GENE FINDING IN EUKARYOTIC GENOMES USING EXTERNAL INFORMATION AND MACHINE LEARNING TECHNIQUES

A Thesis
Presented to
The Academic Faculty

by

Paul D. Burns

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Biomedical Engineering

Georgia Institute of Technology
August 2013

GENE FINDING IN EUKARYOTIC GENOMES USING EXTERNAL

INFORMATION AND MACHINE LEARNING TECHNIQUES

Approved by:

Dr. Mark Borodovsky, Advisor
School of Biomedical Engineering
Georgia Institute of Technology

Dr. King Jordan
School of Biology
Georgia Institute of Technology

Dr. Soojin Yi
School of Biology
Georgia Institute of Technology

Dr. Le Song
School of Computer Science and
Engineering
Georgia Institute of Technology

Dr. Brani Vidakovic
Biomedical Engineering
Georgia Institute of Technology

Date Approved: [May 8, 2013]

ACKNOWLEDGEMENTS

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS AND ABBREVIATIONS

DNA	deoxyribonucleic acid
RNA	ribonucleic acid
mRNA	messenger RNA
NGS	next generation sequencing
nt	nucleotide(s)
ORF	open reading frame
CDS	coding sequence
G+C	percentage of G and C nucleotides in a sequence
UTR	untranslated region of a gene
EST	expressed sequence tag
cDNA	copy DNA, formed by reverse transcriptase acting on mRNA
HMM	hidden Markov model
GHMM	generalized hidden Markov model
SJ	splice junction
Tp	no. of true positives
Fn	no. of false negatives
Fp	no. of false positives
Sn	sensitivity: Tp/(Tp+Fn)
Sp	specificity: Tp/(Tp+Fp)
LR	likelihood ratio
Pr	probability
p	probability density function
PSFM	position specific frequency matrix

SUMMARY

Gene finding in eukaryotic genomes is an essential part of a comprehensive approach to modern systems biology. Most methods developed in the past rely on a combination of computational prediction and external information about gene structures from transcript sequences and comparative genomics. In the past, external sequence information consisted of a combination of full-length cDNA and expressed sequence tag (EST) sequences. Much improvement in prediction of genes and gene isoforms is promised by availability of RNA-seq data. However, productive use of RNA-seq for gene prediction has been difficult due to challenges associated with mapping RNA-seq reads which span splice junctions to prevalent splicing noise in the cell. This work addresses this difficulty with the development of methods and implementation of two new pipelines: 1/ a novel pipeline for accurate mapping of RNA-seq reads to compact genomes and 2/ a pipeline for prediction of genes using the RNA-seq spliced alignments in eukaryotic genomes. Machine learning methods are employed in order to overcome errors associated with the process of mapping short RNA-seq reads across introns and using them for determining sequence model parameters for gene prediction. In addition to the development of these new methods, genome annotation work was performed on several plant genome projects.

CHAPTER 1

INTRODUCTION

Efforts to determine the function of genes of all living organisms have been ongoing for many decades. Today the most powerful method for understanding gene function is by comparative genomics, which requires knowledge of the sequence of nucleotides in and around the protein coding portion of a gene. In this general approach, the nucleotide sequence of a gene (or its protein translation) is simply compared with sequences of genes whose function is already known. Therefore, in this one important sense, accurate prediction of gene sequences is critical for gene functions to be fully understood. Understanding of gene function is also supported by sequences in or nearby gene boundaries, such as transcription factor binding sites, and mobile elements.

Eukaryotic protein coding genes are arranged along DNA sequence in intervals of sequence which are transcribed and included in the mature mRNA. Due to existence of introns which are spliced out of the transcript, a eukaryotic gene consists of alternating intervals of sequence which are part of the mRNA (exons) and those which are spliced out (introns). For those genes which code for proteins, a portion of the mRNA will consist of sequence which will be translated into an amino acid chain by ribosomes. This coding sequence (CDS) lies in the interior of the mRNA, such that at the DNA level CDS is distributed across exonic regions. Any exons and introns upstream of the CDS start or downstream of the CDS stop codon are referred as UTR (untranslated region) exons and introns. The CDS start and stop will typically occur somewhere in the interior of an exon. In computational gene finding, the exon-intron structure of this sequence which codes for the protein is attempted to be predicted. This is a problem of major interest in biology for the simple reason that the implied nucleotide sequence arising from such predictions affects the outcome of comparative genomics analysis. Quite often, predicted gene

1

sequences are incorporated into gene or protein databases which are subsequently used for a variety of applications, including finding genes in novel genomes. Therefore, achievement of accurate gene predictions is crucial for downstream scientific work.

External sequence alignment

Today the most reliable method for accurate gene finding is by sequencing full-length cDNA molecules (a copy of the mRNA created by reverse transcriptase) and aligning them to the reference DNA assembly to determine the gene structures on the genome. Such sequences are typically very long (averaging around 2000 nt) and usually contain the complete gene (or at least the protein coding sequence). The protein coding sequence is reliably found by looking for long ORFs in these sequences. However this method is extremely costly for finding all genes in a genome because the procedure is very laborious and expensive. It is worth mentioning that even with all the resources invested in the human genome over a period of about 15 years, there are today about 17,600 unique human genes having full length cDNAs sequenced (mgc.nci.nih.gov). Of course, most genome projects will never enjoy this exquisite concentration of resources. Therefore, alternate methods must be used in general.

Another method for transcriptome sequencing is by expressed sequence tags (ESTs). ESTs are reads taken from cDNA typically from the 5' or 3' end of the molecule. ESTs may also be taken from random locations if random oligonucleotide tags are used in library preparation. The process was developed in the early 1990's as a possible shortcut to finding all genes in the human genome before a final assembly could be created (Boguski, 1995). While initially cDNA libraries were sequenced with Sanger technology, later such libraries would later be sequenced with next generation sequencing (NGS) technology, such as that provided by machines available from Roche 454 Life Sciences and Illumina.

However, even with a combination of NGS and cDNA libraries created using random primers, only a relatively small percentage of genes may have full coverage of the mRNA with EST sequences. There are several reasons for this, such as non-uniform gene expression level (over-representation of highly expressed genes), the cDNA itself not covering a full gene, and the short read length. In addition, budgetary constraints limit the amount of EST sequencing which can be performed. Scientists tend to give priority for sequencing dollars to be spent on improving the reference DNA assembly of novel genomes, rather than EST sequencing. As an example, in the *R. idaeus* genome project, approximately 25,000 ESTs were used to assist gene finding. From this data set, only 334 protein coding genes were judged to be fully covered by EST sequences.

However, even with many related genomes sequenced, identification of the complete set of genes in even model genomes remains incomplete. For instance, even for extensively studied model genomes such as *C. elegans*, *D. melanogaster*, and *A. thaliana*, there remain thousands of genes and missing gene features which are predicted by computational methods (verify). Furthermore, the problem of gene identification in novel genomes is even more highly dependent on computational prediction, due to very limited full-length cDNA sequencing.

Today most transcriptome sequencing is done with a process known as RNA-seq, in which cDNA molecules are sequenced by shotgun sequencing using NGS technology. mRNA molecules are first fragmented into short lengths, then converted to cDNA (or converted to cDNA, and then fragmented) to be compatible with deep sequencing NGS methods (Wang 2009). This form of shotgun sequencing can enable production of short reads covering the whole transcriptome with a fair degree of uniformity, and with very high coverage and low cost. Such advantages promise to revolutionize gene finding and are the principle reasons for the popularity of RNA-seq today.

Nevertheless, there are many widely acknowledged technical challenges to be solved before such data can be very useful for reliable gene finding. Due to the short read

length (typically 50-150 bp) and high volume of reads, alignment to a reference genome assembly is a difficult computational task. Challenges include the presence of paralogous genes, and transcribed repetitive sequence (such as transposable elements), and reads spanning exon-exon splice junctions. Any of these features in a genome has the potential to make genomic mapping of some reads ambiguous. In particular, the presence of splice junctions created during the formation of mRNA requires a gapped alignment of reads to the genome. Optimal gapped alignment (Needleman-Wunch (Durbin, 1998)) of reads is not feasible to be carried out due to the huge volume of reads to be aligned in this fashion. Therefore, a fast but sub-optimal method is required to map RNA-seq to a reference genome. A new pipeline is proposed below, called UnSplicer, which achieves better accuracy for RNA-seq read mapping than all other comparable programs.

Mathematical sequence models

A third major component of gene finding relies on statistical modeling of genomic sequence. It should be mentioned that the process for gene finding in eukaryotic genomes is very different than a process suitable for prokaryotes. In prokaryotic genomes, due to the absence of introns in the genes, an effective method for finding protein coding regions is to simply search for long open reading frames (ORFs), or intervals in which a conspicuous absence of stop codons is observed. Given that in random sequence with 50% G+C content, a stop codon should be observed in approximately 5% of all nucleotide triplets. So observation of an ORF longer than say 200 nt in this sequence already starts to look like coding sequence interval (albeit a short one).

However in eukaryotic genomes, this procedure cannot be followed due to the presence of introns. When full-length cDNA are sequenced, then ORF finding is a viable strategy. This can be done by reconstructing the mRNA sequence by aligning available reads (EST or RNA-seq) to the genome, thereby finding the introns. Once the introns are found, they can be spliced out digitally, revealing the transcript sequence which can be

4

treated in a manner similar to prokaryotic genes. Of course, this ignores complications such as the presence of alternative isoforms, but for now we narrow our objective to finding the longest isoform of each gene.

It was mentioned above that for a typical novel genome sequencing project, only a certain percentage of genes will have transcriptome reads aligned across the full length of the mRNA. The remaining genes must be found by a combination of partial alignment and mathematical prediction. The most important feature discriminating nucleotide sequences which code for proteins and sequences which are non-coding is the nucleotide composition. It was observed long ago that DNA sequence which codes for proteins has a very different composition compared with non-coding sequence. This difference has led to the development of statistical models for coding sequence composition, and appropriate statistical tests for determining presence or absence of coding sequence over an interval (Borodovsky, 1993). An extensively used probabilistic sequence model is the Markov chain, in which conditional probabilities for each of the four nucleotides (A, C, G, T) are known, given every context sequence of length m (immediately upstream of the nucleotide), where m is the model order. Given the arrangement of coding sequence as a series of 3-nucleotide long codons, 3-periodic Markov models were introduced in 1986 (Borodovsky, 1986) and became an accepted and widely used model for coding sequence.

The evolution of gene prediction algorithms over the years as culminated in generative sequence models (particularly the hidden Markov model, or HMM) as the best performing and most widely adopted method for computational gene prediction (Majoros 2007, Lukashin 1998, Salzberg 1999, Besemer 2001, Stanke 2003, Korf 2004, Brejova 2005, Lomsadze 2005, Delcher 2007). Efforts have been made to find alternative models which yield better performance than the HMM (particularly the generalized HMM (GHMM), or HMM with durations) but so far the evidence is inconclusive. New research in non-generative, discriminately trained sequence models shows promise (Bernal 2007,

5

DeCaprio 2007), but it remains unclear if new methods such as conditional random fields (CRFs) benefit from addition of new features (which could also be incorporated into a GHMM structure) or the discriminative training procedure itself.

Challenges with model parameterization

It is widely understood that a critical task in mathematical gene prediction is proper training of the model parameters. Since so many prediction algorithms rely on a similar sequence model (GHMM), the principal performance limitation seems to be the quality of the model parameters. Most gene prediction programs train coding sequence (CDS) and non-coding sequence models using verified gene examples, which are typically found by comparative genomics and EST alignment (Yeh 2001). However, in order to serve as a highly reliable training set for gene prediction, many hundreds of full genes should be available for training (DeCaprio 2007).

A critical development in gene finding for compact genomes occurred with the introduction of GeneMark-ES (Lomsadze 2005). In this algorithm, an unsupervised machine learning approach to GHMM parameter learning is used for gene finding. In this algorithm, models for CDS and non-coding sequence, donor and acceptor splice site position specific frequency matrices (PSFMs), and feature durations (exons, introns, and intergenic regions) could be found without the need for training examples (which typically require an expert to prepare). Perhaps an even more important feature of GeneMark-ES is that its internally developed training set contains a very large number of genes (many thousands) compared with conventional methods. Conventionally, only highly expressed or highly conserved genes can be found with sufficient confidence to be used for training. This creates a problem of parameter bias, because highly expressed and highly conserved genes tend to have different codon usage patterns compared with more lowly expressed genes (Wald 2012, Mathe 1999). Indeed, it is the genes with low

expression in which computational prediction methods must be relied upon to find. This bias in parameter training could be rather widespread, yet underreported because test sets are typically derived from external sequence alignment to highly expressed genes.

CHAPTER II

GENOME COMPLEXITY

Several major challenges in gene finding are due to features of genome complexity associated with eukaryotes. This includes heterogeneity of G+C content in the genome, codon usage variation, and widespread presence of repetitive sequence.

Limitations of the GeneMark-ES algorithm

At this point it is clear that we rely on computational gene prediction, which for best performance requires a good training set to establish parameters for the sequence model. It turns out that codon usage in a gene varies strongly with its G+C content. GHMMs are well suited to prediction of genes in a sequence having a G+C content which is uni-modal and relatively concentrated around the mean (less than 20% variation in G+C or so). Many whole genomic sequences possess this homogeneity in G+C, such as prokaryotic genomes and many compact eukaryotic genomes like as *A. thaliana* and *D. melanogaster*. In such cases, genes can be predicted with good accurately across the whole genome with a single set of parameters. We refer to such genomes as having homogeneous G+C content. However, many genomes are highly heterogeneous in their G+C composition. Mammalian genomes are an example of heterogeneous genomes. Also grasses, such as *O. sativa* and *B. distachyon*, have a somewhat heterogeneous composition. Some of these genomes possess isochores, which are islands of sequence with a relatively homogeneous composition. Genes within isochores (or regions with somewhat homogeneous G+C composition) may be predicted reliably if parameters appropriate for the G+C of the isochore are used. However, not all genes in heterogeneous genomes are contained in regions identifiable as isochores. In general, genomes with heterogeneous G+C require special procedures for gene prediction,

including the development of multiple parameter sets (from multiple training sets) and sequence segmentation into regions of similar G+C composition.
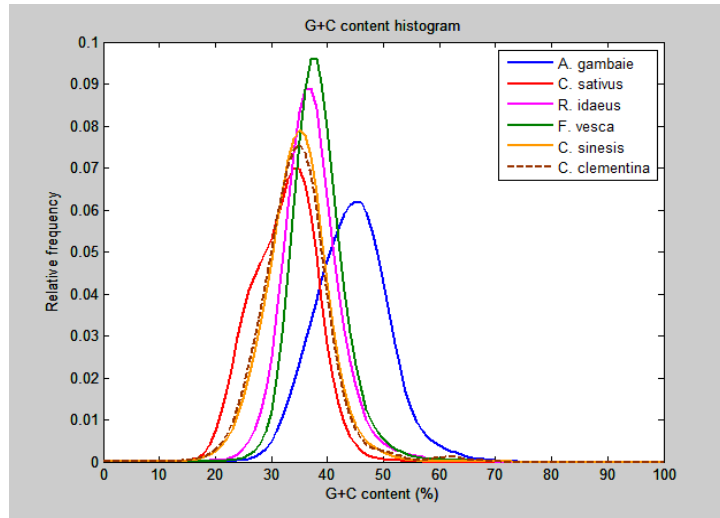
Figure 2.1. G+C content of several homogeneous genomes. Sequences are cut into 2,000 nt segments and %G+C composition is assessed for each segment. Histograms are determined based on the resulting set of G+C values.

It turns out that GeneMark-ES will work successfully on genomes with homogeneous G+C composition, but fails on heterogeneous genomes. The fundamental problem is the algorithm's use of a single set of HMM parameters which does not adequately describe genes in heterogeneous genomes. This limitation restricts the majority of its application to compact eukaryotic genomes. However, even within the realm of homogeneous genomes there remains room for improvement. In Fig. 2.1, G+C content histograms are shown for a number of homogeneous genomes. In Table 2.1, the performance of GeneMark-ES is shown for a test set on each of these genomes. Mean performance is about 85% accuracy (defined here as the average of sensitivity and specificity) for prediction of exon boundaries. For *A. gambiae*, *C. sativus*, and *C. sinensis*, accuracy is closer to 81-82%. Clearly there is room for improvement.

Table 2.1. Exon-level prediction performance of GeneMark-ES on several eukaryotic genomes. Performance was evaluated on test sets derived from EST sequence alignments to the genomes.

genome	exon (Sn+Sp)/2 (%)
F. vesca	85.6
C. sinensis	81.7
C. clementina	86.0
R. idaeus	86.3
A. gambiae	81.5
A. thaliana	91.0
C. sativus	82.0

G+C content and codon usage

In 1999, an evolutionary pattern of codon usage was discovered by analysis of several prokaryotic genomes having different genome G+C composition (Besemer, 1999). In this work, a collection of fully sequenced prokaryotic genomes was used to quantify codon usage characteristics as a function of whole genome G+C content. One motivation for characterizing a universal codon usage was to develop universal Markov models which can be prepared for gene finding in homogeneous sequences with given G+C composition, regardless of how short is the length of the sequence. Codon usage from (Besemer, 1999) was used as the initial starting point models in the unsupervised training program GeneMark-ES.
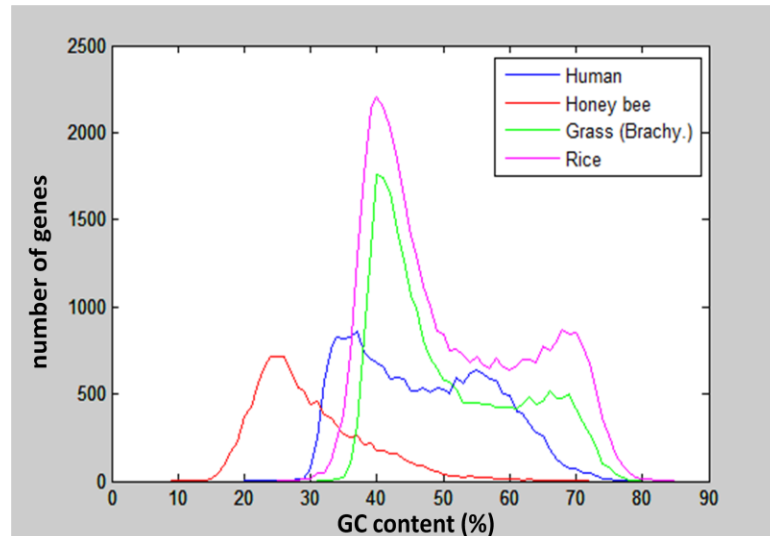
Figure 2.2. G+C compositional distribution of four eukaryotic genomes used to find a shared codon usage pattern. The y-axis represents the number of genes found in a 1% G+C bin.

However, this relationship had not been established for eukaryotic genomes. Therefore, to carry out this task we used a set of 5 eukaryotic genomes: human, rice (*O. sativa*), grass (*B. distachyon*), and honey bee (*A. mellifera*). The distribution of genes per G+C content is shown in Fig. 2 for each genome. Each annotated gene was assigned a G+C value by considering an interval starting at the start codon and ending at the stop codon (including all introns). If more than one isoform was annotated for a gene locus, then only the one with the longest CDS was considered.
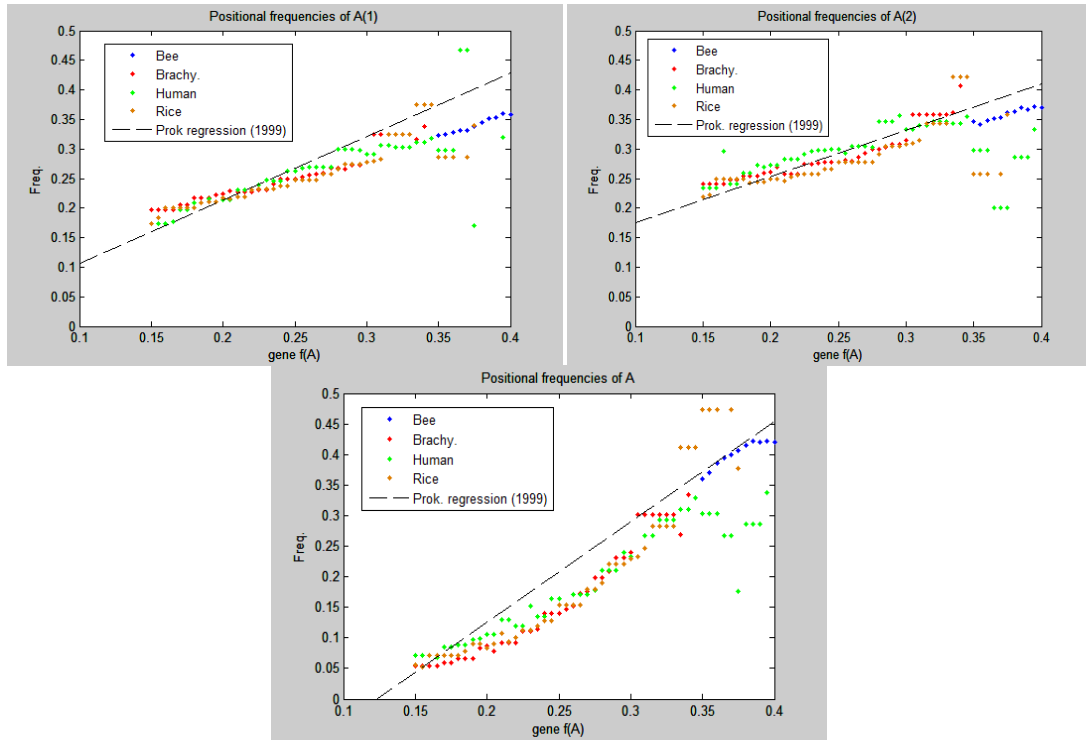
Figure 2.3. Positional frequencies of adenine in eukaryotic gene regions, compared with the result from (Besemer, 1999).
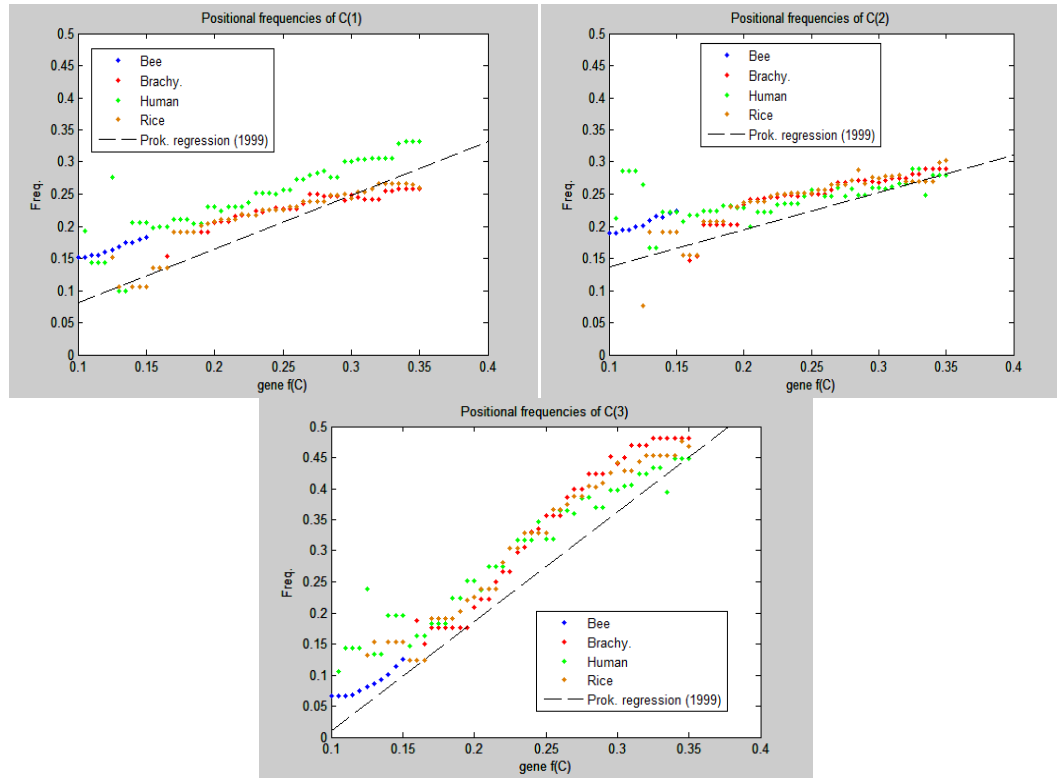
Figure 2.4. Positional frequencies of cytosine in eukaryotic gene regions, compared with the result from (Besemer, 1999).
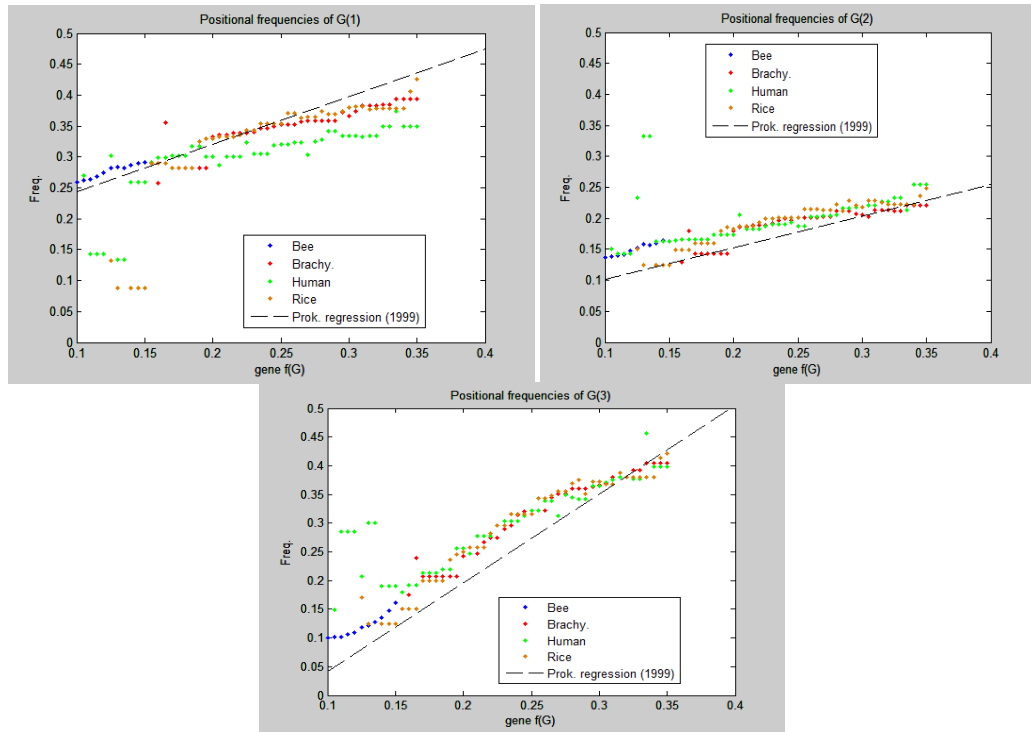
Figure 2.5. Positional frequencies of guanine in eukaryotic gene regions, compared with the result from (Besemer, 1999).
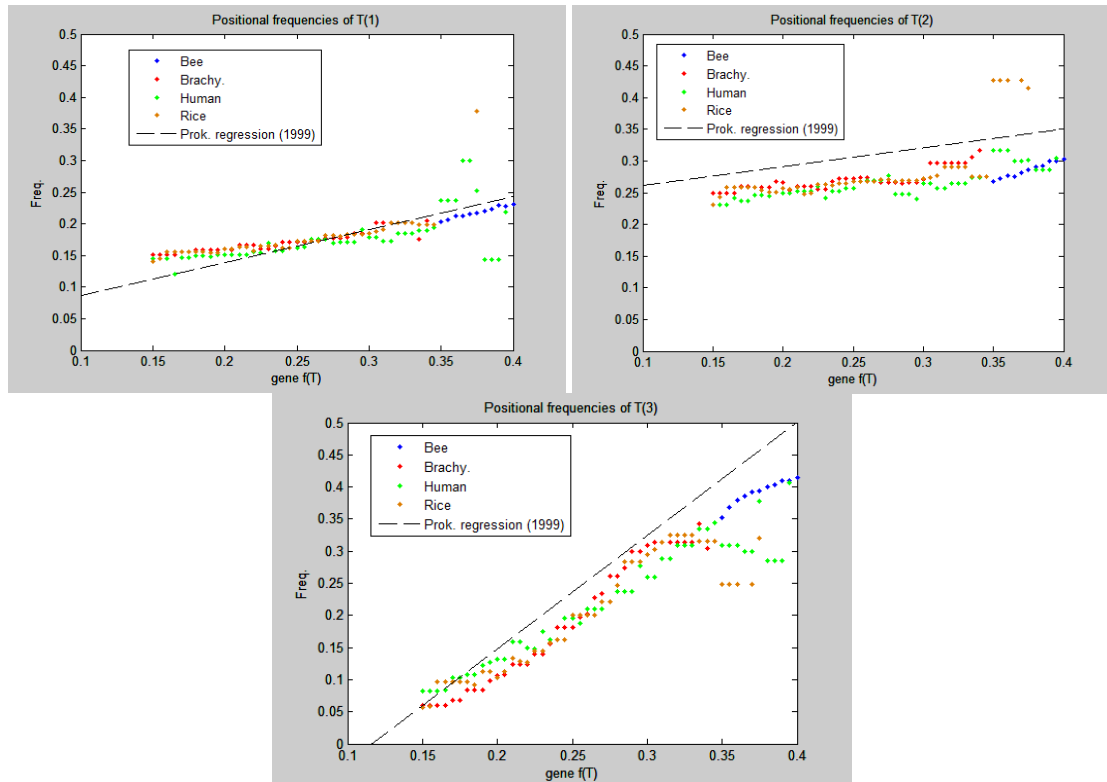
Figure 2.6. Positional frequencies of thymine in eukaryotic gene regions, compared with the result from (Besemer, 1999).

With this set of genes as an input, we compared the positional frequencies of each nucleotide at 3 codon positions, and compared with the findings of the prokaryotic paper (Besemer, 1999). The results are shown in Figs. 2.3-2.6. For each of these plots, the x-axis refers to mean frequency of the nucleotide type in a gene locus, while the y-axis refers to the frequency of the nucleotide type in a specific codon position. We found agreement with the earlier detected general trends in all cases, but with some significant differences. For example, the 1999 linear regression differs in slope with all of the nucleotides in position 1. In position 2, we find a significant difference in the form of a fixed offset. In addition, we observe a wide variation in A(2) in low-G+C genes. Finally, we observe visible nonlinear curvature in frequencies of all nucleotides in the 3rd position.

15

As in the prokaryotic case there are nonlinear relationships between codon usage and G+C composition. A cubic polynomial was fit to the mean value of codon usage frequencies of all 61 codon triplets as functions of G+C content (result shown in Figs. A2-A17). In some cases, such as the arginine codons CGA and CGT, there is a strong difference between native genes in one genome compared with other genomes. In these two codons, we see a strong divergence of honey bee genes from the trend followed by other genomes for these two codons. In addition, for the codon CGA, the *B. distachyon* genes have a significantly different pattern. Regardless, many codons stay close to the general trend, which supports the development of general purpose, pre-calculated sequence models. In this project, 5th order Markov models were developed based on the cubic polynomials fitting each codon usage pattern, in 1% G+C bins in the interval 20-70% G+C. In the construction of the 5th order Markov parameters, sequential codons were assumed to be independent. These parameters are used as the starting point in unsupervised training for version 2.7 of GeneMark-ES. In addition, they have been used in the SeqSweep project (Chapter IV), and the GeneMark-blend+ pipeline (Chapter V).

Repetitive sequence

Despite the fact that many plant genomes have homogeneous G+C composition (see Fig.1), gene finding can be a difficult task due to the prevalence of repetitive sequence. In particular transposable elements (TEs) can occur in genomes with very high copy number, which may severely bias sequence parameters obtained by unsupervised training methods. While only a small number of TEs are actually "alive" in a genome (actively transcribed and translated), most copies in the genome have been inactivated by mutations accumulated over time. In other words, the bulk of copies are pseudogenes. The presence of pseudogenes interferes with eukaryotic gene finding because not only

16

will many of the coding sequences be intact, but they may be present in very large numbers.

Biologists are generally interested in finding native protein coding genes in a genome, as opposed to the proteins associated with TEs. Analysis of the TEs themselves is of interest to a more specialized community. For gene finding, we wish to locate such repetitive elements in order to prevent biasing model parameters in unsupervised training, and also to prevent them from becoming identified as false positive predictions.

To find copies of TEs and other repeats in a DNA sequence, a set of representative TE sequences needs to be established. Subsequently, RepeatMasker (Smit et al http://repeatmasker.org) can be used to identify statistically significant copies of these sequences in the genome. However, a library of query sequences must be established first. To this end, the Repbase (Jurka 2005) database was established to archive a representative set of eukaryotic repetitive sequences. However, the majority of TEs in a genome are dissimilar in sequence to those discovered in genomes of related species. It is thought that TEs originate from RNA retroviruses which lost critical genes (not including the promoter and proteins such as reverse transcriptase). Due to the high mutation rate of viruses in general, it is not surprising that comparative genomics has limited utility in finding TEs in novel genomes.

In order to find native TEs, a number of de novo repeat finder programs have been developed. These include RECON (Bao 2002), PILER (Edgar 2005), RepeatScout (Price 2005), and programs designed to find LTR retrotransposons such as LTR_STRUC (McCarthy 2003) and LTR_FINDER (Xu 2007). These programs employ a variety of strategies and differ greatly in terms of computational resources required to run them. In addition, performance assessment has been quite challenging due to scarcity of high quality annotations, and tremendous diversity in composition of TE families, their copy number, and incomplete representation of TEs in genome assemblies. We have found the

combination of Rebase with RepeatScout to be a good combination of tools for compact

plant genomes.

CHAPTER III

RNA-SEQ: DEEP TRANSCRIPTOME SEQUENCING

The best use of RNA-seq transcriptome sequencing for gene finding remains an unresolved question. This chapter summarizes our progress in this area.

Transcript reconstruction

It was mentioned in Chapter II that transcriptome sequencing in practice has moved to RNA-seq technology. There are several advantages to RNA-seq compared with conventional EST sequencing, including a lower required number of molecules, ability to measure gene expression with high dynamic range, and lower cost (Wang 2009, Trapnell 2010, Roberts 2011). Recent advances in RNA-seq have even enabled single nucleotide resolution of RNA secondary structure (Lucks, 2011). Perhaps the most important feature of RNA-seq is the huge volume of reads which are typically produced. Current generation of Illumina machines (for example, HiSeq 2000) are advertised to produce 3 billion 100 base paired-end reads per run (300 Gb). Use of this technology is becoming widespread, emphasizing the importance of developing efficient algorithms for processing large volume of sequences.

Two well established scientific uses for RNA-seq data are transcript reconstruction and measurement of gene expression. Attention is focused on transcript reconstruction here because it is closely related to gene finding in general.

The two major strategies for transcript and isoform reconstruction from RNA-seq data are: i/ mapping and assembly, and ii/ *de novo* assembly. The former programs align reads to a reference genome as part of the process for transcript construction. Programs which map and assemble include Cufflinks (Trapnell 2010, Roberts 2011) and Scripture (Guttman 2010). Indeed both of these programs use TopHat (Trapnell 2009) to align reads to the genome, then by using a combination of splice junction alignments and paired-end linkage, a transcript graph structure is constructed to allow one or more

19

isoforms to be enumerated. Pure *de novo* transcriptome assemblers (using no reference genome) include ABySS (Simpson 2009, Birol 2009) and Trinity (Grabherr 2011). The *de novo* assemblers are very useful when the reference genome is either unavailable or incomplete. However, despite their advantages the *de novo* pipelines require substantial computational resources. Additional drawbacks to *de novo* assembly include lower sensitivity to genes expressed at a low level, and also susceptibility to creation of false positive "genes" from contaminating sequence in the mRNA library.

Short read alignment

Given that genomic mapping is such an important process in transcript reconstruction (and therefore gene prediction), we focused on this critical step. Due to the presence of exon-exon splice junctions (SJ) in mRNA molecules, sequence reads may be considered to be members of one of two sets: i/ reads which align fully to an exon and ii/ reads which span a splice junction. In general, development of mapping programs has focused on these two kinds of alignments separately.

For mapping reads without a gap, several programs have been developed over the years. However, the winning strategy seems to have emerged: application of the Burrows-Wheeler Transform. This technique has been implemented in software tools such as BWA (Li H 2008, 2009), SOAP (Li R 2009), and Bowtie (Langmead 2009). While full length alignment mapping optimization is by no means a "solved problem," it is arguably solved to a further extent compared to the problem of mapping reads which span splice junctions.

Alignment of reads spanning splice junctions (SJs) is a critical task in transcript reconstruction. Much like full length read alignment, the optimal gapped alignment algorithm (Needleman-Wunch) is impractical due to the large volume of reads. Many programs which carry out various gapped alignment algorithms for RNA-seq reads have been proposed. Software tools of this category include TopHat, MapSplice (Wang 2010),

20

SpliceMap (Au 2010), GSNAP SOAPsplice (Huang 2011), and PASSion (Zhang 2012). TopHat uses an exon inference method using reads mapped without gaps to derive exon (and intron) boundaries. MapSplice uses an "anchor and extend" approach for mapping read segments situated near and over splice junctions. Iterative remapping of reads and read segments in parallel with splice junction inference and filtering of false positives in post processing are the hallmarks of the most recently developed tools. For instance, PASSion and SOAPsplice use paired-end information to eliminate bogus alignments. Despite investing significant efforts in the intense developments, all tools existing to date suffer from a high rate of false positives as compared with tools which align EST sequences such as BLAT (Kent 2001) and GMAP (Wu 2005).

There are two major reasons for the high rate of spurious SJ prediction of RNA-seq mapping programs: 1/ concessions for suboptimal alignment in order to trade accuracy for speed, and 2/ widespread noisy splicing in the cell which is revealed by deep sequencing. Significant efforts have been made to address the major problems with reason 1. As for reason 2, recent work has convincingly demonstrated that most instances of novel SJs, which are observed when mapping high throughput transcriptome sequence data, are manifestations of biochemical noise associated with the splicesosome (Melamud 2009, Pickrell 2010). Indeed, when random models of splicing error are assumed, patterns of experimentally inferred novel introns are closely matched (Melamud 2009). Furthermore, most novel introns show no conservation across species (Pickrell 2010). This suggests that most novel introns found by alignment of RNA-seq are caused by splicing noise. Indeed, even as read lengths achievable by NGS have increased, the number of detected novel isoforms has not dwindled (Trapnell 2010, Marquiz 2012, Daines 2011). This alone does not necessarily imply that there is no biological function associated with such novel transcripts. Rather, we suggest that novel introns should be assessed by the full weight of evidence supporting them—including DNA sequence

patterns. We propose a scoring mechanism for assessing introns based on their similarity to patterns of introns associated with functional gene isoforms.

Using DNA sequence models to improve intron prediction

In 2011, we began investigating the possibility of using probabilistic DNA sequence models determined by unsupervised training (GeneMark-ES) to improve the state-of-the-art in SJ prediction using RNA-seq spliced alignment information. Naturally, this would limit applicability of the new algorithm to genomes with homogeneous G+C composition. However, a large number, a majority, of sequenced genomes satisfy this requirement, so work proceeded on the first algorithm.

SeqSweep

The first concept developed involved filtering SJ predictions made by another pipeline (such as TopHat). A diagram of the pipeline, called SeqSweep, is illustrated in Fig. 3.1. SJ predictions made by an alignment program are filtered for false positives using sequence based models. Fig. 3.2 shows a diagram of a candidate intron, and the four features used to perform discrimination: donor and acceptor splice site models (position specific frequency matrices), intron length, and coding potential on the exon sequences flanking the intron.
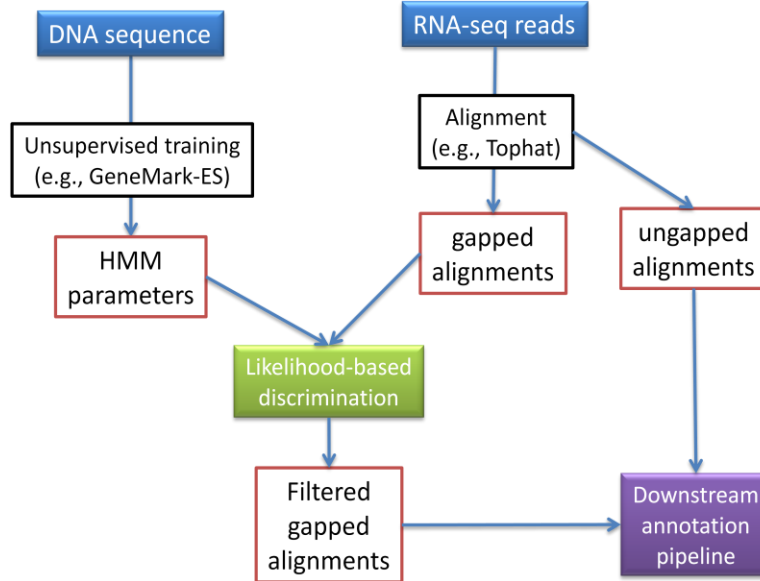
Figure 3.1 Diagram of SeqSweep pipeline. The algorithm in the green block filtered the raw SJ predictions made by the gapped alignment program.
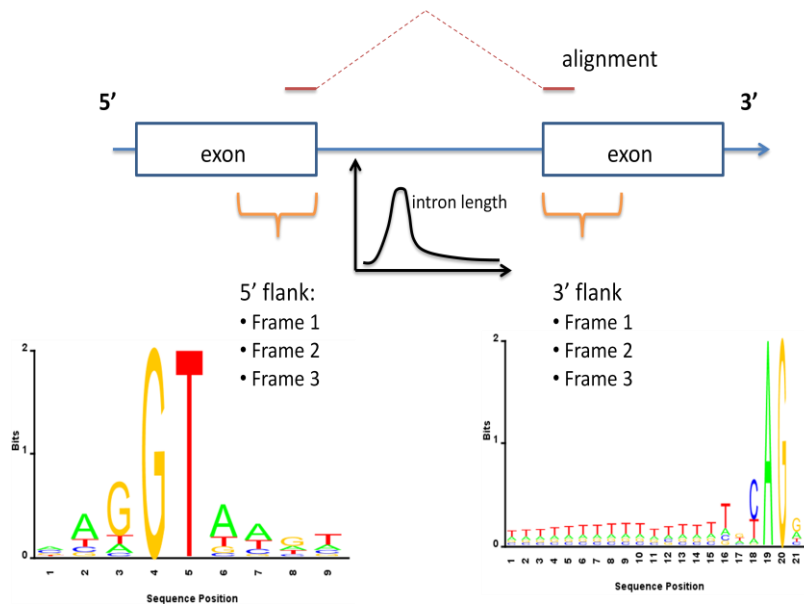
Figure 3.2. Features used by SeqSweep to discriminate true SJs from false positives: donor and acceptor position specific frequency matrices (represented by logos here), intron length distribution, and coding potential found on the intron flanking sequences.

SeqSweep was developed to use all above features, using a naïve statistic for coding potential (no reading frame dependence). The history of coding potential as a feature for SJ classification is a rather complicated one, and it is discussed in detail in

23

Appendix B. In SeqSweep, coding potential was calculated using the eukaryotic heuristic models for CDS (described in Appendix A). A local value of G+C content for each gapped alignment is calculated by averaging the content of sequence 1 Kb upstream of the donor, and 1 Kb downstream of the acceptor. A set of four thresholds (one in each feature dimension) were used so that if any one feature score does not exceed its threshold, the SJ will be rejected. Theshold values were established conservatively to maximize the benefit of this type of classification. Various threshold levels were evaluated by sampling randomly 1,000 values in a parameter hyperbox using limits established based on experience working with spliced alignments of short reads. The discrimination power of SeqSweep was evaluated on an RNA-seq data set SRR100213 aligned to *A. thaliana* (a compact genome) using model parameters found by GeneMark-ES. This data set consisted of 26.3 million single-end reads of length 36 nt. A scatter plot was created in which Sn and Sp are shown for each set of thresholds (one point for each threshold tuplet). In this plot (Fig. 3.3), sensitivity is defined as the number of detected true SJ divided by the total number of annotated SJ in *A. thaliana* (TAIR 10). The TopHat performance without filtering is represented by the most lower-right point on the plot. A substantial improvement in specificity is demonstrated using this method: with an appropriately selected threshold tuple in parameter space, approximately 1,200 false positives may be removed (2% increase in specificity) in this data set with only 130 true positives removed (9.2:1 filtering ratio).
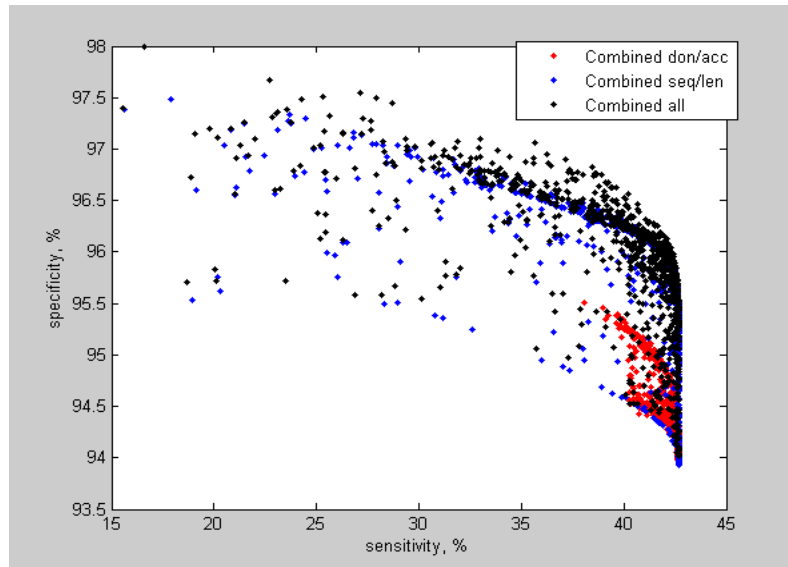
Figure 3.3 Performance of SeqSweep when used to filter false positives from SJ predictd by TopHat. Different colored points correspond to different subsets of features used in filtering. The best performance is obtained by using all features together.

With this promising start, we focused our effort on development of a complete RNA-seq alignment pipeline with collaborators Jian Ma and Yang Li from the University of Illinois Urbana-Champaign. Lessons from SeqSweep were incorporated into two different successful algorithms, described below.

TrueSight

A successful collaboration with scientists at the University of Illinois Urbana-Champaign resulted in the development of TrueSight—the first RNA-seq alignment program to use DNA sequence features to improve SJ prediction accuracy (Li Y 2012). We provide a cursory description of TrueSight because we were not primary authors. The TrueSight pipeline uses Bowtie to map all reads to the genome without gaps. Reads which do not map with Bowtie are mapped using the "anchor and extend" concept used by other programs, such as MapSplice. TrueSight's gapped alignment output includes reads having multiple candidate alignments—ambiguous mappings. All uniquely mapped and ambiguously mapped reads are subsequently classified by an algorithm which

calculates several statistics for each candidate gapped alignment. In the next step, TrueSight collects a number of features associated with each SJ candidate. There are a total of 10 features used for classification. Some features are based on alignment characteristics, such as the number of reads aligned across a SJ, while others are based on DNA sequence models, such as donor and acceptor site Markov model likelihood scores. DNA model parameters are found by unsupervised training by using DNA sequence associated with a subset of SJ candidates as a training set. Classification is performed by a hyperplane boundary in feature space, which is found by expectation maximization on an objective likelihood function. The program is very successful in terms of performance: the publication demonstrates superior performance to several other alignment programs on several data sets.

Many of these features require DNA sequence models (splice site models, and a coding sequence model) which are determined by a subset of initial SJ candidates (those having several read alignments confirming them).

UnSplicer

The work on SeqSweep and TrueSight led to the development of a more effective approach to classification for compact genomes with homogeneous G+C composition. First, for such genomes the proven *de novo* unsupervised training algorithm of GeneMark-ES can be used to find DNA model parameters. Second, the method of selection of training sets for SJ classification in TrueSight could be improved upon. Third, the use of coding potential as a feature could be replaced with an alternative method which does not prevent detection of UTR introns. Finally, the hyperplane boundary itself could be generalized to a non-planar boundary. From these ideas, the concept of UnSplicer emerged.
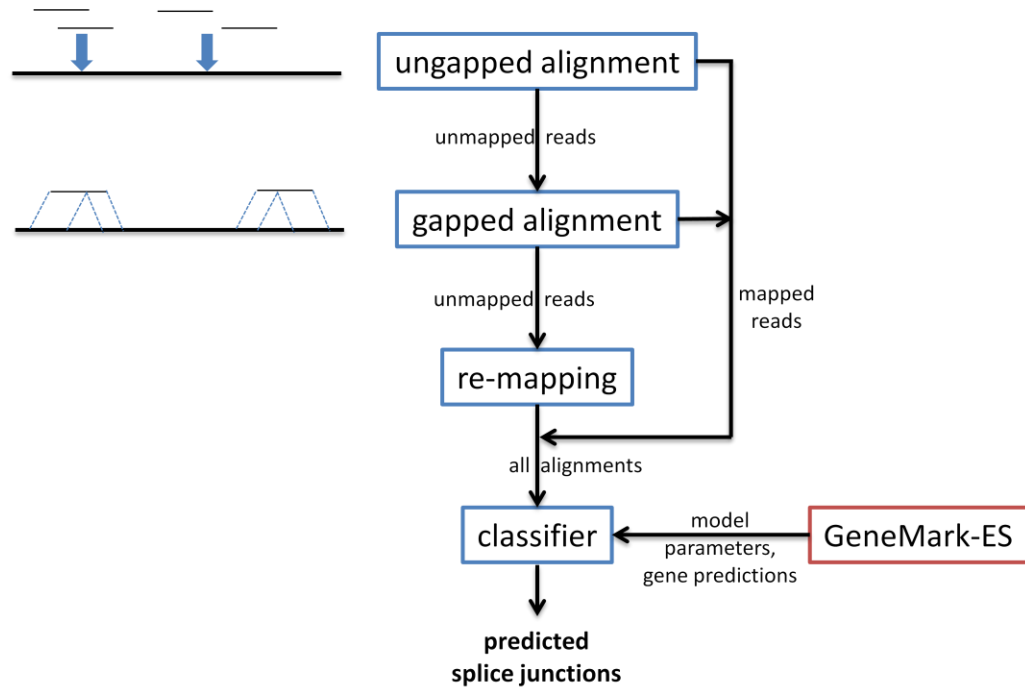
Figure 3.4 The UnSplicer diagram. In the first step RNA-seq reads are attempted to be aligned to the reference genome without gaps (by Bowtie). Second, unmapped reads are attempted to be aligned with gaps by an anchor-extension algorithm (same as in TrueSight). Third, remaining unmapped reads are remapped to pseudotranscripts reconstructed using predicted splice junctions (by Bowtie). Forth, an SVM classifier is used to assign a probability-like score to each splice junction candidate and final predictions are made.

Algorithm description

UnSplicer shares with TrueSight several components such as ungapped (full length) read alignment and "anchor and extend" alignment modules for an initial attempt of a read gapped alignment to genomic sequence. A block diagram of UnSplicer is shown in Fig. 3.4. The ungapped alignment is accomplished using Bowtie (Langmead 2009). Reads unmapped by Bowtie as a whole are divided into short (18-25 nt) non-overlapping segments, with each segment now attempted to be separately aligned by Bowtie. Short segments not aligned by Bowtie are likely to overlap splice junctions. These "missing" segments are attempted to be aligned by a method described in (Li Y 2012), because such segments may overlap splice junctions. An alignment is built on the genome fragment delimited by the alignments of the fragments adjacent to the missing fragment.

27

Alignment starts from the RNA-seq segment borders working into its interior until intron boundaries are delineated.

After finding an initial set of SJ candidates in this fashion, all reads which still remain unmapped to the genome are attempted to be mapped across all splice junctions found from this step. DNA sequence flanking predicted introns are combined to form continuous sequences which serve as targets for alignment to this remaining set of unmapped reads. For details on the remapping methodology, refer to (Burns, 2013).

The second major part of SJ classification in UnSplicer is finding parameters for the sequence models. In fact, sequence parameters are determined prior to running the alignment pipeline. For compact genomes with homogeneous G+C composition, GeneMark-ES predicts intron boundaries with 90% or better sensitivity and specificity (Lomsadze 2005, Ter-Hovhannisyan 2008). GeneMark-ES models donor and acceptor splice sites by hidden states with fixed duration (emitting sequences described by position specific frequency models (PSFMs)) while exons and introns correspond to hidden states with variable duration (emitting sequences described by Markov chains). The donor PSFM spans 3 nt nucleotides upstream of the intron 5' end, and 6 nt downstream. The acceptor PSFM spans 20 nt nucleotide upstream of the intron 3' end, and 1 nt downstream. The PSFM models, considered as non-uniform Markov chains, may be of either zero or first order. Particularly, for all examples discussed below, the donor and acceptor PSFMs are of the first order; all these genomes have at least 5 MB of sequence assembled into long contigs (with N50 larger than 20 kB).
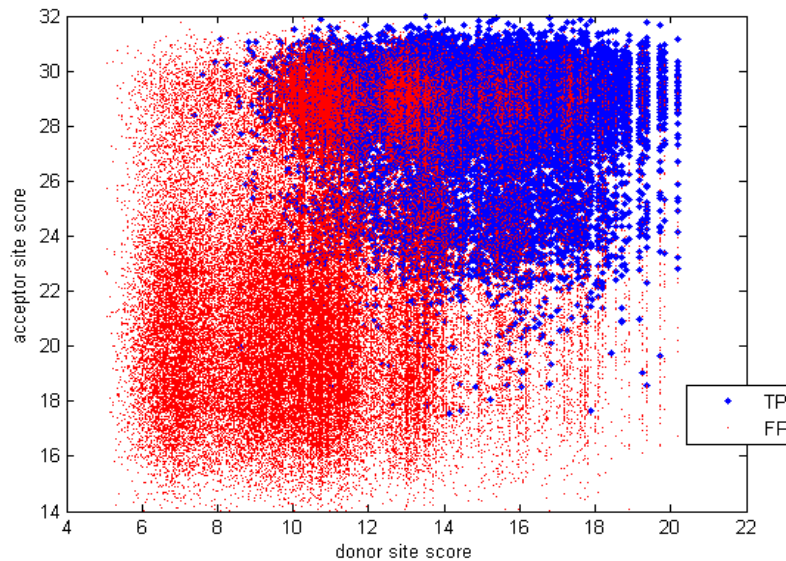
Figure 3.5. Scores of donor and acceptor sites defined by gapped alignments of RNA-seq reads (from SRR360205 set) to genome of *A. thaliana*. The scores are computed with PSFM parameters determined by GeneMark-ES. The blue dots represent splice junctions annotated in TAIR10. Red dots correspond to not annotated splice junctions.

As with SeqSweep, donor and acceptor site PSFMs used by UnSplicer are used to define log likelihood ratio scores of candidate splice junctions. For instance, Fig. 3.5 shows log odds scores of donor and acceptor sites mapped by gapped alignments of RNA-seq reads (SRR360205 set) to genome of *Arabidopsis thaliana*. Thus mapped splice junctions are divided into annotated (shown by blue dots) and not-annotated (shown by red dots) with respect to the TAIR 10 annotation of *A. thaliana* genome. The gene models predicted by GeneMark-ES allow construction of two additional features useful for detection of true of splice junctions by UnSplicer. The first feature is the frame shift indicator taking value one, if a predicted intron is situated inside a coding sequence such that the reading frame shifts upon splicing out the intron. Otherwise, the frame shift indicator takes a value of zero. The second feature is the strand concordance indicator taking value one if the mapped intron appears in the opposite strand of a predicted gene and value zero otherwise. An illustration of these two features for several examples is shown in Fig. C4. Notably, seven of the nine features used by UnSplicer are among the

29

set used by TrueSight. The two new features of UnSplicer are the frame shift and strand concordance indicators. The introns of the *ab initio* gene predictions are not explicitly used by UnSplicer, because prediction accuracy varies from one genome to another. Less variant are coding frame and coding strand, which are much more reliable features for diverse genomes.

Classification of candidate introns is based on construction of a decision boundary in feature space. All nine features are derived from two data sources: read alignments and genomic sequence. Alignment-based features are effectively: i/ gapped (alignment) coverage skew, ii/ gapped alignment depth (the number of alignments confirming a splice junction), iii/ gap (intron) length, iv/ entropy, and v/ read overhang length. While these features were described in detail in (Li Y 2012), there are few minor differences which are described here, in Appendix C.
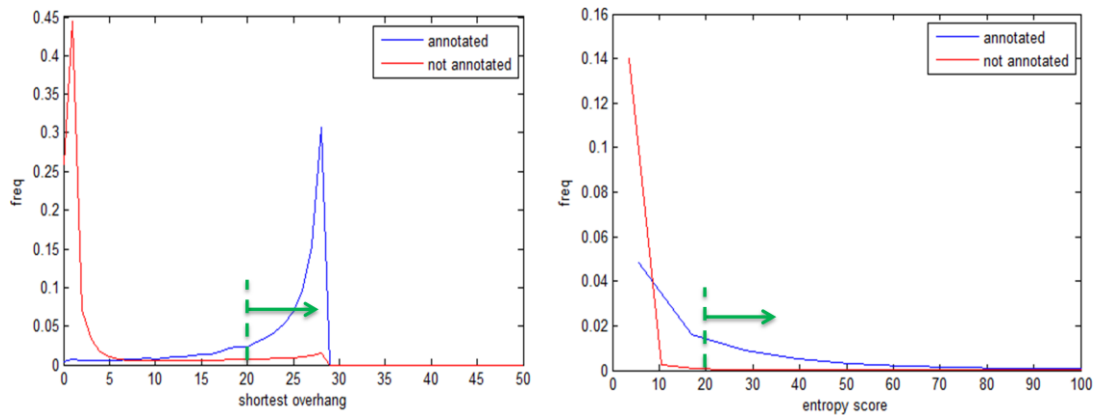
Figure 3**.**6**.** Formation of positive training examples. A set of positive examples was formed from read alignments having both a short overhang longer than 20 nt and an entropy score higher than 20 (as indicated by green arrows). The figures were made for alignments of RNA-seq reads (SRR042297 set) to the *D. melanogaster* genome.

We used a RBF SVM algorithm (Burges 1998) with Gaussian kernel to classify the splice junctions (introns). To determine the SVM parameters and decision boundary, we selected training sets of positive and negative examples. A procedure for selecting the training set was developed with the intention to see positive SJ examples nearly all annotated ("true" SJ) by an external source (e.g. the expert made genome annotation),

and negative examples nearly all not annotated ("false" SJs). The training set consisted of 5000 positive and 5000 negative examples for each genome considered. Positive examples were selected by sampling candidate SJs with high values for shortest overhang (> 20 nt) complemented by high value of entropy (>20). Distributions of these two features for both true and false SJs in *D. melanogaster* are shown in Fig. 34. Out of this set of positive SJs, more than 98% were annotated in Flybase. For reads shorter than 60 nt, the overhang threshold is determined by $0.5 \times (D + 9)$, where D is the half the read length. This heuristic rule is required to maintain a reasonably large number of positive examples in the training set for short reads.
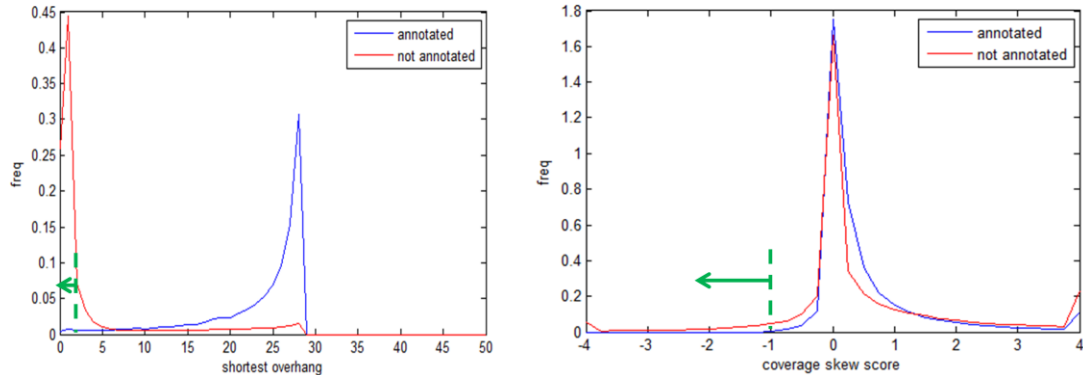
Figure 3.7. Formation of negative training examples. Normalized histograms are shown for shortest overhang values and coverage skew scores for gapped alignments of *A. thaliana* RNA-seq reads (SRR360205 set). The set of candidate splice junctions with the shortest overhang value (< 3) is highly enriched with negative examples. Similarly, enriched with negative examples is the set of candidate splice junctions with coverage skew score less than -1. Splice junctions with scores situated in either of the green regions (shown above) were labeled as negatives.

Negative examples were selected by sampling candidate SJs with either a coverage skew score less than -1 or a shortest overhang length of 2 nt or less. Fig. 35 shows the distribution of shortest overhang length and coverage skew score for true and false junctions obtained by gapped alignment of RNA-seq reads (from SRR360205 set) to the genome of *A. thaliana*. Among this set of negatives, less than 1% were annotated in TAIR 10.

Use of the three features (overhang, entropy, and coverage skew) to define a training set leaves six features for use in the algorithm of classification: donor and acceptor site score, intron length, frame shift indication, strand error indication, and gapped alignment depth. For *A. thaliana* RNA-seq reads from SRR360205 set UnSplicer found 164,373 splice junction candidates with canonical splice sites dinucleotides. As mentioned above we used the LIBSVM package (0) to train an RBF SVM classifier using the training set of 10,000 splice junctions. To find optimal parameters for the Gaussian kernel a grid search was performed for combinations of kernel width (γ) and error cost (c). For each pair of values (γ,c), the SVM was trained and used for prediction on a development set comprised of 10,000 randomly selected splice junction candidates which did not belong to the training set. The label, true or false, for each splice junction in the development set was defined with respect to the *ab initio* gene models made by GeneMark-ES.
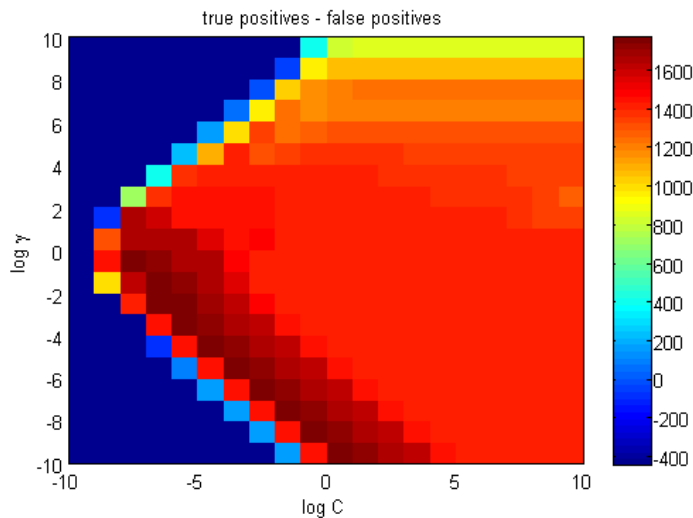
Figure 3.8. A grid search was performed to find the best SVM kernel parameters γ and c, where $\gamma = \frac{1}{\sigma^2}$. The optimal point with respect to Tp-Fp was $(logc, log\gamma)$= (-7,-2).

We searched for the values of γ and c delivering the highest value of classification criterion: the number of true positives minus false positives by with respect to the splice junction labels. Fig. 3.8 visualizes values of the criterion on a (γ,c) grid for the above

mentioned development set of 10,000 splice junctions produced by UnSplicer aligning SRR360205 RNA-seq reads to the *A. thaliana* genome. In this figure, the result of a grid search for *A. thaliana* RNA-seq data is shown, which results in values of $(\log\gamma, \log C)$ to be (-2,-7). It has been shown that good parameters for kernel SVMs tend to lie inside a pocket in the (γ, c) plane (Keerthi 2003). Indeed, this pocket is visible in the lower left plot in Fig. 3.9, where "Tp-Fp" is shown over the parameter plane. The sweet spot runs along a diagonal ridge (colored deep red in the figure). Indeed, as we consider the performance measure Tp-Fp over four data sets, we see a similar pattern (Fig. 3.10). Congruent regions across all four data sets in the (γ, c) parameter plane result in under-fitting (test junctions are labeled the same way), or over-fitting (all test points are labeled as false unless they happen to lie very close to a positive training sample). A pocket of good performance consistently lies on a ridge starting from roughly $(\log\gamma, \log C) = (0, -7)$ and extends from there with a slope of -1 in the log plane as shown in the figure. It was not surprising to observe this feature, as this is reported to be a general property of RBF SVMs (Keerthi 2003). Indeed, because of this consistency, the grid search is restricted to a region in the vicinity of this ridge in order to reduce the computational cost of the search. Following the ridge along its length, with c increasing and γ decreasing, the decision boundary increasingly approximates a hyperplane. We observe most clearly in *C. elegans* that the best performance is obtained by a boundary very different from a hyperplane—specifically in a region near $(\log\gamma, \log C) = (0, -6)$. This demonstrates that the more flexible RBF SVM notably improves performance compared with a hyperplane boundary.
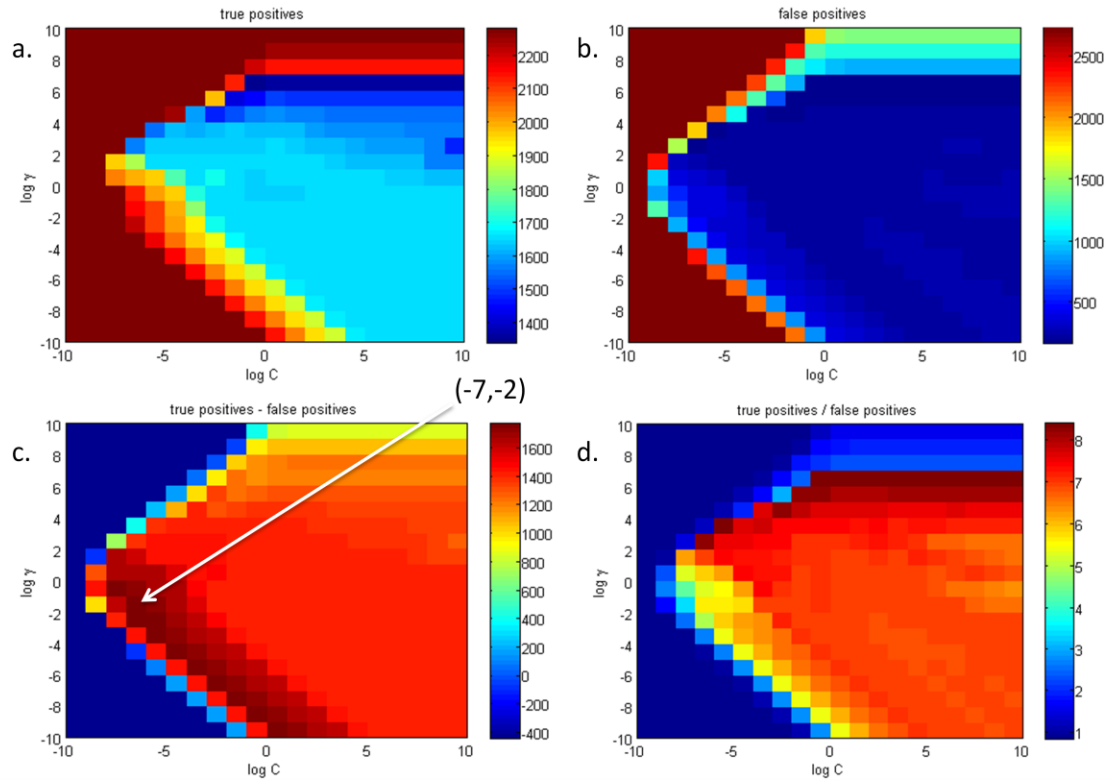
Figure 3.9. Performance of kernels with various values of (γ, c) for a test set derived from SRR360205 (*A. thaliana*). "True" and "false" labels were derived from the *ab initio* gene predictions made by GeneMark-ES. In this example, the grid search found the best pair at log(c) = -7, and log(γ) = -2.
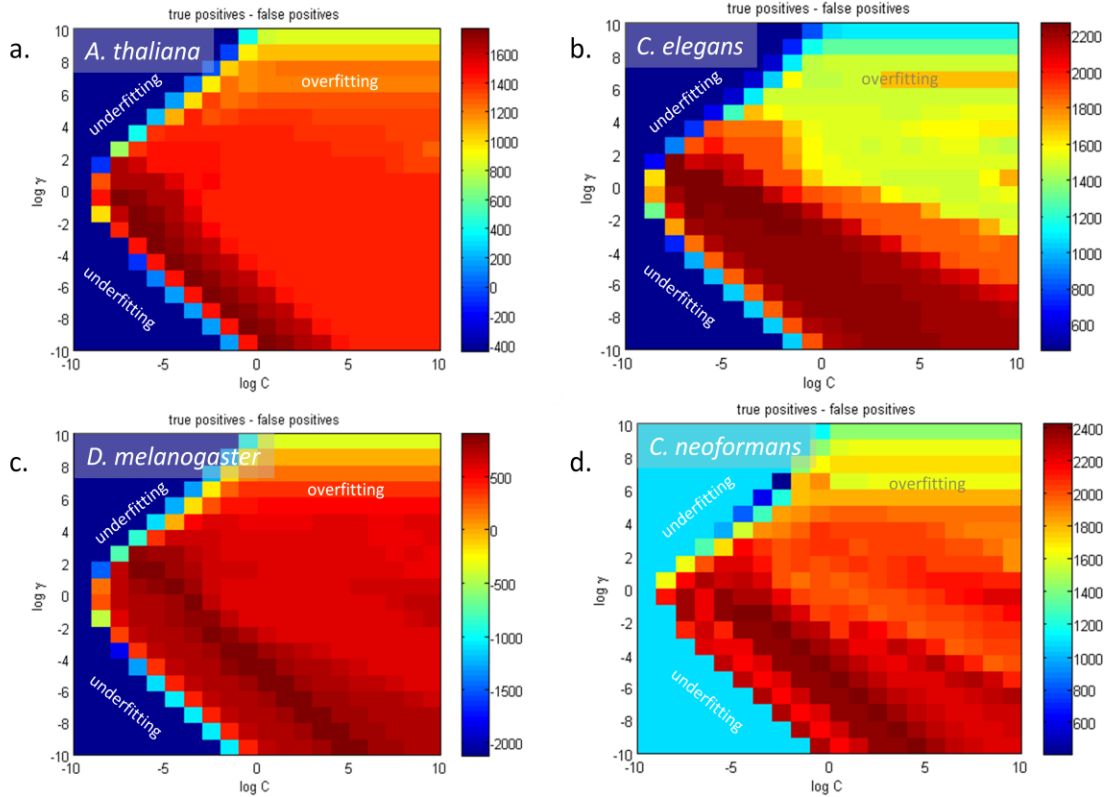
Figure 3.10. We observe that for all examined RNA-seq data sets, regions similarly situated in panels *a,b,c,d* in the (γ, c) parameter plane result in either under-fitting or over-fitting the training data. It is also observed that a similarly situated ridge (with a pocket of highest values of the criterion function) produces the best overall performance in each case.

All training is performed on canonical junctions. After the parameters (γ, c) are found by the grid search, all non-labeled canonical splice junctions are predicted by the SVM. After training and prediction of canonical junctions, prediction of non-canonical junctions is performed using the same decision boundary used for canonical junctions

After the kernel parameters are found, the SVM classification was made for all N (N = 164,373 for data set SRR360205) candidate splice junctions. A probability score $p_i, i = 1, \dots N$ was assigned to each predicted splice junction using the method described in (0). A sigmoid function was used to define the probability score, such that for each input feature vector x

$$P(x \text{ assigned to } k) = \frac{c_k}{1 + e^{A\hat{f}(x)+B}}$$

35

where k $\in$ {0,1} represents class (1 is a "true" splice junction label, and 0 is "false"), $\hat{f}(x)$ is the decision value at point x (+1 or -1, then averaged over a 5-fold cross validation), A, B are constants found by maximizing the likelihood of training data, and c_k is a normalizing constant chosen so that P(x assigned to 0) + P(x assigned to 1) = 1.

Each candidate splice junction was ranked with the probability score. While the default classification threshold was set as 0.5, a full receiver operating curve (ROC) could be defined by variation of the detection threshold over [0,1]. Any point of the ROC could be chosen as an operating point of classification by the user.

Results

We begin with a demonstration of the prevalence of noisy splicing. The ribosomal protein RpS4 in *D. melanogaster* is a highly expressed gene with two isoforms and five annotated introns. Alignment of EST sequences (downloaded from NCBI) using the program BLAT to the RpS4 gene locus results in a set of 33 spliced alignments shown in Table 3.1. The five annotated introns are indicated with a '**' label on the left, and the number to the left of the FlyBase record name (FBgn0011284) is the number of EST alignments across that intron. The 28 spurious introns found by EST alignment are most likely due to a combination of mapping error and splicing noise because their boundaries are dispersed but generally nearby annotated introns. If any of these 28 candidate novel introns are to be predicted, some additional supporting evidence would be required. This is the motivation for using DNA sequence features.

Table 3.1. A list of introns found by spliced alignment of *D. melanogaster* EST sequences to the RpS4 gene locus (FBgn0011284). Columns: 1 (number of alignments), 2 (locus name), 3-4 (intron boundaries inferred by gapped alignment). The introns in the FlyBase annotation are indicated with a ** on the left.

```
**   525  FBgn0011284          83              226
       1  FBgn0011284          84              227
       2  FBgn0011284         258              353
       1  FBgn0011284         259              354
**   628  FBgn0011284         260              354
       1  FBgn0011284         261              356
       1  FBgn0011284         261              357
       2  FBgn0011284         261              355
       1  FBgn0011284         262              356
       1  FBgn0011284         384              405
       1  FBgn0011284         545              972
       1  FBgn0011284         558              569
       1  FBgn0011284         602              685
       1  FBgn0011284         610              645
       1  FBgn0011284         611              694
       1  FBgn0011284         614             1454
       1  FBgn0011284         614              718
**   630  FBgn0011284         614              686
       1  FBgn0011284         650              685
       1  FBgn0011284         781             1408
       1  FBgn0011284         784             1409
**   566  FBgn0011284         785             1410
       1  FBgn0011284        1470             1817
       1  FBgn0011284        1543             1699
       1  FBgn0011284        1580             1817
       4  FBgn0011284        1582             1818
       1  FBgn0011284        1583             1940
**   310  FBgn0011284        1583             1818
      11  FBgn0011284        1584             1821
       2  FBgn0011284        1584             1820
       1  FBgn0011284        1585             1822
       1  FBgn0011284        1587             1822
       1  FBgn0011284        1838             1882
```

The UnSplicer was compared with four other RNA-seq alignment programs: TrueSight v0.06, TopHat v2.0.8, PASSion v1.2.0, and SOAPsplice v1.9. Due to a large number of false positives observed in the runs of MapSplice (Li 2012, Zhang 2012) and GSNAP (Wu 2010), it was not practical to include these results in the comparison. Two different simulation experiments were performed in order to compare the programs, in addition to several real data sets

The five RNA-seq mapping programs were evaluated by mapping simulated reads. Maq (Li 2008) was used (with sequencing error parameter r = 0.02) to simulate reads derived from 41,671 *A. thaliana* full-length cDNA sequences. Three different simulated paired-end read sets were created, each consisting of 5 million paired-end sequences with length 50 nt, 75 nt, and 100 nt. Reads were generated from transcripts according to expression levels found by alignment of a real data set (SRR360205) using TopHat, then using Cufflinks (Trapnell 2010) to quantify expression level. The number of true positives and false positives found by each program, as well as specificity, is shown in Table 3.2. Three results are reported for UnSplicer, corresponding to three different probability score thresholds (0.1, 0.5, and 0.9). The threshold value appears next to the column title in parentheses. It is worth noting that the number of false positives is very effectively reduced by selecting a high threshold for UnSplicer. For UnSplicer (0.9), the false positive incidence is less than 1 per 400 predictions. While some programs perform marginally better than others, it is clear that all programs perform quite well on simulated data. Summarizing these results, UnSplicer has the highest specificity, while TrueSight has the highest overall sensitivity (with SOAPsplice slightly higher for the 75 nt reads).

Table 3.2. Number of true positives (Tp), false positives (Fp), and specificity (Sp) for five programs mapping simulated reads to *A. thaliana*.

Tp	UnSplicer (0.1)	UnSplicer (0.5)	UnSplicer (0.9)	TrueSight	Tophat	SOAPsplice	PASSion
50 nt	81595	79584	68332	**84133**	79056	82216	82113
75 nt	84680	83874	78110	84831	84246	**85137**	84355
100 nt	85168	83646	75936	**85950**	85714	84815	84471

Fp	UnSplicer (0.1)	UnSplicer (0.5)	UnSplicer (0.9)	TrueSight	Tophat	SOAPsplice	PASSion
50 nt	726	**405**	161	1722	2953	502	2159
75 nt	807	**427**	183	858	2857	496	1815
100 nt	811	**430**	155	1614	3259	501	1765

Sp (%)	UnSplicer (0.1)	UnSplicer (0.5)	UnSplicer (0.9)	TrueSight	Tophat	SOAPsplice	PASSion
50 nt	99.12	**99.49**	99.76	97.99	96.40	99.39	97.44

75 nt	99.06	**99.49**	99.77	99.00	96.72	99.42	97.89
100 nt	99.06	**99.49**	99.80	98.16	96.34	99.41	97.95

Table 3.3 reveals the total number of reads aligned by each program. Since there were 5 million paired-end reads given as input, the maximum number of aligned reads is 10 million. Tophat and SOAPsplice emerge as winners in this category, although TopHat predicted the largest number of false positive spliced alignments. Based on simulated read alignment, it would appear that SOAPsplice is the best alignment program. When we examine real data, we see a very different result (below).

Table 3.3. The number of simulated reads aligned by each program (in millions of reads), by read length.

	UnSplicer	TrueSight	Tophat	SOAPsplice	PASSion
50 nt	9.560	9.394	9.905	9.599	8.896
75 nt	8.017	7.924	8.393	8.422	8.299
100 nt	6.953	6.873	7.325	7.424	7.790

We examined the effect of alignment depth on intron prediction accuracy. Figure 3.11 shows the number of correct and spurious predictions made by each program as a function of alignment depth. TopHat makes the largest number of correct predictions as the alignment depth increases (panels a,c,e). However, TopHat also predicts the most spurious introns (panels b,d,f) for all read lengths. TrueSight also has high sensitivity, with fewer spurious predictions compared to TopHat, but more spurious predictions compared to other programs. SOAPsplice predicts the fewest correct introns with high alignment coverage, but also the fewest spurious introns (with depth greater than 10). For greater detail at low coverage depths, Figure 3.12 shows those introns predicted with low alignment coverage. We notice that variation in the pattern of correct predictions decreases with increasing read length. The notable exception is SOAPsplice, which predicts many more correct SJ at low coverage (1-5 alignments) compared to all other programs. A second fact is that UnSplicer predicts the least number of false positives for SJs with only 1-2 reads aligning across them. Both TopHat and PASSion predict a large number of false positives at low alignment depth.
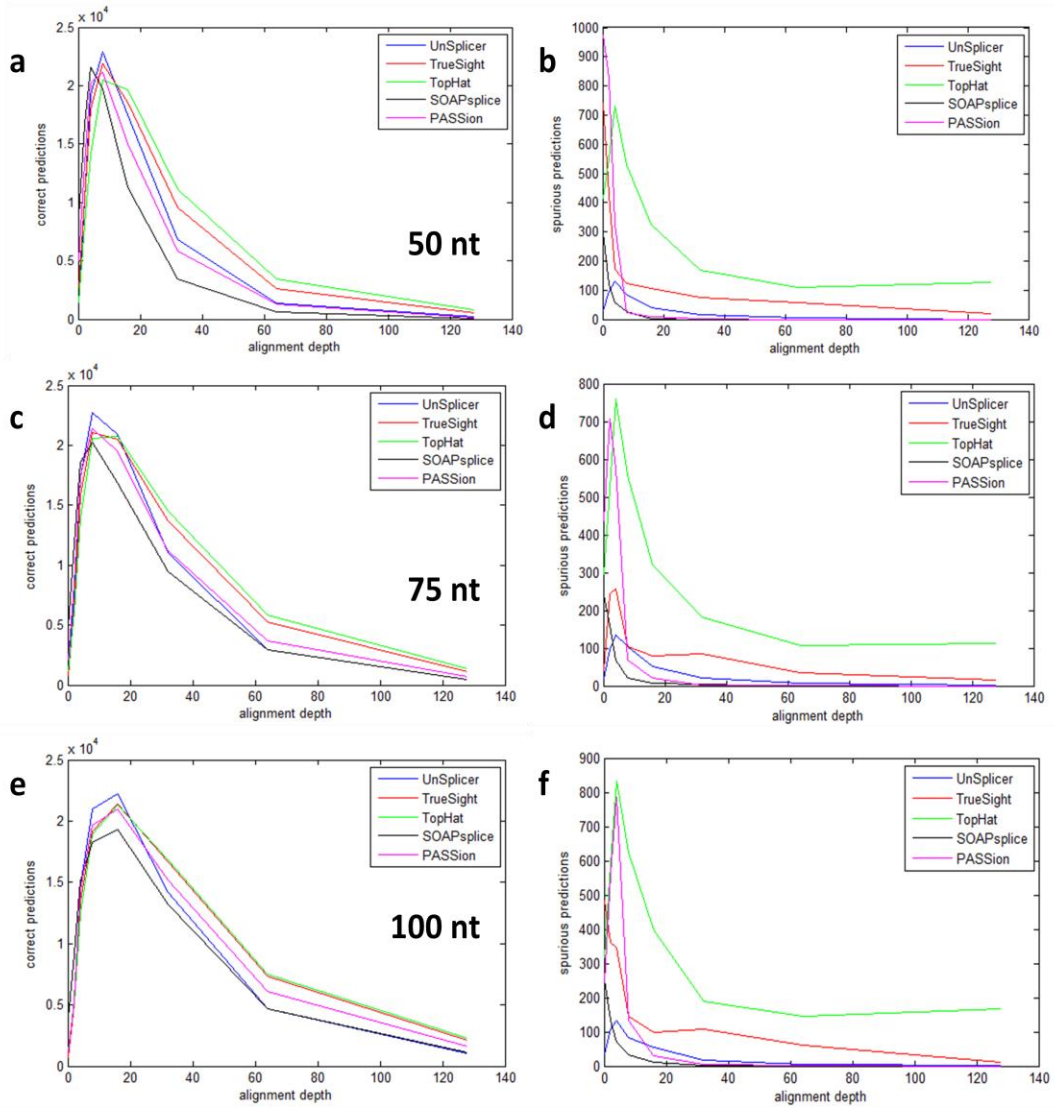
Figure 3.11. Mapping performance of simulated reads with different lengths. Panels a,c,e show the number of correct and panels b,d,f show spurious predictions made by each program on the simulated reads as a function of alignment depth. Panels a and b show the results for the 50 nt paired-end reads, panels c and d correspond to the 75 nt reads, and panels e and f correspond to 100 nt reads.
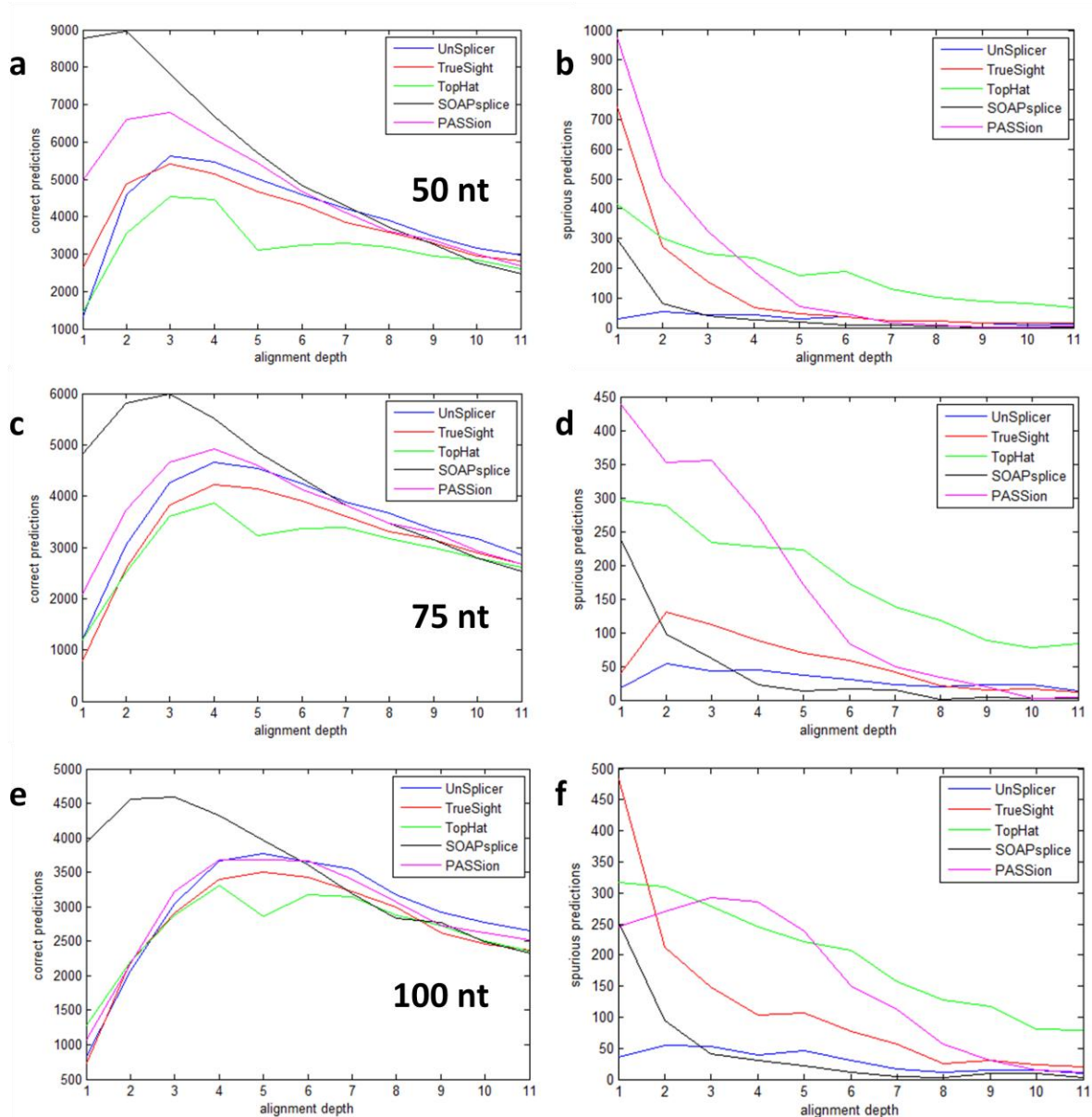
Figure 3.12. Mapping performance of simulated reads with different lengths (low coverage junctions). Correct and spurious predictions made for splice junctions with low alignment depth. Panels a,c,e show the number of correct predictions and panels b,d,f show spurious predictions made by each program on the simulated reads. Panels a and b show the results for the 50 nt paired-end reads, panels c and d correspond to the 75 nt reads, and panels e and f correspond to 100 nt reads.

All experiments were performed on a 16 processor multi-user Linux system. It is not generally feasible to a run program in isolation, nor would this typically be the case in automated pipelines. Therefore, two measures of computational cost are reported: total elapsed time ("wall clock" time), and the CPU time for each program. CPU time is a

41

measure of time in which processors are occupied by the program. All 5 compared

programs are multi-threaded, so we specified 8 threads to be used for each program. A

comparison of computational cost for alignment of the simulated reads are shown in

Table 3.4. Because 8 threads are allowed, the CPU time is often much greater than the

wall clock time. The fastest program are TopHat, with SOAPsplice a close second.

However, for 100 nt long reads, SOAPsplice slows considerably. For 100 nt long reads,

run times for UnSplicer and TrueSight are close to SOAPsplice. It is worth noting that

while TopHat is the speed king in this group, our results indicate that among the 5

programs compared, it is clearly the least accurate for spliced alignment. The most costly

program to run is PASSion, which required 7-8 hours of wall time with 8 threads.

UnSplicer and TrueSight are on par with each other, typically requiring 1-2 hours of wall

time.

Table 3.4. The time and resources required by each of the five programs to map the
simulated reads to the *A. thaliana* reference genome. All times are shown in
(hours:minutes).

Wall clock (h:m)	UnSplicer	TrueSight	Tophat	SOAPsplice	PASSion
50 nt	1:29	1:13	0:22	0:15	7:54
75 nt	1:39	1:31	0:19	0:45	8:43
100 nt	2:08	1:55	0:28	1:45	7:34
SRR360205 (76 nt)	5:37	5:19	1:10	3:55	34:29

CPU time (h:m)	UnSplicer	TrueSight	Tophat	SOAPsplice	PASSion
50 nt	3:51	3:09	0:51	0:50	10:18
75 nt	3:27	4:00	0:58	2:55	15:06
100 nt	4:36	4:53	1:29	5:28	22:02
SRR360205 (76 nt)	13:23	19:49	6:26	23:35	102:42

Table 3.5. Description of real RNA-seq data sets and reference genomes used for comparison.

Species	Data set (SRA no.)	Read length (bp)	No. pairs (millions)	Genome ver.	Annotation ver.
A. thaliana	SRR360205	76	20.9	TAIR10	TAIR10
C. elegans	SRR359066	101	12.2	Ce10	RefSeq, Ensembl
D. melanogaster	SRR042297	75	13.6	r5.42	r5.42
C. neoformans	SRR563164	101	5.7	Broad Institute*	Broad Institute*

* *Cryptococcus neoformans var. grubii* H99 Sequencing Project, Broad Institute of Harvard and MIT

To assess the performance on real data sets for compact genomes, we used RNA-seq data sets available for four different species: *A. thaliana*, *C. elegans*, *D. melanogaster*, and the fungus *C. neoformans* (as summarized in Table 3.5) and counted the number of intron predictions which were in agreement with the annotation and the number which were novel (not annotated). Figs. 3.13-3.16 depict the performance of each program in terms of the number of confirmed introns (previously annotated) compared to the number of predicted novel introns (not annotated). Because both UnSplicer and TrueSight provide probability scores for each splice junction we could build a "receiver operating characteristic," or ROC curve. Predicted introns are ranked by probability score, and the ROC curve may be defined by variation of the detection threshold over [0,1]. Results produced by other programs are represented by single points.
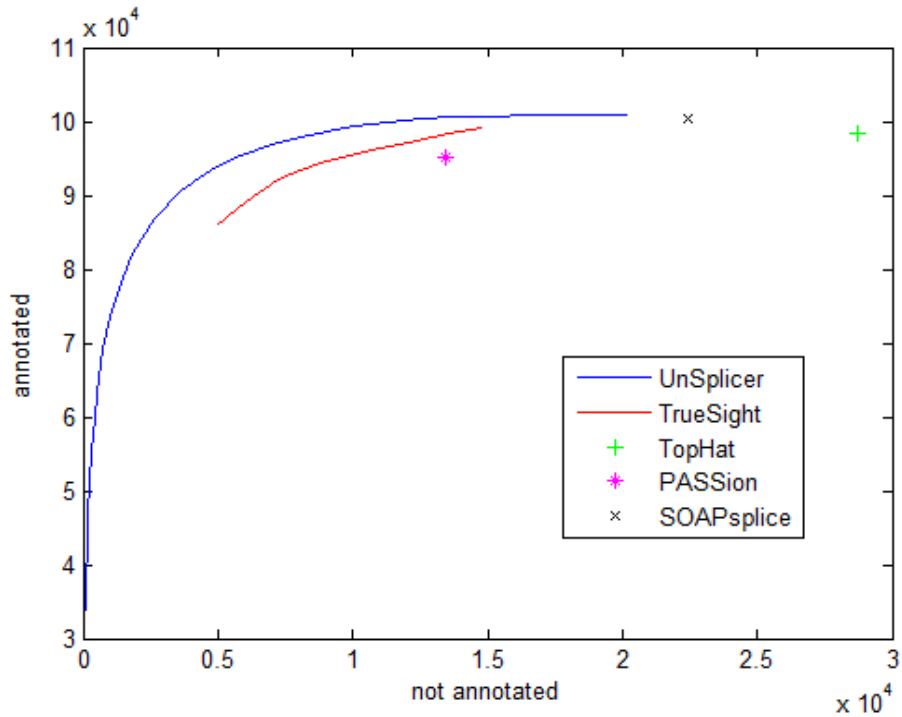
Figure 3.13. Performance comparison of several RNA-seq mapping programs on the *A. thaliana* data set. ROC curve for UnSplicer (blue curve) and TrueSight (red curve); TopHat (green +), PASSION (violet star), and SOAPsplice (black X). The UnSplicer ROC curve shows higher performance, with fewer novel introns predicted for a given number of predicted splice junctions.
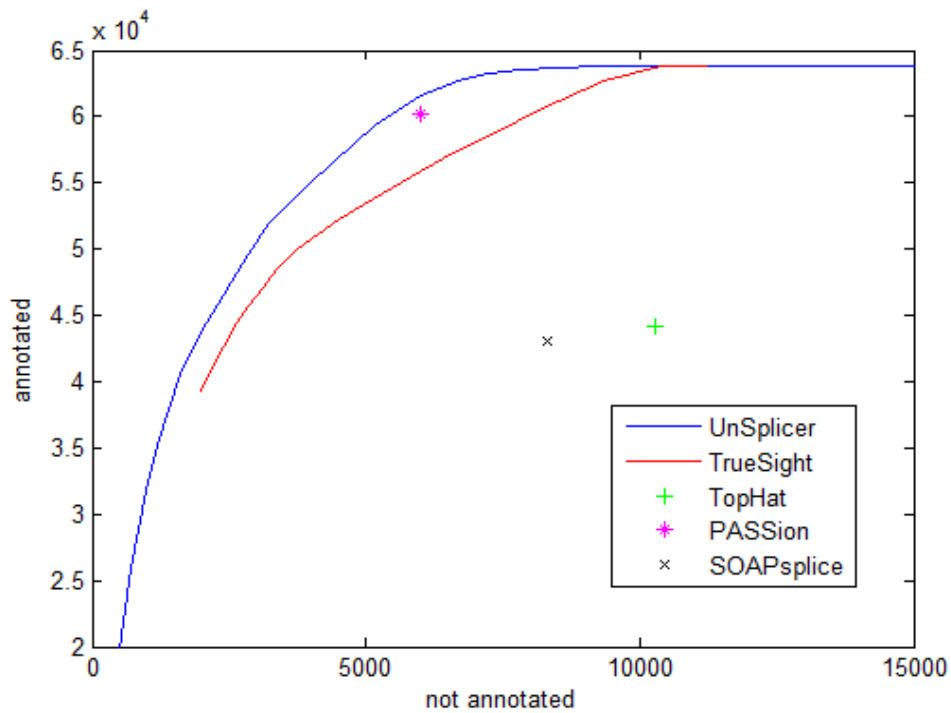
Figure 3.14. Performance comparison of several RNA-seq mapping programs on the *C. elegans* data set. Labels are the same as in the previous figure.
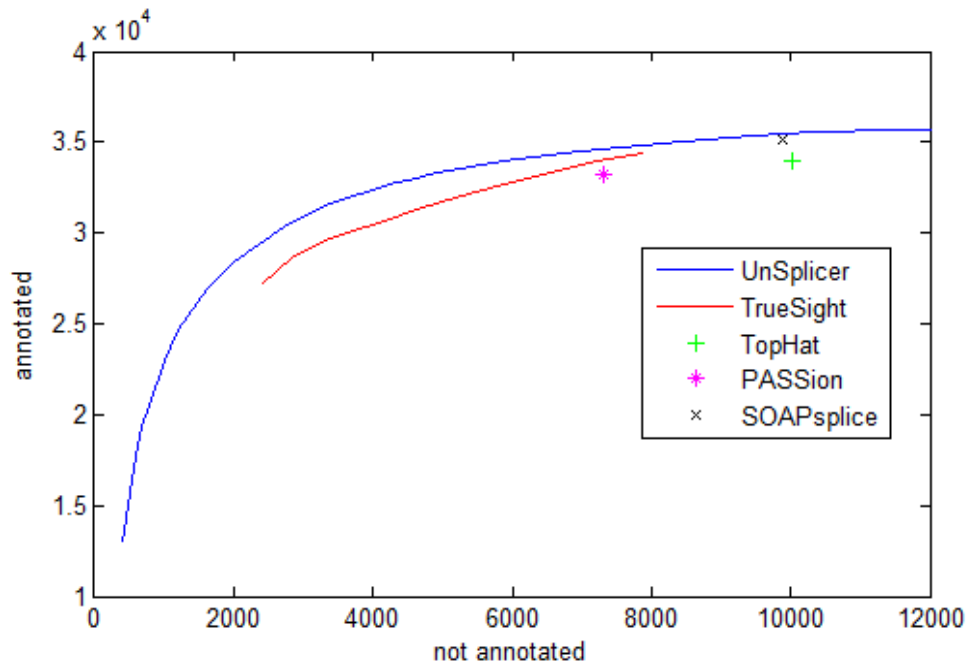
Figure 3.15. Performance comparison of several RNA-seq mapping programs on the *D. melanogaster* data set.
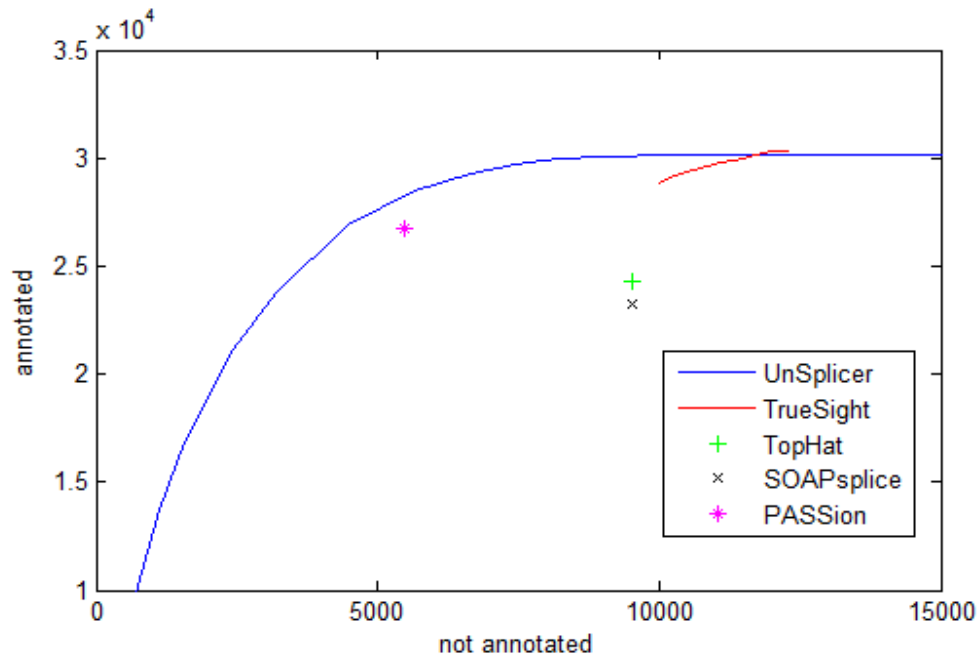
Figure 3.16. Performance comparison of several RNA-seq programs on the *C. neoformans* data set. The UnSplicer ROC curve demonstrates higher level of discrimination between true and false splice junctions compared with all other pipelines, with the exception of marginal improvement by TrueSight when a very low detection threshold is used.

It is expected that the vast majority of introns predicted by mapping programs which have no match in the annotation are spurious. That is, they arose from mapping errors or splicing noise (or both). This is particularly true for the model genomes, which have a large number of EST sequences supporting gene structures. In order to compare the performance of the five programs, we use novel introns as a rough measure of spurious introns. Therefore, we believe prediction of fewer novel introns is strong evidence of better overall prediction performance.

For the RNA-seq data sets in Table 3.5 UnSplicer demonstrated the best performance by a substantial margin (Figs. 3.13-3.16). In *A. thaliana* and *C. neoformans*, the difference with other programs was most pronounced, with UnSplicer predicting the smallest number of spurious introns. UnSplicer reported about 5,500 or 6,600 more introns confirming the respective annotations, compared with TopHat and SOAPsplice, for an equal number of reported novel junctions. TrueSight detected a small number (134) of true positives in *C. neoformans* not found by UnSplicer if the probability threshold is set to zero. However, UnSplicer can filter spurious introns much more effectively, as is evident in Fig. 3.16.

Fig. 3.17 compares the number of novel introns predicted by each program to the number predicted by UnSplicer. The data in Fig. 3.17 are found by finding the difference between the number of novel introns predicted by each program and the number predicted by UnSplicer at the point on the ROC curve matching the corresponding program's annotated predictions. In other words, the difference is considered along the abscissa in Figs. 3.13-3.16. This difference is labeled as a reduction in spurious introns, because it is expected that most such introns are due to splicing noise. For TopHat, PASSion, and SOAPsplice, the reductions are shown as points, while TrueSight appears as a curve. A substantial reduction in spurious predictions is seen compared to all programs and in all data sets, with the lone exception of TrueSight in *C. neoformans* for a very low probability score threshold (mentioned above).

In Fig. 3.18, these results are presented as percent reductions in spurious predictions, relative to the number of spurious predictions made by each program. Most reductions are very substantial, particularly TopHat which can be matched by UnSplicer in sensitivity with 60-80% reduction in spurious predictions in *A. thaliana*, *C. elegans*, and *C. neoformans*. Compared to TrueSight, spurious predictions are reduced by 20-50%, which demonstrates the advantages of UnSplicer on compact genomes.
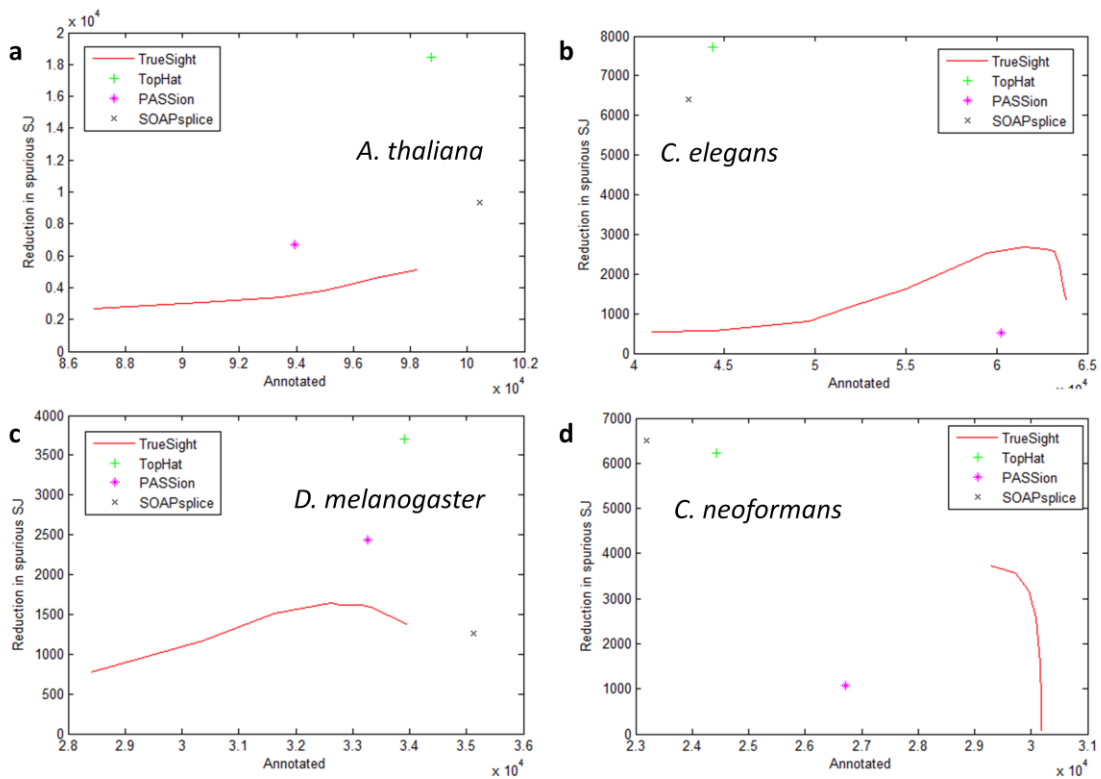
Figure 3.17. Reduction in spurious splice junction (SJ) prediction. Panels a-d compare the number of spurious splice junctions predicted by each program on the four data sets, to the number predicted by UnSplicer. The reduction is evaluated at a point on the UnSplicer ROC curve matching each compared program's predictions confirming the annotation.
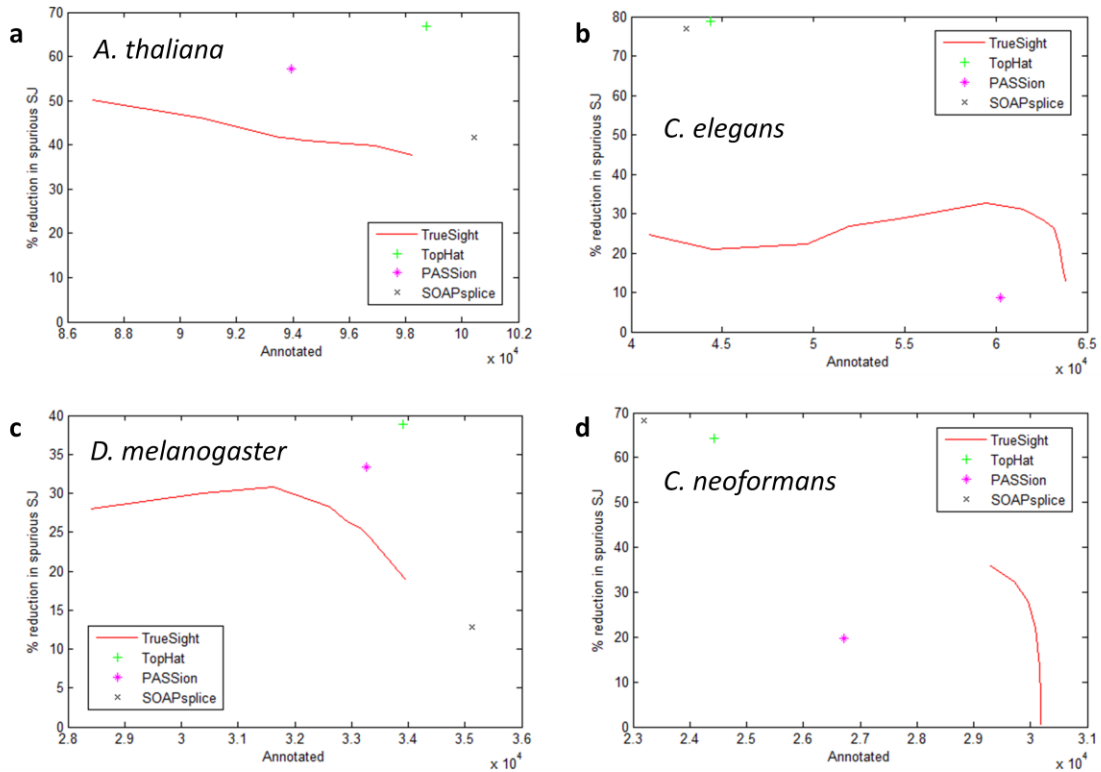
Figure 3.18. Normalized reduction in spurious splice junction (SJ) prediction. Panels a-d compare the number of spurious splice junctions predicted by each program on the four data sets, to the number predicted by UnSplicer. The reduction is calculated as a percent reduction achieved by UnSplicer compared to each program.

Fig. 3.19 compares the number of additional annotated introns found by UnSplicer to the number found by other programs. This is essentially the difference between UnSplicer and the other programs along the vertical direction in the ROC curves (Figs. 3.13-3.16). In virtually all cases, UnSplicer confirms thousands of introns more than other programs when the number of spurious predictions is matched. In Fig. 3.20, these increases are shown as a percentage gain compared to each program. In *D. melanogaster*, UnSplicer finds ~1,500 additional annotated introns compared to the programs TopHat, PASSion, and TrueSight. In *A. thaliana*, UnSplicer finds many thousands of additional confirmed introns compared to the other DNA-aware program TrueSight.
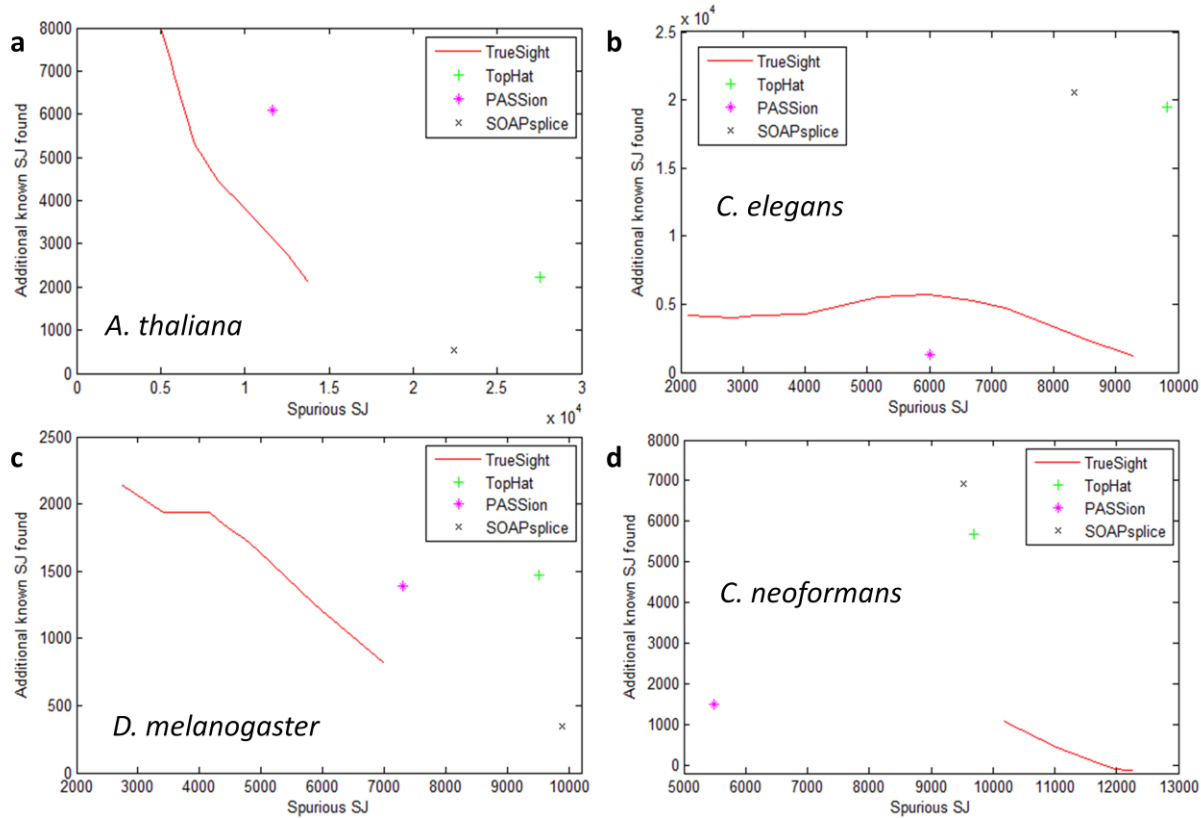
Figure 3.19. Improvement in annotated splice junction (SJ) prediction. Panels a-d show the number of *additional* annotated splice junctions found by UnSplicer compared to other programs on the four data sets.
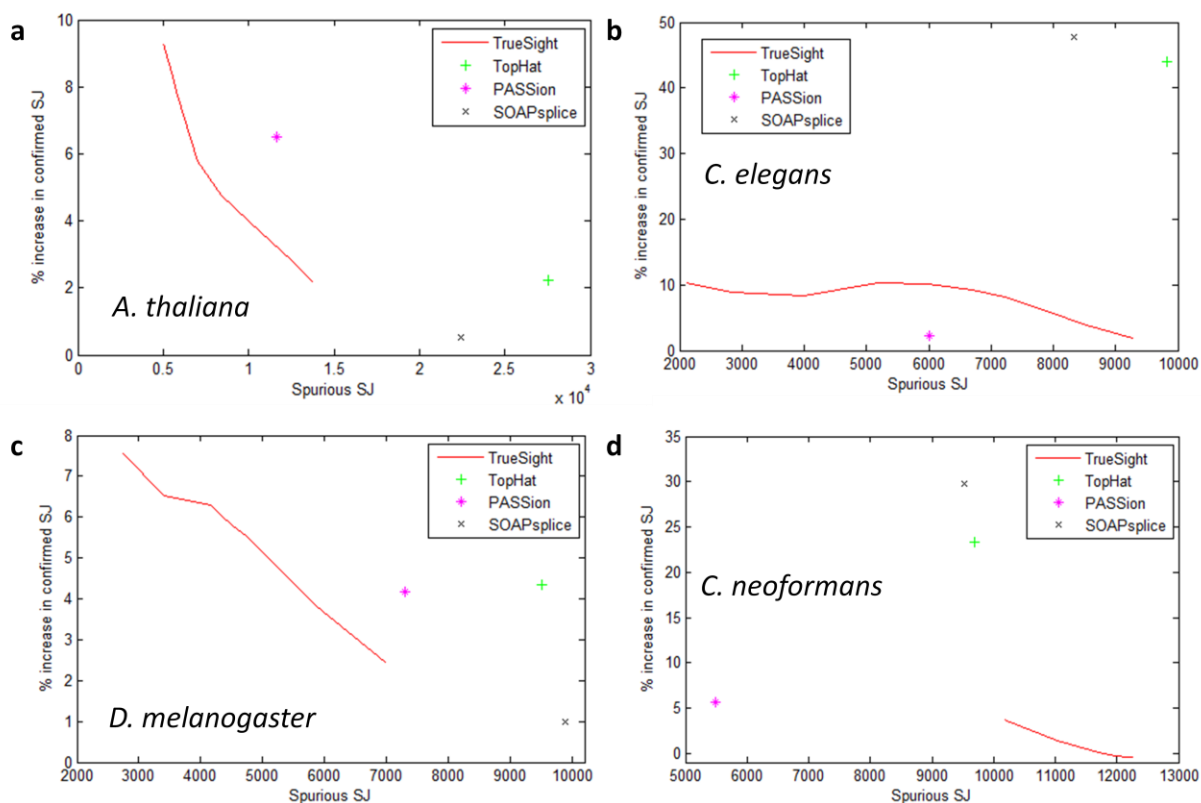
Figure 3.20. Increase in predictions of annotated splice junctions (SJ) by UnSplicer compared to four other programs. The percent increase is determined by (TpU-Tp)/Tp), where Tp is the number of true positives (annotated) SJ predicted by a program, and TpU is the number of true positives predicted by UnSplicer. A threshold value for UnSplicer is selected so that the number of confirmed annotated SJ predictions by UnSplicer matches the number of the corresponding program.

Table 3.6 compares the prediction accuracy of UnSplicer when a threshold value of 0.5 is used with the four other programs. UnSplicer consistently predicts introns with the highest value of annotated predictions (Tp) minus novel predictions (Fp), which is a measure of general performance. UnSplicer predicts the lowest number of novel introns, with the exception of PASSion on the *C. neoformans* data set. In this case, PASSion has slightly fewer novel predictions, but at the cost of about 2,500 fewer annotated predictions compared to UnSplicer. In general, UnSplicer outperforms PASSion because the ROC curve lies above and to the left of the point represented by PASSion (Fig. 3.16). A threshold value of zero was selected for TrueSight in this table because that threshold

was used in the TrueSight publication (Li, 2012). In summary, on all data sets evaluated the best overall performance was achieved by UnSplicer.

Table 3.6. Comparison of the UnSplicer performance (with default threshold of 0.5) to performance of other RNA-seq alignment tools.

		UnSplicer (thresh.=0.5)	TrueSight (thresh.=0)	TopHat	PASSion	SOAPsplice
A. thaliana	Tp	92,818	99,051	98,368	93,934	**100,416**
	Fp	**4,711**	14,770	28,730	11,654	22,418
	Tp-Fp	**88,107**	84,281	69,638	82,280	77,998
	Sp (%)	**95.17**	87.02	77.40	88.96	81.75
C. elegans	Tp	61,529	**63,847**	44,213	60,247	43,043
	Fp	**5,983**	11,201	10,226	6,005	8,327
	Tp-Fp	**55,546**	52,646	33,987	54,242	34,716
	Sp (%)	**91.14**	85.07	81.22	90.94	83.79
D. melanogaster	Tp	32,624	34,441	33,936	33,260	**35,117**
	Fp	**4,168**	7,876	10,021	7,306	9,891
	Tp-Fp	**28,456**	26,555	23,915	25,954	25,226
	Sp (%)	**88.67**	81.39	77.20	81.99	78.02
C. neoformans	Tp	29,273	**30,314**	24,288	26,723	23,202
	Fp	6,637	12,300	9,534	**5,482**	9,531
	Tp-Fp	**22,636**	18,014	14,754	21,241	13,671
	Sp (%)	81.52	71.14	71.81	**82.98**	70.88

Because UnSplicer uses some features derived from gene prediction, it should be examined whether it can detect as many introns which connect non-coding exons as other programs. This would include UTR introns and introns inside non-coding genes. Table 3.7 shows the number of annotated *A. thaliana* introns which were detected by each program, categorized as either "coding" (connecting coding exons) or "non" (connecting non-coding exons). A junction is non-coding only if the nucleotide immediately adjacent

51

to the intron in both exons is non-coding in all annotated isoforms. The ratio of coding-to-non is shown in the bottom row of the table. A high ratio would indicate a bias for detecting introns in coding regions over non-coding. Surprisingly, PASSion has the highest ratio of all programs (excluding the high threshold case with UnSplicer). We attribute this high ratio to PASSion's pattern growth algorithm, which preferentially detects "coding" introns compared to "non-coding". UnSplicer (0.5) also has a relatively high ratio, but only 15% higher compared to the more conventional aligners such as TopHat and SOAPsplice. While this difference is notable, it is a natural artifact of the detection problem in that there is more evidence available to reject spurious junctions in coding regions. This point is more clearly illustrated in Fig. 3.21, where UnSplicer's ratio of coding to non-coding intron detections increases with increasing probability threshold. Also shown is the ratio for TrueSight, which is nearly invariant to probability score. We suspect that the reason for this is that TrueSight's frame-less coding sequence model, which is the basis for its coding potential score, does not play a major role in SJ discrimination in this example.

Table 3.7. Comparison of the number of annotated splice junctions in *A. thaliana* which were found by each program by aligning the data set SRR360205. Junctions connection coding exons are labeled as 'coding' and all others 'non'. The ratio is the number of confirmed annotated coding SJs divided by the number of confirmed non-coding SJs. The UnSplicer numbers are determined by considering all predictions having probability scores greater than the thresholds: 0.1, 0.5, and 0.9.

SJ type	UnSplicer (p=0.1)	UnSplicer (p=0.5)	UnSplicer (p=0.9)	TrueSight	TopHat	SOAPsplice	PASSion
coding	92306	88980	70902	92977	92682	94207	93934
non	5737	5073	3331	6074	6047	6218	5236
coding / non	16.09	17.54	21.29	15.31	15.33	15.15	17.94
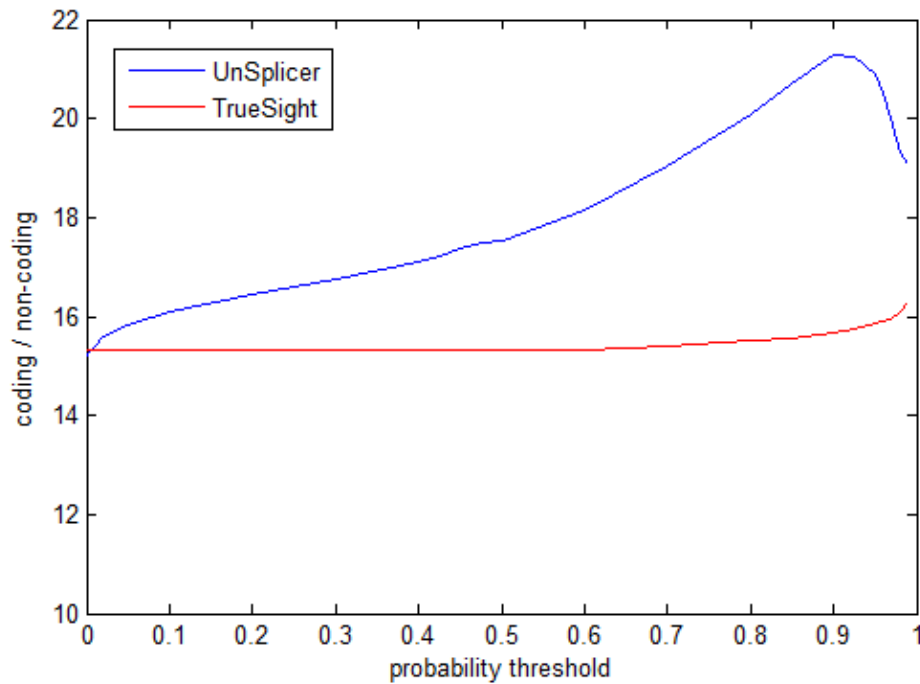
Figure 3.21. Ratio of predicted "coding" introns to "non-coding" as the probability threshold is increased.

The anchor-and-extend technique used by UnSplicer to find gapped alignments will have some impact on detection performance of introns which have short exons flanking them. This is due to the fact that reads aligned in this fashion cannot be mapped across SJs at both ends of a short exon unless the exon is sufficiently long. Reads mapped across only one SJ are not affected. We compared the effect of exon length on detection sensitivity of the five programs on the *A. thaliana* data set. The set of all detections confirming annotated introns was constructed for each program. A separate set of exons was made by including the two flanking exons for each intron in the intron set. A distribution of the lengths of the exons was calculated. In Table 3.8, the number of detected introns is shown for each program for different intervals of flanking exon length. All introns predicted by UnSplicer are considered (the probability score threshold is zero in this case). UnSplicer is the most sensitive for introns next to exons having length 25 nt or shorter. This supports the anchor-and-extend approach for finding introns.

Table 3.8. The distribution of the length of exons flanking every prediction confirming an annotated intron in *A. thaliana*. All introns predicted by UnSplicer are considered (threshold = 0). The anchor-and-extend strategy used by UnSplicer improves detection sensitivity of introns adjacent to short exons compared to other programs

exon length (nt)	PASSion	TrueSight	UnSplicer	TopHat	SOAPsplice
1-24	479	378	482	382	470
25-49	7690	8066	8049	8048	8227
50-99	57102	59904	59483	59757	60678
100-199	64133	67758	67629	67539	68722
200-499	43965	46483	46548	46311	47048
500-	14499	15513	15535	15421	15705

Next we compared the prediction specificity of each program as a function of intron length in the *A. thaliana* data set. The intron predictions made by each program were partitioned into subsets according to their length. The specificity was calculated for each subset of predictions. In general, we expect specificity to drop as with increasing intron length because single-nucleotide precision of feature boundary prediction requires more discrimination power with increasing length. Table 3.9 confirms that each program exhibits a monotonically decreasing specificity with increasing intron length, with the exception of UnSplicer. UnSplicer (with a probability threshold of 0.5) predicts with a high and fairly constant specificity (94-95%) up to a length of 2000 nt. For all introns longer than 2000 nt, specificity drops dramatically. However, even in that case, UnSplicer's specificity is 2.5 times higher compared to the next best program (TrueSight). We observe that most programs severely over-predict long introns, with a ratio of novel to annotated introns from 6.8:1 (TrueSight) to 56:1 (TopHat).

Table 3.9. Specificity of intron predictions is shown as a function of intron length for the *A. thaliana* data set. There were no annotated introns predicted by any program shorter than 50 nt.

intron length (nt)	UnSplicer	TrueSight	TopHat	SOAPsplice	PASSion
50-99	94.8283	**95.7735**	90.0762	89.6227	95.3515
100-199	**93.7958**	93.0739	82.5879	82.1278	91.7287
200-499	**93.7966**	90.6623	74.6497	74.761	86.3691
500-1999	**95.6146**	79.2293	41.8709	49.4626	57.6031
2000-	**38.7097**	14.9718	1.7883	11.1531	4.3651

As a final illustrative example, in Fig. 3.22 we show results of splice junction predictions by several programs, PASSion, SOAPSplice, TopHat, TrueSight and UnSplicer, graphically depicted for a 10 Kb region of chromosome 1 of *A. thaliana* carrying three multi-exon protein coding genes, as annotated in TAIR. UnSplicer predicts no false positive splice junctions in this region. All other programs while predicting the same number of true positives as UnSplicer predict few false positives.
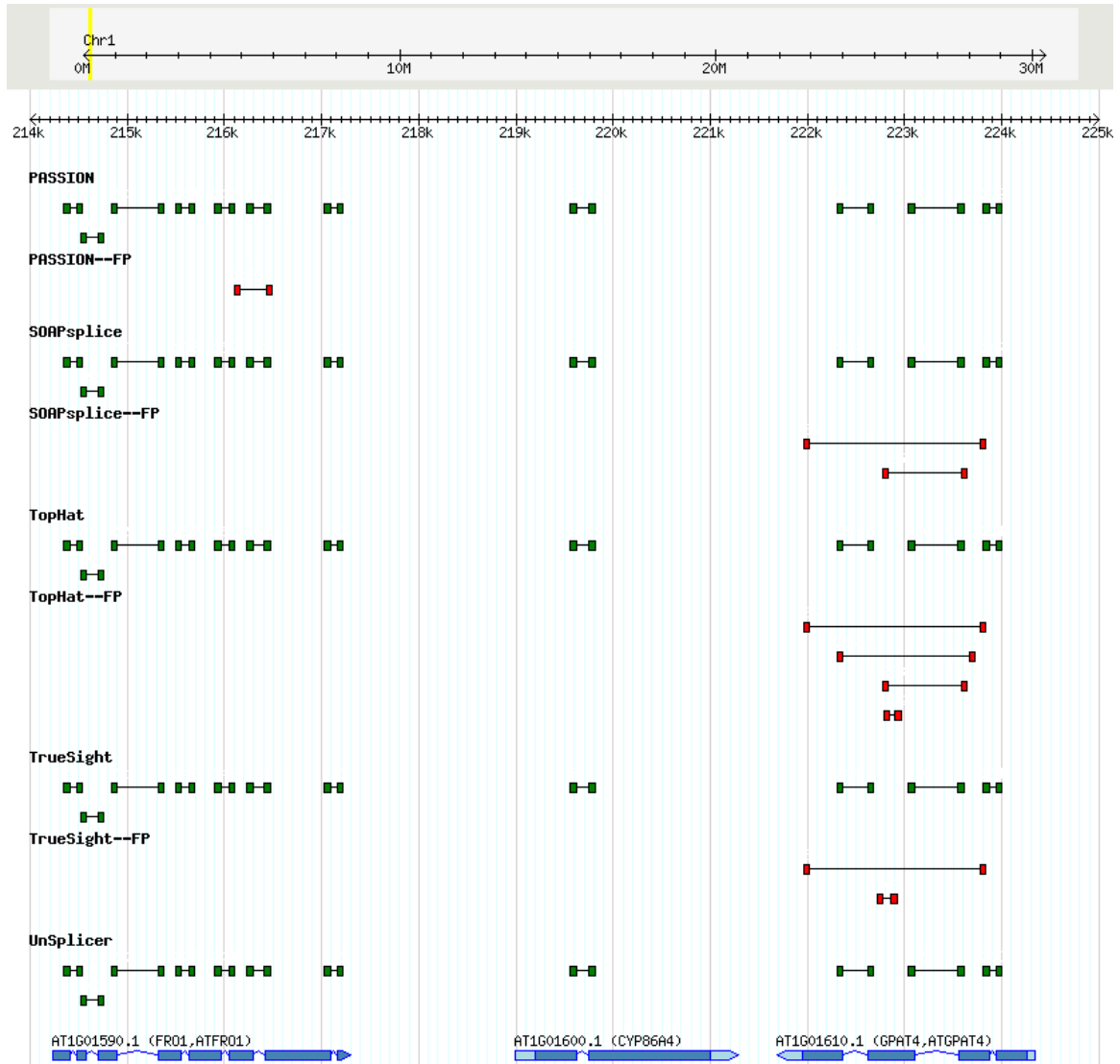
Figure 3.22. Splice junctions predicted by five programs are graphically depicted in 10Kb region of *A. thaliana* chromosome 1. True positive splice junctions (matching annotation) are shown in green while false positives are shown in red in the FP panel for a corresponding program. The three genes annotated in the region are graphically depicted in the bottom. UnSplicer predicts no false positives in this region.

Discussion of simulated data

There is a clear gulf between the results of simulation and alignment of real data. This is caused by the absence of splicesosome noise modeling in read generating simulations. In this comparison, we use well annotated genomes to conclude UnSplicer is much more effective at predicting introns in compact genomes compared to other programs. Simulated data is of limited use in evaluation of spliced alignment programs,

and can lead to incorrect conclusions if used in isolation. Our simulation results suggested that SOAPsplice was a superior alignment program. However, results on real data reveal considerable deficiencies of SOAPsplice. UnSplicer predicts nearly 10,000 fewer spurious introns compared to SOAPsplice in the real *A. thaliana* data set, whereas in simulated data on the same genome the two programs appeared to be very comparable. In addition, with simulated reads SOAPsplice found many more read alignments compared to UnSplicer. However, with real *A. thaliana* data, UnSplicer mapped 33.7 million reads, while SOAPsplice only mapped 26.1 million.

Relationship to TrueSight

UnSplicer and TrueSight have three major differences: i/ in derivation of DNA sequence model parameters, ii/ in training set selection (for classification), iii/ in classification algorithm. UnSplicer derives model parameters by use of GeneMark-ES (conducting unsupervised training on a reference genome), while TrueSight derives parameters directly from the training set of splice junctions. A danger of reliance on RNA-seq alignments for deriving model parameters is a possibility of bias in parameters of the protein-coding sequence model due to overrepresentation of highly expressed genes. It is well known that for in prokaryotic and eukaryotic genomes, codon usage of highly expressed genes differs from codon usage of genes expressed at moderate and low levels (see for example, Duret 1999, Moriyama 1998). UnSplicer avoids this problem because GeneMark-ES trains parameters based on a large number of genes (many thousands).

The training set for the UnSplicer classifier is guided by a different set of heuristic rules (described above), in comparison with rules used in TrueSight. The training set of positive examples in TrueSight is formed from the set of all splice junctions with canonical splice sites for which: i/ no mismatch errors occur in the alignment and ii/ the splice junction is confirmed by 5 or more alignments. The set of negative examples is

taken from the set of gapped alignments spanning introns with non-canonical splice sites, such that: i/ the SJ is confirmed by only one alignment, and ii/ the short side of the aligned RNA read is close to the admitted minimum, either 8 or 9 nt. These training sets are also used for deriving sequence model parameters.

The third major difference is in the design of the classification algorithm. TrueSight finds a decision boundary by the expectation maximization (EM) algorithm maximizing a likelihood of all candidate splice junctions presented to the classifier. The EM algorithm is searching for a hyperplane boundary separating positive and negative examples in the training set. UnSplicer uses the training set to find parameters of a Gaussian kernel SVM that maximizes agreement with *ab initio* gene predictions on a development set.

In addition to the three major differences between UnSplicer and TrueSight, few more minor differences exist. For instance, instead of the coding potential used by TrueSight, we use two binary indicators, strand concordance and frame shift indication. These indicators have zero value for true introns (both protein-coding and UTR introns) and non-zero value for many false positives. Also, UnSplicer uses an intron length distribution provided by GeneMark-ES in the gap length feature which is essentially a log-likelihood that an intron has a given length. Intron length distributions derived by GeneMark-ES show good agreement with empirical distributions determined from transcript sequence alignments. In an example for the strawberry genome (*F. vesca*) shown in Fig. 6 one can see that intron length distribution inferred by GeneMark-ES nearly coincides with one determined from transcript sequence alignments (Fig. 3.23). As the intron length related feature TrueSight uses a score defined as follows: for an intron of length l, the score is zero if $l \leq L_{0.05}$, otherwise it is $log(l - L_{0.05})$, where $L_{0.05}$ is the length for which 95% of candidate introns are shorter. Also, in TrueSight donor and acceptor site log likelihood scores are summed together in a single feature, while UnSplicer has two separate features.
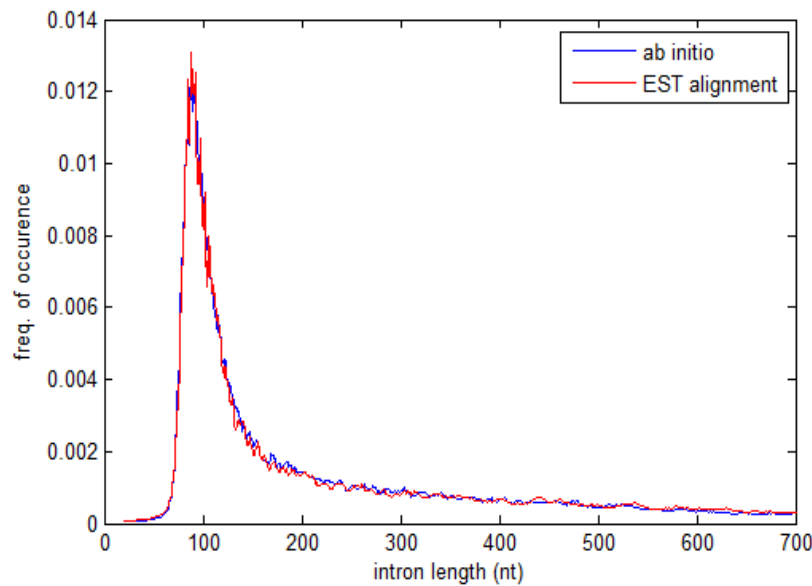
Figure 3.23. Comparison of strawberry (*F. vesca*) intron length distributions found by GeneMark-ES (blue) and one derived from alignment of EST sequences (red).

Use of GeneMark-ES makes UnSplicer applicable to newly sequenced genomes with no annotation where RNA-seq data have become available as well. On the other hand dependence of GeneMark-ES restricts UnSplicer to genomes with homogeneous G+C content. In nonhomogeneous genomes, such as large gnomes of animals, a separate training procedure must to be carried out to derive sequence model parameters needed for UnSplicer.

Lastly, we should note that UnSplicer performs the remapping and filtering steps prior to classification while TrueSight performs these steps after classification. The change of order in UnSplicer was made since the classification algorithm uses the number of alignments across a splice junction as one of the input features.

CHAPTER IV

GENEMARK-NGS: A CONDITIONAL LEARNING ALGORITHM FOR EUKARYOTIC GENE FINDING

Many years of development of eukaryotic gene finding techniques have led to a few basic methodologies which are widely used today. The most reliable systematic method of gene finding today is alignment of full-length cDNA sequences to a reference genome, and testing long open reading frames (ORFs) in the transcript sequences. Such cDNA sequences should ideally be constructed from long-read sequencing technology, due to difficulty in resolving correct isoform structures using short-read sequences. This isoform ambiguity may be alleviated somewhat with paired-end sequencing using carefully chosen fragment length variations. To date, iron clad gene prediction still requires long read sequences for full-length cDNA construction. However, outside of a few extensively studied model organisms, full-length cDNA construction from long-reads is rarely performed due to the relatively high expense of obtaining these sequences.

Comparative genomics (Korf 2001, Allen 2004, Gross 2006) is also commonly used to support gene finding. Conserved protein domains shared by related species are found in the target genome, and used to support exon-intron structures where possible. In (Kellis 2003), gene finding by comparative genomics was demonstrated to find 90% of genes in genomes of several species of novel yeast genomes. The methods rely on a combination of access to many related genome assemblies, and existence of long ORFs in DNA sequence, which is generally not the case in higher eukaryotes which have much shorter exons. In addition, simultaneous sequencing of several larger and more complex genomes is not nearly as practical compared with yeast genomes. Furthermore, in many cases novel genomes are assembled with no available genomes of close biological relatives from which gene structure indications in target genome can be inferred.

Therefore in practice, comparative genomics is of limited value in annotation of full gene structures of eukaryotes.

Today, most transcriptome sequencing is performed by RNA-seq experiments due to production of large volumes of reads for low cost (Martin 2011, Wang 2009). Indeed, sequence production with this new apparatus is so great, efforts have been made to assemble entire transcriptomes with batches of RNA-seq runs (see Martin 2011, and references within). Two fundamentally different approaches have been proposed for this purpose: 1/ mapping reads to a reference genome, and assembling transcripts (Guttman 2010, Trapnell 2010), and 2/ *de novo* assembly of transcripts (Simpson 2009, Birol 2009, Grabherr 2011). The principle advantage of *de novo* assembly is the ability to reconstruct gene transcripts which are not found in the reference genome assembly (or even without a genome assembly altogether). Disadvantages include lower sensitivity to genes with low expression, false transcripts made from contaminating sequences in the RNA-seq experiment, and a requirement for much greater computational resources compared to mapping-and-assembly methods. If it were possible to assemble the entire transcriptome with RNA-seq reads, then the problem of gene finding in eukaryotes would be greatly simplified due to removal of exon-intron structures from the detection problem. Of course, this can only work on those transcripts which have been sufficiently reconstructed so that the protein coding region may be correctly identified in its entirety. The set of assembled transcripts for which coding sequences are fully covered will represent some fraction of all genes in a genome—a fraction dependent on the depth of sequencing performed as well as the variety of cell conditions and tissue types used for library construction. A successful application of *de novo* transcriptome assembly was reported (Grabherr 2011) in which roughly 90% of protein coding genes of *S. pombe* were assembled into transcripts by the Trinity pipeline. Less successful was the transcriptome assembly from mouse sequences, resulting in assembly of a little over half of known genes. In (Guttman 2010), a similar result of roughly 55% of known mouse genes were

recovered in full length with the program Scripture. The difference in genome coverage could be related to more stable or more extensive heterochromatin in the mouse genome compared to yeast.

Whether 10% or 50% of genes remain to be found, these recent results demonstrate computational gene finding methods are still required even with available deep transcriptome sequencing technology. Perhaps the most widely adopted and successful algorithm class for eukaryotic gene prediction is the generalized hidden Markov model (GHMM), which has been incorporated into many gene finders, such as GenomeScan (Yeh 2001) and Augustus (Stanke 2003), and GeneMark.hmm (Lukashin 1998). The GHMMs used by these programs require a training set to obtain parameters for prediction. A traditional approach to preparing a training set would involve alignment of full-length cDNA or other available long-read sequences of cellular RNA to a reference genome in order support highly reliable prediction of a small percentage of protein coding genes, which may then serve as a training set to find parameters for sequence models used by gene prediction algorithms.

Even when limited long-read transcript sequences are available, their scarcity tends to provide support to the most highly expressed genes in a genome. Derivation of sequence model parameters from a subset of genes which is overrepresented with highly expressed genes results in biased parameters. This bias is caused by differences in codon usage in genes having different constitutive expression level (Sharp 2010). This fact has been pointed out before (Rogic 2001), yet overlooked by many scientists because test sets used to measure gene prediction performance tend to be overrepresented by highly expressed genes.

A challenge for computational gene prediction is to incorporate the wealth of RNA-seq information in a productive fashion for finding functional protein coding genes over a whole genome. RNA-seq data tends to be somewhat error prone, with contributions originating from natural cell splicing errors, steps associated with cDNA

library preparation, sequencing, and the error-prone process of spliced alignment. The challenge is to find a method for parameter training which prevents such errors associated with RNA-seq to be incorporated into GHMM model parameters. We propose a new pipeline called GeneMark-NGS, which solves this problem using a combination of the RNA-seq mapping program UnSplicer (Burns, submitted for pub.) and GeneMark.hmm. GeneMark-NGS is the first gene prediction program which uses RNA-seq spliced alignments to improve GHMM model parameters for more accurate gene prediction.

Conditional learning using RNA-seq alignments

In GeneMark-NGS, RNA-seq spliced alignments to a reference genome are used to assist in training GHMM parameters for gene prediction. GHMM parameters are found by an iterative machine learning approach which incorporates spliced alignments into predicted exon confirmations, and predicted exons into spliced alignment predictions.

The basic concept we propose is a form of conditional learning illustrated in Figure 4.1. Splice junction detection is performed by UnSplicer and gene prediction with GeneMark.hmm. The two programs are used together in a ping-pong iterative fashion— UnSplicer providing intron confirmations and intron sub-models to GeneMark.hmm, and GeneMark.hmm providing gene predictions and improved model parameters back to UnSplicer. Each predictive algorithm is conditioned to some extent on the input provided by the other.
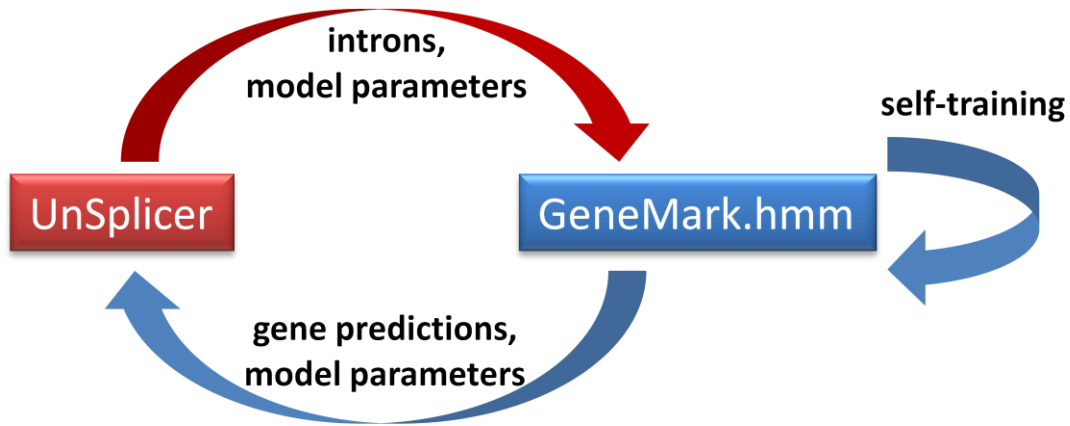
Figure 4.1. A diagram of the basic concept of conditional learning applied in GeneMark-NGS.

Due to available deep sequencing of transcriptomes, support for tens of thousands of introns are obtained with most RNA-seq data sets. In this new method, introns found by UnSplicer are compared with exons predicted by GeneMark.hmm. If an exon is in agreement with a gapped alignment—that is, if a predicted exon shares an intron boundary predicted by a gapped alignment—then the sequence in that exon will be included in the set of coding sequence used to train a Markov model for the next iteration. Such exons are highly likely to be biological exons containing coding sequence because two different sources of evidence confirm a boundary with single-nucleotide precision. The sequence within all such exons is used to train a Markov chain which models coding sequence (CDS). The CDS Markov chain parameters are part of a larger set of GHMM parameters used for the next iteration of predictions.

The process begins with alignment of available RNA-seq to the reference genome using UnSplicer in a special conservative mode. In the conservative mode, UnSplicer does not make use of features derived from DNA sequence to classify candidate splice junctions. Instead, only the following features are used to select the initial intron predictions: coverage skew, entropy, and alignment depth (the number of alignments across a splice junction). Only gapped alignments across canonical introns are considered in this process, and only canonical introns are predicted by GeneMark.hmm. A set of

64

"positives" and "negatives" are selected according to the training set selection criteria described in Chapter III. Additionally, introns having 5 or more gapped alignments confirming them are added to this set of "positives." All splice junctions in the positive set are given non-zero probability scores. All those remaining are given a probability score of 0. The positive set establishes an initial set of introns from which empirical donor and acceptor position specific frequency matrices (PSFMs) and intron length distribution are derived.

Next, the first prediction step is performed by GeneMark.hmm using GHMM parameters comprised if the intron sub-models described above, along with heuristic parameters for coding and non-coding sequence Markov chains (Besemer 1999).

The final step of the first iteration is the update of model parameters. In this step, new Markov chain parameters describing coding and non-coding sequence and PSFM parameters for gene start and stop contexts are derived from the set of predictions. The parameters are found by maximum likelihood estimation, using example sequences for each type. Gene start and stop contexts are derived from all predicted genes having more than 300 nt of coding sequence. Non-coding sequences are found by collecting predicted intergenic and intron sequences. Coding sequences, however, are selected very carefully in order to minimize the amount of non-coding sequence (or out-of-frame coding sequence) which may contaminate the set of predicted coding sequences. Coding sequences are found in two ways: 1/ sequence inside all open reading frames (ORFs) longer than 1000 nt and 2/ all predicted exons which border any intron boundary supported by UnSplicer (regardless of probability score). All remaining GHMM parameters (transition probabilities) are calculated according to the method adopted in GeneMark-ES.

In Table 4.1, a number of GHMM sub-model parameter types utilized by GeneMark.hmm are listed down the rows. The columns correspond to GeneMark-NGS iterations. In each cell, the method of deriving each parameter is indicated. There are

65

three (3) principal methods for obtaining model parameters: 1/ heuristic rules (for initial CDS and non-coding sequence Markov chain parameters), 2/ estimation based on the prior round of GeneMark.hmm predictions (conditioned on agreement with UnSplicer intron predictions in the case of CDS), and 3/ estimation based on a prior round of UnSplicer intron predictions. In the initial model, we mentioned that the donor and acceptor PSFMs and intron duration are derived from intron predictions made by UnSplicer (in a conservative mode). These parameters are labeled "UnSplicer (ini)" in Table 4.1. Splice site models in all frames are represented by the same parameters in this step. Exon durations (length distributions) are modeled as uniform in the first iteration.

Prediction and update steps are carried out in iterations 0 and 1 with no changes to the intron sub-models. The method of GeneMark-ES is used to update gene start and stop context models, and the non-coding sequence Markov model. Starting with iteration 1, exon durations are also updated by the method of GeneMark-ES. Coding sequence model parameters are based on sequence found in those predicted exons either having support from UnSplicer alignments or single-exon genes predicted longer than 1000 nt.

Gene predictions and the model parameters from iteration 1 are given as input to UnSplicer to support the second round of intron predictions from RNA-seq alignment. UnSplicer is run in the usual fashion in this step, discriminating candidate splice junctions using a set of 9 different features. Two of those features (strand concordance and frame shift indication) require gene predictions on the whole genome. The gene predictions are made by GeneMark.hmm using the model parameters in labeled "iter 1" in Table 1. UnSplicer assigns new probability scores to each candidate intron. This results in a more accurate set of likely introns (with probability score ≥ 0.5) from which new donor and acceptor PSFMs and intron duration are derived for iteration 2. In this step, the set of introns is partitioned into three different sets according to their inferred reading frame (their position in a codon) in order to derive frame-dependent models for

66

splice sites. Two additional GeneMark-NGS iterations are carried out (iterations 3 and 4), in order to allow the model parameters to reach convergence to a final point.

Table 4.1. The GHMM sub-models and method of updating in each iteration of the algorithm. Key: heuristic = initial parameters parameters based on (Besemer 1999), ES = GeneMark-ES algorithm, UnSplicer (ini) = initial UnSplicer (used in conservative mode), UnSplicer = parameters based on intron predictions made by UnSplicer, uniform = uniform length distribution, unch = unchanged from last iteration.

model	initial	iter 0	iter 1	iter 2	iter 3	iter 4	Iter 5
CDS seq	heuristic	GeneMark.hmm, UnSplicer	GeneMark.hmm, UnSplicer	GeneMark.hmm, UnSplicer	GeneMark.hmm, UnSplicer	GeneMark.hmm, UnSplicer	GeneMark.hmm, UnSplicer
NCOD seq	heuristic	ES	ES	ES	ES	ES	ES
start context	ATG	ES	ES	ES	ES	ES	ES
donor site	UnSplicer (ini)	unch	unch	ES, UnSplicer	unch	unch	unch
acceptor site	UnSplicer (ini)	unch	unch	ES, UnSplicer	unch	unch	unch
stop context	TAA,TAG,TGA	ES	ES	ES	ES	ES	ES
intron duration	UnSplicer (ini)	unch	unch	UnSplicer	unch	unch	unch
exon durations	uniform	uniform	ES	ES	ES	ES	ES

Results

GeneMark-NGS was run on the *D. melanogaster* genome and four mosquito genomes downloaded from VectorBase (Megy, 2012): *A. gambiae*, *A. stephensi*, *A. aegypti*, and *C. quinquefasciatus* genomes. These five genomes were selected due to their size variation, and availability of public RNA-seq data sets (refer to Table 4.2). The genome sizes vary from 158 MB (*A. stephensi*) to 1.31 GB (*A. aegypti*). The genomes all generally have homogeneous G+C content, as shown in Fig. 4.2. The genome of *C. quinquefasciatus* is a possible exception, due to the presence of a low G+C bump in its composition. However, the low G+C content in this genome is caused by repetitive elements, and not variation in composition of native genes.

Table 4.2. Genomes and associated RNA-seq data sets used for gene finding.

	genome					RNA-seq		
	assembly	genome size (MB)	assembly size(MB)	unmasked sequence (MB)	no. assembly gaps	source	reads	read count (millions)
Anopheles gambiae	AgamP3	264	273	207	16,817	SRR520428	85 nt, paired	36.9 pair
Aedes aegypti	AegL1	1,310	528	168	12,023	SRR388682	83 nt, single	27.8
Culex quinquefasciatus	CpipJ1	540	528	204	41,472	SRR364516	50 nt, paired	37.2 pair
Anopheles stephensi	Astel1	158	207	147	27,474	SRR643416	84 nt, paired	18.0 pair
Drosophila melanogaster	5.42	172	208	120	8	SRR042297	75 nt, paired	13.6 pair
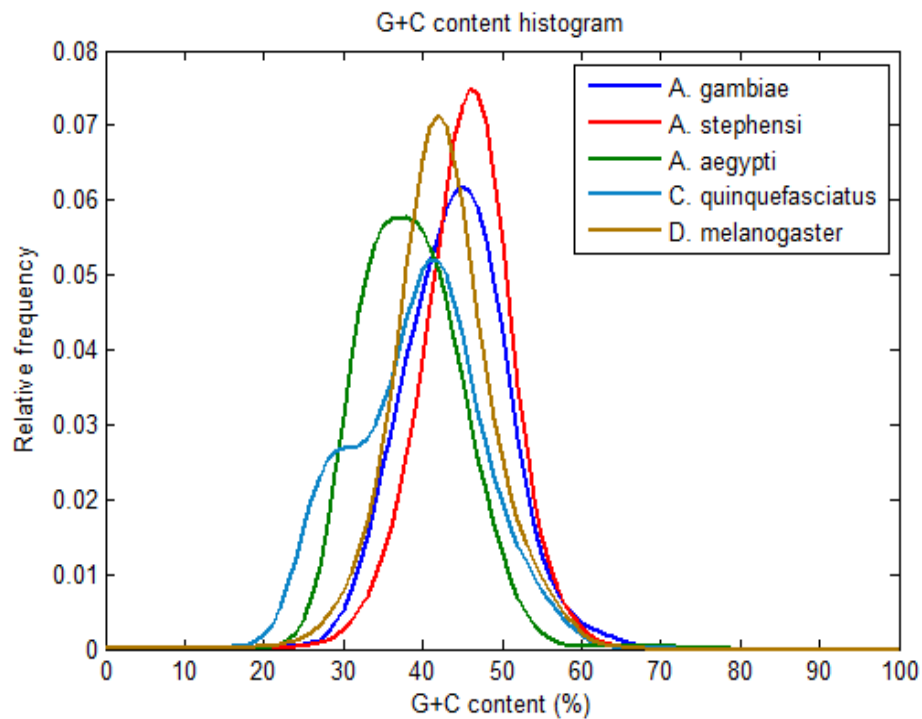
Figure 4.2. G+C content histograms of each genome processed by GeneMark-NGS, made by 2,000 nt non-overlapping intervals of the assembly. Each genome in this group is homogeneous in G+C content, with the exception of *C. quinquefasciatus*, which has a significant mode in a low G+C region.

In each iteration of GeneMark-NGS, a number of predicted exons will be confirmed by introns found by UnSplicer. If the conditional learning is successful, it is expected that the number of confirmed exons will increase with each iteration. Indeed, this is the case as shown in Fig. 4.3. Exons confirmed on just one side or on both sides are categorized. In general, a steady monotonic increase in the number of confirmed exons is achieved with each iteration, until the number levels off. At iteration 2, the second round of intron predictions are provided by UnSplicer. At this point, a sharp increase in the number of confirmed exons is seen in *A. gambiae* and *D. melanogaster*. This trend is also apparent in the amount of coding sequence contributing to training the coding sequence Markov model parameters (Fig. 4.4). The amount of sequence in training levels off at iteration 2 for three of the four genomes. For *A. gambiae*, there is a significant improvement on the last step, suggesting that additional iterations may result in better performance for that genome.
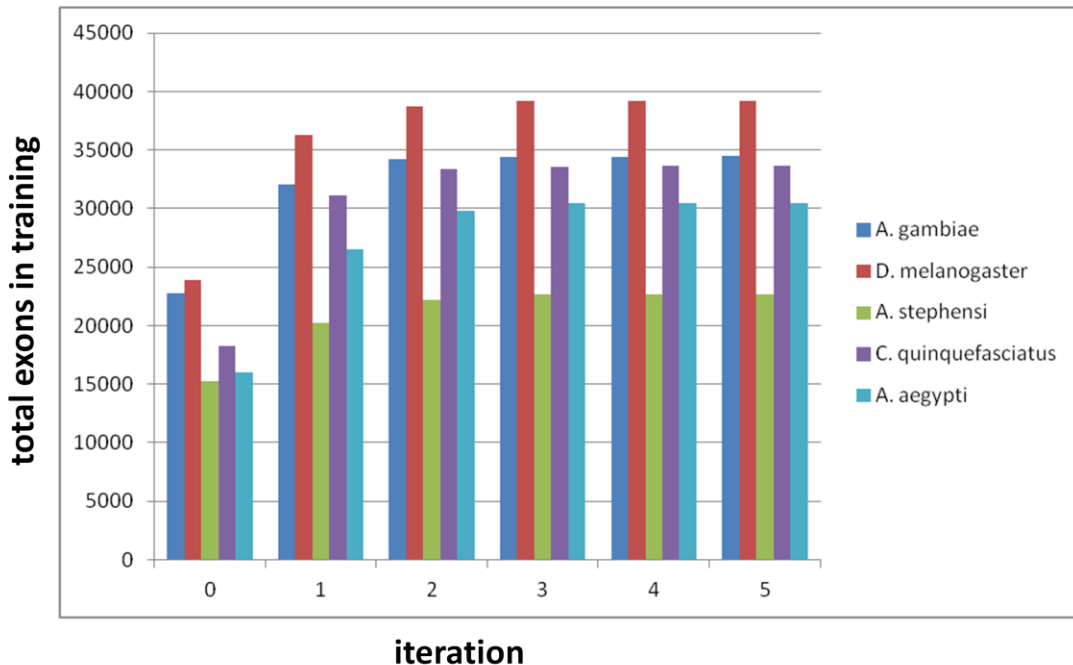
Figure 4.3. The number of exons included in training for each genome, as a function of iteration number.
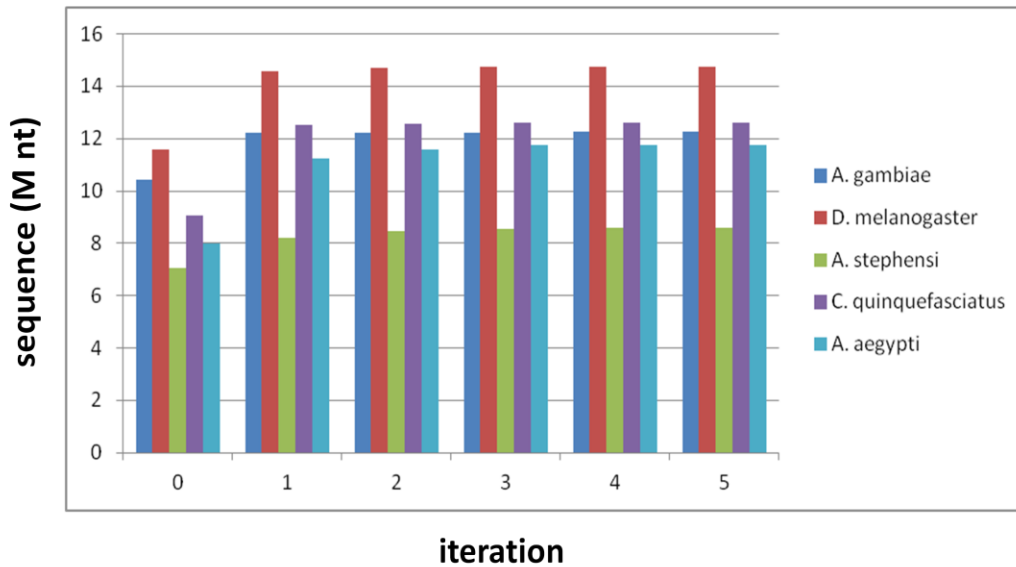
Figure 4.4. The quantity of coding sequence included in training, as a function of iteration.

Gene prediction performance of GeneMark-ES and GeneMark-NGS was compared on test sets for each of the five genomes listed in Table 4.3. Sensitivity (Sn) and specificity (Sp) were calculated for each of eight different gene features: internal exons, gene start (initiation), gene stop (termination), introns, donors, acceptors, and nucleotides. The numbers are shown normalized to percentages. Bold face indicates the better score for each feature. GeneMark-NGS scores higher in both sensitivity and specificity for many features, particularly in *A. gambiae* and C. *quinquefasciatus.* In cases when GeneMark-ES has a higher Sn (or Sp) than GeneMark-NGS, GeneMark-NGS will have higher score for Sp (or Sn). When we compare the two programs using the performance measure (Sn+Sp)/2 (Table 4.4), we see that GeneMark-NGS scores higher in the vast majority of cases. GeneMark-NGS most dramatically outperforms GeneMark-ES in prediction of exon boundaries and gene starts. Improvement in internal exon performance varies from 3.5-13.1%. Internal exon prediction performance improvement is quite dramatic for the 4 VectorBase genomes—ranging from 5.85-16.3% marginal increase in performance.

70

Table 4.3 Gene prediction performance on test sets derived for each genome. Columns: ES = GeneMark-ES, SM = GeneMark-NGS, diff = GeneMark-NGS score – GeneMark-ES score. Sn and Sp are given as percentages.

		D. melanogaster			A. gambiae			A. aegypti			A. stephensi			Culex q.		
		ES	NGS	diff	ES	NGS	diff	ES	NGS	diff	ES	NGS	diff	ES	NGS	diff
internal	Sn	88.3	86.7	-1.6	74.8	77.5	2.7	78.6	82.3	3.7	83.2	84.3	1.1	80.6	87.1	6.5
	Sp	85.4	92	6.6	79.7	89.2	9.5	57.9	87.3	29.4	72.2	84.5	12.3	61	87.1	26.1
intron	Sn	86	87.3	1.3	77.1	80	2.9	74.3	82.5	8.2	84.2	88.1	3.9	73.4	87.5	14.1
	Sp	92.2	91.2	-1	80.7	88.6	7.9	79.3	86.4	7.1	89.6	88.9	-0.7	81	86.2	5.2
donor	Sn	89.3	89.7	0.4	80.9	83.2	2.3	82.2	85.4	3.2	87.5	90.4	2.9	76.6	89.1	12.5
	Sp	89.8	93.5	3.7	84.6	92.1	7.5	76.9	89.2	12.3	87.8	91	3.2	81.7	87.7	6
acceptor	Sn	90.8	89.7	-1.1	81.9	82.8	0.9	84.4	86.7	2.3	90.9	91.6	0.7	92.2	95.3	3.1
	Sp	87.1	92.5	5.4	85.5	91.5	6	77.2	90.6	13.4	85.6	91.8	6.2	77.6	92.4	14.8
all exons	Sn	81	83.8	2.8	73.6	76.9	3.3	67.8	80.4	12.6	73.5	82.2	8.7	73.1	88.5	15.4
	Sp	80.7	85.8	5.1	75.9	82.6	6.7	69.3	82.3	13	75.7	82.3	6.6	76	86.8	10.8
initiation	Sn	64	74.8	10.8	72.3	74.9	2.6	64.1	80	15.9	54.3	74.8	20.5	50	80	30
	Sp	73.6	74.5	0.9	72.3	75.1	2.8	82	79.3	-2.7	77.5	74.8	-2.7	83.3	80	-3.3
termination	Sn	82.4	89.8	7.4	84.9	86.3	1.4	73.6	90	16.4	78.2	91.5	13.3	97.5	100	2.5
	Sp	79.4	86.5	7.1	84.5	86.1	1.6	89.5	89.2	-0.3	87.3	90.3	3	97.5	97.6	0.1
nucleotide	Sn	94.4	92.9	-1.5	89.9	90.5	0.6	96.6	93.5	-3.1	96.2	96.8	0.6	97.8	99	1.2
	Sp	96.1	97.7	1.6	97.4	97.8	0.4	95.7	96.8	1.1	98	98.4	0.4	96.1	97	0.9

CHAPTER V

GENOME ANNOTATION PROJECTS

Over the past several years, the genomes of many important plants and animals have been sequenced and annotated. We have participated in several such projects, notably *Fragaria vesca* (woodland strawberry), *Citrus sinensis* (sweet orange), *Citrus clementina* (clementine orange), and *Rubus idaeus* (raspberry). These genome projects are more complex compared to genomes of model organisms which in the past have been the focus of gene prediction algorithms (Lomsadze, 2005).

A major challenge for gene finding in these novel genomes is detection of repetitive elements in the DNA sequence. The presence of repetitive elements interferes with algorithms for finding native protein coding genes due to their high copy number and contamination of the sequence with pseudogenes. Most repetitive sequence in eukaryotes is composed of transposable elements (TEs) (Feschotte 2009). In plant genomes, most TEs use a retroviral mechanism to introduce new copies in the genome. These elements are called retrotransposons, e.g. frequent in plant long terminal repeat (LTR) retrotransposons. The presence of TEs can be disruptive to gene finding algorithms such as GeneMark-ES because high copy number TEs can bias GHMM model parameters found by unsupervised training. This has the effect of reducing the performance of the GHMM for native gene finding. TEs are known to be responsible for a tremendous amount of DNA remodeling and rapid genome evolution. Indeed, while comparative genomics is useful for finding native protein coding genes, it is not a very effective method for finding TEs due to their rapid evolution. This places much of the burden of finding TEs in novel genomes on *de novo* algorithms. Many such *de novo* programs have been developed to find TEs in genomes, such as RepeatScout (Price 2005) and PILER (Edgar 2005). Programs which attempt to find LTR retrotransposons using

basic structural knowledge have also been developed, such as LTR_STRUC (McCarthy 2003), LTR_FINDER (Xu 2007), and LTRharvest (Ellinghaus 2008). Each of these programs can be used to develop libraries of consensus sequences, which can then be augmented with TE sequences found in closely related species (available at Repbase). The combined library may then be used to mask the DNA sequence for gene finding with GeneMark-ES.

Fragaria vesca (**woodland strawberry**)

For gene prediction in the *F. vesca* project, a combined approach to repeat masking was adopted. The structural LTR detection program LTR_STRUC was used in addition to RepeatScout. The output sequences from these programs were combined with a collection of TE sequences from Repbase taken from the rosaceae clade. All combined, the collection masked 31.8% of the *F. vesca* assembly (68 MB out of 218 MB assembled).

After masking, the program GeneMark-ES was used to predict genes. The *F. vesca* genome is quite homogeneous in G+C content (much like *A. thaliana*), making it a good candidate for the machine learning algorithm used by GeneMark-ES. Next, the alignment of available EST sequences was performed using BLAT. Alignments were clustered and gene stops and starts predicted on mapped transcript sequences, as described in the modeling pipeline in GeneMark-HB+ (Appendix D). This produced a reliable test set of 633 genes to be used for assessment of gene prediction accuracy. The prediction performance on the test set is shown in Table 5.1.

A team at the University of Georgia (UGA) (Wang, Bennetzen) also performed work attempting to identify LTR retrotransposons in *F. vesca*. We therefore developed a pipeline to find the best set of gene predictions combining the UGA repeat annotations with our own. In Fig. 5.1, the process is illustrated with the pipelines labeled "GT" and "UGA. The idea is to mask the DNA sequence with both libraries repeats (separately), and recover genes which are masked by one pipeline but not the other. In the example

shown in Fig. 5.1, there are 4 such genes. Each of the 4 genes are aligned to protein sequences in the non-redundant (NR) protein database (from NCBI) using BLASTp. A gene will be recovered if has at least one significant hit in the alignment, and that hit satisfies two conditions 1/ not labeled as a hypothetical gene, and 2/ not labeled as reverse transcriptase (commonly found in LTR retrotransposons).
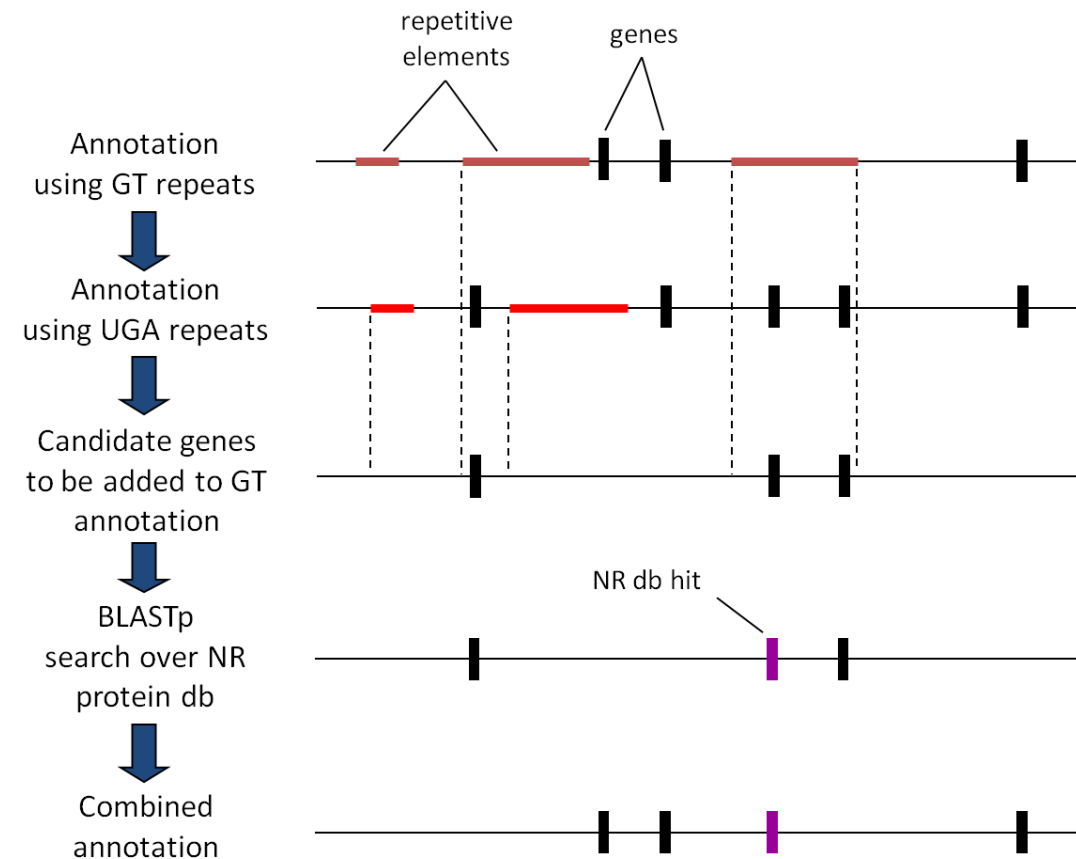
Figure 5.1. Combining two sets of repetitive sequence annotations (GT and UGA) to form a set which minimizes overlap with predicted native genes.

A test set consisting of 666 complete genes was constructed by alignment of 47,230 EST sequences to the genome using BLAT. Gene prediction performance on the test set was evaluated using GHMM parameters found by GeneMark-ES on the repeat masked sequence. The performance (shown in Table 5.1) is specified for individual gene features: exons (initial, terminal, and internal), introns, and donor and acceptor boundaries. Overall nucleotide prediction accuracy is shown as well. The prediction

performance is quite good for all features—in particular we found over 90% Sn and Sp

for donor and acceptor site preditions.

Table 5.1. Gene prediction performance on *F. vesca* test set consisting of 633 genes.

Feature	Sensitivity (%)	Specificity (%)
internal exons	88.4	88.0
introns	85.8	86.5
donor	90.7	91.4
acceptor	90.7	91.2
all exons	83.9	84.4
initial exons	81.8	82.4
terminal exons	82.9	82.7
nucleotide	96.7	94.9

A procedure for gene prediction was established based on earlier unpublished

work (Lomsadze, Ter-Hovhannisyan, Borodovsky). The algorithm, called GeneMark-

ES+, is similar to the method of incorporating external information for gene prediction

used by Augustus. In this method, high quality EST alignments to the genome are given a

very high likelihood score (see Fig. 5.2). The Viterbi parse of the DNA sequence by the

GHMM will select exon-intron boundaries confirmed by alignment with high likelihood.

The resulting parse is sometimes called a conditional maximum likelihood parse. This

procedure resulted in 34,809 genes predicted on the 214 MB genome. There were 5915

single exon genes predicted and an average of 5.0 exons per gene (4.0 introns per gene).

The average gene length was 1160 nt and the average intron length was 407 nt.
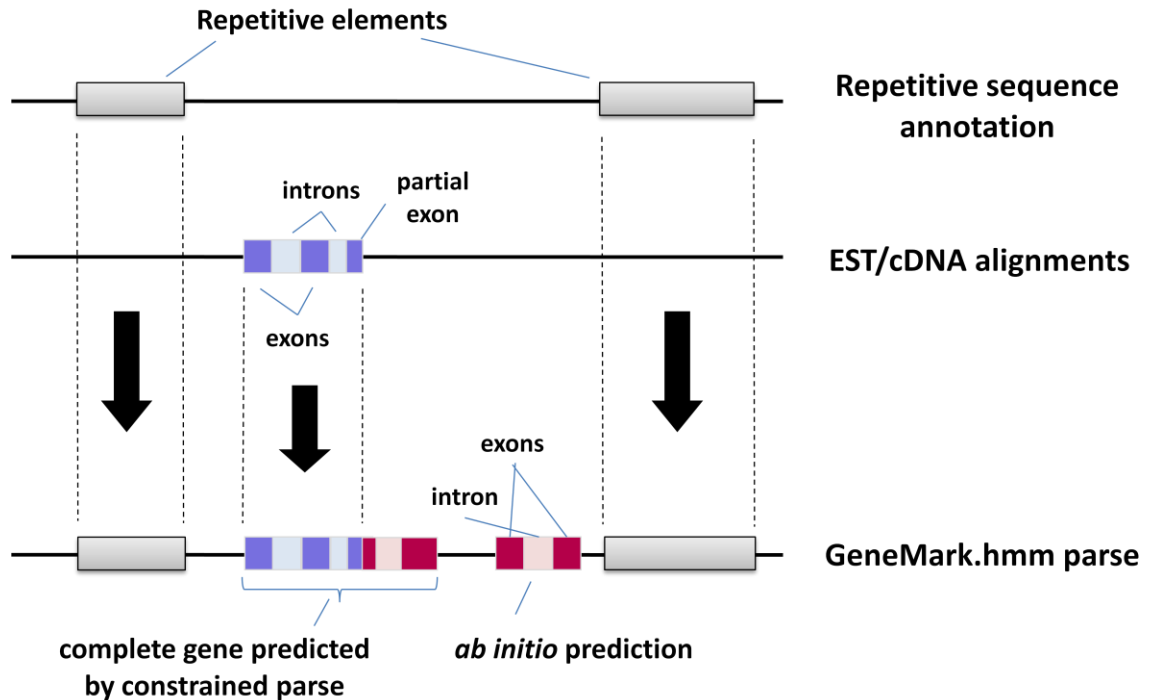
Figure 5.2. Prediction step by GeneMark-ES+. The output parse is constrained to predict features found from EST alignments, thereby producing a conditional maximum likelihood parse.

Rubus idaeus (**raspberry**)

The genome of *Rubus ideaus* (raspberry) of the Heritage variety was sequenced with NGS technology in 2011. Assembly of the genome was a challenge due to a very high degree of heterozygous composition of the chromosomes (the plant is a diploid). Gene finding on the raspberry assembly sequences involved some trial and error, since it was not known what kind of assembly would be optimal for gene prediction. An assembly which was more faithful to an individual haplotype was highly fragmented, making unsupervised training less accurate. A merged genome assembly was preferred, since it resulted in a much larger number of long contigs which could be used for unsupervised training. The gene sequences in the "merged" assembly were not necessarily faithful to a haplotype, but the gene predictions are more reliable.

The GeneMark-HB+ pipeline was used to predict genes for this assembly (Appendix D). 352 complete gene structures were found by alignment of EST sequences,

76

using the modeling pipeline in GeneMark-HB+. 175 (about half) genes were used as a development set for semi-discriminative training to find the optimal blending parameter for construction of the final parameters used for gene prediction. The accuracy of the final model is quite good—about 90% Sn and Sp predicting donor sites, 91% for acceptor sites, and 1-2% better than GeneMark-ES for predicting internal exons (Fig. D.5, panel b). Prediction of initial exons is significantly better compared to GeneMark-ES.

APPENDIX A

CODON USAGE PATTERNS IN HETEROGENEOUS

EUKARYOTIC GENOMES

The concept of heuristic model parameters was proposed and developed for prokaryotic genomes in 1999 (Besemer, 1999). The main idea is to derive a set of universal parameters (depending on G+C content only) which can be used to discriminate CDS from non-coding sequence. Such parameters have application in metagenomic gene finding, for example. In eukaryotes, a similar concept is useful for detection of protein coding splice junctions and gene finding in heterogeneous genomes.

Codon Usage

Prokaryotic genomes are different from eukaryotic genomes in many ways, but a difference we focus on here is that prokaryotic genomes are almost all homogeneous in G+C content, while only some eukaryotic genomes are homogeneous. Besemer analyzed genes in 17 prokaryotic genomes having whole genome G+C content collectively over a broad range. All genes in a genome were analyzed to determine the mean codon usage (distribution of codon frequency for each of the 63 coding codons). This distribution defines a codon usage at a specific G+C level. Due to heterogeneous G+C content, this codon usage in eukaryotes must be calculated differently.

We define the G+C level of a eukaryotic gene as the percentage of G+C nucleotides in the genome in the interval from the start codon to the stop codon. This includes intron sequences. Whenever multiple isoforms are annotated at a locus, the one having the longest CDS length is selected as the representative isoform. All genes are assigned to a G+C bin, varying from 20-70% in increments of 1%. Mean codon usage is then calculated for each bin, according to the genes assigned to that bin. For

78

heterogeneous genomes, this results in a set of codon usage distributions (one for each G+C bin) over a range of G+C bins found in that genome. In order to capture additional genomic diversity, and broaden the range of observed G+C, four (4) different genomes were considered: human, *A. mellifera* (honey bee), *B. distachyon* (grass), and *O. sativus* (rice). A histogram of the number of genes in each genome found in each G+C bin is shown in Fig. A1. Codon usage for each of 63 codons as a function of G+C content is shown in Figs. A1-A16. Mean codon usage derived from each genome is shown as a dot of the same color (refer to the legend in the graphs).
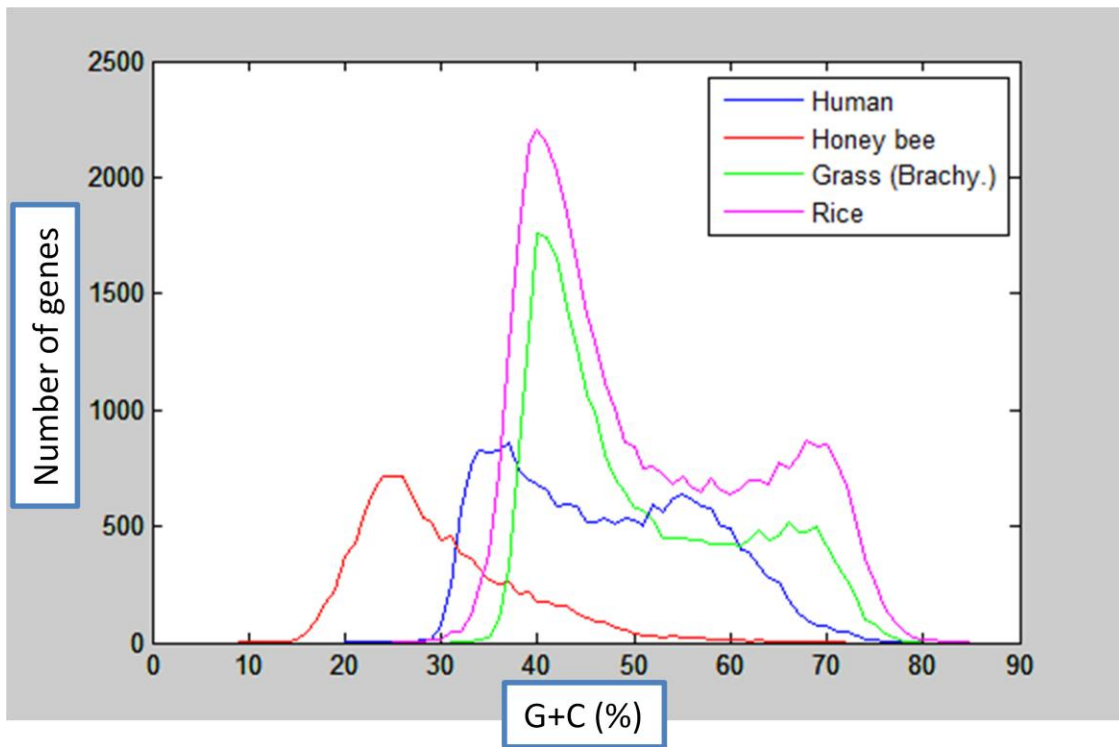
Figure A.1. The number of genes found in each G+C bin from each of the four genomes analyzed.

Several features are noteworthy. First of all, the honey bee genome has a very low G+C content in general, and all annotated genes have a calculated G+C value from 20-30%. Other genomes have few genes in those bins. Second, the relationship between codon frequency and G+C is clearly nonlinear in most cases. A third degree polynomial fit to all the shown data (by least-squares with equal weight) is shown in each graph as a

black dashed line. Third, some codons have usage relationships which are well conserved over all four genomes while others have very notable differences. For example, the lysine codon AAA (Fig. A.2) and isoleucine codon ATT (Fig. A.5) are closely patterned in all genomes. However, consider the arginine codons CGA and CGT (Fig. A.8). In these examples, not only is the variation in usage spread significantly, but in honey bee an increase in usage of these codons is seen with an increase in G+C (as expected), while a *decreasing trend* is seen in all other genomes with increasing G+C. Both the values of codon usage and the trend differ among the genomes in those examples.

Eukaryotic heuristic parameters

Heuristic coding sequence Markov model parameters are found by using a least-squares curve fit (3rd degree polynomials) to determine codon usage frequency for all coding codons. A 6-mer frequency table is then derived by assuming adjacent codons are independent. 5th order Markov model parameters are derived from the 6-mer frequency table inside GeneMark.hmm. For each G+C content bin, frequencies for each of 64 codons (where pseudocounts are assigned in all cases where stop codons occur) are multiplied with all other codon frequencies to yield 64^2=4096 6-mer frequencies for each frequency bin. As a final step, each set of 4096 frequencies are normalized to sum to 1 to form the parameters for each G+C bin.
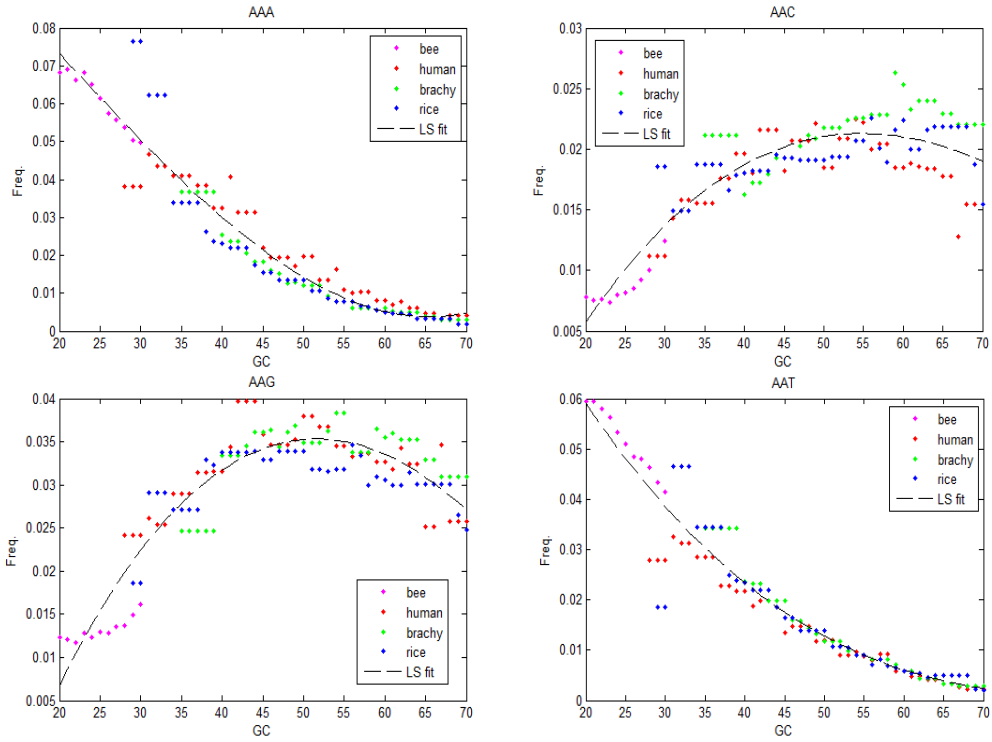
Figure A.2. Codon usage patterns for AAA, AAC, AAG, AAT.
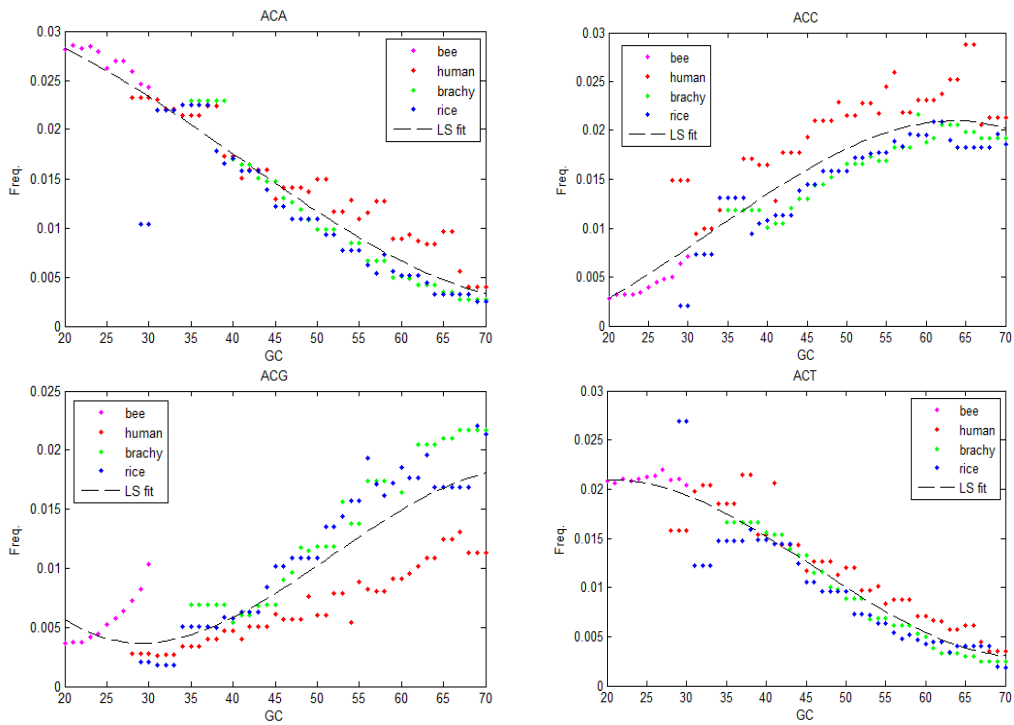
Figure A.3. Codon usage patterns for ACA, ACC, ACG, ACT.
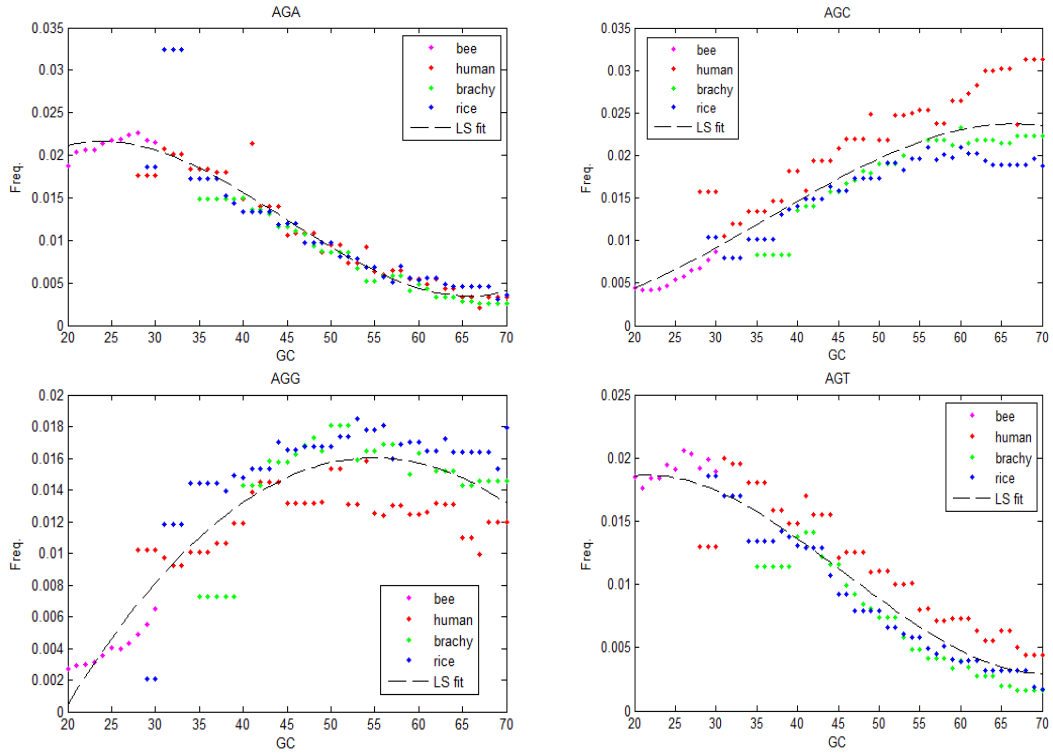
Figure A.4. Codon usage patterns for AGA, AGC, AGG, AGT.
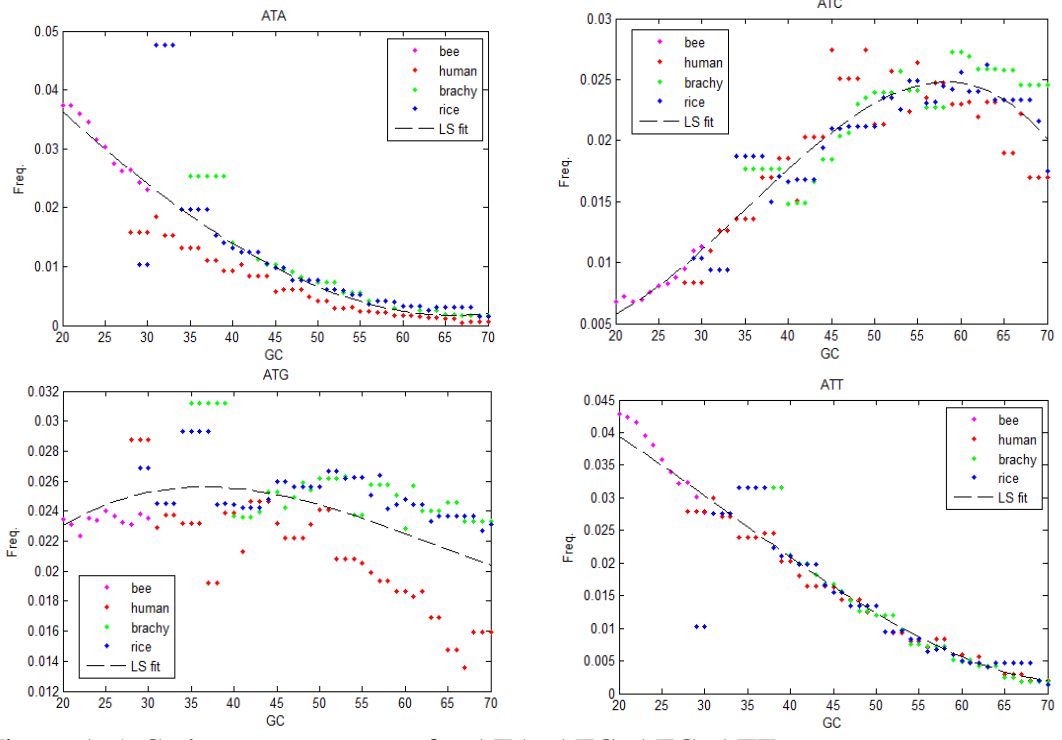
Figure A.5. Codon usage patterns for ATA, ATC, ATG, ATT.
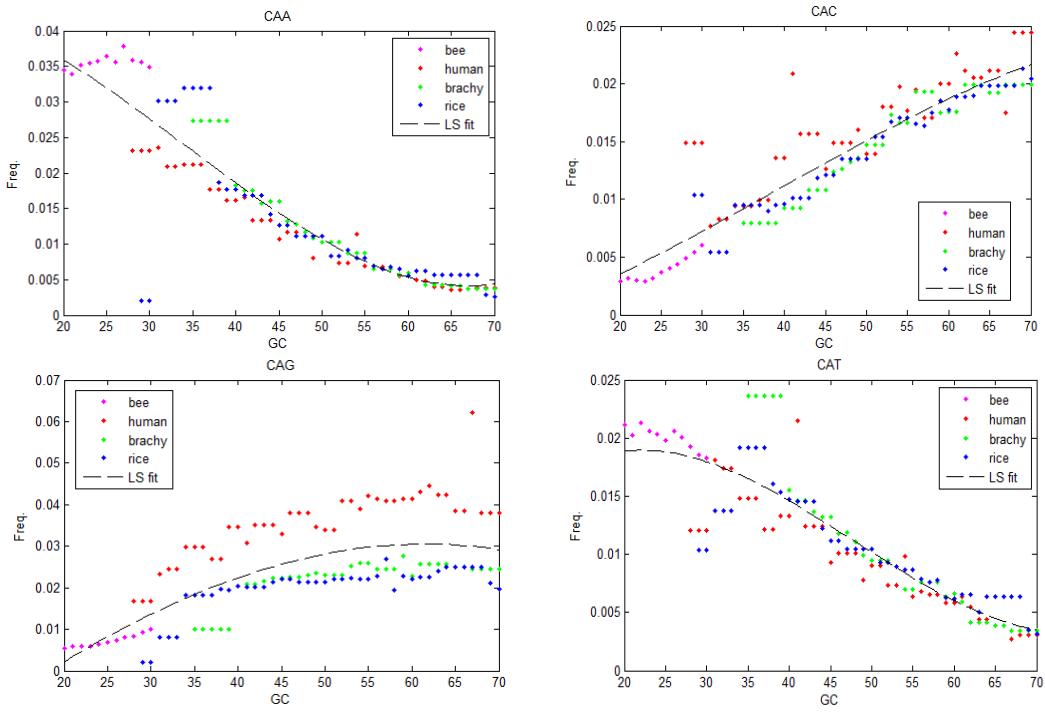
Figure A.6. Codon usage patterns for CAA, CAC, CAG, CAT.
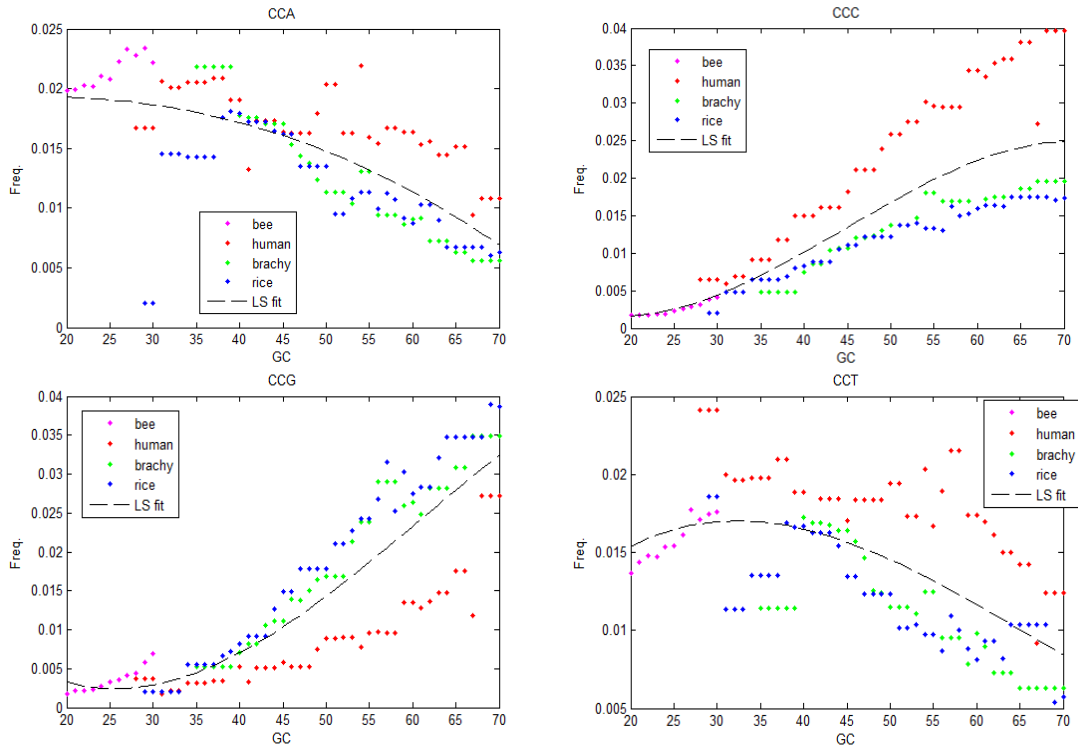
Figure A.7. Codon usage patterns for CCA, CCC, CCG, CCT.
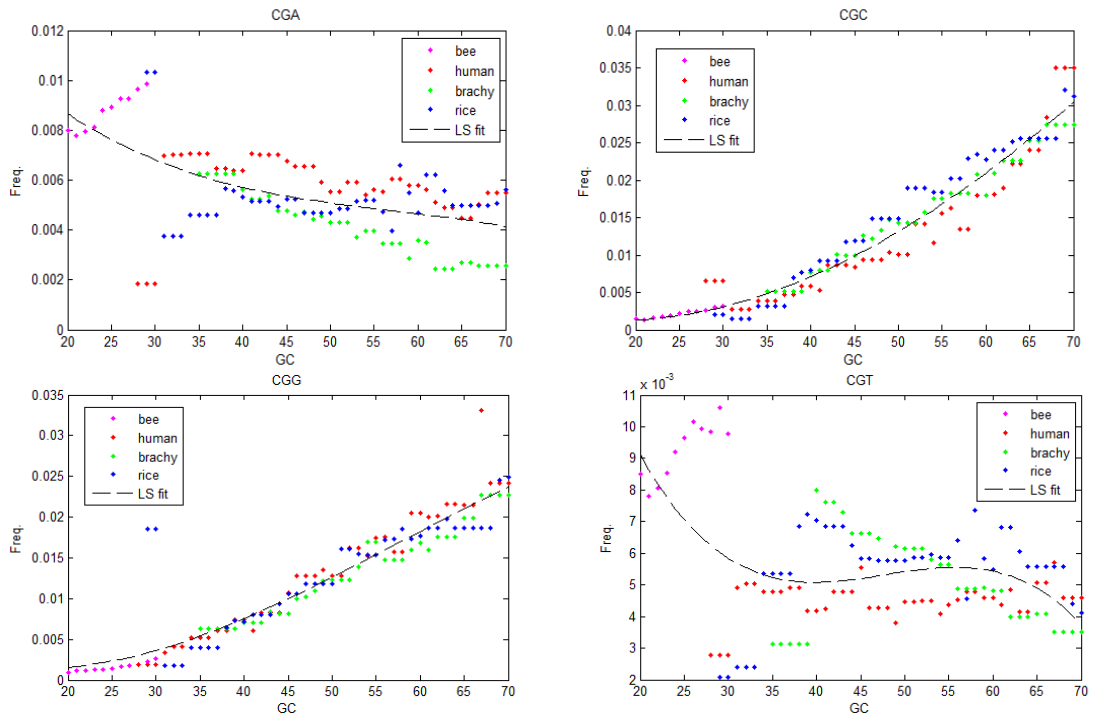
Figure A.8. Codon usage patterns for CGA, CGC, CGG, CGT (arginine codons).

84
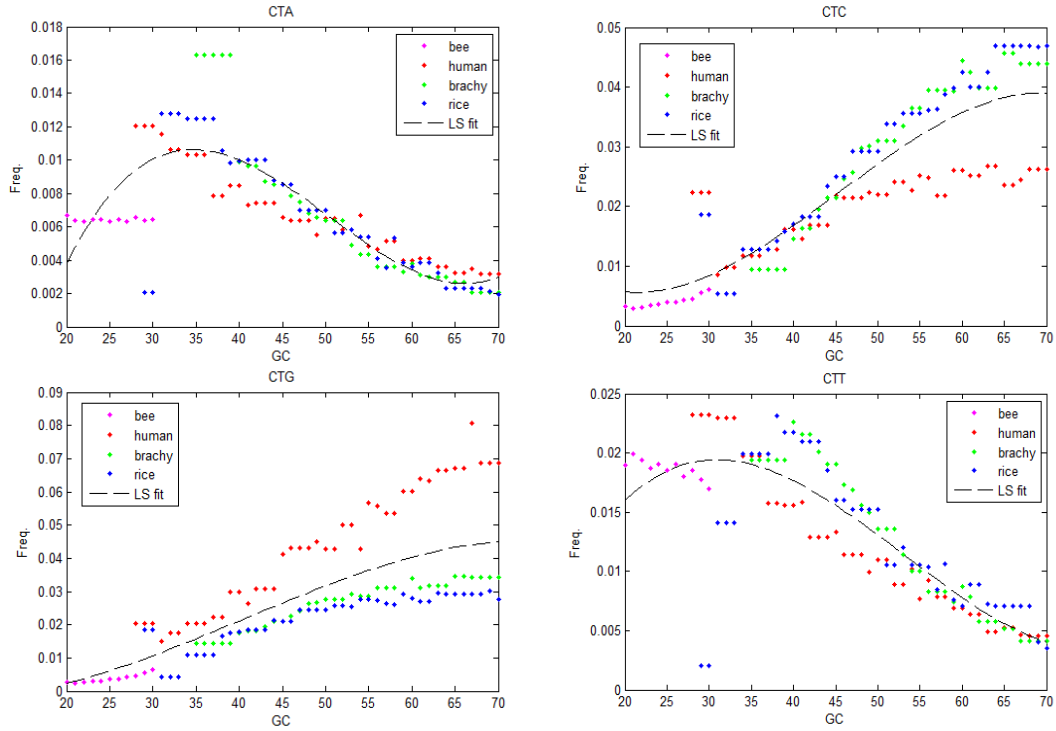
Figure A.9. Codon usage patterns for CTA, CTC, CTG, CTT.
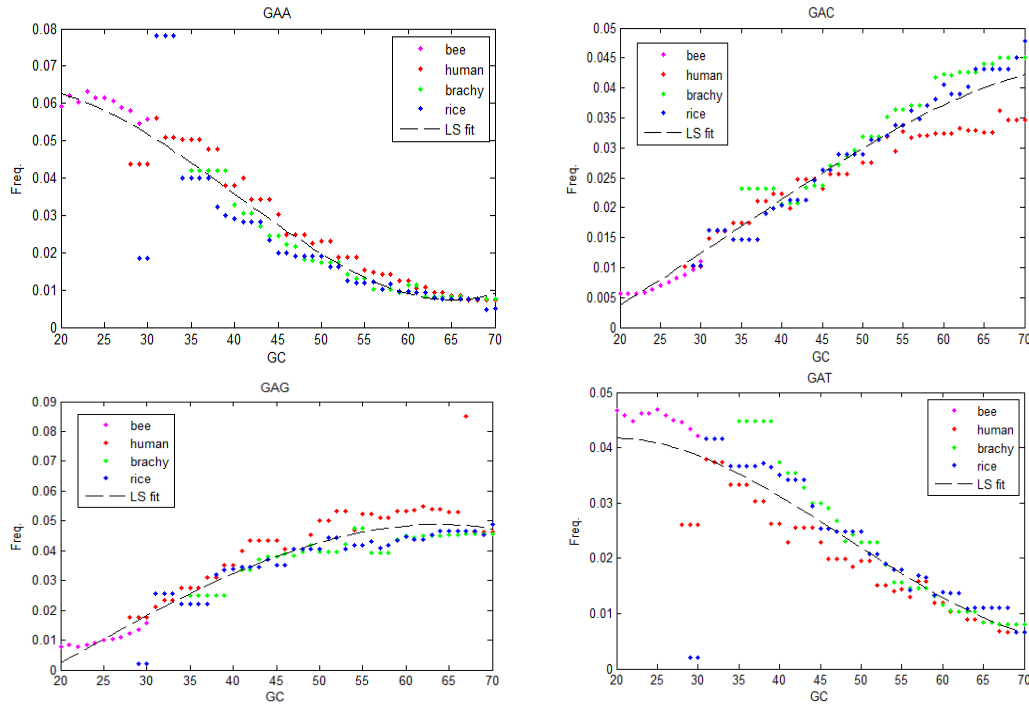
Figure A.10. Codon usage patterns for GAA, GAC, GAG, GAT.
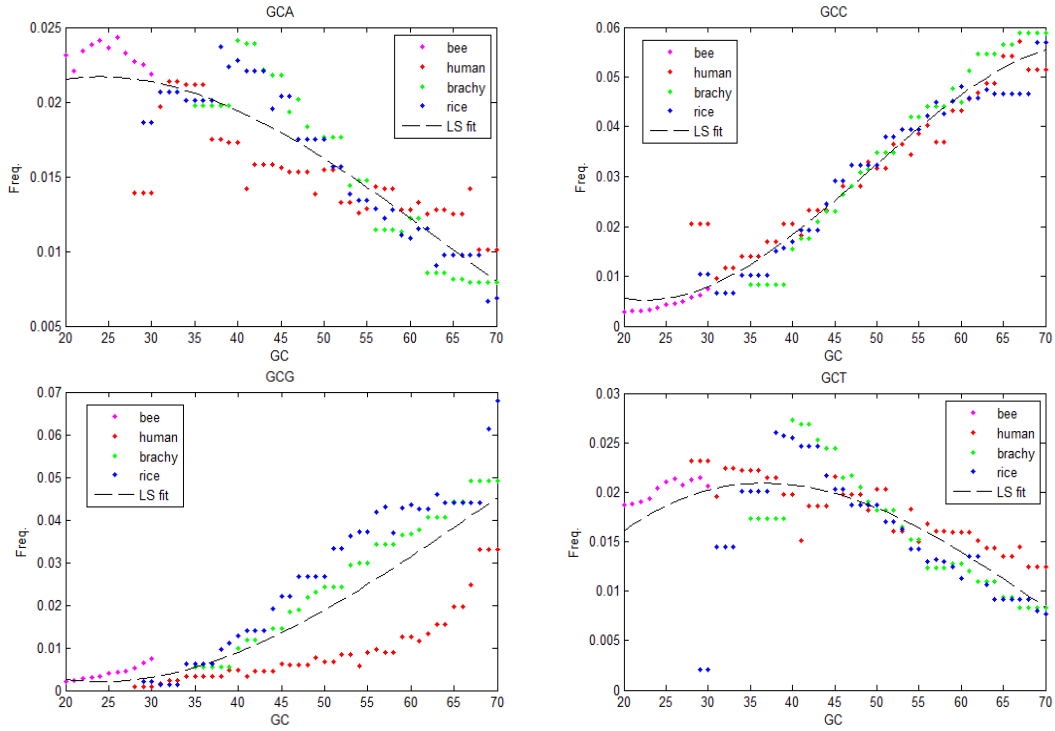
Figure A.11. Codon usage patterns for GCA, GCC, GCG, GCT.
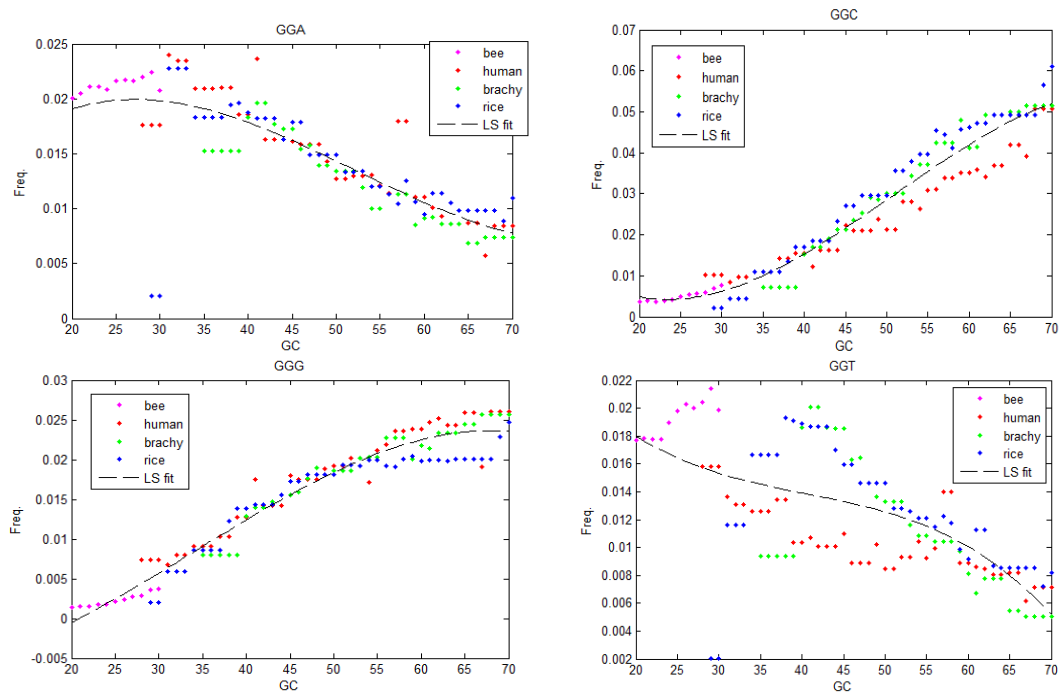
Figure A.12. Codon usage patterns for GGC, GGC, GGG, GGT.
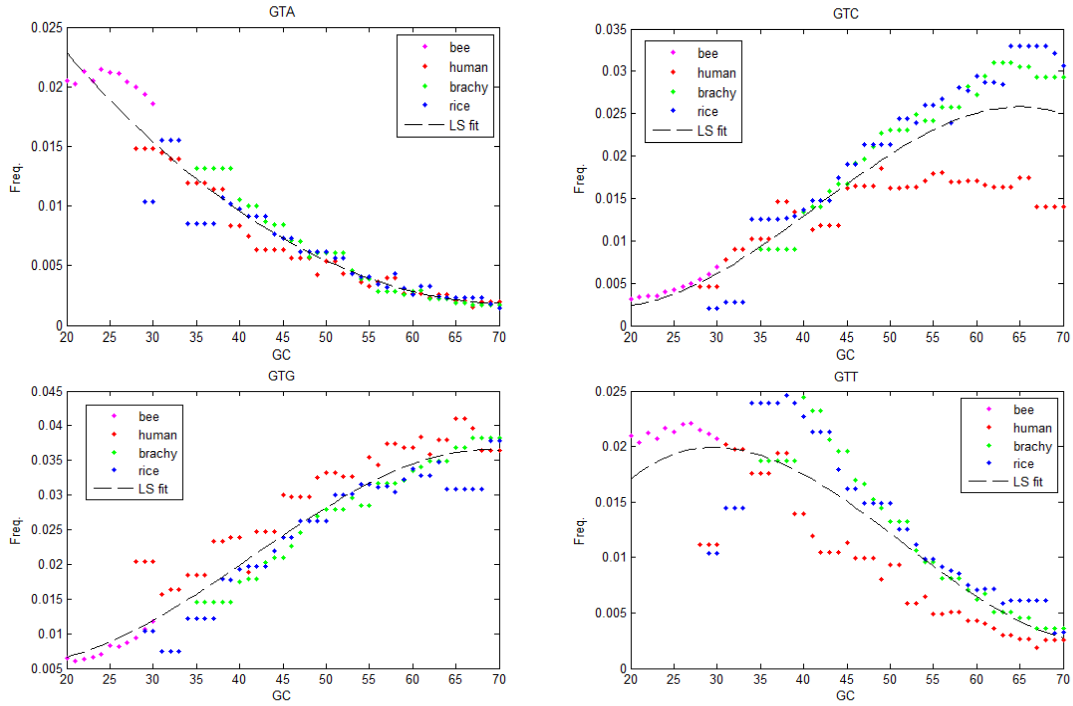
Figure A.13. Codon usage patterns for GTA, GTC, GTG, GTT.
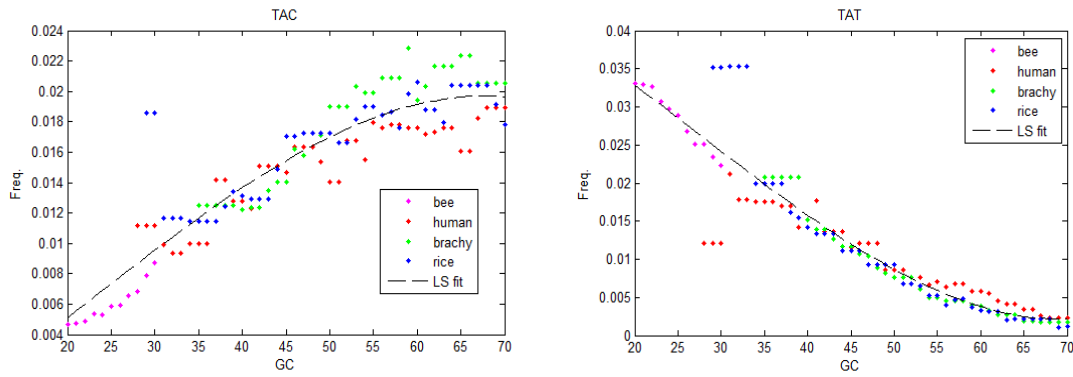
Figure A.14. Codon usage patterns for TAC and TAT (TAA and TAG are stop codons).
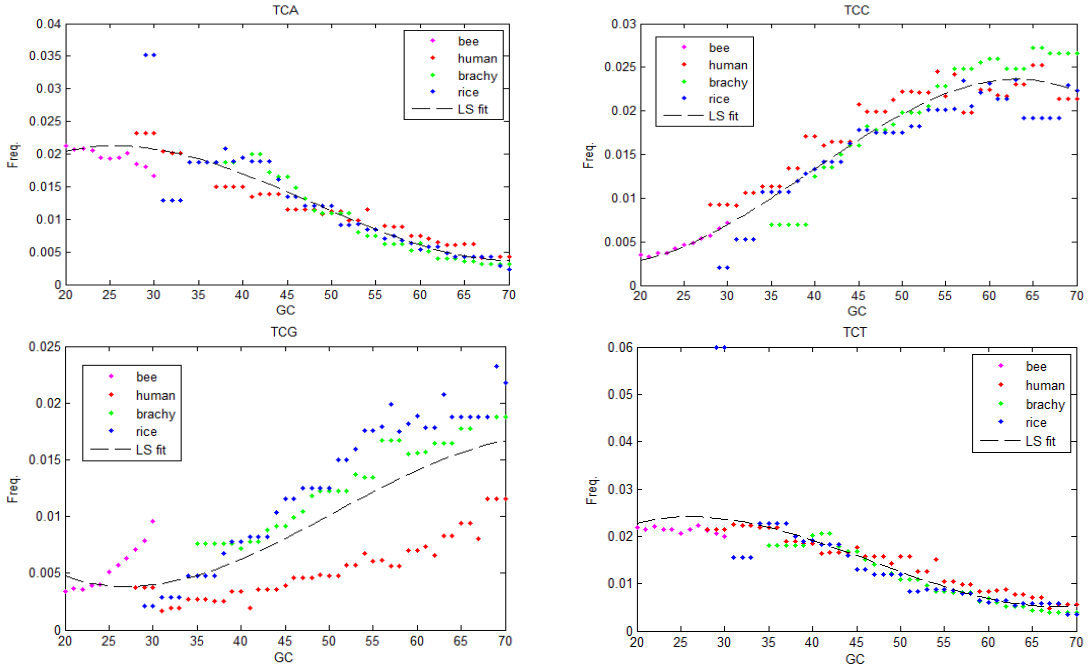
Figure A.15. Codon usage patterns for TCA, TCC, TCG, TCT.
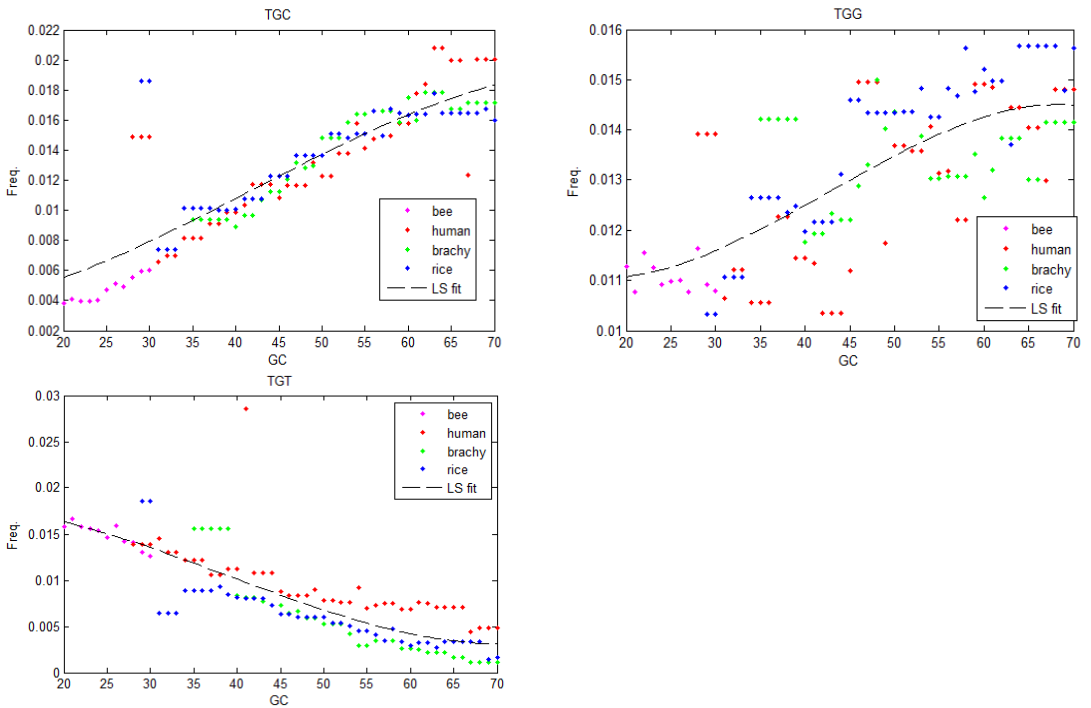
Figure A.16. Codon usage patterns for TGC, TGG, TGT (TGA is a stop codon).
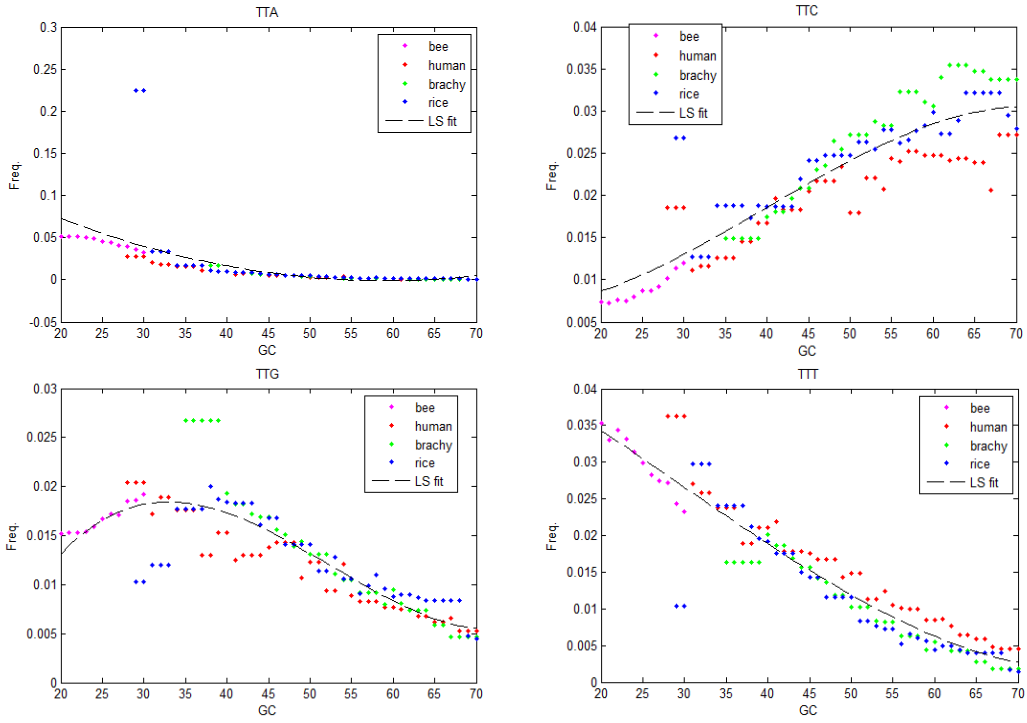
Figure A.17. Codon usage patterns for TTA, TTC, TTG, TTT.

APPENDIX B

CODING POTENTIAL FEATURE ANALYSIS IN SEQSWEEP

There are many reasonable options in selection of a feature derived from intron flanking sequence. It is desirable to account for the 3-periodic nature of the coding sequence, and the fact that the intron can interrupt a codon in any of 3 positions. A comparison of several candidate test statistics was performed on SJ predictions made by TopHat on four different RNA-seq data sets. The statistics (and detectors) are listed in Fig. B.1. Upstream (5') and downstream (3') sequences flanking an intron (of length L) are denoted by by $x^{5'}$ and $x^{3'}$, respectively. Sequences in the interior of the intron, but at the 5' and 3' ends are labeled $y^{5'}$ and $y^{3'}$, respectively. Each of these sequences also has length N (equal to 96 nt in examples here). cod_i is the coding sequence model starting in frame i, and non is a non-coding sequence model. cod (without a subscript) is a frameless coding sequence model, formed by a weighted average of models in all three frames such that frame 1 is given 50% weight and frames 2 and 3 are weighted 25% each. Prior probabilities P_{cod} and P_{non} (in statistic 5) are set to 0.5 by default.
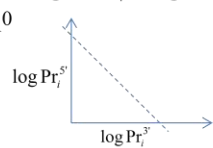
Statistic:	Detector:

1. $LR_{Ma/Li}^{no\,frame} = \dfrac{p(x^{5'}x^{3'}\mid cod)\,p(y^{5'}y^{3'}\mid non)}{p(x^{5'}x^{3'}\mid non)\,p(y^{5'}y^{3'}\mid cod)}$

$d = \begin{cases} 1 & \log LR_{Ma/Li}^{no\,frame} > t \\ 0 \end{cases}$

2. $LR_{Ma/Li}^{3\,frame} = \arg\max_i \dfrac{p(x^{5'}x^{3'}\mid cod_i)\,p(y^{5'}y^{3'}\mid non)}{p(x^{5'}x^{3'}\mid non)\,p(y^{5'}y^{3'}\mid cod_i)}$

$d = \begin{cases} 1 & \arg\max LR_i^{Ma/Li5'} = \arg\max LR_i^{Ma/Li3'},\ \log LR_{Ma/Li}^{3\,frame} > t \\ 0 \end{cases}$

3. $LR_{Bor/Burns} = \arg\max_i \left(\dfrac{p(x^{5'}x^{3'}\mid cod_i)}{p(x^{5'}x^{3'}\mid non)}\right)^{1/N} \left(\dfrac{p(y\mid non)}{p(y\mid cod_i)}\right)^{1/L}$

$d = \begin{cases} 1 & \arg\max LR_i^{5'} = \arg\max LR_i^{3'},\ \log LR_{Bor/Burns} > t \\ 0 \end{cases}$

N = flank seq length (=96 in these examples)
L = intron length

4. $LR_{flank}^{5'} = \dfrac{p(x^{5'}\mid cod_i)}{p(x^{5'}\mid non)} \qquad LR_{flank}^{3'} = \dfrac{p(x^{3'}\mid cod_i)}{p(x^{3'}\mid non)}$

$d = \begin{cases} 1 & \arg\max LR_i^{5'} = \arg\max LR_i^{3'},\ \log LR_i^{5'} + \log LR_i^{3'} > t \\ 0 \end{cases}$

5. $\Pr_i^{5'} = \dfrac{p(x^{5'}\mid cod_i)P_{cod}}{\left(\sum_i p(x^{5'}\mid cod_i)\right)P_{cod} + p(x^{5'}\mid non)P_{non}}$

$\Pr_i^{3'} = \dfrac{p(x^{3'}\mid cod_i)P_{cod}}{\left(\sum_i p(x^{3'}\mid cod_i)\right)P_{cod} + p(x^{3'}\mid non)P_{non}}$

$d = \begin{cases} 1 & \arg\max \Pr_i^{5'} = \arg\max \Pr_i^{3'},\ \log \Pr_i^{5'} + \log \Pr_i^{3'} > t \\ 0 \end{cases}$

Figure B.1. Five test statistics (and associated detectors) which were compared for effectiveness in false positive filtering of TopHat SJ predictions. A description of the notation is provided in the text. A detection value of $d = 1$ indicates a SJ confirmation, otherwise it is a false positive.

The coding and non-coding sequence models were taken from the eukaryotic heuristic models described in another chapter. Since these models vary with G+C content, appropriate models must be selected for each candidate intron. Here we calculate G+C by counting the number of 'C' and 'G' nucleotides in the 1Kb sequence upstream and 1Kb downstream of the intron, and divide by 2,000. Then converting the ratio to a percentage, and rounding to the nearest whole 1%, the appropriate model parameters are selected.

annotated
introns

introns predicted
by alignment

T_p

F_p

ΔT_p

ΔF_p

shaded: introns
removed by filtering
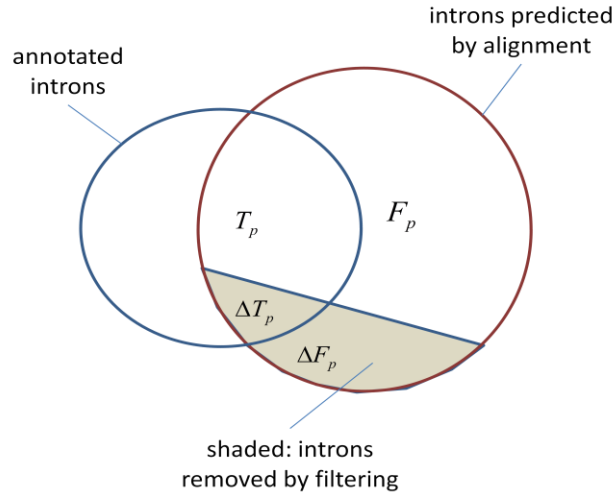
Figure B.2. Venn diagram picturing the quantities involved in evaluation of marginal improvement.

The likelihood ratio $\mathrm{LR}_{\mathrm{Ma/Li}}^{\mathrm{no\ frame}}$ is the same as the coding potential feature used in the program TrueSight (discussed in the next section). The statistic $\mathrm{LR}_{\mathrm{Ma/Li}}^{3\,\mathrm{frame}}$ generalizes it to a frame sensitive model. All remaining statistics were concepts evaluated for possible replacement of the Ma/Li coding potential statistic.

Each of these detectors was evaluated on sets of SJ predictions made by TopHat. A measure of filtering performance called marginal improvement is defined (refer to the Venn diagram in Fig. B.2). The SJ predictions are comprised of some number of true positives $\mathbf{T_p}$ (agreement with annotation) and false positives $\mathbf{F_p}$ (not annotated). The process of filtering will remove some number in each set: $\mathbf{\Delta T_p}$ and $\mathbf{\Delta F_p}$ (shaded region). For a given detection threshold, we assess the filtering performance by monitoring the ratios $\mathbf{\Delta T_p/T_p}$ and $\mathbf{\Delta F_p/F_p}$ which together we call marginal improvement. As the detection threshold is varied, we trace out marginal improvement ROC curves (Fig. B.3). The ideal point on these plots is the lower right-hand corner (filtering all false positives, and no filtering true positives). The best statistic in terms of marginal improvement is arguably the $\mathbf{LR_{Bor/Burns}}$ statistic, which uses likelihood ratios of sequences flanking the

92

intron as well as the full intron sequence itself. However, in every example this detector removes about 20% of true positives, even with accommodating threshold levels. The reason for this is probably due to filtering out true UTR introns, which have no coding sequence flanking them. However, the Ma/Li detector (with no frame) does not have this strong filtering effect on UTR introns, probably because the statistic has weaker discrimination power. Unfortunately, a 20% reduction in sensitivity will make this filtering procedure uncompetitive with other methods. Based on this result, we tested a classification method in which candidate UTR and coding introns were attempted to be separated, so that different classifiers could be applied to each group.
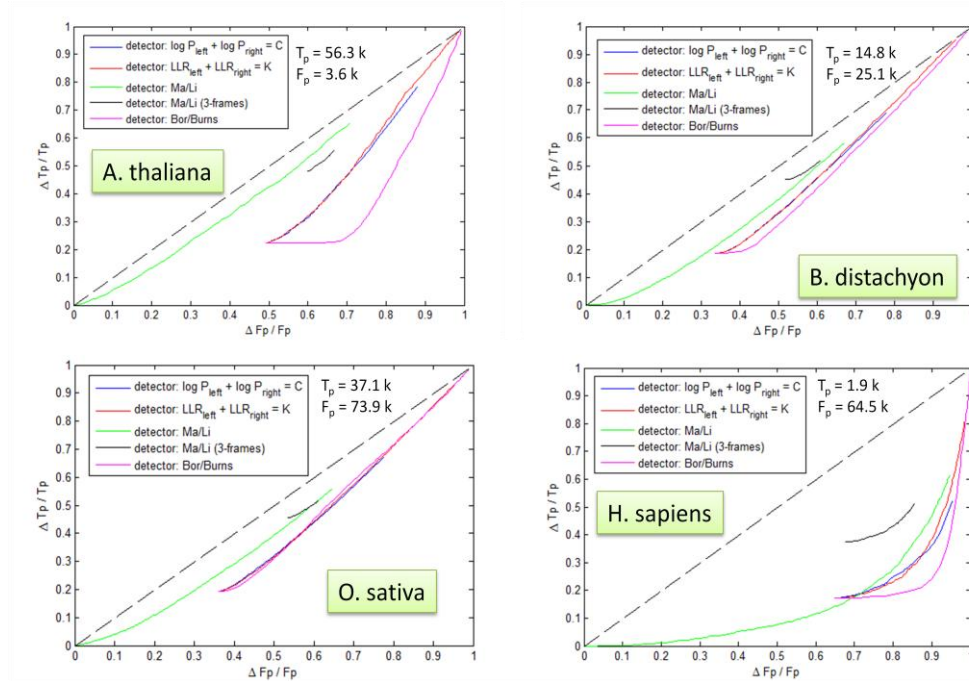
Figure B.3. Marginal improvement ROC curves of five detectors on four data sets. The accuracy of the SJ predictions by TopHat vary tremendously due to differences in the number of reads, read length, and sequencing technology used.

APPENDIX C

DEFINITION OF FEATURES USED FOR SPLICE JUNCTION

CLASSIFICATION IN UNSPLICER

UnSplicer calculates nine features for each candidate splice junction: five alignment features, and four sequence derived features. Some of the UnSplicer features have also been described in the TrueSight publication (Li, 2013).

Alignment depth score

The alignment depth D is the number of reads aligned across the splice junction. This number also includes the reads aligned across the junction in the second attempt, with the initially predicted intron spliced out. The alignment depth score is equal to $\log(D)$.

Max shortest overhang length

Let a read of length L be aligned across a splice junction so that n nucleotides are aligned on the upstream to the junction and L-n are aligned downstream. The length of the shorter side of the alignment equals $\min[n, L - n]$. We are concerned about reliability of splice junctions that have consistently short shorter overhangs regardless of the side. Let say a junction has D alignments, with overhang lengths n_i on the 5' end and $L - n_i$, on the 3' end, with $i = \{1, ..., D\}$. We define the max shortest overhang length as

$$\text{overhang} = \max_i\big(\min(n_i, L - n_i)\big).$$

The gapped alignment pipeline has a hard-coded minimum threshold of 8nt for overhang for any splice junction candidate; this feature is defined as:

$$\text{overhang score} = |8 - \text{overhang}|$$

Entropy

A read which spans a splice junction may be split by the splice junction at some position i (i nt aligned on the 5' side of SJ and $L - i$ nt on the 3' side of SJ, see Fig. C.1). Considering a population of D reads aligned across a given SJ, we construct an empirical

94

distribution of frequencies $f_i = n_i/D$, where n_i is the number of reads split at position i.

The entropy is then determined by

$$H = \sum_{i=1}^{L} f_i \log f_i$$

read position i

Figure C.1. Alignment of a read of length L to genomic DNA is split over an intron bordered by two exons. The read is split at positions i and i+1, where $1 \leq i < L$. Position i is aligned to p-1 and position i+1 is aligned to q+1.

Coverage skew

The coverage skew is a measure of the difference in RNA expression level of the DNA sequence flanking a putative intron and of the sequence in the intron interior. A diagram illustrating the calculation of coverage skew is shown in Figure C.2. A count of full length alignments is tallied in four intervals: upstream and downstream of p (the 5' end of putative intron), and intervals upstream and downstream of q (the 3' end). The coverage skew score is simply the difference in the number of reads aligned to the intron exterior flanking intervals of length L and the intron interior intervals of length L (L is the length of the reads).

Figure C.2. The coverage skew is based on a comparison of the number of full length alignments on the flanks of a spliced intron (p,q) to the number of alignments in the immediate interior. For true splice junctions, it is expected that the sequence on the flanks will generally be expressed more highly than the sequence in the interior. Therefore, the quantities Ap-Bp and Aq-Bq would generally be positive, and the skew score will be positive for nearly all true splice junctions. However, many false positives have a negative skew score (Figure 3 in the article), which makes this feature useful for negative set selection.

Gap (intron) length

The intron length score is obtained from the intron length distribution determined by GeneMark-ES. The length score is simply the log of frequency of introns having given length. The intron length distribution is calculated from the set of introns predicted by GeneMark-ES in the last iteration of unsupervised derivation of the training set. A uniform kernel is used to smooth the distribution function. For genomes with homogeneous G+C content we have observed that the intron length distribution derived from EST gapped alignments is close to the length distribution derived by GeneMark-ES, see Fig. C.3, showing the example of *Fragaria vesca* genome.
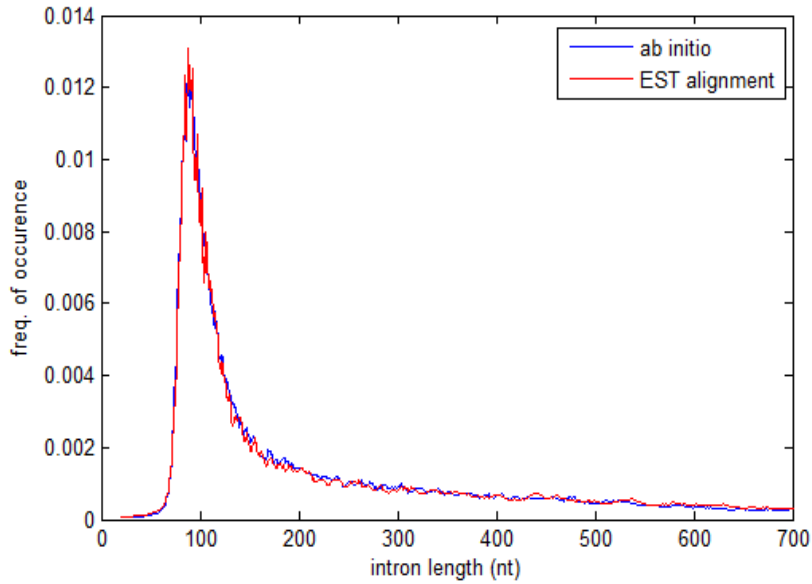
Figure C.3. Comparison of strawberry (*F. vesca*) intron length distributions found by GeneMark-ES (blue) and one derived from alignment of EST sequences (red).

Donor and acceptor splice site score

The donor and acceptor splice sites are modeled by two separate position weight matrices (PWMs). The PWM score is defined as a logarithm of the ratio of the likelihood that the site sequence is generated by the position dependent site model and the likelihood that the site sequence is generated by a background model. The likelihood score for a zero-order site and background models can be written as

$$S = \sum_{i=1}^{N} log \frac{P_i^0(x_i)}{Q_i^0(x_i)}$$

where N is the width of the model and i is the site position. For the first-order site and background models, the score will be found as

$$S = log \frac{P_i^0(x_1)}{Q_i^0(x_1)} + \sum_{i=1}^{N-1} log \frac{P_i^1(x_{i+1}|x_i)}{Q_i^1(x_{i+1}|x_i)}.$$

The order of the site model is dependent on the number of the size of the training set defined by GeneMark-ES. Normally, the models will have the first order. Note that the splice site canonical dinucleotides do not contribute to the PWM score of either the

donor or acceptor. The site model P_i^j is found by calculating positional nucleotide (or conditional dinucleotide) frequencies.

Strand concordance indicator

The strand concordance indicator is a binary feature which equals to one if one or both ends of the read are aligned with the genomic sequence which resides in the opposite strand to *ab initio* predicted (annotated) gene. There are three conditions that should be fulfilled. The first condition is that the splice junction strand is identifiable from the terminal dinucleotide sequences. If the canonic splice site dinucleotide sequence is one of GT-AG, G+C-AG, AT-AC, then the strand is identifiable. Otherwise, no strand is inferred. The second condition is that either p or q, the intron border position in genomic DNA, must lie inside a predicted gene. This means that either p or q (or both) falls between the first nucleotide of a predicted start codon and last nucleotide of the gene stop codon. The third condition is that the inferred strand of the splice junction is different from a strand of a predicted gene overlapping with positions p or q (or both). If all three conditions are met, the strand error indicator equals 1. Otherwise it is zero. This strand concordance indicator feature (as well as the frame shift indicator feature) assignments is illustrated in Fig. C.4.

Frame shift indicator

The frame shift indicator feature is similar to the feature above. It is a binary feature that requires three conditions to hold for value of one to be assigned. First, for a junction (p,q), *both* p and q must lie interior to a gene predicted by GeneMark-ES delimited by the first nucleotide of a predicted start codon, and the last nucleotide of the predicted stop codon. Second, the upstream and downstream regions from the junction (p,q) are predicted as coding exons. Third, the predicted reading frame in these exons connected by the splice junction (p,q) are in agreement. The frame shift feature equals to one if all three conditions hold, otherwise it is zero.
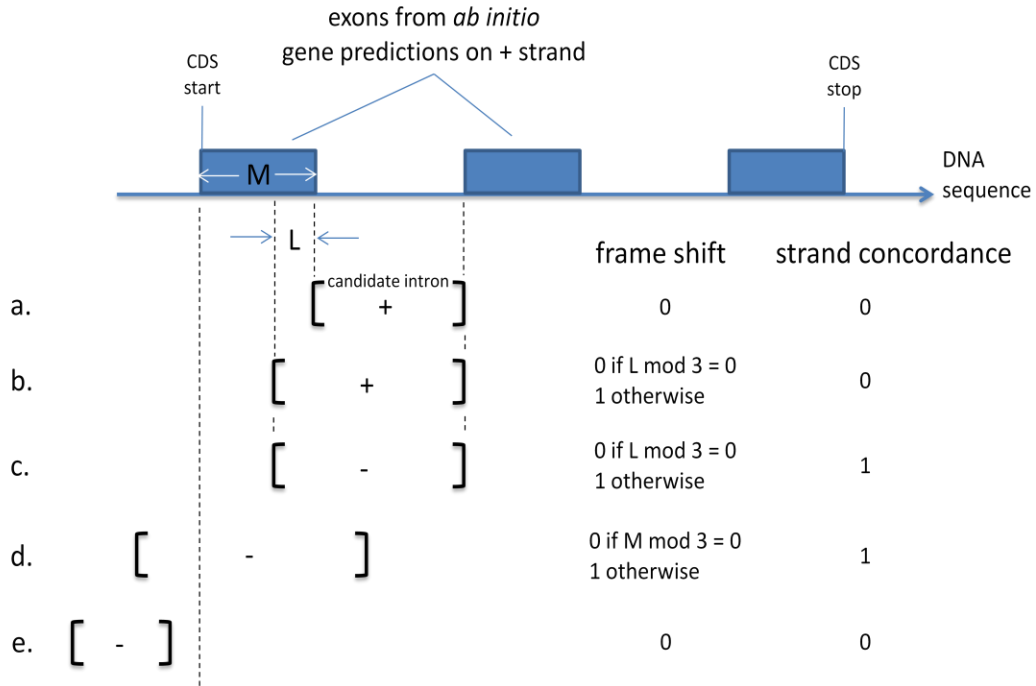
Figure C.4. Frameshift and strand concordance features: a/ the candidate SJ agrees with an *ab initio* predicted intron, both indicators are equal to zero; b/ one boundary falls into the interior of a predicted coding exon, a frameshift will occur if the interval length L is not a multiple of three, c/ same as in (b), except that the intron is located in the opposite strand to the predicted gene, therefore the strand concordance feature is equal to one; d/ the intron includes predicted exon of length M, resulting in a frame shift if M is not a multiple of 3; e/ both indications are zero because the predicted intron does not overlap with a predicted gene.

APPENDIX D

GENEMARK-HB+: A PIPELINE DEVELOPED FOR PREDICTION

OF GENES IN PLANT GENOMES

A new method for gene finding was developed which incorporates external evidence (EST alignments) for training GHMM parameters, in addition to a semi-discriminative parameter blending with heuristic parameters. The name GeneMark-HB+ is given to emphasize the heuristic blending performed in the semi-discriminative training step in the pipeline. The pipeline consists of 4 main steps: 1/ EST pipeline, 2/ semi-supervised training, 3/ semi-discriminative training, and 4/ final prediction step. In Figure D1, a block diagram of the pipeline is shown.
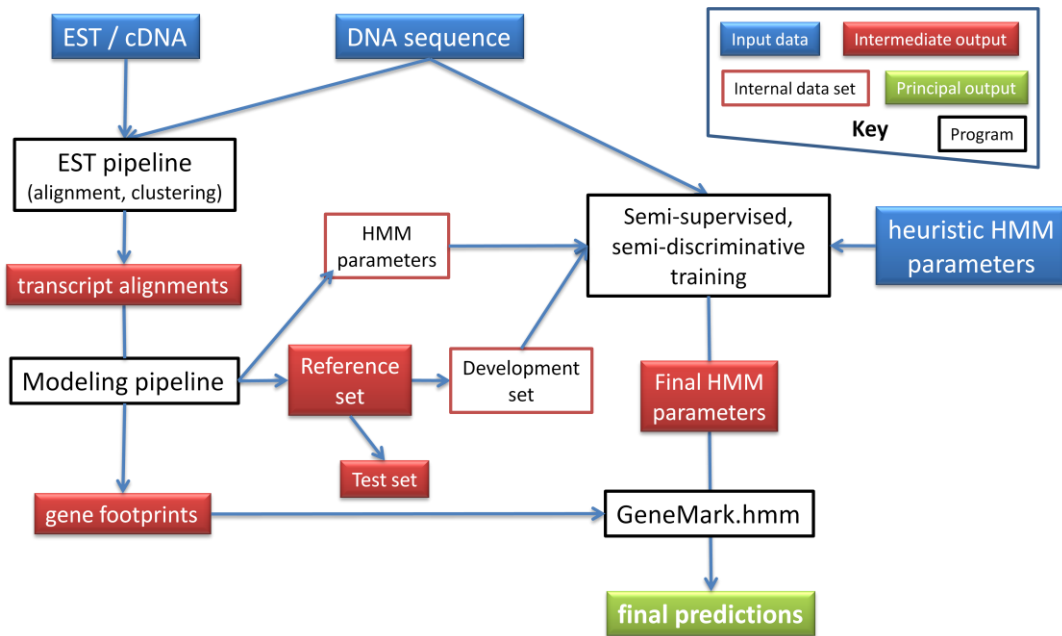
Figure D.1. Diagram of GeneMark-HB+. The pipeline includes an EST alignment and clustering pipeline, a sequence modeling pipeline, semi-discriminative training, and final prediction step.

The EST pipeline

The EST pipeline is a collection of programs which perform EST alignment and clustering, unsupervised Markov model training on mapped transcripts, then subsequent development of submodels used in later steps.

The pipeline begins by removing noise from the raw EST sequences in the form of vectors, adaptors, and bacterial sequences using programs SeqClean (http://compbio.dfci.harvard.edu/tgi/software/) and UniVec (http://www.ncbi.nlm.nih.gov/VecScreen/UniVec.html). These are sequences which contaminate a sequencing library caused by inevitable sample preparation and biochemical errors in the experiment. Subsequently, the sequences are aligned to the genome using the program BLAT (Kent, 2001). Alignments are then post-processed to minimize errors. For instance, all EST alignments shorter than 100 nt, or which have less than 90% sequence identity in the alignment to the genome are eliminated. Gapped alignments are also filtered in order to minimize intron prediction errors. Alignment gaps (introns) longer than 10 kb are eliminated for small and medium sized genomes, since the error rate for longer gapped alignments is quite high. Intron terminal dinucleotides are scored according to prior probabilities, so that any score less than a threshold will result in elimination of the gapped alignment. Clustering alignments by locus results in a set of candidate transcripts and transcript fragments.

Unsupervised training and Start/stop annotation

The mapped transcript sequences are subsequently provided to the program GeneMarkS (Besemer, 2001) with introns spliced out. GeneMark-S uses an unsupervised training technique to find parameters of a GHMM designed to parse sequences into single-exon genes. The program was designed to be used for gene finding in prokaryotic genomes, but it includes an option to make it suitable for use on eukaryotic mRNA sequences (having no ribosomal binding site). These parameters are then used by

GeneMarkS to predict all genes or partial genes in the transcript sequences. The process is illustrated in Fig. D.2.

The existence and location of such predicted CDS start and stop coordinates within a consensus fragment can occur in 4 distinct scenarios: 1/ a transcript with one complete gene coding sequence (start and stop predicted inside) 2/ a partial gene with either a CDS start or stop predicted inside (but not both), 3/ a transcript fragment with no CDS predicted inside, 4/ a fragment with more than one gene CDS predicted (any combination of more than one complete or partial gene). The only fragment types in which the predicted start and stop coordinates are not thrown out are 1 and 2. However, even in types 1 and 2, the predicted CDS is required to be longer than 300 nt. In addition, all predicted CDS starts must occur than 100 nt downstream from the 5' end of the sequence. However, in some cases the CDS start has an upstream in-frame stop codon in the transcript. In such cases the 100 nt distance requirement is nullified. This filter is designed to prevent truncating genes with unsequenced 5' ends. Another filter is the requirement for canonical splice sites (GT-AG dinucleotides at the 5' and 3' terminal ends of each intron). The reason for this is that not only are the vast majority of introns canonical in this sense, but also the probability of error for non-canonical splice junction alignment is much higher compared with canonical for all EST alignment programs (van Nimwegen, 2006), In addition, any sequence for which the gene strand implied by alignment does not match the strand implied by GeneMarkS is thrown out.
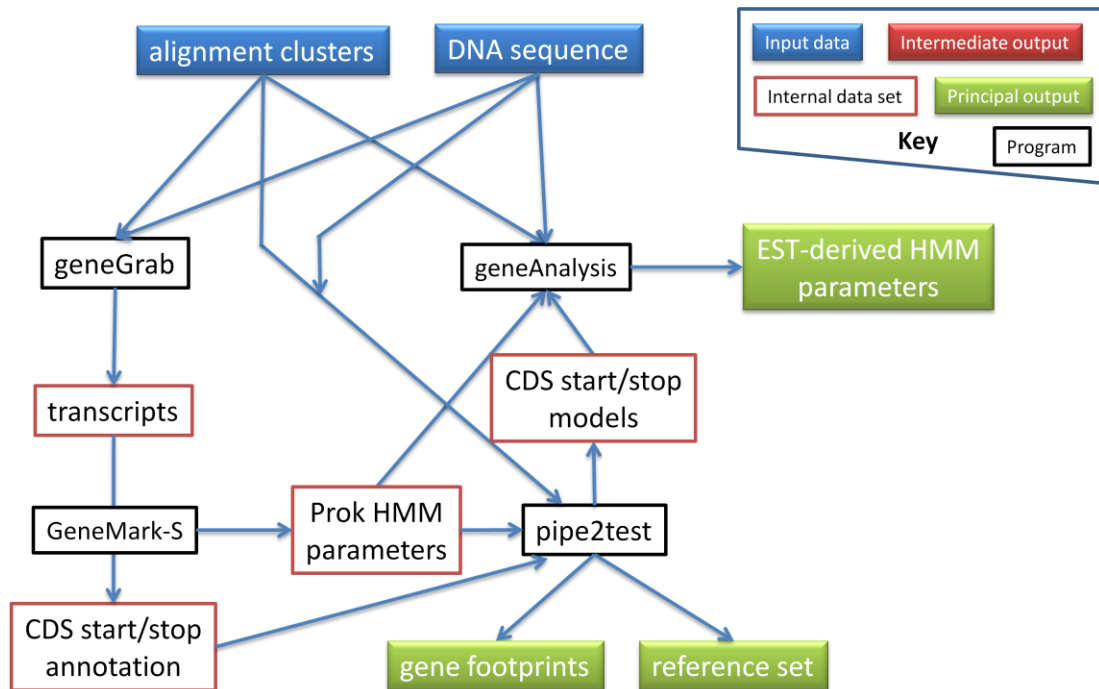
Figure D.2. Diagram of the modeling pipeline. Clusters of alignments of EST to the genome are used as input to GeneMark-S for unsupervised training.

Modeling pipeline: GHMM submodel calculation

All introns, exons, and CDS start and stop boundaries found by the EST pipeline are compiled into a list called *gene footprints*. The footprints are used not only for GHMM parameter estimation, but also to provide support to these features in the final prediction step.

The introns found by EST alignment are used to calculate position specific frequency matrices (PSFMs) for donor and acceptor sites. If more than 2000 introns are found by EST alignment (meeting the quality criteria discussed above), then these PSFM models will be first order. Otherwise, zero order models are calculated. The GeneMark-S Markov model (obtained by unsupervised parameter learning on transcripts) provides 3-periodic, n^{th} order parameters which characterize CDS, in addition to an nth order non-coding sequence model. The default order of the Markov models (n) is two (2) in this pipeline. Higher orders are possible if the quantity of CDS found by GeneMark-S is sufficient. If enough introns are found by alignment, an accurate intron length distribution

103

may be obtained as well. The length distribution is always calculated, requiring the program user to verify that the quality is suitable for gene prediction. Thus, five submodels may be derived from the EST pipeline: donor and acceptor PSFMs, coding (3-periodic) and non-coding sequence Markov models, and the intron length distribution.

A set of all externally verified genes is found by selecting those satisfying three criteria: 1/ predicted start and stop codons are found by the EST pipeline on the transcript, 2/ the gene length is at least 300 nt, and 3/ all introns have canonical terminal dinucleotides (GT-AG). This full set of genes is partitioned into two sets: a development set and a test set. The development set is used for semi-discriminative training, while the test set is used for final evaluation.

Semi-discriminative training

Two different training strategies are followed, with the supervised model parameters derived in the previous step as a starting point. The two strategies are called Type 0 and Type 1 (Fig. D3). In Type 0 training, unsupervised training is carried out by the GeneMark-ES algorithm, using a starting model with parameters developed from the analysis of eukaryotic genomes described in Appendix A. The G+C% content of the target genome is calculated and the appropriate heuristic model for that level is used as the starting point for training. In Type 1 training, the externally derived model (from the modeling pipeline) is used as a starting point, and 3 iterations of machine learning are performed so that all parameters are updated. In the first iteration, donor and acceptor splice site models are derived for each reading frame separately (for introns found in each position in a codon), in addition to all feature durations. All GHMM parameters are updated in the two subsequent iterations. After Type 0 and Type 1 training, the better model is found by scoring them on exon-level (Sn+Sp)/2 on the development set. The selected model is referred to as M_{ES}^*.
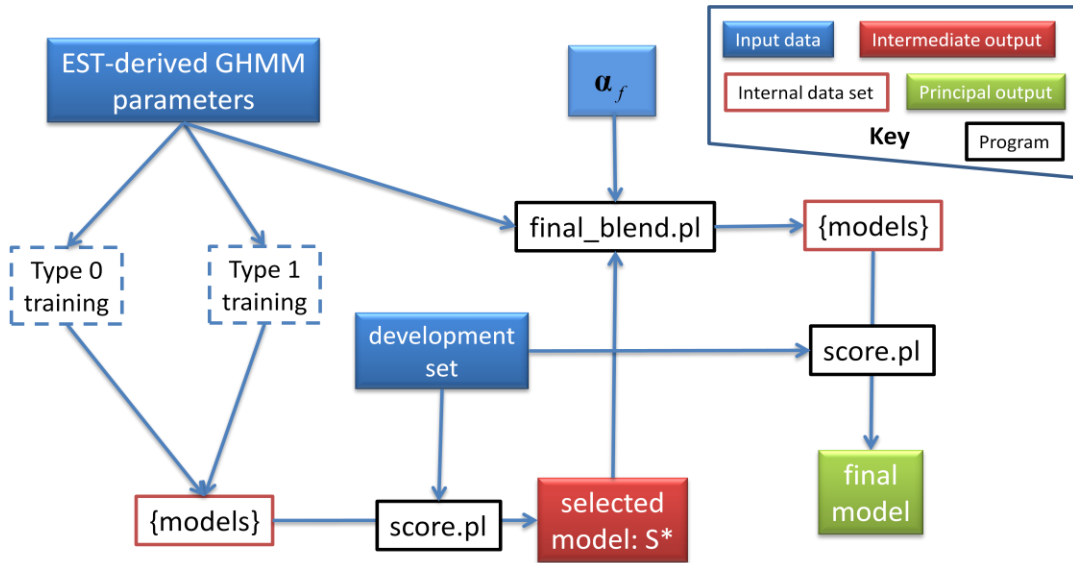
Figure D.3. Diagram of semi-discriminative training in GeneMark-HB+. Starting with model parameters derived from EST alignment, two training strategies are attempted and scored on a development set. The better parameters are chosen as input for blending with heuristic models. The final model is chosen as the blended model scoring highest against the development set.

The quality of submodels derived from alignment of EST sequences depends on the number of gene structures are revealed by alignment, and whether the codon usage of these genes is representative of the full set of genes in the genome. In general, the genes found by alignment will favor the highly expressed genes, which tend to have slightly different codon usage compared than genes expressed at lower levels (Mathe, 1999). Therefore, coding sequence models derived from EST alignments will tend to be biased. This may lead to a bias in M_{ES}^*, particularly if Type 1 training is used because it undergoes only 3 iterations of training. A more general approach to formation of a starting point model for parameter learning is a blended model, which consists of submodels found by

$$M_{blend} = aM_{ES}^* + (1 - a)M_{heur}$$

where, M_{heur} is the eukaryotic heuristic starting model (Appendix A), and M_{blend} is the combined model. In general, the best value of the scalar blending factor $(0 < a < 1)$ is closer to 0 when little or no alignment information is available, and closer to 1 when

alignment information is plentiful. Substantial differences in gene expression can lead to biases in GHMM parameters derived from alignments (highly expressed genes will be overrepresented).

From the set of EST alignments, a test set is constructed. The test set consists of all alignments which form transcripts containing one (and only one) complete CDS with a minimum length of 300 nt as predicted by GeneMark-S. The full test set is partitioned into two sets: a development set, and a final test set. The development set is a set of genes which will be used for semi-discriminative training. The test set is only used for final evaluation.

In semi-discriminative training, the "optimal" blending (α) parameters are sought which maximize prediction performance on the development set. The 3 blending parameters used in the final blending step are donor PSFM, acceptor PSFM, and the coding sequence Markov chain model. In practice, this search for the optimal 3-dimensional vector of blending parameters may be found by a simplified brute force grid search approach, in which a collection of pre-defined samples in this space are scored and the "optimal" point is declared as that point which yields the best score. Practically, a 3-dimensional grid search is not feasible due to the time required to evaluate each point. Instead, a small set of 21 points in this space is evaluated. The 21 points are found on a diagonal cut along donor and acceptor PSFM dimensions (10 points), and another diagonal cut along all 3 dimensions (10 points), and one additional point for the model M_{ES}^* (no blending). The score is defined as (Sn+Sp)/2 for all predicted exons. The parameters with the highest score are taken as the final GHMM parameters, which are used for the prediction step of the pipeline.
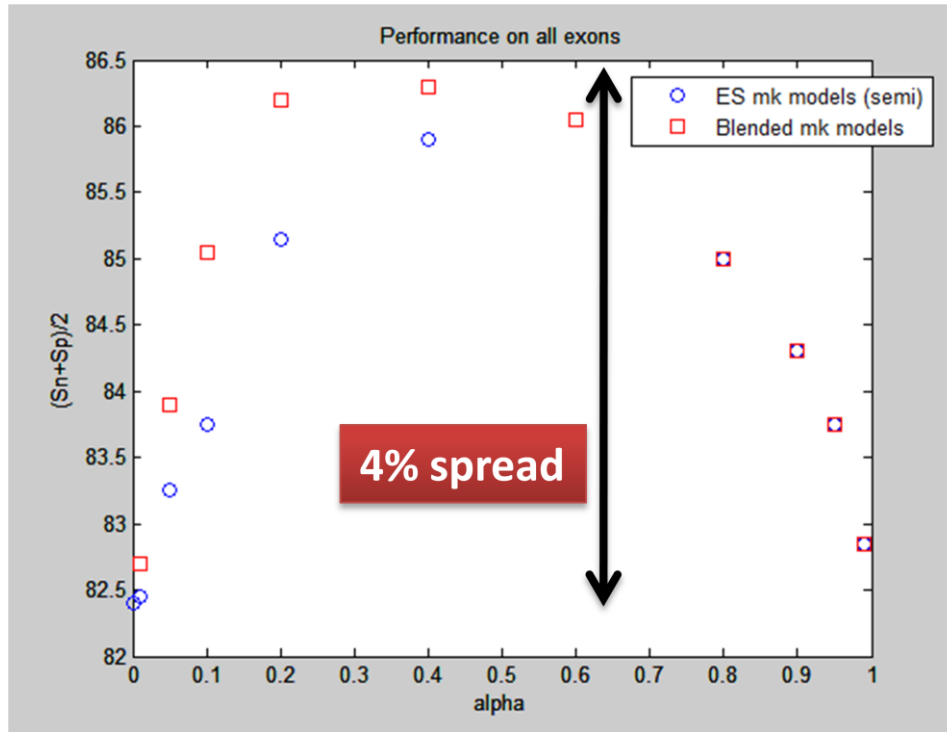
Figure D.4. Effect of variation of blending parameter alpha on prediction performance of coding exons of *F. vesca*. Blue circles correspond to performance when M_{ES}^* Markov model parameters are used (blending occurs on donor and acceptor PSFM models only). Red squares correspond to performance when blending is applied to the Markov model parameters in addition to the PSFMs. The lower left corner of the graph corresponds to regular GeneMark-ES performance.

Fig. D.4 shows an example from *Fragaria vesca*, in which final blending performance on the 21 sets of blending parameters is compared. 11 of the 21 blending parameter sets do not incorporate any Markov model (coding and non-coding sequence) information from the ESTs (colored in blue), while 10 parameter sets do include sequence models (colored red) with blending values equal to that used for donor and acceptor PSFMs. This figure shows a 4% exon prediction accuracy improvement is possible on the development set merely by searching for suitable blending parameters. In this particular example, an alpha value of 0.4 is approximately optimal, but this value changes depending on the genome, and quantity of available EST sequences.

In Fig. D.5 (panels a and b), the performance of GeneMark-HB+ is compared to GeneMark-ES and a model which is fully derived from external sequence aligment (the

output of the modeling pipeline). The genomes evaluated are nearly all plant genomes, due to involvement in many plant genome projects. The mosquito genome *A. gambiae* is included due to its similarity to the plant genomes in terms of size and G+C composition. In Fig. D5, each genome has two yellow numbers next to it, labeled as (m/n). m is the number of genes in the development set and n is the number of genes in the test set. The test set is used to assess the performance of gene prediction, which is evaluated using 8 different performance metrics: internal exon, donor boundaries, acceptor boundaries, introns, all exons, initial exons, terminal exons, and nucleotide coverage. In general, there is no clear winner between ES and HB+ in this comparison, although both are clearly superior to the externally derived model. In Fig. D6, ES and HB+ are compared by the number of features each is found to score higher for all 8 features across all 7 genomes considered. A point is given for each feature a program scores highest (Sn+Sp)/2, or ties with highest. The result is a measure of aggregate performance. HB+ scores higher than ES at a ratio of about 3:2.

a.

		Anopheles gambiae (251/253)			Fragaria vesca (290/293)			Citrus sinensis (245/247)			Citrus clementina (1003/1005)		
		ES	HB+	Ext	ES	HB+	Ext	ES	HB+	Ext	ES	HB+	Ext
Internal	Sn	85.1	**87.7**	82.0	**89.2**	87.8	59.7	**86.6**	86.8	60.8	90.4	**90.3**	65.9
	Sp	83.3	**82.0**	84.2	**89.9**	89.0	90.4	**85.7**	85.0	78.8	87.4	**89.9**	90.4
Intron	Sn	85.3	**87.6**	81.4	**87.3**	87.3	67.1	**82.7**	82.5	65.2	87.8	**89.0**	69.9
	Sp	84.2	**83.5**	87.6	**88.8**	88.3	90.2	**82.4**	81.7	77.0	85.8	**88.1**	89.7
Donor	Sn	89.3	**91.9**	85.3	91.5	**91.9**	71.2	**89.1**	89.2	73.2	93.1	**93.4**	73.8
	Sp	87.6	**87.2**	90.1	93.1	**93.0**	95.6	**88.7**	88.4	86.5	91.0	**92.4**	94.5
Acceptor	Sn	90.8	**93.4**	87.8	**91.7**	90.9	69.4	**89.6**	89.8	71.7	**93.4**	93.0	72.9
	Sp	88.2	**86.5**	90.3	**92.8**	90.6	92.1	**89.3**	88.8	84.0	**91.1**	91.4	93.1
All exons	Sn	82.4	**85.4**	79.8	85.1	**85.5**	65.1	**81.9**	81.9	64.5	86.9	**87.2**	67.2
	Sp	80.5	**81.7**	82.5	86.1	**86.1**	80.8	**81.5**	81.1	71.8	85.1	**86.4**	81.3
Initiation	Sn	75.5	**75.9**	72.3	79.1	**81.2**	75.1	**79.4**	79.4	72.5	**81.7**	79.7	71.2
	Sp	74.6	**77.1**	75.6	80.2	**85.0**	76.4	**79.0**	79.0	70.5	**81.2**	82.0	69.1
Termination	Sn	91.7	**94.5**	91.3	**87.4**	86.3	75.8	82.2	**83.0**	75.3	**86.9**	86.3	79.0
	Sp	88.9	**91.6**	90.9	**87.1**	86.1	75.0	81.9	**82.3**	71.8	**85.7**	85.9	76.0
Nucleotide	Sn	93.2	94.0	**94.1**	**96.4**	95.0	86.8	95.0	**95.4**	88.5	**97.8**	96.4	91.1
	Sp	90.9	91.0	**92.0**	**95.8**	95.2	95.2	94.0	**93.8**	88.5	**96.9**	96.4	96.5

b.

		Rubus idaeus (175/177)			Arabidopsis thaliana (1611/1613)			Cucumis sativus (45/48)		
		ES	HB+	Ext	ES	HB+	Ext	ES	HB+	Ext
Internal	Sn	85.2	**86.7**	62.0	94.0	**93.6**	75.4	92.9	**91.4**	57.1
	Sp	88.9	**90.1**	89.7	92.1	**93.3**	91.8	83.3	**87.7**	64.5
Intron	Sn	85.1	**86.4**	66.7	93.3	**93.5**	78.6	**91.3**	90.4	53.9
	Sp	87.9	**88.4**	88.9	92.2	**93.5**	92.3	**86.8**	87.4	89.9
Donor	Sn	88.7	**89.5**	69.8	96.0	**95.9**	82.0	94.8	**95.7**	55.7
	Sp	91.7	**91.6**	92.6	94.8	**95.8**	96.3	90.1	**92.4**	92.8
Acceptor	Sn	89.2	**90.1**	70.7	**96.2**	95.8	80.6	**94.8**	93.0	68.7
	Sp	92.1	**91.6**	94.0	94.8	**95.2**	94.2	**88.6**	88.4	72.5
All exons	Sn	81.3	**83.1**	65.0	91.4	**91.4**	75.9	83.4	**84.0**	44.2
	Sp	83.3	**84.5**	81.1	90.5	**91.4**	86.4	80.5	**82.0**	61.5
Initiation	Sn	72.2	**76.3**	75.0	85.1	**84.9**	76.9	68.8	**70.8**	0.0
	Sp	71.7	**77.9**	72.0	86.2	**87.5**	77.1	71.7	**73.9**	0.0
Termination	Sn	**83.2**	82.5	77.8	**92.3**	92.3	84.9	**89.6**	83.3	75.0
	Sp	**82.0**	81.9	75.2	**92.2**	92.3	83.5	**89.6**	83.3	75.0
Nucleotide	Sn	**96.0**	95.7	86.3	**98.7**	98.3	93.0	98.3	**97.6**	76.9
	Sp	**95.3**	94.8	94.1	**97.8**	98.0	96.8	91.8	**92.6**	88.5

Figure D.5. Performance comparison of GeneMark-ES, GeneMark-HB+, and predictions based on models derived from external sequence alignment only (Ext). Performance is shown for 8 different features for 7 different genomes. Bold face Sn,Sp pairs indicate the highest level of (Sn+Sp)/2.

109

Genome	ES	HB+
F. vesca	5	3
C. sinensis	6*	3*
C. clementina	4	4
R. idaeus	2	6
A. gambaie	0	7
A. thaliana	3*	7*
C. sativus	3*	5*
Total	**23**	**35**

Figure D.6. Summary of prediction performance on 7 genomes. Each number represents the number of features for which that program achieved the highest value of (Sn+Sp)/2. An asterisk (*) indicates that at for at least one feature, the two programs had the same value of (Sn+Sp)/2. The number of features for *A. gambiae* do not sum to 8 because in that case the external model scored the highest nucleotide-level performance.

REFERENCES

[1]   Boguski M, "The Turning Point in Genome Research," *Trends in Biochemical Sciences* 20:295-6, 1995.

[2]   Durbin R, Eddy S, Krogh A, and Mitchinson G, *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*, Cambridge University Press, 1998.

[3]   Borodovsky M and McIninch J, "GeneMark: parallel gene recognition for both DNA strands," Computers & Chemistry, 1993, Vol. 17, No. 19, pp. 123-133.

[4]   Delcher A, Bratke K, Powers E, and Salzberg S, "Identifying bacterial genes and endosymbiont DNA with Glimmer," *Bioinformatics*, 2007.

[5]   Yeh RF, Lim LP, and Burge CB, Computational inference of homologous gene structures in the human genome. *Genome Research* (2001) May; 11(5):803-16.

[6]   Salzberg S, Pertea M, et al, "Interpolated Markov models for eukaryotic gene finding," *Genomics* 59(1): 24-31, 1999.

[7]   Besemer J, Lomsadze A, and Borodovsky M, "GeneMarkS: a self-training method for prediction of gene starts in microbial genomes. Implications for finding sequence motifs in regulatory regions," *Nucleic Acids Research*, 2001, Vol. 29, No. 12, 2607-2618.

[8]   Lomsadze A, Ter-Hovhannisyan V, Chernoff Y, and Borodovsky M, "Gene identification in novel eukaryotic genomes by self-training algorithm," *Nucleic Acids Research*, 2005, Vol. 33, No. 20, 6494-6506.

[9]   Ter-Hovhannisyan V., Lomsadze A., Chernoff Y. and Borodovsky M. "Gene prediction in novel fungal genomes using an ab initio algorithm with unsupervised training." *Genome Research*, 2008, Dec 18(12):1979-90.

[10] Lukashin A and Borodovsky M, "GeneMark.hmm: new solutions for gene finding," *Nucleic Acids Research*, 1998, Vol. 26, No. 4, pp. 1107-1115.

[11] DeCaprio D, Vinson J, Pearson M, Montgomery P, Doherty M, and Galagan J, "Conrad: gene prediction using conditional random fields," *Genome Research*, 2007 Sep; 17(9): 1389-98.

[12] Bernal A, Crammer K, Hatzigeorgiou A, and Pereira F, "Global discriminative learning for higher-accuracy computational gene prediction," *PLOS Computational Biology* 2007 March; 3(3): e54.

[13] Stanke M and Waack S, "Gene prediction with a hidden Markov model and a new intron submodel," *Bioinformatics* 2003 Oct; 19.

[14] Stanke M, "Gene Prediction with a Hidden Markov Model," Ph.D. Dissertation, Gottingen, 2003.

[15] Mathe C, Peresetsky A, Dehais P, Van Montagu M, Rouze P, "Classification of Arabidopsis thaliana Gene Sequences: Clustering of Coding Sequences into Two Groups According to Codon Usage Improves Gene Prediction," *Journal of Molecular Biology* (1999) 285, 1977-1991.

[16] Wald N, Alroy M, Botzman M, and Margalit H, "Codon usage bias in prokaryotic pyrimidine-ending codons is associated with the degeneracy of the encoded amino acids," *Nucleic Acids Research*, 2012 August; 40(15): 7074-7083.

[17] Majoros WH, *Methods for computational gene prediction*, Cambridge University Press, 2007.

[18] Wang Z, Gerstein M, and Snyder M, RNA-seq: a revolutionary tool for transcriptomics, *Nature Review Genetics* 2009 January; 10(1): 57-63.

[19] Besemer J and Borodovsky M, Heuristic approach to deriving models for gene finding (1999), *Nucleic Acids Research*, Vol. 27, No. 19, pp. 3911-3920.

[20] A.F.A. Smit, R. Hubley & P. Green RepeatMasker at http://repeatmasker.org

[21] Jurka, J., Kapitonov, V.V., Pavlicek, A., Klonowski, P., Kohany, O., Walichiewicz, J. Repbase Update, a database of eukaryotic repetitive elements. (2005) *Cytogentic and Genome Research* 110:462-467.

[22] McCarthy EM and McDonald JF, LTR_STRUC: a novel search and identification program for LTR retrotransposons, *Bioinformatics* (2003), Vol. 19, no. 3, pp. 362-367.

[23] Xu Z and Wang H, LTR_FINDER: an efficient tool for the prediction of full-length LTR retrotransposons (2007), *Nucleic Acids Research*, Jul;35(Web Server issue):W265-8. Epub 2007 May 7.

[24] Bao Z and Eddy SR, Automated *de novo* identification of repeat sequence families in sequenced genomes (2002), *Genome Research*, 12:1269-1276.

[25] Price AL, Jones, NC, and Pevzner PA, *De novo* identification of repeat families in large genomes, *Bioinformatics* (2005), Vol. 21, Suppl. 1, pp. i351-i358.

[26] Edgar RC and Myers EW, PILER: identification and classification of genomic repeats (2005), *Bioinformatics*, Vol. 21 Suppl. 1 2005, pp.. i152-i158.

[27] Grabherr MG, Haas BJ, Yassour M *et al*, Full-length transcriptome assembly from RNA-seq data without a reference genome (2011) *Nature biotechnology* 29, pp. 644-652.

[28] Roberts A, Trapnell C, Donaghey J, Rinn J, Pachter L, Improving RNA-seq expression estimates by correcting for fragment bias (2011) Genome Biology, 12:R22.

[29] Trapnell C, Williams B, Pertea G, Mortazavi A, Kwan G, van Baren M, Salzberg S, Wold B, Pachter L, Transcript assembly and quantification by RNA-seq reveals unannotated transcripts and isoform switching during cell differentiation (2010) *Nature biotechnology*, Vol. 28, no. 5, pp. 511-515.

[30] Simpson J, Wong K, Jackman S, Schein J, Jones S, Birol I, ABySS: A parallel assembler for short read sequence data (2009), *Genome Research* (June); 19(6): 1117-1123.

[31] Birol I, Jackman S, Nielsen C, Qian J, Varhol R, Stazyk G, Morin R, Zhao Y, Hirst M, Schein J, Horsman D, Connors J, Gascoyne R, Marra M, Jones S, De novo transcriptome assembly with ABySS (2009) Bioinformatics 25(21):2872-2877.

[32] Guttman M, Garber M, Levin J Donaghey J, Robinson J, Adiconis X, Fan L, Koziol M, Gnirke A, Nusbaum C, Rinn J, Lander E, Regev A, *Ab initio* reconstruction of cell type-specific transcriptomes in mouse reveals the conserved multi-exonic structure of linkRNAs (2010) *Nature biotechnology*, 28, pp. 503-510.

[33] Trapnell, C., Pachter, L. & Salzberg, S.L. TopHat: discovering splice junctions with RNA-Seq. (2009) *Bioinformatics* **25**, 1105–1111.

[34] Li H and Durbin R. (2009) Fast and accurate short read alignment with Burrows-Wheeler transform. *Bioinformatics*, July 15; 25(14): 1754-1760.

[35] Li H, Ruan J, and Durbin R. (2008) Mapping short DNA sequencing reads and calling variants using mapping quality scores. *Genome Research*, 18:1851-1858.

[36] Li R, LiY, Kristiansen K, and Wang J. (2008) SOAP: short oligonucleotide alignment program. *Bioinformatics*, 24:713-714.

[37] Langmead B, Trapnell C, Pop M, Salzberg SL. (2009) Ultrafast and memory-efficient alignment of short DNA sequences to the human genome. *Genome Biol* 10:R25.

[38] Wang K, Singh D, Zeng Z, Coleman SJ, Huang Y, Savich GL, He X, Mieczkowski P, Grimm SA, Perou CM *et al.* (2010) MapSplice: accurate mapping of RNA-seq reads for splice junction discovery. *Nucleic acids research*, 38, e178.

[39] Zhang Y, Lameijer EW, t Hoen PA, Ning Z, Slagboom PE and Ye K. (2012) PASSion: a pattern growth algorithm-based pipeline for splice junction detection in paired-end RNA-Seq data. *Bioinformatics*, 28, 479-486.

[40] Huang S, Zhang J, Li R, Zhang W, He Z, Lam T, Peng Z, and Yiu S. (2011) SOAPsplice: genome-wide ab initio detection of splice junctions from RNA-seq data. *Frontiers in Genetics*, 2:46.

[41] Au KF, Jiang H, Lin L, Xing Y, Wong WH. (2010) Detection of splice junctions from paired-end RNA-seq data by SpliceMap. *Nucleic Acids Research*, Aug; 38(14):4570-8, Epub 2010 Apr 5.

[42] Kent WJ, (2001) BLAT—The BLAST-like alignment tool, *Genome Research*. 12:656-664.

[43] Wu TD and Watanabe CK. (2005) GMAP: a genomic mapping and alignment program for mRNA and EST sequences. Vol. 21 no. 9, pp. 1859-1875.

[44] Li Y, Li-Byarlay H, Burns P, Borodovsky M, Robinson GE, Ma J. TrueSight: a new algorithm for splice junction detection using RNA-seq. (2012) *Nucleic Acids Research*, doi: 10.1093/nar/gks1311. First published online: December 18.

[45] Burges C. (1998) A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery* 2, 121-167.

[46] Chang C and Lin C. (2011) LIBSVM : a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1--27:27. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[47] G. Pertea, et al (2003). "TIGR Gene Indices clustering tools (TGICL): a software system for fast clustering of large EST datasets." *Bioinformatics* Mar 22(19(5)): 651-652.

[48] van Nimwegen, et al. (2006). "SPA: A Probabilistic Algorithm for Spliced Alignment." PLoS Genetics 2(4): 587-605.

[49] Lucks J, Mortimer S, Trapnell C, Luo S, Aviran S, Schroth G, Pachter L, Duodna J, and Arkin A. (2011). "Multiplexed RNA structure characterization with selective 2'-hydroxyl acylation analyzed by primer extension sequencing (SHAPE-seq)." PNAS June 3.

[50] Allen J and Salzberg S (2005). "JIGSAW: integration of multiple sources of evidence for gene prediction" *Bioinformatics*, Vol. 21 no. 18, pp. 3596-3603.

[51] Allen J, Pertea M, and Salzberg S (2004). "Computational gene prediction using multiple sources of evidence" *Genome Research*, 14(1).

[52] Gross S and Brent M (2006). "Using multiple alignments to improve gene prediction" *Journal of Computational Biology*, 13(2), pp. 379-393.

[53] Megy, K. *et al*. (2012) VectorBase: improvements to a bioinformatics resource for invertebrate vector genomics. *Nucleic Acids Research* 40: D729-734; PMID: 22135296.

[54] Feschotte C, Keswani U, Ranganathan N, Guibotsy M, and Levine D (2009) "Exploring Repetitive DNA Landscapes Using REPCLASS, a Tool That Automates the Classification of Transposable Elements in Eukaryotic Genomes," *Genome Biology and Evolution*, Vol. 1, pp. 205-220.

[55] Ellinghaus D, Kurtz S, and Willhoeft U (2008) "LTRharvest, an efficient and flexible software for de novo detection of LTR retrotransposons," *BMC Bioinformatics*, 9:18.

[56] McCarthy EM and McDonald JF (2003) "LTR_STRUC: a novel search and identification program for LTR retrotransposons," *Bioinformatics* Vol 18, issue 3, pp. 362-367.

[57] Pontaroli AC, Rogers R, Zhang Q, Shields M, Davis T, Folta K, SanMiguel P, and Bennetzen J (2009), "Gene content and distribution in the nuclear genome of *Fragaria Vesca*," The Plant Genome, Vol. 2, No. 1.

[58] Martin J and Wang Z (2011), Next-generation transcriptome assembly, *Nature Reviews Genetics* 12, 671-682 (October 2011).

[59] Melamud E and Moult J. (2009) Stochastic noise in splicing machinery. *Nucleic Acids Research*, Vol. 37, No. 14, pp. 4873-4886.

[60] Pickrell JK, Pai AA, Gilad Y, Pritchard JK (2010) Noisy Splicing Drives mRNA Isoform Diversity in Human Cells. *PLoS Genet* 6(12): e1001236.

[61] Marquiz Y, Brown J, Simpson C, Barta A, and Kalyna M. (2012) Transcriptome survey reveals increased complexity of the alternative splicing landscape in *Arabidopsis*. *Genome Research* June; 22(6), pp. 1184-1195.

[62] Daines B, Wang H, Wang L, Li Y, Han Y, Emmert D, Gelbart W, Wang X, Li W, Gibbs R, and Chen R. (2011) The *Drosophila melanogaster* transcriptome by paired-end RNA sequencing. *Genome Research* February; 21(2), pp. 315-324.

[63] Wu TD and Nacu S. (2010) Fast and SNP-tolerant detection of complex variants and splicing in short reads, *Bioinformatics*. Vol. 26, no. 7, pp. 873-881.

[64] Moriyama EN, and Powell J R. (1998) Gene length and codon usage bias in *Drosophila melanogaster*, *Saccharomyces cerevisiae* and *Escherichia coli*. *Nucleic Acids Research*, Vol 26(13); July 1.

[65] Duret L and Mouchiroud D. (1999) Expression pattern and , surprisingly, gene length shape codon usage in *Caenorhabditis*, *Drosophila*, and *Arabidopsis*, *PNAS* vol. 96 no. 8.

[66] Keerthi SS and Lin CJ. (2003) Asymptotic behaviors of support vector machines with Gaussian kernel. *Neural Computation*, 15(7):1667-1689.

[67] Zhang Y, Lameijer EW, t Hoen PA, Ning Z, Slagboom PE and Ye K. (2012) PASSion: a pattern growth algorithm-based pipeline for splice junction detection in paired-end RNA-Seq data. *Bioinformatics*, 28, 479-486.

[68] Sharp PM, Emery LR, Zeng K (2010), "Forces that influence the evolution of codon bias," *Philosophical Transactions of the Royal Society B: Biological Sciences*, v.365 (1544).

[69] Rogic S, Mackworth A, Ouellette F (2001), "Evaluation of gene-finding programs on mammalian sequences," Genome Research 11, pp. 817-832.

[70] Kellis M, Patterson N, Endrizzi M, Birren B, and Lander E (2003), Sequencing and comparison of yeast species to identify genes and regulatory elements, *Nature* 423, pp. 241-254.

[71] Korf I (2004) Gene finding in novel genomes, *BMC Bioinformatics* 5; 59.

[72] Korf I, Flicek P, Duan D, Brent MR (2001) Integrating genomic homology into gene structure prediction, *Bioinformatics* 17; Suppl 1.

[73] Brejova B, Brown DG, Li M, Vinar T (2005) ExonHunter: a comprehensive approach to gene finding, *Bioinformatics* 21; Suppl. 1.

[74] Borodovsky M.Yu., Sprizhitskii Y.A., Golovanov E.I., and Aleksandrov A.A. "Statistical Patterns in Primary Structures of the Functional Regions of the Genome in Escherichia Coli." *Molekular Biology*, 1986, Vol. 20, pp. 826-833, 833-840, 1144-1150.

VITA

PAUL D. BURNS

BURNS was born in Huntsville, Alabama. He studied electrical engineering at Auburn University, where he received BSEE and MSEE degrees in 1992 and 1995, respectively. He worked in the field of radar systems for 14 years, working at Dynetics, Magnacom, and Georgia Tech Research Institute. While at GTRI, he became recognized as a subject matter expert in multi-sensor, multi-target tracking, and he taught short course lectures on advanced tracking concepts and sensor registration. In 2009 he was lured into the world of biology to pursue a doctorate in Bioinformatics at Georgia Tech.