# PALMA: Perfect Alignments using Large Margin Algorithms

G. Rätsch,[a] B. Hepp,[b] U. Schulze,[a] and C.S. Ong[a,c]

[a] Friedrich Miescher Lab., Max Planck Society, Spemannstr. 39, 72076 Tübingen, Germany,
[b] Fraunhofer FIRST, Kekulèstr. 7, 12489 Berlin, Germany,
[c] Max Planck Institute for Biological Cybernetics, Spemannstr. 38, 72076 Tübingen, Germany

**Abstract:** Despite many years of research on how to properly align sequences in the presence of sequencing errors, alternative splicing and micro-exons, the correct alignment of mRNA sequences to genomic DNA is still a challenging task. We present a novel approach based on large margin learning that combines kernel based splice site predictions with common sequence alignment techniques. By solving a convex optimization problem, our algorithm – called PALMA – tunes the parameters of the model such that the true alignment scores higher than all other alignments. In an experimental study on the alignments of mRNAs containing artificially generated micro-exons, we show that our algorithm drastically outperforms all other methods: It perfectly aligns all 4358 sequences on an hold-out set, while the best other method misaligns at least 90 of them. Moreover, our algorithm is very robust against noise in the query sequence: when deleting, inserting, or mutating up to 50% of the query sequence, it still aligns 95% of all sequences correctly, while other methods achieve less than 36% accuracy. For datasets, additional results and a stand-alone alignment tool see `http://www.fml.mpg.de/raetsch/projects/palma`.

## 1 Introduction

Many genomes have been sequenced recently. This is only a first step to understand what the genome actually encodes. Fortunately, most of them also come with rather large amounts of expressed sequence tags (ESTs; sequenced parts of mRNA), which help to accurately recognize genes and to identify the exon/intron boundaries as well as alternative splice forms (see [ZG06] and references therein).

Many methods for aligning ESTs to genomic DNA have been proposed, including approaches based on *blast* [AGM+90], *spliced alignments* [GMP96], *sim4* [FHZ+98], *GeneSeqer* [UZB00], *Spidey* [WS01], *blat* [Ken02], an approach to find additional microexons [VHS03] and most recently *exalin* [ZG06]. The identification of exon/intron boundaries is important for finding the correct alignment. Therefore most approaches try to find an alignment that prefers splice site consensus signals in the identified introns (usually `GT/AG`, considerably less often `GC/AG` and in some organisms also `AT/AC`) that help to accurately identify these boundaries. This is done by employing either dynamic programming or sophisticated heuristics.

[ZG06] used an information theoretic approach to combine the two types of information available during alignment: the sequence similarity and splice site predictions. Given this model, dynamic programming is used to compute the maximum-log likelihood alignment. Our algorithm, called *PALMA*, is based on similar ideas as *exalin*. The main difference is

the modeling of splice sites using support vector machines, the modeling of intron lengths and the large margin based combination of the different types of information. Our approach does not include any probabilistic models and hence does not return probabilities for a particular alignment. It is, however, able to very accurately and robustly align sequences as will be seen in the experimental section where we consider the problem of aligning modified EST sequences to genomic DNA (here of the model organism *C. elegans*) using the most difficult setup: We consider artificially generated short internal exons (2-50nt) combined with small to large amounts of noise in the query sequence. We show that our method perfectly aligns all sequences while other methods fail as soon as the exons become too short or the amount of noise too large.

## 2 Method

The idea of our algorithm is to compute an alignment by dynamic programming that uses a scoring function. We tune the parameters of the scoring functions such that the true alignment does not only achieve a large score (to be "most likely"), but also that all other alignments score considerably lower than the true alignment (to obtain a "large margin between the alignments"). Similar ideas are used in other large margin algorithms such as Support Vector Machines [Vap95] and Boosting [FS97]. Also, a similar approach for aligning protein sequences (without intron related gaps) has been proposed independently by [JGE05]. The resulting scoring function can then be maximized using dynamic programming in order to obtain the optimal alignment. Our method consists of three independent parts: the splice site prediction model, the dynamic programming algorithm and the optimization of the scoring function which we describe in the following sections.

Training the splice site model and also the large margin combination requires separate sequence data sets. For the splice site model, we used genes that were EST confirmed but without full length cDNA support (set 1). We consider a random subset of 40% of all cDNA confirmed genes without evidence for alternative splicing for training the large margin combination (set 2). The remaining 20% and 40% were used for hyper-parameter tuning (set 3) and final evaluation (set 4) respectively.

### 2.1 Splice Site Predictions

From the set of EST sequences (set 1) we extracted sequences of confirmed donor (intron start) and acceptor (intron end) splice sites (see Appendix A for details). For acceptor splice sites we used a window of 80bp upstream to 60bp downstream of the site (on the DNA). For donor sites we used 60bp upstream and 80bp downstream. Also from these training sequences we extracted non-splice sites that are within an exon or intron of the sequence and have `AG` (acceptor) or `GT`/`GC` (donor) consensus. In order to recognize acceptor and donor splice sites, we trained two Support Vector Machine classifiers [Vap95] with soft-margin using the so-called "weighted degree" kernel [SRJM02, RSS06]. The kernel mainly takes positional information (relative to the splice site) about the appearance of certain motifs into account. It computes the scalar product between two sequences $\mathbf{x}$ and $\mathbf{x}'$:

$$\mathrm{k}(\mathbf{x}, \mathbf{x}') = \sum_{j=1}^{d} v_j \sum_{i=1}^{N-j} \mathbf{I}(x_{[i,i+j]} = x'_{[i,i+j]}), \tag{1}$$

where $N = 140$ is the length of the sequence and $x_{[a,b]}$ denotes the substring of $x$ from position $a$ to (excluding) $b$. The function $\mathbf{I}$ is the indicator function, $\mathbf{I}(\text{true}) = 1$, $\mathbf{I}(\text{false}) = 0$ and the weights $v_j := d - j + 1$. We used a normalization of the kernel $\tilde{k}(s_1, s_2) = \frac{k(s1,s_2)}{\sqrt{k(s_1,s_1)k(s_2,s_2)}}$ and $d = 22$ for the recognition of splice sites. Additionally, the regularization parameter of the Support Vector Machine was set to be $C = 2$ and $C = 3$ for acceptor and donor sites, respectively. All parameters (including the window size, regularization parameters etc) have been tuned on data set 2 (cf. [RSS05]).

Given a DNA sequence as target of an alignment we can now use the two SVMs to compute scores for each position with corresponding consensus[1] for being a splice acceptor or donor site, respectively. Since we consider *C. elegans* where U12 splicing is extremely rare or not present, we exclude the `AT/AC` splice sites from our splice site model.

### 2.2 Needleman-Wunsch Alignments with Intron Model

The classical deterministic and exact alignment algorithm is the Needleman-Wunsch algorithm and is based on dynamic programming. Its running time is $\mathcal{O}(m \cdot n)$, where $m$ is the length of the EST sequence $S_E$, and $n$ is the length of the DNA sequence $S_D$. It builds up a $m \cdot n$ matrix and hence has the same space complexity.

The main idea of the algorithm is to compute an overall alignment by determining the maximum over all alignments of all prefixes $S_E(1:i) := (S_E(1), \ldots, S_E(i))$ and $S_D(1:j) := (S_D(1), \ldots, S_D(j))$ of the two sequences $S_E$ and $S_D$, respectively. An alignment is given by a sequence of pairs $(a_r, b_r)$, $r = 1, \ldots, R$, where $R \leq m + n$ depends on the alignment and $a_r, b_r \in \Sigma := \{A, C, G, T, N, -\}$. A pair consists either of the two corresponding letters of the two sequences or a single letter in one sequence paired with a gap in the other sequence. The alignment is scored using a substitution matrix $M$, which we interpret as a function $M : \Sigma \times \Sigma \to \mathbb{R}$. Then the score for the alignment $\mathcal{A} = \{(a_r, b_r)\}_r$ is simply $\sum_r M(a_r, b_r)$.

We define $V(i, j)$ to be the score of the best possible alignment of prefixes $S_E(1:i)$ and $S_D(1:j)$. Then $V(n, m)$ can be computed using the following recurrence formula (for all $i = 1, \ldots, m$ and $j = 1, \ldots, n$):

$$V(i,j) = \max \begin{cases} V(i-1, j-1) + M(S_E(i), S_D(j)) \\ V(i-1, j) + M(S_E(i), '-') \\ V(i, j-1) + M('-', S_D(j)) \end{cases} \tag{2}$$

The recurrence is initialized with $V(0,0) := 0$, $V(i,0) := 0$ and $V(0,j) := 0$ for all $i = 1 \ldots m$ and $j = 1 \ldots n$. There are three possibilities:

- $S_E(i)$ and $S_D(j)$ are aligned to each other (possibly a mismatch).
- $S_E(i)$ is aligned to a gap in the DNA sequence.
- $S_D(j)$ is aligned to a gap in the EST sequence.

In the original setting there are only these three possibilities and one can straightforwardly fill the matrix from left to right and top to bottom to finally compute $V(n, m)$. The optimal alignment can then be obtained by backtracking [DEKM98].

[1] `AG` for acceptor sites and `GT` or `GC` for donor sites.

The Needleman-Wunsch algorithm only aligns the single bases of two sequences and does not distinguish between exons and introns – it essentially treats everything as exons. We therefore propose to extend the Needleman-Wunsch algorithm to better model introns: We allow an additional "intron transition" that is separately scored based on its length and the scores of splice sites at its ends. We denote by $f_I(i_1, i_2)$ the intron scoring function for an intron starting at $i_1$ and ending at $i_2$. The intron scoring function $f_I(i_1, i_2)$ is computed based on the intron length $i_2 - i_1$, the donor SVM output $g_{don}(i_1)$ for position $i_1$ and the acceptor SVM output $g_{acc}(i_2)$ for position $i_2$. During learning we determine three functions $f_\ell$, $f_{acc}$ and $f_{don} : \mathbb{R} \to \mathbb{R}$ to combine these three values:

$$f_I(i_1, i_2) = f_\ell(i_2 - i_1) + f_{don}(g_{don}(i_1)) + f_{acc}(g_{acc}(i_2)). \tag{3}$$

When there is no donor consensus at position $i_1$, then we define $f_{don}(g_{don}(i_1)) := -\infty$ (similarly for $f_{acc}(g_{acc}(i_2))$). Given the intron scoring function $f_I$ we can now restate the recurrence formula (for all $i = 1, \ldots, m$ and $j = 1, \ldots, n$):

$$V(i,j) = \max \begin{cases} V(i-1, j-1) + M(S_E(i), S_D(j)) \\ V(i-1, j) + M(S_E(i), '-') \\ V(i, j-1) + M('-', S_D(j)) \\ \max_{1 \le k \le j-1}(V(i,k) + f_I(k,j)) \end{cases} \tag{4}$$

where we consider the additional possibility of an intron starting at position $k$ and ending at $j$. Please note that the above recurrence formula is considerably more computationally expensive than the previous one: every step involves finding the optimal intron start ($\mathcal{O}(n)$). However, one only needs to consider those positions where the intron start and end exhibit the corresponding splice consensus sites and also the splice site predictors output large enough scores. Additional strategies for speeding up such algorithms are discussed in [ZG06].

For completeness we need to extend our notation for alignments with introns. We use again alignment pairs $\mathcal{A} = \{(a_r, b_r)\}_r$, but extend the alphabet for $a_r$ to $\Sigma \cup \{+\}$ ("intron sequence missing") and for $b_r$ to $\Sigma \cup \{*\}$ ("intron sequence"). Note that $b_r$ should only contain strings of length greater than one if $a_r = '+'$. Then the score $f(\mathcal{A})$ for an alignment $\mathcal{A}$ with intron is computed as before, i.e. $\sum_r M(a_r, b_r)$, except when $a_r = +$: In this case the intron score function is used to score the corresponding intron.

### 2.3 Large Margin Combination

In the previous section we assumed that the functions $f_{acc}$, $f_{don}$ and $f_\ell$ as well as the substitution matrix $M$ were given. We now describe a algorithm to determine these parameters based on the training set of sequences and their true alignments.

Two methods based on a similar idea have been independently proposed in [JGE05] and [KK06]. They both present a simpler algorithm for learning the substitution matrix required for aligning protein sequences. [KK06] presents an algorithm–based on the method from [GBN94]–that can learn hundreds of parameters simultaneously and is able to model affine gap-costs. [JGE05] propose an algorithm related to support vector machines. However, both approaches do not model introns or splice sites explicitly and are therefore expected to fail in identifying microexons.

Note that our proposed algorithm is two-layered: First one learns the splice site model

and then how to combine the different pieces of information. In principle these two steps can be combined to one step. Then the piecewise linear functions can be replaced with linear combinations of kernel elements as similarly done in [ATH03]. However, this makes training much slower and is not expected to improve the results in our case.

Since the three functions are one-dimensional, it suffices to use a simple piecewise linear model: Let $s$ be the number of supporting points $x_i$ (satisfying $x_i < x_{i+1}$) and $y_i$ their values, then the piecewise linear function is defined by

$$f(x) = \begin{cases} y_1 & x \leq x_1 \\ \frac{y_i(x_{i+1}-x)+y_{i+1}(x-x_i)}{x_{i+1}-x_i} & x_i \leq x \leq x_{i+1} \\ y_s & x \geq x_s \end{cases}. \tag{5}$$

After having appropriately chosen supporting points on the $x$-axis we only need to optimize the corresponding $y$-values. For $f_{acc}$ and $f_{don}$ we use 30 supporting points uniformly sampled between $-5$ and $+5$ (our SVM outputs are typically not larger). For $f_\ell$ we use 30 log-uniformly sampled supporting points between 30nt and 1000nt.[2] Given the three functions and the substitution matrix, the alignment scoring function $f(\mathcal{A})$ is fully specified. Moreover, for a given alignment $\mathcal{A}$, it can be verified that $f(\mathcal{A})$ is linear in all parameters that we denote by $\boldsymbol{\theta}$, i.e. in the values of the substitution matrix and the $y$-values of the three piecewise linear functions, $\boldsymbol{\theta} := (\boldsymbol{\theta}_{acc}, \boldsymbol{\theta}_{don}, \boldsymbol{\theta}_\ell, \boldsymbol{\theta}_M)$.

### 2.3.1 Optimization

For training we are given a set of $N$ true alignments $\mathcal{A}_i^+$, $i = 1, \ldots, N$. The goal is to find the parameters $\boldsymbol{\theta}$ of the alignment scoring function $f$ such that the difference of scores $f_\theta(\mathcal{A}_i^+) - f_\theta(\mathcal{A}^-)$ is large for *all* wrong alignments $\mathcal{A}^- \neq \mathcal{A}_i^+$. This can be done by solving the following convex optimization problem:

$$\min_{\boldsymbol{\xi}\geq\mathbf{0},\boldsymbol{\theta}} \frac{1}{N}\sum_{i=1}^{N}\xi_i \qquad \text{s.t.} \qquad f_\theta(\mathcal{A}_i^+) - f_\theta(\mathcal{A}^-) \geq 1 - \xi_i \quad \forall i \text{ and } \mathcal{A}^- \neq \mathcal{A}_i^+. \tag{6}$$

Here we introduced so-called slack-variables $\xi_i$ to implement a soft-margin [CV95]. The above optimization problem has exponentially many constraints and cannot be easily solved directly. Instead one adopts a column generation technique (cf. [HK93] and references therein) and for every true alignment one maintains a set of wrong alignments $\mathcal{A}_{i,j}^- \neq \mathcal{A}_i^+$, for all $j$. Initially this set is empty but it can easily be filled by running the dynamic programming algorithm discussed in the previous section to generate wrong alignments (based on some arbitrary initial parameters). Then one solves the following optimization problem

$$\min_{\boldsymbol{\xi}\geq\mathbf{0},\boldsymbol{\theta}} \frac{1}{N}\sum_{i=1}^{N}\xi_i \qquad \text{s.t.} \qquad f_\theta(\mathcal{A}_i^+) - f_\theta(\mathcal{A}_{i,j}^-) \geq 1 - \xi_i \quad \forall i,j \tag{7}$$

Given a set of wrong alignments one can now determine the intermediate optimal parameters $\boldsymbol{\theta}$, and further use the dynamic programming algorithm to find other wrong alignments to be included in the set of wrong alignments. The procedure is iterated and provably converges to the solution of (6) in a finite number of steps (in our application often not more than 10 iterations).

[2] For other organisms one might want to choose a larger range.

### 2.3.2 Regularization

In empirical inference it is common to regularize the parameters in order not to overfit. We implement this by adding a regularization term $C\mathbf{P}(\boldsymbol{\theta})$ to (6), where $C$ is the regularization constant and $\mathbf{P}$ a regularization function. Recall that the parameter vector consists of four parts, and we define the regularization term as follows:

$$\mathbf{P}(\boldsymbol{\theta}) = \sum_{i=1}^{n-1}(\theta_{acc,i+1}-\theta_{acc,i})^2+\sum_{i=1}^{n-1}(\theta_{don,i+1}-\theta_{don,i})^2+\sum_{i=1}^{n-1}(\theta_{\ell,i+1}-\theta_{\ell,i})^2+\sum_{a,b} M(a,b)^2.$$

It implements the idea that the piecewise-linear functions should be smooth and the values in the substitution matrix small.

## 3 Results and Discussion

Most alignment algorithms work very well for aligning mRNA sequences against genomic DNA when query and target perfectly match and the matching blocks are long enough. In our experimental study we are interested in the most difficult cases, where most algorithms start to fail. If an algorithm works in such case we expect that it will also return correct alignments for easier cases.

We evaluate our proposed method, *PALMA*, and other methods such as (*exalin*, *sim4* & *blat*). We consider the alignment of mRNA sequence fragments containing three exons where we artificially shortened the middle exon (final length of 2-50nt, see Appendix B for details). Artificially generating the data has the benefit of knowing exactly what the correct alignment has to be. Additionally, we add considerable amounts of noise ($p = 0\%, 1\%, 10\%, 20\%, 50\%$ of random mutations, deletions and insertions) to the query sequence. We then measure how often the methods find the middle exons and the whole alignment correctly. The evaluation is done on a separate test set which was not used during training of our method (set 4, cf. Appendix B).

The model selection for the splice site predictors have been performed on separate validation sets (set 2). Model selection of regularization parameter $C$ in our method (cf. Section 2.3.2) was done by simple validation on a separate validation set (set 3). While the method was trained on noise-free data, we applied it to the noisy versions during validation since otherwise the validation error rate was always zero, almost independently of the choice of $C$. We determined $C = 0.01$ as optimal regularization constant. To analyze the importance of the splice site predictions relative to the sequence similarity for correct alignments, we additionally trained a second model that does not use splice site information (but only intron lengths and the substitution matrix). We call it *PALMA* without splice sites (SS).

Figure 1 shows the alignment error rate for different methods on the 4358 test sequences. Here we counted an alignment as a mistake if the exon boundaries deviated by at least one nucleotide. [3] We also looked at how often the middle exon has been correctly identified.

[3] For *exalin* we noticed that the alignment is very often off by 2nt. We assume that this is a fixable bug in the *exalin* implementation. For fairness we therefore allowed deviations of $\pm$2nt for exalin only. The problem often occurs for high noise levels. For instance at $p = 20\%$ we find 20% error rate for the strict evaluation, while only 6% error when using the relaxed criterion.

We observed that in most cases an alignment error is induced by the inaccurate identification of the middle exon.[4] From our results in Figure 1 we observe that there are drastic differences between the methods. Almost all methods perform reasonably well when the query perfectly matches the target – with the exception of *sim4* which has problems aligning at least 18% of the sequences. For *blat* and *sim4* the error rates drastically increase when adding noise to the query sequence. Only *exalin* and *PALMA* (with and without splice site information) have low error rates for noise levels of at most 20%. When deleting, inserting or mutating up to 50% of the query sequence, PALMA (with splice sites) still aligns 95% of all sequences correctly, while the other methods achieve less than 36% accuracy. For high noise levels the splice site information helps to reduce the error rate considerably. But also in the low noise cases the splice site predictions help to accurately identify very short exons that can be found ambiguously in the intronic regions (0.4% of the test sequences).

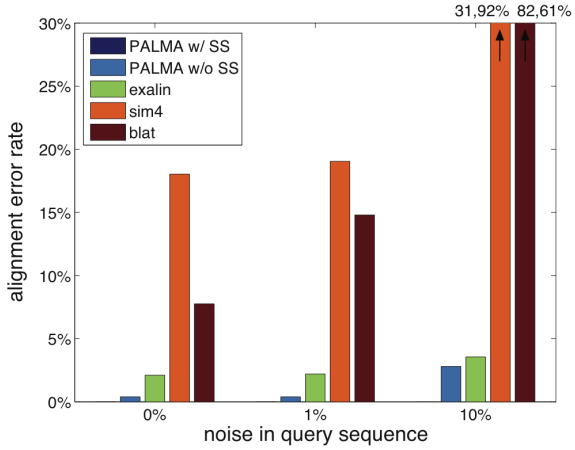



Figure 1: Comparison of different methods for aligning mRNAs to genomic DNA: We considered the particularly difficult task of aligning exon triples with short middle exons (2-50nt) in the presence of noise. Although an alignment is already declared as true if the intron boundaries are correct, only PALMA (with splice sites) achieves 0% error rate for aligning queries with up to 10% noise.

Figures 2-4 show the optimized parameters determined by our algorithm. For the piecewise linear functions in 2 we obtain smooth sigmoid-shaped functions ("differences between large score values do not matter"). Comparing with Figure 4 we observe that the difference between a weak and a strong splice site is worth about 3-4 matches, since the substitution matrix contains values between $-0.4$ and $+0.4$. Figure 3 illustrates the piecewise linear function for scoring intron lengths. We observe that the maximum coincides with the most frequent intron length of around 50nt. The optimized substitution matrix is essentially diagonal, which is not surprising as there was no preference for substitutions in our data.

## 4 Conclusion

We have proposed a new alignment algorithm that computes the optimal alignment of mRNA sequences to genomic DNA while exploiting existing very accurate kernel-based splice site predictions. In a simulation study on aligning sequences with very short exons and considerable amounts of noise we have shown that our method achieves significantly

[4] Since it gives a very similar figure, we omitted it from the manuscript.

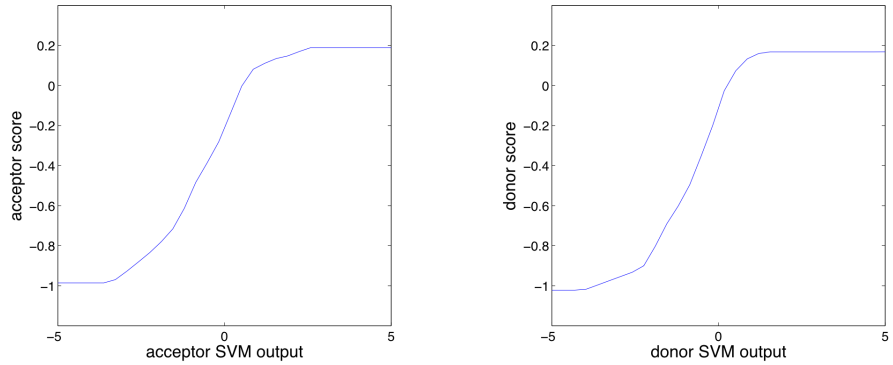



Figure 2: *PALMA*'s optimized functions $f_{acc}$ and $f_{don}$ scoring acceptor and donor SVM outputs.

lower error rates than other methods. This indicates that the proposed method would be more effective than current approaches for discovering microexons, i.e. exons between 2-25nt in length. This is especially true in the presence of sequencing errors or mutations which may render current approaches and heuristics inaccurate. In addition, by combining it with other methods such as *blast* we can reduce the computational cost in order to apply our method for alignments of ESTs to whole-genomes.

## Acknowledgments

We thank G. Schweikert and N. Toussaint for proofreading the manuscript.

## A Processing of Sequence Databases

We collected all known *C. elegans* ESTs from Wormbase [HCC$^+$04] (release WS120; 236,893 sequences) and dbEST [BT93] (as of February 22, 2004; 231,096 sequences). Using *blat* [Ken02] we aligned them against the genomic DNA (release WS120). The alignment was used to confirm exons and introns. We refined the alignment by correcting typical sequencing errors, for instance by removing minor insertions and deletions. If an intron did not exhibit the consensus `GT`/`AG` or `GC`/`AG` at the 5' and 3' ends, then we tried to achieve this by shifting the boundaries up to 2 base pairs (bp). If this still did not lead to the consensus, we split the sequence into two parts and considered each subsequence separately. In a next step we merged consistent alignments, if they shared at least one complete exon or intron. This lead to a set of 124,442 unique EST-based sequences.

We repeated the above procedure with all known cDNAs from Wormbase (release WS120; 4,855 sequences). These sequences only contain the coding part of the mRNA. We used their ends as annotation for start and stop codons.

We clustered the sequences in order to obtain independent training, validation and test sets. In the beginning each of the above EST and cDNA sequences were in a separate cluster. We iteratively joined clusters, if any two sequences from distinct clusters match to the same genomic location (this includes many forms of alternative splicing). We obtained

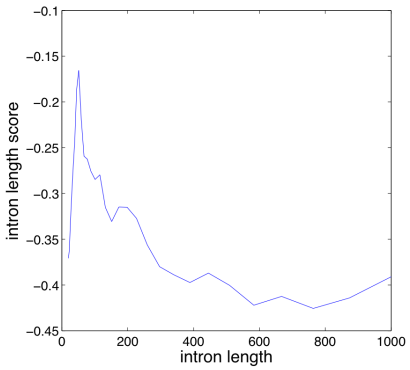



Figure 3: Shown is the optimized intron length scoring function $f_\ell$: The maximum is located at around 50nt, which is also the most frequent intron length in *C. elegans*.

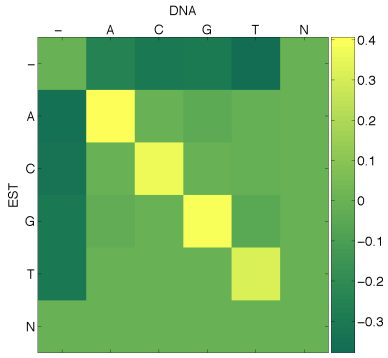



Figure 4: Shown is the optimized substitution matrix: matches score high and gaps low. Mismatch scores are all close to zero and do not contribute much.

21,086 clusters, while 4072 clusters contained at least one cDNA.

For *set 1* we chose all clusters not containing a cDNA (17215), for *set 2* we chose 40% of the clusters containing at least one cDNA (1536). For *set 3* we used 20% of clusters with cDNA (775). The remaining 40% of clusters with at least one cDNA (1,560) were used as *set 4*. Sets 2-4 were filtered to remove confirmed alternative splice forms. This left 1,177 cDNA sequences for *testing* in set 4 with an average of 4.8 exons per gene and 2,313bp from the 5' to the 3' end.

## B Artificial Microexon Dataset

Based on sets 2-4 described in the last section we created sets of consecutive exon triples from the confirmed transcripts in these sets. This lead to 4604, 2257 and 4358 triples. In a first processing step we shortened the middle exons to a random length between 2nt and 50nt (uniformly distributed). To do so, we removed the correct number of nucleotides from the center of the middle exon – from the query as well as the DNA. This leaves the splice sites mostly functional. In a second step we added varying amounts of noise. For a given noise level $p$ and a query sequence of length $L$, we first replaced $p \cdot L/3$ positions with a random letter ($\Sigma = \{A, C, G, T, N\}$). Then we deleted the same number of non-overlapping positions in the sequence and added the same number of random nucleotides at random positions. We used $p = 0\%, 1\%, 10\%, 20\%, 50\%$.

## References

[AGM+90] S.F. Altschul, W. Gish, W. Miller, E.W. Myers, and D.J. Lipman. Basic local alignment search tool. *Journal Molecular Biology*, 215(3):403–10, 1990.

[ATH03] Y. Altun, I. Tsochantaridis, and T. Hofmann. Hidden Markov Support Vector Machines. In *Proc. 20th International Conference on Machine Learning*, 2003.

[BT93] M.S. Boguski and T.M. Lowe C.M. Tolstoshev. dbEST–Database for "Expressed Sequence Tags". *Nat Genet.*, 4(4):332–3, 1993.

[CV95] C. Cortes and V.N. Vapnik. Support Vector Networks. *Machine Learning*, 20:273–297, 1995.

[DEKM98] R. Durbin, S. Eddy, A. Krogh, and G. Mitchison. *Biological Sequence Analysis: Probabilistic models of protein and nucleic acids*. Cambridge University Press, 1998. 7th edition.

[FHZ+98] L. Florea, G. Hartzell, Z. Zhang, G.M. Rubin, and W. Miller. A computer program for aligning a cDNA sequence with a genomic DNA sequence. *Genome Research*, 8:967–974, 1998.

[FS97] Y. Freund and R.E. Schapire. A Decision-theoretic Generalization of On-line Learning and an Application to Boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997.

[GBN94] D. Gusfield, K. Balasubramanian, and D. Naor. Parametric Optimization of Sequence Alignment. *Algorithmica*, 12:312–326, 1994.

[GMP96] M.S. Gelfand, A.A. Mironov, and P.A. Pevzner. Gene recognition via spliced sequence alignment. *Proc. Natl. Acad. Sci.*, 93(17):9061–6, 1996.

[HCC+04] T.W. Harris, N. Chen, F. Cunningham, et al. WormBase: a multi-species resource for nematode biology and genomics. *Nucl. Acids Res.*, 32, 2004. Database issue:D411-7.

[HK93] R. Hettich and K.O. Kortanek. Semi-Infinite Programming: Theory, Methods and Applications. *SIAM Review*, 3:380–429, September 1993.

[JGE05] T. Joachims, T. Galor, and R. Elber. Learning to Align Sequences: A Maximum-Margin Approach. In B. Leimkuhler, C. Chipot, R. Elber, A. Laaksonen, and A. Mark, editors, *New Algorithms for Macromolecular Simulation*, number 49 in LNCS, pages 57–71. Springer, 2005.

[Ken02] W. J. Kent. BLAT–the BLAST-like alignment tool. *Genome Res*, 12(4):656–664, April 2002.

[KK06] J. D. Kececioglu and E. Kim. Simple and Fast Inverse Alignment. In *RECOMB*, pages 441–455, 2006.

[RSS05] G. Rätsch, S. Sonnenburg, and B. Schölkopf. RASE: Recognition of Alternatively Spliced Exons in *C. elegans*. *Bioinformatics*, 21(Suppl. 1):i369–i377, June 2005.

[RSS06] G. Rätsch, S. Sonnenburg, and C. Schäfer. Learning Interpretable SVMs for Biological Sequence Classification. *BMC Bioinformatics*, 7(Suppl 1):S9, February 2006.

[SRJM02] S. Sonnenburg, G. Rätsch, A. Jagota, and K.-R. Müller. New Methods for Splice-Site Recognition. In *Proc. International Conference on Artificial Neural Networks*, 2002.

[UZB00] J. Usuka, W. Zhu, and V. Brendel. Optimal spliced alignment of homologous cDNA to a genomic DNA template. *Bioinformatics*, 16(3):203–211, 2000.

[Vap95] V.N. Vapnik. *The nature of statistical learning theory*. Springer Verlag, New York, 1995.

[VHS03] N. Volfovsky, B.J. Haas, and S.L. Salzberg. Computational Discovery of Internal Micro-Exons. *Genome Research*, 13:1216–1221, 2003.

[WS01] Ostell JM. Wheelan SJ, Church DM. Spidey: a tool for mRNA-to-genomic alignments. *Genome Research*, 11(11):1952–7, 2001.

[ZG06] M. Zhang and W. Gish. Improved spliced alignment from an information theoretic approach. *Bioinformatics*, 22(1):13–20, January 2006.