREB	5131	85	5	>500	-	7.34	27	19	8	66	2
84	HUMKAL2	6139	260	5	-	-	9.20	504	164	340	96	5
85	HUMIL9A	4663	140	5	145	6.07	7.16	162	117	45	23	6
86	HUMNKG5PRO	5000	144	5	-	-	9.98	220	85	135	59	6
87	HUMGHG	4452	246	5	11	11.66	12.42	248	199	49	47	4
88	HUMGHN	2657	217	5	1	12.03						
89	HUMGOS8PP	7345	211	5	4	6.98	7.06	359	211	148	0	6
90	HUMAZCDI	5002	251	5	>500	-	14.85	358	126	232	125	4
91	HUMCHYMASE	4124	247	5	88	8.07	9.00	247	151	96	96	3
92	HUMCHYMB	3279	247	5	17	8.30	8.80	221	151	70	96	3
93	HUMIGERA	7659	239	5	-	-	8.54	317	234	83	5	6
94	HUMCSPA	4791	246	5	19	11.66	12.42	248	199	49	47	4
95	HUMGARE	4754	447	5	15	11.83	12.58	526	447	79	0	4
96	HUMPRS6B	4990	249	6	7	7.03	7.25	267	249	18	0	6
97	HUMTDGF1A	7000	189	6	-	-	7.57	236	75	161	114	6
98	HUMTPALBU	6172	175	6	-	-	11.49	417	124	293	51	6
99	HUMEDHB17	4845	328	6	2	12.82	12.88	329	328	1	0	6
100	HUMMHDOB	5447	273	6	33	10.32	11.36	217	217	0	56	3
101	HUMSAACT	3778	377	6	1	18.86						
102	HUMAK1	8000	194	6	-	-	12.10	242	179	63	15	6
103	HUMIL1B	7824	269	6	2	12.46	12.52	231	219	12	50	6
104	HUMTFPB	13865	295	6	2	9.21	9.28	248	248	0	47	6
105	HUMTROC	4567	161	6	10	14.48	14.77	197	161	36	0	6

[a](1) Fragment number. (2) GenBank ID. (3) Sequence length (in nucleotides). (4) Encoded protein length (in amino acids). (5) Number of translated exons. (6) Rank of the true structure (dash means that the true structure is not contained in the Pareto set). (7) Score of the true structure (dash means that the true structure is not among the 500 top ones or is not contained in the Pareto set). (8) Score of the leader structure (if different from the true structure, same for the remaining columns). (9) Length of the leader structure (in amino acids). (10) Number of amino acids common to the true structure and the leader structure. (11) Overpredition (false amino acids). (12) Underprediction (missed amino acids). (13) Number of exons in the leader structure.

initial random walk form of the coding potential. After the learning stage the value of μ_C providing the best recognition was fixed and this value was used for testing.

The program GREAT (Genome Recognition and Exon Assemby Tool) together with the necessary parameter files and the source code can be obtained from the authors at **misha@imb.imb.ac.ru** (M.G.) or **roytberg@impb.serpukhov.su** (M.R.)

3. RESULTS

The first version of the program GREAT was tested on an independent random sample of 105 completely sequenced human genes (Table 1). The total length of the sequences was 480,243 nucleotides; the average fragment length was 4578 nucleotides. The total length of the coding regions was 69,852 nucleotides (23,284 codons), and thus the coding regions constituted 14.5% of the sample.

TABLE 2. RANKS OF TRUE STRUCTURES IN THE TEST SET OF COMPLETE HUMAN GENES[a]

Rank	1	2	3-5	6-10	11-20	21-100	101-500	>500	-
Number of sequences	28	10	10	3	8	8	13	4	21
Number of sequences at the top of the list	28	38	48	51	59	67	80	84	NA
Percent of sequences at the top of the list	27%	36%	46%	49%	56%	64%	76%	80%	NA

[a]The last column corresponds to structures not contained in the Pareto set.

The quality of prediction can be measured by two kinds of characteristics. First, we can consider the rank of the true structure in the list of all structures ordered by decrease of the scoring function *R* (column 6 in Table 1). This shows how many suboptimal structures we have to retain in order to get the true structure with some given probability. The summary is presented in Table 2. In 28 cases (27%) the true structure has rank 1, that is, coincides with the leader; in 59 cases (56%) it is among the top 20 structures from the Pareto set. However, in 21 cases (20%) the true structure was not in the Pareto set. This usually happened if a structure contained an extremely weak site not found by the site recognition procedure (the threshold had been set so as not to lose more than 1% of true sites from the learning sample), or if there was a relatively strong alternative site situated at a short distance (divisible by 3) from one of the true splicing sites. In the latter case the true structure was dominated by the structure that used this alternative site.

Another characteristic is the size of overlap between the leader and the true structure (columns 10–12 in Table 1). The average results were as follows: 88% of the coding region was found (sensitivity 88%) and 79% of the predicted coding region was really coding (specificity 79%). In four cases the overlap between the true structure and the leader was less than 50% (including one case where there was no overlap between the leader and the true structure), in four additional cases more than half of the true structure was missing, and in eight cases more than half of the leader was not coding. On the other hand, in addition to 28 exact predictions, in 21 cases nothing was lost and in 7 cases nothing was added.

The same set of sequences was submitted to GRAIL-2 file-server (Xu *et al.*, 1994). Results of the testing were as follows. Specificity was 90%, sensitivity was 82%; there were four exact predictions, nothing was added in 37 more cases (including two cases when no coding region was predicted at all), nothing was lost in 18 cases. In one case the overlap between the leader and the true structure was less than 50%, in one additional case more than half of the predicted coding region was not coding, in 13 cases more than half of the coding region was missing (including the two cases of no-coding prediction mentioned above).

We analyzed also whether GREAT could distinguish the situation when a sequence did not contain any coding regions at all. To do that, 39 intronic or intergenic sequences were analyzed (Table 3). It turned out that the scores of leaders in such noncoding situations were substantially lower than in analysis of complete genes (Table 4). In four cases scores of the leader were much higher than those ordinarily observed in noncoding sequences. It turned out that these fragments contained parts of protein-coding genes with descriptions not conforming to the standards of the GenBank feature tables. Consequently, these cases were not included into the tables. On the other hand, it illustrates the sensitivity of the algorithm.

TABLE 3. RESULTS OF PREDICTION FOR THE SET OF SEQUENCES NOT CONTAINING PROTEIN-CODING REGIONS[a]

NUM	ID	SeqL	L.Sc.	L.L	LNE
1	2	3	4	5	6
1	HUMCRYGBC	4000	5.10	142	2
2		4000	6.57	105	3
3		4840	3.55	164	6
4	HUMCSN2A	4803	2.98	65	2
5	HUMFMR1S	4000	4.05	26	1
6		4000	3.40	66	1
7		5961	5.27	106	2
8		6000	3.50	14	1
9		4984	4.04	40	2
10	HUMHP2HPR	2360	4.86	94	5
11		8000	5.56	103	2
12		5684	3.23	15	1
13		4000	4.40	106	2
14		4000	4.30	231	2
15		6921	3.36	105	1
16	HUMODC1A	4769	3.87	297	6
17	HUMPCI	4000	5.50	150	6
18		4000	4.86	175	3
19		2781	5.46	251	4
20	HUMHPRTB	4000	3.14	119	3
21		5000	5.17	49	2
22		3960	6.38	203	6
23		5000	3.48	197	6
24		6000	4.28	88	2
25		4000	4.02	208	5
26		4000	3.41	31	1
27		3040	4.23	120	6
28	HUMRIGBCHA	3975	2.71	10	1
29	HUMTHB	3500	3.79	152	6
30		3500	3.39	24	1
31		4000	5.63	133	5
32	HUMVITDBP	4000	3.25	10	1
33		4000	3.25	17	1
34		5000	3.05	2	2
35		5915	5.00	73	5
36		5234	3.52	40	1
37		5000	4.92	53	2
38		4000	4.16	35	2
39		3992	3.25	153	1

[a] Fragment number. (2) GenBank ID. (3) Sequence length (in nucleotides). (4) Score of the leader structure. (5) Length of the leader structure (in amino acids). (6) Number of exons in the leader structure.

TABLE 4. SCORES OF LEADER STRUCTURES[a]

(1)	2	3	4	5	6	7	8	9	10	11	12	>=13	Total
(2)	0	0	5	6	3	13	12	16	12	15	12	11	105
(3)	0	0	0	0	3	2	5	5	3	3	2	5	28
(4)	0	0	4	1	0	9	6	9	8	9	10	5	61
(5)	0	0	1	5	0	2	1	2	1	3	0	1	16
(6)	2	16	11	8	2	0	0	0	0	0	0	0	39

[a](1) Score (integer part). (2) Number of cases when the predicted leader had this score. (3) Same, only cases with exact prediction. (4) Same, only cases with less than 50% over- or underprediction. (5) Same, only cases with more than 50% over- or underprediction. (6) Same for the sample of noncoding fragments.

4. DISCUSSION

The overall results demonstrated by the program are encouraging. It is possible to diagnose the noncoding situation, and to make good predictions in most cases. The "gray area" of scores, where both coding and noncoding sequences occur, is at the same time responsible for most major errors, and this area is rather narrow. This means that by allowing the "no opinion" outcome we can avoid most errors of both under- and overprediction type.

Performance of the program is at least comparable to that reported for other algorithms (Guigo *et al.*, 1992; Snyder and Stormo, 1993; Xu *et al.*, 1994). In particular, detailed comparison with the most recent of the available versions of GRAIL (Xu *et al.*, 1994) demonstrates that GREAT has lower specificity, but higher sensitivity. GREAT makes more predictions that are exact and rarely misses more than half of the coding region. On the other hand, the GREAT leader more often contains false positives, that is, regions not coding in reality. They either form additional exons, or are adjacent to true exons, that is, have boundaries shifted to the intron. It should be noted here that GREAT uses much fewer parameters than GRAIL.

In testing the approach we deliberately used the simplest splice recognition and coding potential functions. These parameters have clear statistical sense, and it is not difficult to extend the prediction range to other species using simple training procedures or just the published data. On the other hand, the use of more sophisticated procedures can improve performance of the algorithm. In particular, it is possible to take into account intron statistics and to apply coding potentials that account for the statistical inhomogeneity of the human genome. Another possibility is to combine the above approach with similarity searches. It can be done either in the standard way, considering results of similarity searches as one more parameter, or by forcefully causing regions having strong similarity with protein-coding regions to participate in predicted structures.

For technical reasons (insufficient speed and memory of the PC at our disposal) the testing was confined to genes with not more than 6 exons. However, since preliminary analysis shows that application of the program to incomplete genes produces roughly similar results, longer sequences can be analyzed even now after partition into several fragments.

The main computational problem arising at implementation of the vector dynamic programming is keeping the intermediate Pareto set of incomplete structures. The number of these structures still might be exponential, although with a much smaller constant than if the complete search is performed (typically the final Pareto set contains up to several thousand complete structures, whereas the total number of complete structures can exceed millions). The size of the intermediate Pareto set can be decreased by linear transformation of input parameters (Roytberg, 1994), preliminary filtering of exons, or transformation of the main dynamic programming graph (Roytberg *et al.*, 1996b).

One of the major advantages of the suggested approach is uncoupling of the time-consuming step of Pareto set construction from the fast scoring step. Thus, Pareto sets constructed once can be used for experiments with various scoring schemes using suitable modified pattern recognition or neural network techniques.

Finally, we have observed that in many cases the correct exons appear in all or almost all structures from the top of the list. Thus some sort of "consensus of exons" might be used to delineate regions guaranteed to code for proteins. Besides employing purely empirical approaches, it is possible to perform this kind of analysis in a consistent way, making use of the dynamic programming duality between the search for the optimal structure and the search for regions contained in most suboptimal structures (Finkelstein and Roytberg, 1993; Stormo and Haussler, 1994; Roytberg *et al.*, 1996a).

ACKNOWLEDGMENTS

We gratefully acknowledge discussions with Drs. M. Borodovsky, A.V. Finkelstein, P.A. Pevzner, and J.C. Wootton, and help with the data from Dr. E.V. Koonin. This work was supported by Grant MTW300 from International Science Foundation and the Russian Government to M.G., Grants 70-95 from State Program "Human Genome" (Russia) and 94-01-12330 from the Russian Fund of Basic Research to M.R., and in part by Grants HG00783 from NIH and DE-FG02-ER61919 from DOE (USA).

REFERENCES

Berg, O.G., and von Hippel, P.H. 1987. Selection of DNA binding sites by regulatory proteins. I. Statistical-mechanical theory and application to operators and promoters. *J. Mol. Biol.* 193, 723–750.

Dong, S., and Searls, D.B. Gene structure prediction by linguistic methods. *Genomics* 23, 540–551.

Fickett, J.W., and Tung, C.-G. 1992. Assessment of protein coding measures. *Nucl. Acids Res.* 20, 6441–6450.

Fields, C.A., and Soderlund, C.A. 1990. gm: A practical tool for automating DNA sequence analysis. *Comput. Appl. Biosci.* 6, 263–270.

Finkelstein A.V., and Roytberg M.A. 1993. Computation of biopolymers: A general approach to different problems. *BioSystems* 30, 1–19.

Gelfand, M.S. 1989. Statistical analysis of mammalian pre-mRNA splicing sites. *Nucl. Acids Res.* 17, 6369–6382.

Gelfand, M.S. 1990. Computer prediction of exon-intron structure of mammalian pre-mRNA. *Nucl. Acids Res.* 18, 5865–5869.

Gelfand, M.S. 1995. Prediction of function in DNA sequence analysis. *J. Comput. Biol.* 2, 87–117.

Gelfand, M.S., and Roytberg, M.A. 1993. Prediction of the exon-intron structure by a dynamic programming approach. *BioSystems* 30, 173–182.

Gelfand, M.S., Podolsky, L.I., Astakhova, T.V., and Roytberg, M.A. 1995. Prediction of the exon-intron structure and multicriterial optimization, 173–180. *In* Lim, H.A., and Cantor C.R., eds., *Bioinformatics and Genome Research (Proceedings of the 3rd International Conference)*, World Scientific, Singapore.

Guigo R., Knudsen, S., Drake, N., and Smith, T. 1992. Prediction of gene structure. *J. Mol. Biol.* 226, 141–157.

Milanesi, L., Kolchanov, N.A., Rogozin, I.B., Ischenko, I.V., Kel, A.E., Orlov, Y.L., Ponomarenko, M.P., and Vezzoni, P. 1993. GenViewer: A computing tool for protein-coding regions prediction in nucleotide sequences, 573–587. *In* Lim, H.A., Fickett, J.W., Cantor, C.R., and Robbins, R.J., eds., *Proceedings of the 2nd International Conference "Bioinformatics, Supercomputing and Complex Genome Analysis."* World Scientific, Singapore.

Roytberg, M.A. 1994. The method of dynamic programming in problems of molecular biology. Preprint. Institute of Mathematical Problems of Biology, Russian Academy of Sciences (in Russian).

Roytberg, M.A., Astakhova, T.V., and Gelfand, M.S. 1996a. Algorithm for higly specific recognition of protein-coding regions in higher eukaryote DNA sequences. Submitted to *Mol. Biol. (Moscow)*.

Roytberg, M.A., Astakhova, T.V., and Gelfand, M.S. 1996b. Combinatorial approaches to gene recognition. Submitted to *Computers and Chemistry.*

Snyder, E.E., and Stormo, G.D. 1993. Identification of coding regions in genomic DNA sequences: An application of dynamic programming and neural networks. *Nucl. Acids Res.* 21, 607–613.

Snyder, E.E., and Stormo, G.D. 1995. Identification of protein coding regions in genomic DNA. *J. Mol. Biol.* 248, 1–18.

Staden, R. 1984a. Graphic methods to dertermine the function of nucleic acid sequences. *Nucl. Acids Res.* 12, 521–538.

Staden R. 1984b. Measurements of the effect that coding for a protein has on a DNA sequence and their use for finding genes. *Nucl. Acids Res.* 12, 551–567.

Stormo, G.D., and Haussler, D. 1994. Optimally parsing a sequence into different classes based on multiple types of evidence, 369–375. *In* Altman, R., Brutlag, D., Karp, P., Lathrop, R., and Searls, D., eds., *Proceedings of the 2nd International Conference on Intelligent Systems for Molecular Biology.* AAAI Press, Menlo Park, CA.

Uberbacher, E., and Mural, R. 1991. Locating protein coding regions in human DNA sequences by a multiple sensor—neural network approach. *Proc. Natl. Acad. Sci. U.S.A.* 88, 11261–11265.

Xu, J., Einstein, J.R., Mural, R.J., Shah, M., and Uberbacher, E.C. 1994. An improved system for exon recognition and gene modeling in human NA sequences, 376–383. *In* Altman, R., Brutlag, D., Karp, P., Lathrop, R., and Searls, D., eds., *Proceedings of the 2nd International Conference on Intelligent Systems for Molecular Biology.* AAAI Press, Menlo Park, CA.

Address reprint requests to:
M.S. Gelfand
Institute of Protein Research
Russian Academy of Sciences
Pushchino, Moscow Region, 142292, Russia

Received for publication July 7, 1995; accepted as revised January 24, 1996.