Gene-Finding Approaches for Eukaryotes

Gary D. Stormo

Genome Res. 2000 10: 394-397
Access the most recent version at doi:10.1101/gr.10.4.394

Gene-Finding Approaches for Eukaryotes

Gary D. Stormo[1]

Department of Genetics, Washington University School of Medicine, St. Louis, Missouri 63110-8232 USA

The goal of this paper is to introduce the methods commonly used for predicting protein-coding regions in eukaryotic DNA, primarily for the benefit of those not familiar with the topic. This is not meant as a comprehensive review, nor do I describe in detail the underlying mathematical formalism. Those seeking additional information are encouraged to read some recent reviews (Fickett 1996; Claverie 1997; Burge and Karlin 1998; Haussler 1998). Most of the papers in this issue from the recent Genome Annotation Assessment Project (GASP) attempt to identify protein-coding genes using one or more of the methods I describe. I do not assess the success of the different methods, as that is done in the accompanying paper (Reese et al. 2000) and by each paper individually.

There are two important aspects to any program for gene identification: one is the type of information used by the program, and the other is the algorithm that is employed to combine that information into a coherent prediction. Three types of information are used in predicting gene structures: "signals" in the sequence, such as splice sites; "content" statistics, such as codon bias; and similarity to known genes. The first two types have been used since the early days of gene prediction, whereas similarity information has been used routinely only in recent years. One of the reasons that the accuracy of gene-prediction programs have improved in the last few years is the enormous increase in the number of examples of known coding sequences. This much larger sample size allows for more reliable statistical measures to be developed, as well as a much greater likelihood of encountering a gene that is related to one that has been identified previously.

[1]E-MAIL stormo@ural.wust1.edu; FAX (314) 362-7855.

Types of Information Used

Signals

The most important features to identify are the splice junctions—the donor and acceptor sites. If these could be reliably detected from the genomic DNA the difficulty in identifying the coding regions would be greatly reduced because most genes could be recognized simply by finding the long ORFs. It would still be somewhat more difficult than for prokaryotes simply because genes are much less dense in eukaryotes, but a high degree of accuracy could be obtained easily. Unfortunately, splice junctions are not reliably detectable in the genomic sequence. The most common method for predicting them has been the "weight matrix." This is simply a matrix with a score for each possible base at every position within a "site" (Stormo 1990; Gelfand 1995). There are separate weight matrices for acceptor and donor sites, and the scores for each base depend on the frequencies of each base at each position in the known sites. The method of Staden (1984a) simply used the logarithm of the frequency, but it is more common now to use a log-odds ratio between the frequency of each base in the collection of sites and the expected frequency of that base in the genome (Stormo 1990). This gives positive scores to the bases that are preferred in the sites and negative scores to bases that are discriminated against. Using weight matrices to identify donor and acceptor sites is much more reliable than a consensus sequence but still predicts a large excess over the correct sites; that is, there are many false positives for every true positive prediction (Senapathy et al. 1990). More complicated site descriptors have also been tried. For example, one can use a "weight array matrix" that has a score for each dinucleotide and thereby takes into account the nonindependence of adjacent positions in the sites (Zhang and Marr 1993; Salzberg 1997). Recently, Burge and Karlin (1997) used a maximal dependence decomposition (MDD) method for representing splice sites. This takes into account nonadjacent correlations in the splice site positions. In addition, neural networks have been employed to detect splice sites (Brunak et al. 1991; Reese et al. 1997). Neural networks are a pattern recognition technique that takes as input positive and negative examples (i.e., true splice sites and similar sites that are not functional splice sites) and discover the features that distinguish the two sets. The essential distinguishing features may include correlations in the positions of the sites. Each of these more complicated methods improves the detection of splice sites only slightly compared to the standard weight matrix methods. If used with thresholds that are low enough to correctly predict all, or nearly all, true splice sites (i.e., they have a high sensitivity) they still predict many false positives. However, if one can narrow the region in which a splice site is expected to occur, then the accuracy increases significantly. This means that the unreliable splice site prediction methods can be combined with methods for identifying exons based on content measures and increase the exon prediction accuracies significantly (Burge and Karlin 1998).

Other signals can also be useful in predicting exons. (Although technically incorrect, I will use the convention of the gene-finding field for "exon" to mean only the protein-coding portion.) The start and stop codons are essential in predicting the correct gene. Unfortunately they are fairly uninformative without knowing the reading frame. But they are essential in categorizing exons into four classes: single exon genes that begin with a start codon and end with a

10:394–397 ©2000 by Cold Spring Harbor Laboratory Press ISSN 1088-9051/00 $5.00; www.genome.org

stop codon; initial exons that begin with a start codon and end with a donor site; terminal exons that begin with an acceptor site and end with a termination codon; and internal exons that begin with an acceptor site and end with a donor site (see Guigó et al. 1992). Initial and terminal exons tend to be the most difficult to identify, both because the signals are less informative and because they are often much shorter than internal exons and therefore harder to identify by content measures.

Some programs also look for sites associated with promoters, such as TATA boxes, transcription factor (TF) binding sites, and CpG islands (Uberbacher et al. 1996). Although identifying promoters on their own is a difficult problem (Fickett and Hatzigeorgiou 1997), they can sometimes add information that is useful for predicting genes. Poly(A) addition signals are also used sometimes to aid in identifying the proper carboxyl terminus of the gene. In general, the use of these other types of signals provides a marginal improvement over methods that do not use them (Solovyev and Salamov 1997).

Content Measures

Coding regions have statistical properties that can help to distinguish them from noncoding regions. In prokaryotes, simply the length of most coding ORFs is statistically significant. In eukaryotes, the lengths of typical exons are not especially significant, but they have other properties that are useful. For example, every species employs a bias in its choice of codons, such that synonomous codons are not used with the same frequency. So knowing the codon bias for a species can help to identify the genes from the DNA sequence (Staden and MacLachlan 1982). Fickett (1982) also showed that coding regions have asymmetries and periodicities that help to distinguish them from noncoding sequences. Other statistical tests have also been applied to the problem of distinguishing coding from noncoding sequences based on their sequences (for review, see Fickett and Tung 1992). In general, the strengths of these content measures increases with the length of the exons, so that long exons are fairly easy to identify whereas short ones remain difficult even after applying these tests.

Neural networks have also been used to distinguish coding from noncoding sequences. In Brunak et al. (1991) the network was trained to classify whether a particular nucleotide was coding or not based on the surrounding nucleotides, using regions of 100–400 bases. In the `GRAIL` method, a region of sequence, typically ~100 bases, was first analyzed by various statistical tests of coding potential (Uberbacher and Mural 1991). The neural network was then trained, using both coding and noncoding sequences, to find combinations of those statistical tests that had a high accuracy of predicting exons. The success of GRAIL was a significant advance and helped stimulate new research in methods for gene identification.

Similarity Measures

A region of genomic DNA that is significantly similar to a known sequence will usually have the same, or very similar, function. This can be used as both positive and negative evidence about the coding likelihood of the region. For example, if the region matches well to a known repetitive sequence it is unlikely to be protein coding. Some repetitive sequences contain coding regions, but usually we are interested in identifying other genes and would like to ignore the repetitive elements. Programs like `RepeatMasker` (Smit and Green 1999) use a database of known repetitive elements to locate their positions in the genomic DNA, which can then be ignored by the gene-finding program.

If a region of DNA is similar, after translation, to a known protein or protein family, that is strong evidence that the region codes for a protein, and even gives you information about its likely function. This information has been used for a long time to compare the predicted genes with protein databases and provide added confidence for predictions with matches. But it is also possible to include database searches in the initial analysis and use the resulting matches to help guide the prediction process (Snyder and Stormo 1995). One approach uses protein homology exclusively to identify probable coding regions in genomic DNA (Gelfand et al. 1996). EST and cDNA databases can be used similarly. When a region of genomic DNA matches a sequenced cDNA that is very strong evidence that it is transcribed and likely to be part of a coding region. The same can be said of EST databases, although they tend to contain more artifacts that can be misleading. In general, similarities between genomic DNA and sequences that correspond to genes, whether from protein, cDNA, or EST databases, can provide useful evidence for the occurrence of protein coding regions (Xu and Uberbacher 1997). Several of the papers from GASP use similarities to aid the predictions, and they usually show improvements compared to ignoring that information.

Algorithms

Staden (1984b) provided a graphic interface that displayed both signal information, including splice site predictions and start and stop codons, and content measures, such as codon bias. From that, a user could choose exon combinations that appeared likely to code for genes. That remained the state of the art for many years, and while useful, was not the automated system that is required for the current high throughput sequencing projects. Fields and Soderlund (1990) introduced a truly automated gene prediction program, `GeneModeler`, that combined many of the types of information described above. It identified all predicted exons that exceeded some set scoring thresholds and then generated all compatible (i.e., in-frame) combinations into possible gene products. They chose not to rank the various predictions, of which there may be many for any particular DNA sequence. Their program was designed and optimized for *Caenorhabditis elegans* and so did not receive much use on other organisms. `GeneID` used a similar approach on vertebrate sequences (Guigó et al. 1992). It identified all exons, of all four classes, that exceeded some scoring thresholds determined by a mixture of signal and content measures. All of the compatible combinations were determined and ranked by the same scoring

system. In many cases, the correct gene structure was identified near, if not at, the top of the ranked list. Of course, in other cases the correct structure was not even included in the list because the score of one or more of the exons was less than the specified threshold. In general, this was an important step toward fully automated gene prediction. The biggest limitation was that there could be an enormous number of compatible predictions (>200,000, in some cases) even for sequences that are relatively short, <15 kb, by today's standards.

Dynamic Programming

The problem of having too many models to exhaustively enumerate and analyze was solved by the application of dynamic programming methods (Gelfand and Roytberg 1993; Snyder and Stormo 1993). These are recursive optimization techniques that are guaranteed to find the highest scoring prediction without examining all possible ones. Dynamic programming approaches are used on a wide variety of sequence analysis problems, such as sequence alignment, RNA structure prediction, and others (for examples, see Sankoff and Kruskal 1983). In the context of gene prediction it uses the grammatical rules of gene structure (Searls 1992). These are the constraints on the order in which different segments can occur. For example, an initial exon must occur before any introns, an internal exon is bounded on both sides by introns, and a terminal exon is preceded by an intron but not followed by one. Given these constraints and a collection of potential exons and introns, each with an associated score, it is possible to scan across the sequence once and determine the highest scoring predicted gene structure. For example, when considering some particular internal exon candidate, you have to account only for the highest scoring solution that ends with an intron preceding it, and not all possible solutions. So by starting at one end and keeping track of the best solutions ending with each potential exon or intron, the most preferable solution is guaranteed to be found quickly. A modification of the method even allows one to determine ranked suboptimal solutions, or the best solution containing each possible intron or exon (Snyder and Stormo 1995). Most of the previously mentioned gene-finding methods added a dynamic programming step to combine features and determine the optimum predictions based on their own scoring systems (Dong and Searls 1994; Xu et al. 1994; Solovyev et al. 1995; Guigó 1998). While these methods are guaranteed to find the highest scoring predictions, they do not always find the correct predictions. In the earliest adaptation of dynamic programming methods the overall prediction accuracy was not much improved over the previous methods (Burset and Guigó 1996). But because that approach was guaranteed to efficiently find the highest scoring prediction given the scoring system and the information used, researchers could focus their efforts on identifying more useful types of information and improving the scores assigned to them.

Hidden Markov Models

Hidden Markov Models (HMMs) were developed in the field of linguistics research, but they have found applications in many biological problems (Churchill 1989; Krogh et al. 1994a,b; Baldi and Brunak 1998; Durbin et al. 1998). They have three very useful attributes that have contributed to their popularity. First, the models have an intuitive analogy to the things that are being modeled—in this case, gene structures. Second, they have a consistent mathematical formalism that allows for rigorous analysis. Third, over the years, many algorithms have been developed that allow for the efficient determination of many of their properties. Generalized HMMs (GHMMs) extend the basic idea in ways that are especially convenient for gene modeling (Stormo and Haussler 1994; Kulp et al. 1996; Burge and Karlin 1997), but the distinctions are not described here.

The intuitive appeal of HMMs has to do with a natural analogy between the structure of the models and the things being modeled. An HMM has several "states," and in gene prediction these correspond to exons, introns, and any other classes of sequences desired (such as 5′ and 3′ UTRs, promoter regions, intergenic regions, repetitive DNA, etc.). There are probabilities for transitions between the different states that correspond to the allowed changes in state, for example, an intron can only be followed by an internal exon or a terminal exon. The probability of changing from an intron to an exon depends on the local sequence such that it is high only at plausible splice junctions. While the HMM is in any particular state, it "emits" DNA sequence, which is visible. The "hidden" in HMMs denotes the fact that we see only the DNA sequence directly, and the state that generated the sequence (exon, intron, etc) is not visible. But the different states emit DNA with different characteristics. For example, exons emit DNA that must have an ORF, tends to have a certain codon bias, tends to have a certain length distribution, etc. DNA emitted by intron states has different characteristics.

At this point HMMs appear very much like the models for gene structure described above. The main difference, and the main advantage, is that all parameters of the model are probabilities. There are transition probabilities between the states, and emission probabilities from the states. Any "parse" of a sequence (i.e., assignment of its bases to specific states) has an associated probability. The probabilities of any two, or more, parses can be compared directly. Most importantly, there are efficient methods, usually dynamic programming, to perform all of the essential tasks. Given a collection of known correct parses, that is, examples where we know both the genomic DNA and the correct assignment of each nucleotide to its proper class (state), we can find the set of parameters (the probabilities of the model) that maximize the probability of those example sequences. So a "training set" of correct examples is sufficient (but the more examples the better) to find the optimal values of the parameters. Then those parameters are sufficient to determine the optimum (highest probability) parse of any new sequence. A number of other useful features, such as near optimal parses, can also be obtained easily.

GENSCAN (Burge and Karlin 1997) is a very successful example of a GHMM.

Although the general principles it employs are those described here, great care went into a number of details. For example, instead of using weight matrices for splice site detectors, Burge and Karlin (1997) used MDDs, which can account for correlations in the positions. They also tuned the splice site detectors to the local sequence composition. Human DNA is made of isochores with distinct nucleotide compositions; consequently, splice junction sequences vary with the surrounding composition. Also GENSCAN modeled exon length distributions explicitly. And unlike most other programs available at that time, the model allowed multiple genes to occur within the sequence on either or both strands of the DNA. These specific features make it a prototype for the kind of gene finder needed to do whole genome annotation. In this GASP exercise it was used by the annotators to help determine the standard by which the other methods were compared, although additional information and expertise also contributed. However, it should be kept in mind that GENSCAN predictions are not error free, so when other methods predict alternative solutions they might be correct instead, particularly when several other methods agree. One disadvantage of GASP compared to CASP [critical assessment of methods of protein structure prediction (Moult et al. 1999)] is that the true gene structure for the entire region is not known. However, it still serves to point out distinctions, and in some cases weaknesses, of the various approaches and should lead to improved methods. It can serve to stimulate further experimental work to determine the actual collection of genes in that region as well.

Conclusions

Ever since the development of efficient DNA sequencing methods, >20 years ago, there has been a need for programs to automatically identify the proteins encoded in genomic DNA. The last 10 years, and especially the last 5, have seen significant advances, but there is still much room for improvement. The good news is that >90% of the nucleotides can be identified correctly as either coding or noncoding, but the exact boundaries of the exons and their assemblies into complete coding regions are much more difficult to predict. As the papers in this issue demonstrate, even the best methods get less than half of the genes exactly correct. For many purposes, having the exact coding prediction is not necessary, and even partially correct predictions can help focus experiments that can determine the true gene structures faster than would be possible by experimental methods alone. Synergy between computational and experimental methods of gene identification will facilitate the full analysis of the Drosophila genome, and will be essential to extract the full information from the human genome sequence.

ACKNOWLEDGMENTS

G.D.S. is supported by NIH grant HG00249 and DOE grant ER62575. I thank the organizers of GASP, and especially Martin Reese, for encouraging me to write this article.

REFERENCES

Baldi, P. and S. Brunak. 1998. *Bioinformatics: The machine learning approach*. MIT Press, Cambridge, MA.

Brunak, S., J. Engelbrecht, and S. Knudsen. 1991. *J. Mol. Biol.* **220:** 49–65.

Burge, C.B. and S. Karlin. 1997. *J. Mol. Biol.* **268:** 78–94.

Burge, C.B. and S. Karlin. 1998. *Curr. Opin. Struct. Biol.* **8:** 346–354.

Burset, M. and R. Guigó. 1996. *Genomics* **34:** 353–367.

Claverie, J.-M. 1997. *Hum. Mol. Genet.* **6:** 1735–1744.

Churchill, G.A. 1989. *Bull. Math. Biol.* **51:** 79–94.

Dong, S. and D.B. Searls. 1994. *Genomics* **23:** 540–551.

Durbin, R., S. Eddy, A. Krogh, and G. Mitchison. 1998. *Biological sequence analysis: Probabilistic models of proteins and nucleic acids*. Cambridge University Press, Cambridge, UK.

Fickett, J.W. 1982. *Nucleic Acids Res.* **10:** 5303–5318.

———. 1996. *Trends Genet.* **12:** 316–320.

Fickett, J.W. and C.-S. Tung. 1992. *Nucleic Acids Res.* **20:** 6441–6450.

Fickett, J.W. and A. Hatzigeorgiou. 1997. *Genome Res.* **7:** 861–878.

Fields, C.A. and C.A. Soderlund. 1990. *Comput. Applic. Biosci.* **6:** 263–270.

Gelfand, M.S. 1995. *J. Comput. Biol.* **2:** 87–115.

Gelfand, M.S. and M.A. Roytberg. 1993. *BioSystems* **30:** 173–182.

Gelfand, M.S., A. Mironov, and P. Pevzner. 1996. *Proc. Natl. Acad. Sci.* **93:** 9061–9066.

Guigó, R. 1998. *J. Comput. Biol.* **5:** 681–702.

Guigó, R., S. Knudsen, N. Drake, and T. Smith. 1992. *J. Mol. Biol.* **226:** 141–157.

Haussler, D. 1998. *Trends Guide Bioinformatics*, pp. 12–15.

Krogh, A., M. Brown, I.S. Mian, K. Sjolander, and D. Haussler. 1994a. *J. Mol. Biol.* **235:** 1501–1531.

Krogh, A., I.S. Mian, and D. Haussler. 1994b. *Nucleic Acids Res.* **22:** 4768–4778.

Kulp, D., D. Haussler, M.G. Reese, and F.H. Eeckman. 1996. *Proc. Intell. Syst. Mol. Biol.* **4:** 134–142.

Moult, J., T. Hubbard, K. Fidelis, and J.T. Pedersen. 1999. *Proteins* (Suppl.) **3:** 2–6.

Reese, M.G., F.H. Eeckman, D. Kulp, and D. Haussler. 1997. *J. Comput. Biol.* **4:** 311–323.

Reese, M.G., G. Hartzell, N.L. Harris, U. Ohler, and S.E. Lewis. 2000. *Genome Res.* (this issue).

Salzberg, S. 1997. *Comput. Applic. Biosci.* **13:** 365–376.

Sankoff, D. and J.B. Kruskal. 1983. *Time warps, string edits, and macromolecules: The theory and practice of sequence comparison.* Addison-Wesley, Reading, MA.

Searls, D.B. 1992. *Am. Sci.* **80:** 579–591.

Senapathy, P., M.B. Shapiro, and N.L. Harris. 1990. *Methods Enzymol.* **183:** 252–278.

Smit, A.F.A. and P. Green. 1999. http://repeatmasker.genome.washington.edu/.

Snyder, E.E. and G.D. Stormo. 1993. *Nucleic Acids Res.* **21:** 607–613.

Snyder, E.E. and G.D. Stormo. 1995. *J. Mol. Biol.* **248:** 1–18.

Solovyev, V.V. and A.A. Salamov. 1997. *Proc. Intell. Syst. Mol. Biol.* **5:** 294–302.

Solovyev, V.V., A.A. Salamov, and C.B. Lawrence. 1995. *Proc. Intell. Syst. Mol. Biol.* **3:** 367–385.

Staden, R. 1984a. *Nucleic Acids Res.* **12:** 505–519.

———. 1984b. *Nucleic Acids Res.* **12:** 521–538.

Staden, R. and A.D. McLachlan. 1982. *Nucleic Acids Res.* **10:** 141–156.

Stormo, G.D. 1990. *Methods Enzymol.* **183:** 211–221.

Stormo, G.D. and D. Haussler. 1994. *Proc. Intell. Syst. Mol. Biol.* **2:** 369–375.

Uberbacher, E.C. and R.J. Mural. 1991. *Proc. Natl. Acad. Sci.* **88:** 11261–11265.

Uberbacher, E.C., Y. Xu, and R.J. Mural. 1996. *Methods Enzymol.* **266:** 259–281.

Xu, Y. and E.C. Uberbacher. 1997. *J. Comput. Biol.* **4:** 325–338.

Xu, Y., R.J. Mural, and E.C. Uberbacher. 1994. *Comput. Applic. Biosci.* **6:** 613–623.

Zhang, M.Q. and T.G. Marr. 1993. *Comput. Applic. Biosci.* **9:** 499–509.