BIOINFORMATICS

Vol. 20 no. 17 2004, pages 3055–3063
doi:10.1093/bioinformatics/bth364

Predicting the efficacy of short oligonucleotides in antisense and RNAi experiments with boosted genetic programming

Pål Sætrom

Interagon AS, Medisinsk teknisk senter, N-7489 Trondheim, Norway

Received on March 26, 2004; revised and accepted on June 10, 2004
Advance Access publication June 17, 2004

ABSTRACT

Motivation: Both small interfering RNAs (siRNAs) and antisense oligonucleotides can selectively block gene expression. Although the two methods rely on different cellular mechanisms, these methods share the common property that not all oligonucleotides (oligos) are equally effective. That is, if mRNA target sites are picked at random, many of the antisense or siRNA oligos will not be effective. Algorithms that can reliably predict the efficacy of candidate oligos can greatly reduce the cost of knockdown experiments, but previous attempts to predict the efficacy of antisense oligos have had limited success. Machine learning has not previously been used to predict siRNA efficacy.

Results: We develop a genetic programming based prediction system that shows promising results on both antisense and siRNA efficacy prediction. We train and evaluate our system on a previously published database of antisense efficacies and our own database of siRNA efficacies collected from the literature. The best models gave an overall correlation between predicted and observed efficacy of 0.46 on both antisense and siRNA data. As a comparison, the best correlations of support vector machine classifiers trained on the same data were 0.40 and 0.30, respectively.

Availability: The prediction system uses proprietary hardware and is available for both commercial and strategic academic collaborations. The siRNA database is available upon request.

Contact: paal.saetrom@interagon.com

INTRODUCTION

Antisense oligonucleotides and RNA interference are the two technologies that are available for sequence-specific knockdown of mRNA. Antisense oligonucleotides (ODNs) typically consist of 15–20 nt that inhibit gene expression by complementary base-pairing to target mRNA. The result is a simple mechanistic blocking of the ribosome thereby inhibiting protein translation, or activation of the RNase H enzyme that subsequently induces cleavage and degradation of mRNA (Kurreck, 2003). Modern ODNs have been chemically modified in various ways to increase their potency.

More recently, RNA interference (RNAi)—a natural biological pathway for mRNA depletion—was discovered: 21–23 nt long double-stranded RNA with characteristic $3'$ overhangs is incorporated into a ribonucleoprotein complex called RNA-induced silencing complex, which cleaves mRNA with complementarity to its RNA component (McManus and Sharp, 2002). The RNA agent is called short interfering RNA (siRNA).

Scherer and Rossi (2003) reviews the relative strengths and weaknesses of ODNs and siRNAs as well as catalytically active classes of oligonucleotides (oligos) for mRNA knockdown that will not be discussed here.

The question of oligo efficacy arises because not all oligos are equal; common to both ODNs and siRNAs are that only a fraction of the oligos are effective in biological assays. Methods that can predict the efficacy of potential oligos are important since these would reduce the cost of carrying out antisense and siRNA knockdown experiments.

There has been some previous work on *in silico* methods that predicts oligo efficacy. The work range from considering thermodynamic properties (Matveeva *et al*., 2003; Khvorova *et al*., 2003; Schwarz *et al*., 2003), to using sequence motifs (Matveeva *et al*., 2000) and artificial neural networks (Giddings *et al*., 2002; Chalk and Sonnhammer, 2002) for predicting efficacy. Since the RNAi pathway is a more recent discovery, however, most of this work has been done on ODNs. Here, we bridge the gap by (i) collecting a database of publicly available siRNA efficacy data and (ii) developing and evaluating a machine learning method that can predict both ODN and siRNA efficacy.

Our main contribution is a machine learning system that predicts ODN and siRNA efficacy and is consistently better than previously published methods. Furthermore, we present a uniform analysis of the thermodynamic properties of ODNs and siRNAs.

METHODS

In the following sections, we will describe the databases used in our experiments and present the algorithms used in

	Production	Semantic Rule
(1)	$R \rightarrow \{C : \mathrm{p} >= N\}$	$R.\mathit{hit} := C.\mathit{count} \geq N.\mathit{cutoff}$
(2)	$C \rightarrow (C_1)(C_2)$	$C.\mathit{count} := C_1.\mathit{count} + C_2.\mathit{count}$
(3)	$C \rightarrow A$	$C.\mathit{count} := A.\mathit{count}$
(4)	$A \rightarrow A_1 \mid A_2$	$A.\mathit{count} := A_1.\mathit{count} + A_2.\mathit{count}$
(5)	$A \rightarrow L$	$A.\mathit{count} := L.\mathit{count}$
(6)	$L \rightarrow \mathbf{a}$, for some $\mathbf{a} \in \{\mathrm{a}, \mathrm{c}, \mathrm{g}, \mathrm{t}\}$	$L.\mathit{count} := \mathit{match}(\mathbf{a})$
(7)	$N \rightarrow \mathbf{n}$, for some $\mathbf{n} \in \{1, 2, \ldots\}$	$N.\mathit{cutoff} := \mathbf{n}$
	(a)	(b)

Fig. 1. Our genetic programming solution language. (**a**) Grammar. The grammar is shown in Backus–Naur form, with non-terminals represented by uppercase letters and terminals represented by boldface letters. Syntactical elements in the language, such as parentheses and operators, are in normal typeface. Alternatives are represented as separate productions. Adjacent symbols are concatenated. (**b**) Semantics. An expression matches a string if $R.hit$ is true. $match(\mathbf{a})$ returns 1 if **a** is identical to the character it is compared with, and 0 otherwise.

our classification system. This includes sections on genetic programming (GP), boosting and boosted GP. We also briefly discuss the inherent stochastic properties of GP, and describe how support vector machines (SVMs) can be used to predict ODN and siRNA efficacy. SVMs are often regarded as state-of-the-art among the different machine learning methods, and will be used as a benchmark throughout this work. Finally, we describe a procedure for estimating the predictive accuracy of classifiers, and detail the methods used to compute oligo thermodynamics.

Databases

A collection of experimentally evaluated siRNAs were collected from different sources (Vickers *et al*., 2003; Kawasaki *et al*., 2003; Harborth *et al*., 2003; Holen *et al*., 2002; Khvorova *et al*., 2003). We cross-referenced the siRNAs with their corresponding GenBank sequences when possible, and otherwise used the sequences as given in the articles.

We assigned an efficacy score based on the reported mRNA or protein knockdown levels to each siRNA. For the human cyclophilin sequences in Khvorova *et al*. (2003), where individual efficacies were missing, we assigned 0.05 and 0.75 knockdown to the effective and ineffective sequences. The efficacy score measures the fraction of mRNA or protein that remains after the knockdown experiment. Hence, effective and ineffective siRNAs are assigned low and high efficacy scores. The same measure was used in Giddings *et al*. (2000).

For our antisense experiments, we used the set of antisense sequences collected by Giddings *et al*. (2000).

We partitioned both the siRNA database and the antisense database into a set of positive and negative sequences. We used a cut-off score of 0.5; i.e. the sequences that scored above the threshold were classified as negative and vice versa. After duplicates were removed, a total of 101 positive and 103 negative siRNAs, and 125 positive and 186 negative ODNs remained.

Genetic programming

Genetic programming (Koza, 1992), as the genetic algorithms of Holland (1975), use evolution in a population of candidate solutions to solve problems. The main difference between genetic algorithms and GP is that the former operate on a population of (bit) strings, whereas the GP populations usually consist of syntax trees in some solution language. The solution language defines the structure of the possible solutions. Other solution representations such as linear or graph-based GP exist (see Banzhaf *et al*., 1997 for an overview and a good introduction).

Our GP algorithm is designed to solve two-class classification problems where the positive and negative data consists of strings. It uses subtree swapping crossover, tree generating mutation and reproduction as generic operators. Individuals are chosen for participation in new generations using tournament selection.

Each individual in the population is a syntax tree in a formal query language (Interagon AS, 2002, http://www.interagon.com/pub/whitepapers/IQL.reference-latest.pdf). To ensure that all individuals are legal expressions, we use a set of production rules when generating and combining the individuals, as Montana (1995). These production rules correspond to the grammar of the query language.

In the experiments described in this paper, only a subset of the full query language is used. Figure 1 gives a grammar for this subset along with a formal definition of its semantics.

The semantics can be explained as follows: the numeral N in the $p >= N$ part indicates the minimum number of terminals in the C-production that must match. This means that if the fourth production is removed from the language, the expressions search for strings with Hamming distance less than $|C| - N$, where $|C|$ is the number of characters in the expression. For example, the expression {gcggtt : $p >= 4$} will match the strings *gcgatt* and *acgatt*, but not *acgata* and *actatt*.

Alternatives (production 4) enables, in addition to alternative bases at the same string position, the construction of expressions where some positions in the string are considered more important than others. To illustrate this, consider the expression {gc(g|g)gtt : $p >= 4$}, where (g|g) matches g in the same string position twice (we will in the following use the shorthand (g|g) $= g_2$ for weighted positions). This expression now matches the string *acgata*, but still misses *actatt*.

Note that if the string is longer than the expression, the expression reports a match if it matches any of the string's possible substrings. Thus, the expression {gcggtt : $p >= 4$} will match strings *aagcaatt* and *gcgcgtt*, but will miss *agcaaatt* and *aattgcgg*.

The Interagon pattern matching chip (PMC) (Halaas *et al.*, 2004) is an application specific integrated circuit designed to provide orders of magnitude higher performance than comparable regular expression matchers. In our GP system, we use the PMC to evaluate the individual expressions in the GP population, thus reducing the GP system's total computation time. This increased performance becomes important when the datasets are large, or when many GP runs must be done, as for instance in cross-validation experiments or when GP is used as the base learner in a boosting algorithm.

Given an oligo sequence, the expressions generated by our GP system will only be able to answer 'yes' or 'no' as to whether the sequence is effective or not. The following section describes a method that creates classifiers with a more refined output.

Boosting

Boosting algorithms combine a set of simple rules to form a single model. The algorithms construct the combined model so that the performance of the model is increased compared to each of the single rules. Given a learning algorithm that generates a set of hypotheses $h_1, h_2, \ldots, h_T$, the boosting algorithms construct a combined hypothesis f on the form

$$f(x) = \sum_{t=1}^{T} \alpha_t \cdot h_t(x). \qquad (1)$$

α_t is the weight of hypothesis h_t, and both weights and hypotheses are learned by the boosting algorithm.

The AdaBoost algorithm, introduced by Freund and Schapire (1997), was the first step towards practical boosting algorithms. AdaBoost constructs classifiers that maximize the minimal margin in the training data (Meir and Rätsch, 2003), but as a result, the algorithm does not handle noise or outliers in the dataset very well. To solve this, regularized boosting algorithms, such as AdaBoost$_{\text{Reg}}$ (Rätsch *et al.*, 2001) and ν-Arc (Rätsch *et al.*, 2000), have been developed. Boosting algorithms are related to other large margin classifiers such as SVMs (Vapnik, 1995; Müller *et al.*, 2001); for instance, Rätsch *et al.* (2002) constructed a boosting algorithm for detecting outliers from a corresponding support vector algorithm.

Generally, boosting algorithms works as follows (Meir and Rätsch, 2003): given a training set S of m examples (x, y), the algorithms start by assigning weights $d_{i=1,\ldots,m}$ to each element in the set. Then the algorithms iteratively construct the T hypothesis in (1), so that for each iteration t the algorithms:

(1) Train a basic hypothesis h_t on the weighted data.
(2) Find the hypothesis weight α_t, which is found by minimizing the loss function $G(f_t + \alpha \cdot h_t, S)$.
(3) Set $f_{t+1} = f_t + \alpha_t \cdot h_t$.
(4) Update each data weight so that $d_i^{(t+1)} = \partial/[\partial f_{t+1}(x_i)]\, G(f_{t+1}, S), i \in \{1, \ldots, |S|\}$.

See Meir and Rätsch (2003) for further information on the loss functions used by different boosting algorithms.

Boosted GP

The idea of combining boosting algorithms with GP was first presented by Iba (1999). Here the training set for the GP algorithm was constructed by sampling from S, using the weight distribution D. Paris *et al.* (2001) extended this, by training on the complete set S, and including the data weights in the fitness function

$$\varepsilon(h, D, S) = \sum_{i=1}^{|S|} d_i \cdot |h(x_i) - y_i|. \qquad (2)$$

The GPboost algorithm presented by Paris *et al.* (2001) is based on AdaBoost, and was used to solve simple benchmark regression problems without noise. Since GPboost is based on the AdaBoost algorithm, it is expected that GPboost will have difficulties on datasets that contain noise and outliers. To alleviate this, we propose a regularized algorithm, GPboost$_{\text{Reg}}$, which is based on the AdaBoost$_{\text{Reg}}$ algorithm of Rätsch *et al.* (2001).

Our implementation uses the GP system described in the Genetic programming section as the base learner, with (2) as the fitness function. It produces classifiers of the form given in (1).

Variance in GP

Genetic programming is inherently a stochastic method. Consequently, two separate GP runs are not guaranteed to produce identical results. This also goes for boosted GP, hence there will be some variance in the predictions of different GPboost classifiers. This inherent variance can, however, be used to create even better classifiers. Hansen and Salamon (1990) showed that diverse and accurate classifiers can be combined to produce a more accurate classifier, which also has the added benefit of a reduced variance compared with the individual classifiers.

In our experiments, we create combined classifiers by taking the simple average of the individual classifiers. That is, for a set of classifiers $f_1, f_2, \ldots, f_n$, the combined classifier F is given by

$$F(x) = \frac{1}{n}\sum_{i=1}^{n} f_i(x). \tag{3}$$

Support vector machines

The classic SVM algorithm (C-SVM) was introduced by Vapnik (1995). In our experiments, we compare the performance of this algorithm and the ν-SVM algorithm by Schölkopf *et al.* (2000) with the performance of our GP system. Both SVM algorithms use radial basis functions as kernels. The implementation used in this study is part of the LIBSVM library (available at http://www.csie.ntu.edu.tw/∼cjlin/libsvm/).

While our GP system uses the oligo sequences as input, the input to the SVMs must be numerical data. In our experiments we solve this by using three different encodings, where the first encoding simply transforms the sequences into numerical data by mapping *a*s to 0, *c*s to 0.33, *g*s to 0.67 and *t*s to 1. This encoding ensures that the letters are evenly distributed in the $[0, 1]$ attribute range suggested in the LIBSVM documentation.

The second encoding uses the relative number of each nucleotide and dinucleotide in the sequence. That is, the input is a 20-dimensional vector where the first element in the input is the relative number of *a*s in the sequence, the fifth element is the relative number of *aa*s, and so forth.

Our third encoding use the relative number of tetramers in each sequence. We use this representation because Matveeva *et al.* (2000) showed that certain tetramers correlate with antisense efficacy. In addition, Giddings *et al.* (2002) found that using the relative number of tetramers in the sequence as input to an artificial neural network classifier gave better results than using dimers or trimers.

Classifier accuracy

The predictive accuracy (or generalization accuracy) of a classifier is important, because it gives a measure of how well the classifier will perform on unseen data. That is, it estimates the probability that the classifier will correctly predict the class of unseen data. The classifier accuracy is given by

$$\mathrm{Acc} = \frac{\mathrm{Tp} + \mathrm{Tn}}{\mathrm{Tp} + \mathrm{Tn} + \mathrm{Fp} + \mathrm{Fn}}, \tag{4}$$

where Tp, Fp, Tn and Fn are the number of true positive, false positive, true negative and false negative predictions.

A 10-fold cross-validation is known to create a good estimate of the predictive accuracy of classification methods (Martin and Hirschberg, 1996). The average of the 10 testing accuracies from the cross-validation procedure estimates the predictive accuracy of a classifier trained on the original dataset.

In our experiments, we use cross-validation for both accuracy and parameter estimation. That is, we do a separate 10-fold cross-validation experiment on each of the 10 training sets to determine the optimal parameters of the classification algorithm. This is done to ensure that the training process—both parameter estimation and model generation—is completely independent of the test data.

An alternative to using the accuracy as the 10-fold estimate is to use the Matthews correlation:

$$M = \frac{\mathrm{Tp}\cdot\mathrm{Tn} - \mathrm{Fp}\cdot\mathrm{Fn}}{\sqrt{(\mathrm{Tn}+\mathrm{Fn})(\mathrm{Tn}+\mathrm{Fp})(\mathrm{Tp}+\mathrm{Fn})(\mathrm{Tp}+\mathrm{Fp})}}. \tag{5}$$

Matthews (1975) points out that this correlation measure is often a more informative quality measure than the accuracy.

The receiver operating characteristic (ROC) is an even more general quality measure (Hanley and McNeil, 1982). A ROC curve is a plot of the sensitivity Se versus the specificity Sp of a given classifier, where

$$\mathrm{Sp} = \frac{\mathrm{Tn}}{\mathrm{Tn} + \mathrm{Fp}}, \quad \text{and} \tag{6}$$

$$\mathrm{Se} = \frac{\mathrm{Tp}}{\mathrm{Tp} + \mathrm{Fn}}. \tag{7}$$

A ROC curve can be characterized by the area under its curve: perfect classification gives an area of 1.0, and random classification gives an area of 0.5. The ROC area or ROC score gives a good indication of whether one classifier has a higher overall performance than another classifier. One can also use sign tests on the classifiers's output (Salzberg, 1997) for varying specificity levels to test whether one classifier is significantly better on the given specificity level. In this work, we focus these tests on the high specificity levels, since in practical applications, users will in most cases only be interested in the oligos that are highest ranked. That is, they want few false positives among the highest ranked oligos (on high specificity levels), and do not care if there are some false negatives among the oligos of low rank (on low specificity levels).

All the above measures assumes that the examples to be classified have been characterized as being either positive or negative sequences. The correlation coefficient R between the predicted and the observed efficacy is an alternative quality measure that does not depend on such a characterization. R^2 represents the proportion of variation in the observed efficacy that can be explained by the classifiers. A Student's t-test gives the statistical significance of a given correlation.

Oligo thermodynamics

We use the nearest-neighbour model (Xia *et al.*, 1998) to compute the oligo/mRNA duplex stability. The model is used in several of our custom applications that in addition to the overall duplex stability, also compute the oligo 5′ stability and the difference in oligo 5′ and oligo 3′ stability.

The oligo 5′ stability is computed from the first five nucleotides in the oligo (antisense) sequence. When computing the difference in 5′ and 3′ stability, we compute the stability of

Table 1. The average results from 10 different 10-fold cross-validation experiments using the genetic programming algorithms on the antisense data

Algorithm	ROC	R	P	M	Se	Sp
GP	N/A	-0.18 ± 0.06	1×10^{-3}	0.15 ± 0.05	0.34 ± 0.05	0.79 ± 0.04
GPboost	0.71 ± 0.01	-0.37 ± 0.02	2×10^{-11}	0.31 ± 0.03	0.61 ± 0.03	0.70 ± 0.03
GPboost$_{\text{Reg}}$	0.71 ± 0.02	-0.38 ± 0.04	4×10^{-12}	0.29 ± 0.05	0.59 ± 0.05	0.71 ± 0.02

R is the correlation and P is its statistical significance. M, Se and Sp are the Matthews correlation, sensitivity and specificity when using the the standard cutoff for the algorithms (≥ 0 for the boosted GP algorithms). Note that because the standard GP classifiers are binary, no ROC score is given for this algorithm.

the first four (5′) and the last four (3′) nucleotides in the oligo sequence. The difference is given by the 5′ stability minus the 3′ stability, and a negative value means that the 5′ end of the antisense strand is more stable than the 3′ end.

We use the parameters of Xia *et al*. (1998) in all our RNA/RNA nearest-neighbour model computations, except for the 5′ stability calculations where we use the same parameters as that of Khvorova *et al*. (2003). For DNA/RNA interactions, we use the parameters of Sugimoto *et al*. (1995).

The Mfold web server of Zuker (2003) is used to compute the oligo secondary structure stability.

RESULTS AND DISCUSSION

The following sections detail the results from our experiments at predicting siRNA and antisense efficacy. Furthermore, we show how different thermodynamic properties correlate with the observed efficacies sequences, and that effective antisense and siRNA oligos have different thermodynamic properties. We also show that the genetic programming classifiers for the antisense and siRNA oligos differ. Finally, we discuss the possibilities of using our predictors in the genomewide oligo design.

Antisense predictions

We analyzed the antisense data using the GP, GPboost and GPboost$_{\text{Reg}}$ algorithms, and Table 1 shows the average results of 10 different 10-fold cross-validation experiments for these algorithms. The GP solutions were binary classifiers of the form given in Figure 1 and the boosted GP classifiers had the form given in (1). Although all three methods have a significant correlation between the predicted and observed efficacy, the boosted GP algorithms get better results than the standard algorithm.

Table 2 compares the accuracy of the three input encodings for the two SVM algorithms with the accuracy of the GP-based algorithms. The GP-based algorithms use the average of 10 different models as output (3); i.e. the basic GP models are the average of 10 binary classifiers of the form given in Figure 1, and the boosted GP classifiers are the average of 10 classifiers of the form given in (1). ROC curves for the GP-based algorithms were generated by measuring the sensitivity and specificity for different thresholds of the average classifiers (the average GP classifier has 10 different thresholds; the average boosted GP classifiers have more). This table shows that the two boosting algorithms have the best overall results in terms of ROC score and correlation. They also get the best Matthews correlation. Figure 2 shows the ROC graphs for the GPboost$_{\text{Reg}}$ algorithm and the input-encodings to the ν-SVM and C-SVM algorithms that have the highest ROC-score. To test whether the GPboost$_{\text{Reg}}$ algorithm had a significant better performance than the other two classifiers, we used a sign test (Salzberg, 1997) on specificities 0.95, 0.9, 0.8 and 0.6 (for a motivation see Classifier Accuracy section). The GPboost$_{\text{Reg}}$ algorithm was significantly better ($P \leq 0.05$) than the ν-SVM

Table 2. Antisense results

Algorithm	ROC	R	P	M	Se	Sp
GP	0.61	-0.28	5.2×10^{-7}	0.22 ± 0.20	0.35	0.83
GPboost	0.75	-0.45	8.1×10^{-17}	0.38 ± 0.20	0.64	0.75
GPboost$_{\text{Reg}}$	0.76	-0.46	1.7×10^{-17}	0.43 ± 0.18	0.66	0.77
ν-SVM(seq)	0.54	-0.04	0.45	0.01 ± 0.14	0.09	0.93
C-SVM(seq)	0.52	-0.01	0.86	0.03 ± 0.13	0.10	0.94
ν-SVM(1–2)	0.68	-0.35	5.6×10^{-10}	0.28 ± 0.15	0.45	0.82
C-SVM(1–2)	0.67	-0.37	1.6×10^{-11}	0.16 ± 0.24	0.26	0.89
ν-SVM(4)	0.66	-0.31	2.8×10^{-8}	0.23 ± 0.17	0.50	0.72
C-SVM(4)	0.69	-0.40	2.2×10^{-13}	0.26 ± 0.13	0.36	0.85

See Table 1 for table header explanations.

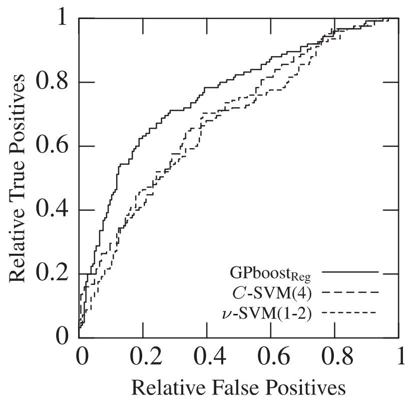

Fig. 2. ROC graphs for the GPboost$_{\text{Reg}}$, and best SVM classifiers on the antisense data. The GPboost$_{\text{Reg}}$ algorithm has the highest sensitivity for almost all specificity levels.

Table 3. The average results of 10 different runs of the genetic programming based algorithms on the siRNA data

Algorithm	ROC	*R*	*P*	*M*	Se	Sp
GP	N/A	-0.19 ± 0.05	7×10^{-3}	0.16 ± 0.05	0.47 ± 0.05	0.68 ± 0.05
GPboost	0.65 ± 0.03	-0.31 ± 0.03	6×10^{-6}	0.21 ± 0.07	0.58 ± 0.04	0.63 ± 0.06
$\text{GPboost}_{\text{Reg}}$	0.67 ± 0.03	-0.33 ± 0.05	1×10^{-6}	0.22 ± 0.08	0.59 ± 0.04	0.63 ± 0.05

See Table 1 for table header explanations.

algorithm on specificities 0.95 and 0.9, and significantly better than the *C*-SVM algorithm on specificities 0.9 and 0.8.

Another point worth considering is that using (3) to combine several GP-based classifiers do result in better classifiers. This is evident from comparing the results in Tables 1 and 2: the results for the GP-based classifiers in Table 2 are consistently better than the corresponding results in Table 1; the GP-based classifiers in Table 2 use the algorithm from (3).

Giddings *et al*. (2002) observed that using the complete tetramer vector gave poor results. To reduce the risk of overfitting, they went on to consider only the 40 most significant tetramers in the dataset. However, the cross-validation procedure does not remain independent when the complete dataset is used to select the input. Consequently, one would expect that the cross-validation accuracy estimate becomes positively biased; i.e. the estimates are better than they would be if tested on a truly independent test set. Indeed, when subsequently testing their classifiers on an independent test set, Giddings *et al.* found that the accuracy was lower than the reported 10-fold cross-validation accuracy.

To test this hypothesis of a possible bias, we did an experiment where we first estimated the 10-fold cross-validation accuracy using the 40 most significant tetramers in the global dataset, as measured by the Mathews correlation coefficient (5). This is analogous to the experiments of Giddings *et al*. (2000). We then did a 10-fold cross-validation experiment where we used the 40 most significant tetramers in each local training set as input to the SVM classifiers. When we compared the results of these two experiments, we found that although the local top 40 estimates gave results similar to using the complete tetramer input, the global top 40 estimate gave much higher accuracy estimates (ROC score, correlation and Matthews correlation of 0.77, −0.47 and 0.39). Hence, using the 40 most significant tetramers in the dataset does give a biased estimate of the classifier accuracy.

siRNA predictions

We did the same analysis on the siRNA data as we did on the antisense data. Table 3 shows the average results of 10 different runs of the GP, GPboost and $\text{GPboost}_{\text{Reg}}$ algorithms (please confer the relevant Methods sections and the comments to Tables 1 and 2 for details on the different GP classifiers). The results are similar to the antisense results (confer Table 1).

Table 4. siRNA results

Algorithm	ROC	*R*	*P*	*M*	Se	Sp
GP	0.62	−0.27	9.4×10^{-5}	0.19 ± 0.09	0.45	0.73
GPboost	0.70	−0.42	4.0×10^{-10}	0.27 ± 0.18	0.56	0.70
$\text{GPboost}_{\text{Reg}}$	0.72	−0.46	4.5×10^{-12}	0.24 ± 0.24	0.50	0.73
ν-SVM(seq)	0.57	−0.05	0.48	0.09 ± 0.23	0.52	0.56
C-SVM(seq)	0.53	−0.02	0.78	0.05 ± 0.20	0.35	0.70
ν-SVM(1–2)	0.70	−0.21	2.6×10^{-3}	0.31 ± 0.19	0.61	0.68
C-SVM(1–2)	0.64	−0.30	1.3×10^{-5}	0.26 ± 0.14	0.45	0.80
ν-SVM(4)	0.62	−0.14	0.046	0.25 ± 0.16	0.49	0.73
C-SVM(4)	0.65	−0.30	1.3×10^{-5}	0.19 ± 0.20	0.35	0.82

See Table 1 for table header explanations.

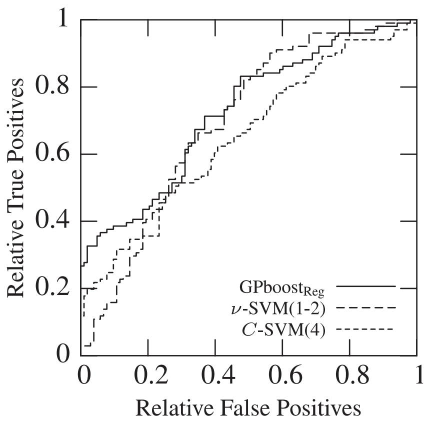

Fig. 3. ROC graphs for the $\text{GPboost}_{\text{Reg}}$, and best SVM classifiers on the siRNA data.

Table 4 summarizes the performance of the SVM algorithms and GP-based algorithms. Just as in the antisense analysis, the GP-based algorithms use the average of 10 different models as the final output. We also used the same input to the SVM algorithms here as previously. The two boosting algorithms achieve the best overall results in terms of ROC score and correlation, which was also the case for the antisense experiments. But in terms of the Matthews correlation, the ν-SVM classifiers that use the nucleotide and dinucleotide percentage input are now the highest. Figure 3 shows the ROC graphs for the $\text{GPboost}_{\text{Reg}}$ algorithm and the input-encodings to the ν-SVM and C-SVM algorithms that have the highest

Table 5. The correlation of structural features with siRNA and antisense efficacy (see Methods section for details on thermodynamic computations)

Structural feature	siRNA sequences			Antisense sequences		
	oligos	R	P	oligos	R	P
oligo 5′ stability	204	−0.21	2.6×10^{-3}	311	0.11	0.053
oligo 5′ − oligo 3′ stability	204	**−0.28**	5.0×10^{-5}	311	0.051	0.37
ΔG°_{37} oligo 2D structure	203	**−0.26**	1.8×10^{-4}	302	−0.10	0.083
ΔG°_{37} oligo/target duplex	204	**−0.14**	0.046	311	**0.20**	3.9×10^{-4}
Oligos with oligo/target duplex stability $\Delta G^\circ_{37} \leq -30$ kcal/mol						
ΔG°_{37} oligo 2D structure	176	**−0.28**	1.7×10^{-4}	84	**−0.44**	2.8×10^{-5}
ΔG°_{37} oligo/target duplex	177	**−0.23**	2.1×10^{-3}	89	−0.011	0.92
Oligos with oligo/target duplex stability $\Delta G^\circ_{37} > -30$ kcal/mol						
ΔG°_{37} oligo 2D structure	27	−0.35	0.073	218	0.055	0.42
ΔG°_{37} oligo/target duplex	27	−0.26	0.19	222	**0.37**	1.3×10^{-8}

Significant correlations R ($P = 0.05$) are marked in boldface.

ROC score. A sign test on specificities 0.95, 0.9, 0.8 and 0.6 showed that the GPboost$_{\text{Reg}}$ algorithm was significantly better ($P \leq 0.05$) than the ν-SVM algorithm on specificities 0.95 and 0.9, and significantly better than the C-SVM algorithm on specificity 0.95.

As a final note, comparing Tables 3 and 4 again shows that combining several GP-based classifiers results in better classifiers.

Structural feature predictions

Several researchers have pointed at correlations between different structural features of an oligo and its efficacy. For antisense oligos, Matveeva *et al*. (2003) have identified a significant correlation between the antisense efficacy and the free energy ΔG°_{37} in the antisense and RNA target duplex formation. They also found a significant correlation between the antisense efficacy and the free energy in the predicted antisense secondary structure. Finally, they found that oligos that form less stable duplexes with target RNA depend more on high duplex stability to be effective. Conversely, ODNs that form highly stable duplexes depend on smaller self-interaction potentials to be effective.

For siRNA oligos, Khvorova *et al*. (2003) found that the 5′ antisense region is significantly less stable in effective siRNAs than in ineffective siRNAs. Moreover, they observed different patterns in the internal stability profile for effective and ineffective siRNAs. At the same time, Schwarz *et al*. (2003) found that the 5′ antisense region was less stable than the 5′ sense region in functional siRNAs.

We tested these different structural features on both the antisense and siRNA data sets. Table 5 lists the correlation between each structural feature and the antisense and siRNA efficacy as well as its statistical significance. The details on the different thermodynamic computations are given in the Methods section.

Table 5 shows that siRNA and antisense oligos share some physical characteristics: both become less active when the self-interaction potential of an oligo is large. However, this seems to be the only common feature. The antisense and siRNA oligos seem to have complete opposite characteristics, both regarding 5′ and overall duplex stability. To determine whether the opposite characteristics could be explained by the differences in oligo length for the antisense data, we did a separate analysis on the antisense sequences of length 20 in the database (178 out of 311 sequences). But this analysis gave the same overall results as reported in Table 5 (data not shown).

In addition, we investigated the relationship between duplex stability and self interaction potential, found by Matveeva *et al*. (2003). Table 5 support these results, since it shows that the efficacy of antisense oligos that are more stable ($\Delta G^\circ_{37} \leq -30$) is negatively correlated with the free energy in the predicted secondary structure of oligos. Likewise, the efficacy of less stable oligos ($\Delta G^\circ_{37} > -30$) is positively correlated to the free energy in the oligo/mRNA duplex. We found a similar trend in the smaller set of antisense sequences of length 20. For the more stable duplexes, the P-values were 0.75 for the duplex stability and 0.085 for the self-interaction potential; for the less stable duplexes, the corresponding P-values were 2.4×10^{-5} and 0.83. We did not, however, find a similar trend in the siRNA sequences.

We note that a possible extension to the method presented here is to try to combine our sequence-based, boosted GP predictors with the thermodynamic characteristics. This is, however, left as an extension for further work.

Antisense and siRNA classifiers differ

As the analysis in the previous chapter showed, effective antisense and siRNA oligos have very different thermodynamic characteristics. To determine whether this difference also was reflected in the classifiers produced by our algorithm, we examined the binary classifiers created by the standard GP

Table 6. Examples of GP classifiers generated on the antisense and siRNA database

Data	Classifier	Example matches
ODNs	$\{c(g_6 \mid a_5 \mid c_8)(g_4 \mid a_2 \mid c_6)(g_3 \mid a_2 \mid c_8)(a_6 \mid c_2)tc : p >= 25\}$	*.ccca.., .gcca.., .acca.., .cgca..*
siRNAs	$\{gcct(c_1 \mid t_1)ct(a_1 \mid t_1)ct(t_2)(c_4 \mid t_1)(c_3 \mid t_1)(c_3 \mid t_1) : p >= 13\}$	.......*a..tccc*, ...*c.....tccc*

The example matches list some of the least complex subsequences matched by the corresponding expression (. represents any letter). Thus, we have not listed matches involving the characters that have the least weight, as for instance *cccaatc* for the ODN expression or *gcctcctacttt..* for the siRNA expression. These longer matches are more likely to be specific to the training set than to represent general characteristics of ODNs or siRNAs. Note that the expressions use the subscript operator to denote the weight of each letter in alternatives (see production 4 in Fig. 1 and the accompanying examples in the text).

algorithm (Fig. 1). Because of the complexity of the boosted GP classifiers [these classifiers are a weighted combination of the basic expressions—see (1)], a detailed analysis of these classifiers was not done.

Table 6 shows two typical examples of the expressions generated on the antisense and siRNA data, along with some of the subsequences matched by each expression. The two expressions illustrate the main difference between the antisense and siRNA expressions: the subsequences matched by the antisense expressions are shorter than the subsequences matched by the siRNA expressions. Indeed, in an experiment where we greatly increased the number of generations that the GP system was run, the length of the subsequences matched by the siRNA expressions increased to 19. This equals the length of the siRNA oligos. The length of the subsequences matched by the antisense expressions, however, remained the same (data not shown).

This difference in motif length between the antisense and siRNA expressions suggests that for effective siRNAs, the base composition at different positions in the oligo is important. This dependence on base composition is also reflected in the 5′ and 3′ thermodynamics of the siRNA oligos (Table 5). For effective antisense oligos, on the other hand, the occurence of specific, short motifs is important. This was also observed by Matveeva *et al.* (2000).

Genomewide oligo design

One of the important questions in genomewide oligo design is the tradeoff between oligo efficacy and oligo uniqueness. That is, for each gene in a genome one would like to have at least one oligo that is sufficiently unique so that it does not target any other genes, and has a high probability of being effective. As not all oligos predicted to be effective will be effective, one will need a set of candidate oligos for each gene.

To test how many potential candidate oligos our boosted GP classifiers predict to be effective, we downloaded 10 human mRNAs at random from the NCBI RefSeq database. We then tested all candidate 19mer oligos on the antisense and siRNA efficacy predictors, and counted the number of oligos where the predictor output were above a threshold corresponding to 90% specificity. This resulted in that on average 22% of the candididate siRNAs and 16% of the candidate antisense oligos for each mRNA were predicted to be effective. The SD in both cases were 4%.

We also determined the number of candidate oligos that were unique in the human transcriptome. To do this we computed the Hamming distance between each candidate oligo and every 19mer in the Ensembl cDNA database, NCBI build 34. Oligos that had a Hamming distance greater than two to all other 19mers in the database were considered unique. The relative number of unique oligos for each mRNA varied between 0.5 and 55%, with an average of 24%.

Each mRNA had, however, candidate oligos that were both unique and predicted to be effective, both in RNAi and antisense experiments. The number ranged from one candidate to 11% of all possible oligos. The number of candidate oligos can be increased by decreasing the specificity threshold, but as the result of the uniqueness screening shows, the limiting factor on the number of candidate oligos is the uniqueness screening. One should expect that this would also be the case for true, genomewide oligo design.

CONCLUSION

We have developed a GP-based machine learning system that successfully predicts the efficacy of oligos used in antisense and siRNA experiments. The system is consistently better than both SVMs and physical property-based classifiers at predicting the efficacy of both antisense and siRNA oligos.

We have also showed that siRNA and antisense sequences do not seem to share the same physical characteristics. The only common feature is a sensitivity to a high self-interaction potential of the oligo. That is, oligos that form more stable secondary structures are less likely to be effective when used in both antisense and RNAi experiments.

Antisense and RNAi are very different technologies, and, as shown here, antisense and siRNA oligos have very different physical characteristics. Despite this, our novel, sequence-based machine learning system can sucessfully predict the efficacy of both antisense and siRNA oligos. This suggests that the oligo efficacy for both technologies is in a large part determined by the oligo sequence itself.

REFERENCES

Banzhaf,W., Nordin,P., Keller,R.E. and Francone,F.D. (1997) *Genetic Programming: An Introduction: On the Automatic Evolution of Computer Programs and Its Applications*. Morgan Kaufman, San Francisco, CA.

Chalk,A.M. and Sonnhammer,E.L. (2002) Computational anti-sense oligo prediction with a neural network model. *Bioinformatics*, **18**, 1567–1575.

Freund,Y. and Schapire,R.E. (1997) A decision-theoretic generalization of on-line learning and an application to boosting. *J. Comput. Sys. Sci.*, **55**, 119–139.

Giddings,M.C., Matveeva,O.V., Atkins,J.F. and Gesteland,R.F. (2000) ODNBase—a web database for antisense oligonucleotide effectiveness studies. *Bioinformatics*, **16**, 843–844.

Giddings,M.C., Shah,A.A., Freier,S., Atkins,J.F., Gesteland,R.F. and Matveeva,O.V. (2002) Artificial neural network prediction of antisense oligodeoxynucleotide activity. *Nucleic Acids Res.*, **30**, 4295–4304.

Halaas,A., Svingen,B., Nedland,M., Sætrom,P., Snøve,O. and Birkeland,O. (2004) A recursive MISD architecture for pattern matching. IEEE Trans. Very Large Scale Integrat. (VLSI) Syst., **12**, 727–734.

Hanley,J. and McNeil,B.J. (1982) The meaning and use of the area under a receiver operating characteristic (ROC) curve. *Radiology*, **143**, 29–36.

Hansen,L.K. and Salamon,P. (1990) Neural network ensembles. *IEEE Trans. Pattern Anal. Machine Intell.*, **12**, 993–1001.

Harborth,J., Elbashir,S.M., Vandenburgh,K., Manninga,H., Scaringe,S.A., Weber,K. and Tuschl,T. (2003) Sequence, chemical and structural variation of small interfering RNAs and short hairpin RNAs and the effect on mammalian gene silencing. *Antisense Nucleic Acid Drug Dev.*, **13**, 83–106.

Holen,T., Amarzguioui,M., Wiiger,M.T., Babaie,E. and Prydz,H. (2002) Positional effects of short interfering RNAs targeting the human coagulation trigger Tissue Factor. *Nucleic Acids Res.*, **30**, 1757–1766.

Holland,J.H. (1975) *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor, MI.

Iba,H. (1999) Bagging, boosting and bloating in genetic programming. In Banzhaf,W., Daida,J., Eiben,A.E., Garzon,M.H., Honavar,V., Jakiela,M. and Smith,R.E. (eds). *Proceeding of GECCO'99*, Volume 2. Morgan Kaufmann, pp. 1053–1060.

Interagon AS (2002) The Interagon query language: a reference guide.

Kawasaki,H., Suyama,E., Iyo,M. and Taira,K. (2003) siRNAs generated by recombinant human Dicer induce specific and significant but target site-independent gene silencing in human cells. *Nucleic Acids Res.*, **31**, 981–987.

Khvorova,A., Reynolds,A. and Jayasena,S.D. (2003) Functional siRNAs and miRNAs exhibit strand bias. *Cell*, **115**, 209–216.

Koza,J.R. (1992) *Genetic Programming: On the Programming of Computers by Natural Selection*. MIT Press, Cambridge, Massachusetts.

Kurreck,J. (2003) Antisense technologies. Improvements through novel chemical modifications. *Eur. J. Biochem.*, **270**, 1628–1644.

Martin,J.K. and Hirschberg,D.S. (1996) Small sample statistics for classification error rates. I: error rate measurements. *Technical Report 96-21*, ICS Department, UC Irvine.

Matthews,B.W. (1975) Comparison of the predicted and observed secondary structure of T4 phage lysozyme. *Biochim. Biophys. Acta*, **405**, 442–451.

Matveeva,O.V., Tsodikov,A.D., Giddings,M., Freier,S.M., Wyatt,J.R., Spiridonov,A.N., Shabalina,S.A., Gesteland,R.F. and Atkins,J.F. (2000) Identification of sequence motifs in oligonucleotides whose presence is correlated with antisense activity. *Nucleic Acids Res.*, **28**, 2862–2865.

Matveeva,O., Mathews,D., Tsodikov,A., Shabalina,S., Gesteland,R., Atkins,J. and Freier,S. (2003) Thermodynamic criteria for high hit rate antisense oligonucleotide design. *Nucleic Acids Res.*, **31**, 4989–4994.

McManus,M. and Sharp,P. (2002) Gene silencing in mammals by small interfering RNAs. *Nat. Rev. Genet.*, **3**, 737–747.

Meir,R. and Rätsch,G. (2003) An introduction to boosting and leveraging. In Mendelson,S. and Smola,A. (eds), *Advanced Lectures on Machine Learning*, Volume 2600. Springer-Verlag, pp. 118–183.

Montana,D.J. (1995) Strongly typed genetic programming. *Evol. Comput.*, **3**, 199–230.

Müller,K.-R., Mika,S., Rätsch,G. and Tsuda,K. (2001) An introduction to kernel-based learning algorithms. *IEEE Trans. Neural Networks*, **12**, 181–201.

Paris,G., Robilliard,D. and Fonlupt,C. (2001) Applying boosting techniques to genetic programming. In Collet,P., Fonlupt,C. Hao,J.-K. Lutton,E. and Schoenauer,M. (eds), *Proceeding of Artificial Evolution: 5th International Conference,* Springer-Verlag, pp. 267–280.

Rätsch,G., Mika,S., Schölkopf,B. and Müller,K.-R. (2002) Constructing boosting algorithms from SVMs: an application to one-class classification. *IEEE Trans. Pattern Anal. Machine Intell.*, **24**, 1184–1199.

Rätsch,G., Onoda,T. and Müller,K.-R. (2001) Soft margins for AdaBoost. *Machine Learning*, **42**, 287–320.

Rätsch,G., Schökopf,B. Smola,A., Müller,K.-R., Onoda,T. and Mika,S. (2000) ν-arc: Ensemble learning in the presence of outliers. In Kearns,M.S., Solla,S.A. and Cohn,D.A. (eds), *Advances in Neural Information Processing Systems 12: Proceedings of NIPS'99*. MIT Press.

Salzberg,S. (1997) On comparing classifiers: pitfalls to avoid and a recommended approach. *Data Mining Knowl. Discov.*, **1**, 317–328.

Scherer,L. and Rossi,J. (2003) Approaches for the sequence-specific knock-down of mRNA. *Nat. Biotechnol.*, **21**, 1457–1465.

Schölkopf,B., Smola,A., Williamson,R. and Bartlett,P.L. (2000) New support vector algorithms. *Neural Comput.*, **12**, 1207–1245.

Schwarz,D.S., Hutvágner,G. Du,T., Xu,Z., Aronin,N. and Zamore,P.D. (2003) Asymmetry in the assembly of the RNAi enzyme complex. *Cell*, **115**, 199–208.

Sugimoto,N., ichi Nakano,S., Katoh,M., Matsumura,A., Naka-muta,H. Ohmichi,T., Yoneyama,M. and Sasaki,M. (1995) Thermodynamic parameters to predict stability of RNA/DNA hybrid duplexes. *Biochemistry*, **34**, 11211–11216.

Vapnik,V. (1995) *The Nature of Statistical Learning Theory*. Springer, NY.

Vickers,T.A., Koo,S., Bennett,C.F., Crooke,S.T., Dean,N.M. and Baker,B.F. (2003) Efficient reduction of target RNAs by small interfering RNA and RNase H-dependent antisense agents. A comparative analysis. *J. Biol. Chem.*, **278**, 7108–7118.

Xia,T., SantaLucia,Jr,J., Burkard,M.E. Kierzek,R. Schroeder,S.J., Jiao,X. Cox,C. and Turner,D.H. (1998) Thermodynamic parameters for an expanded nearest-neighbor model for formation of RNA duplexes with Watson-Crick base pairs. *Biochemistry*, **37**, 14719–14735.

Zuker,M. (2003) Mfold web server for nucleic acid folding and hybridization prediction. *Nucleic Acids Res.*, **31**, 3406–3415.