Published online July 20, 2004

3826–3835 *Nucleic Acids Research, 2004, Vol. 32, No. 13*
doi:10.1093/nar/gkh713

Discovery of sequence motifs related to coexpression of genes using evolutionary computation

Gary B. Fogel, Dana G. Weekes, Gabor Varga[1], Ernst R. Dow[1], Harry B. Harlow[1], Jude E. Onyia[1] and Chen Su[1,*]

Natural Selection, Inc., 3333 N. Torrey Pines Ct., Suite 200, La Jolla, CA 92037, USA and [1]Lilly Research Laboratories, A Division of Eli Lilly and Company, Lilly Corporate Center, Indianapolis, IN 46285, USA

Received April 21, 2004; Revised May 28, 2004; Accepted July 1, 2004

ABSTRACT

Transcription factors are key regulatory elements that control gene expression. Recognition of transcription factor binding site (TFBS) motifs in the upstream region of coexpressed genes is therefore critical towards a true understanding of the regulations of gene expression. The task of discovering eukaryotic TFBSs remains a challenging problem. Here, we demonstrate that evolutionary computation can be used to search for TFBSs in upstream regions of genes known to be coexpressed. Evolutionary computation was used to search for TFBSs of genes regulated by octamer-binding factor and nuclear factor kappa B. The discovered binding sites included experimentally determined known binding motifs as well as lists of putative, previously unknown TFBSs. We believe that this method to search nucleotide sequence information efficiently for similar motifs will be useful for discovering TFBSs that affect gene regulation.

INTRODUCTION

Microarray analysis of gene expression in different cells provides a means to identify genes that share similar expression over time and/or biochemical treatment. These expression patterns may be important biomarkers for cell type and/or phenotypic behavior. Under the assumption that coexpressed genes share coregulatory elements, examination of the upstream regions for these genes may lead to the identification of common regulatory mechanisms (1–7). These regulatory signals may take the role of common transcription factor binding sites (TFBSs) that facilitate common gene expression.

Several computational methods for the discovery of TFBSs have been offered in the literature (8). These can be divided into two main approaches (9). The first approach makes use of an exhaustive calculation to evaluate the frequency of occurrence of all possible sequences of length n (n-mers) using an iterative process to update a nucleotide probability matrix. A background sequence distribution (of n-mers found in a set of non-coregulated genes) is used as a basis for comparison and any n-mers that are more overly abundant than expected are saved for further analysis as putative TFBSs (10–15). This approach has several deficiencies, including a lack of allowing flexible substitutions in the matching segments and the maximum length of $n \leqslant 7$ nt to achieve computation in reasonable time. However, this method is guaranteed to find n-mers with the highest Z-scores (a length-and-composition corrected measure of the similarity) as all possible length $n \leqslant 7$ nt are examined exhaustively (9).

A second approach specifies the n-mer as a probability matrix and utilizes an iterative algorithm, typically represented as an expectation maximization (16) or Gibbs sampling procedure (9,17–22). These iterative methods can extend the window size to lengths $n > 8$ nt but are susceptible to entrapment in locally optimal solutions.

Both of these approaches are sensitive to the low signal-to-noise ratio of the short TFBS n-mer sequences relative to the length of the upstream region. Portions of the upstream region that do not contain true TFBSs are considered as noise. When applied to sequence data from higher eukaryotes, the signal-to-noise ratio can be quite low due to the considerable length of many intergenic regions. Improved models of the background sequence distribution using higher-order Markov models (23) to represent the intergenic sequences can assist in increasing the signal-to-noise ratio presented to the discovery tool, but these models are organism (or perhaps even gene cluster) specific and need to be regenerated for every cluster that is to be interrogated. N-th order Markov models that work particularly well for some sequences might work poorly on others. In some cases, the number of correctly predicted motifs can only be marginally affected by the complexity of the background model (23). An approach to motif identification that is not organism-specific, does not require exhaustive calculation, and can report back to the user putative TFBS in terms of a nucleotide likelihood matrix or n-mer alignment could avoid some of these pitfalls.

In this paper, we present a method to find similar nucleotide motifs in upstream regions of previously clustered, coexpressed genes with motif length $n > 7$, where the output is to be viewed either as a nucleotide likelihood matrix or as a non-gapped multiple sequence alignment. The number of possible nucleotide likelihood matrices P increases as

$$P = (L - x)^S, \quad \mathbf{1}$$

where S is the number of sequences to be interrogated, L is the length of the sequences and x is the length of the window being

*To whom correspondence should be addressed. Tel: +1 317 277 9657; Fax: +1 317 276 6009; Email: chen_su@lilly.com

used. A method to search this space and converge rapidly on the set of optimal motifs is required. One way to approach the task of *de novo* identification of conserved TFBSs is to search for common sequence motifs across multiple upstream sequences without pre-alignment using methods of evolutionary computation.

Evolutionary computation

Simulated evolution on a computer provides a method of optimization suitable for a wide variety of complex problems that may have multiple local optima, might vary with time or have a low signal-to-noise ratio. Evolutionary computation (EC) includes methods of genetic algorithms (GAs) (24,25), evolutionary programming (EP) (26), evolution strategies (ESs) (27) and other derivatives of these techniques, such as genetic programming (GP) (28) and classifier systems (29,30). Although each of these methods is slightly different, they are broadly similar. All evolutionary algorithms (EAs) maintain a population of contending solutions to be generated (31–33). Each solution in the population is then scored with respect to a measure of its worth or 'fitness'. The solutions of low fitness are most likely to be removed from the population and not used in future generations than solutions of higher fitness during a process of selection. Following selection, the surviving solutions are used to generate new contending solutions with random variation until the population size is re-established. The process of variation and selection is iterated for a specified number of generations or until the population has converged into a solution of adequate worth. The methods of EC differ on the choices of representation for solutions, the manner in which variation is applied (e.g. recombination and/or mutation operators), and the forms of selection that are used [e.g. proportional selection, truncation selection and tournament selection (30)]. These algorithms have been shown to generate robust performance across a wide range of functional optimization problems (32,34,35).

More specifically, EC proceeds as follows:

(i) An initial population of solutions is generated. This can be accomplished by selecting solutions at random and/or by accepting solutions provided by other algorithms or human operators. These 'hints' can be incorporated directly into the search.
(ii) These parent solutions are scored in light of an objective function.
(iii) The offspring solutions are generated from the parent solutions. Various operators are used to generate offspring. Typical operators include (a) random variation of individual components of a single parent solution and (b) random recombination of multiple components from two (or more) parent solutions.
(iv) The offspring solutions are scored in a manner similar to that of their parents (see Step ii).
(v) Selection is applied to the collection of parents and offspring. According to a selected rule, a subset of this collection is chosen to become parents of the next generation. Possible selection rules include truncation selection (i.e. eliminate solutions below a certain threshold ranking in the population) and proportional selection (i.e. the expected number of copies of each solutions is directly proportional to their relative fitness) among many others.
(vi) If the available time has expired or a solution of sufficient worth has been discovered, then terminate the process; else proceed to Step iii.

The speed of EC, in terms of rate of convergence to optimal solutions, can be accelerated by optimizing a number of different evolutionary parameters, including (i) the population size, (ii) the type and frequency of random variation imposed, (iii) the degree of stringency of selection applied, (iv) the evaluation function and (v) the representation specified. The parameters act in concert to generate an overall system performance.

DATABASE

The TRANSFAC (36) and COMPEL (37) databases were used to identify two test cases each with multiple sequences and experimentally derived TFBSs that could be used for the purpose of algorithm evaluation. TRANSFAC provides information for single transcription factors, providing information on cofactors if these are known. COMPEL provides composite regulatory elements containing two closely located TFBSs, which are considered as minimal functional units in combinatorial transcription regulation. Thus, COMPEL can be considered as a subset of TRANSFAC with additional annotation with respect to complexes of TFBSs and their function. COMPEL contains 178 experimentally validated composite elements from which we chose two transcription factors, octamer-binding factor (Oct) and nuclear factor kappa B (NF-κB) as test examples. These two test cases were chosen because their binding mechanisms are well studied and many known genes in the downstream regulatory pathways have been verified experimentally. In addition, both of them represent families with multiple family members binding to slightly different DNA motifs, thus increasing the difficulty of the motif search. These two factors interact with other factors of different regulatory roles, however, our goal was to focus on one factor at a time as a first step toward the prediction of putative TFBSs. In these two test cases, we selected genes with known binding sites for Oct and NF-κB in their 1 kb upstream region (Table 1). The test cases contained nucleotide sequences representing the 1 kb region upstream to the transcription start site (TSS) for each of these genes. Occasionally, the binding sites occurred downstream of the TSSs and in these cases the 1 kb region was shifted in the 3′ direction 100 nt at a time until the binding sites were within the 1 kb region. The seven Oct motifs in Table 1 share a high degree of similarity (92.9%) and these sequences can be represented by the consensus motif ATGYAAAD. The nine NF-κB motifs in Table 1 were less conserved (68.8%) and can be loosely represented by the consensus motif GDVNWDYY.

ALGORITHM

The code for EC was written in C++ for the evolution of similar TFBS motif 'windows', one per each sequence in each of the control examples. The sequence information contained in the grouping of these windows was used to calculate

Table 1. Genes with experimentally validated Oct or NF-κB binding sites

Transcription factor	Gene name	COMPEL entry	Species	Position	Strand	Sequence motif
Oct	U2 small nuclear RNA	C00039	*Homo sapiens*	−230 to −214	(+)	ATGCAAAT
Oct	Unknown protein	C00043	Mouse mammary tumor virus	−89 to −49	(+)	ATGTAAAT
Oct	Surface antigen	C00048	Human hepatitis B virus	−86 to −51	(−)	ATGTAAAT
Oct	Immunoglobulin heavy chain	C00049	*Mus musculus*	−116 to −98	(+)	ATGCAAAT
Oct	Immunoglobulin heavy chain	C00050	*Mus hortulans*	−68 to −48	(+)	ATGCAAAG
Oct	Interleukin-2	C00158	*M.musculus*	−86 to −71	(+)	ATGTAAAA
Oct	Interleukin-3	C00169	*H.sapiens*	185 to 206	(+)	ATGCAAAT
NF-κB	ELAM-1	C00097	*H.sapiens*	−154 to −84	(+)	GGGGATTT
NF-κB	Interferon-beta	C00099	*H.sapiens*	−98 to −53	(+)	GGGAAATT
NF-κB	Serum amyloid A2	C00100	*H.sapiens*	−179 to −82	(+)	GGACTTTC
NF-κB	Serum amyloid A1	C00101	*Rattus norvegicus*	−117 to −73	(−)	GACTTTCC
NF-κB	Interleukin-6	C00152	*H.sapiens*	−158 to −63	(+)	GGATTTTC
NF-κB	Serum amyloid A3	C00153	*M.musculus*	−166 to −67	(−)	GAAATGCC
NF-κB	ICAM-1	C00155	*H.sapiens*	−199 to −178	(−)	GAAATTCC
NF-κB	GM–CSF	C00156	*M.musculus*	−91 to −47	(+)	GAAATTCC
NF-κB	Interleukin-2	C00165	*H.sapiens*	−167 to −142	(−)	GTAGTTCC

For each listing, the COMPEL database number '(C00###)' is provided, as well as binding site sequence motif. The consensus sequence for Oct is ATGYAAAD. The consensus sequence for NF-κB is GDVNWDYY.

a nucleotide likelihood matrix. Fitness was measured with respect to a basis matrix where both a metric for similarity and a metric for complexity were used. Several strategies for selection and variation were evaluated. The following sections detail this process of TFBS discovery by EC.

Population initialization

For the purpose of this experimentation, we used a fixed TFBS window size of $w = 8$. A single candidate solution represents a set of windows placed (initially randomly) over the 1 kb upstream sequences with only one window per sequence. A user-defined number of P 'parent' solutions were created in this manner. A user-defined number of O 'offspring' solutions were created from these parent solutions using three methods of variation described below. The resulting P and O solutions were considered as one evolving 'population'.

Variation operators

When generating offspring solutions from parent solutions, the number of variation operators to apply was chosen at random from the range [1, 3]. This range ensured a minimum number of one and a maximum number of three variations relative to the parent (under the assumption that more than this number of mutations would be akin to developing a new individual at random rather than making use of the information contained in a parent solution). Three variation operators were developed: *Window Shift*, *Window Recombination* and *G+C% Slide*. Each of these variation operators was associated with a user-defined probability of use in generating a new offspring solution. These variation operators and their associated probabilities of use are outlined below. These probabilities were established through iterative testing and evaluation using 10 different settings with respect to the known TFBSs information. Although there is no guarantee that these same settings will work over all examples, they do work well for these test cases. Further refinement and testing of these settings on additional examples is the subject of further investigation.

Each time a variation operator was required, the choice on the type of variation was made with respect to the probabilities associated with the frequency of use for each variation type, defined by the user. The chosen variation operator was applied and if the number of variations to apply was >1, this process was repeated up to a user-defined maximum of three times to generate an offspring solution.

Variation operator #1—Window Shift

When generating an offspring solution from a parent solution, one window in the parent solution was chosen uniformly at random for modification. This window was moved either to the left or to the right at random with equal probability. Following a decision of a direction, a choice of movement distance was chosen at random from [1, N], where N represents the maximum number of nucleotides in that direction. For example, given a window of 8 nt placed at the 5′ end of a 1 kb sequence, the remaining sequence is represented by $N_{max} = 992$ nt (total sequence length − window length).

Variation operator #2—G+C% Slide

The G+C% for all upstream sequences was calculated using a sliding window of length 8 nt. A single window from a parent solution was chosen uniformly at random for modification. For the remaining windows in the parent solution, the average G+C% was calculated. A percentage similarity to this average was calculated for all possible window positions in the full 1000 nt sequence for the window being modified. Locations of window positions with G+C% similarity equal to or greater than a user-defined threshold percentage similarity (for instance >80%) were stored. These positions, representing regions in the sequence that most closely resembled the average G+C% content of the remaining windows in the parent solution, provided the highest potential for increased similarity in the resulting offspring solution. From this set of possible window placements, a new location for the window was chosen with equal probability for the offspring solution.

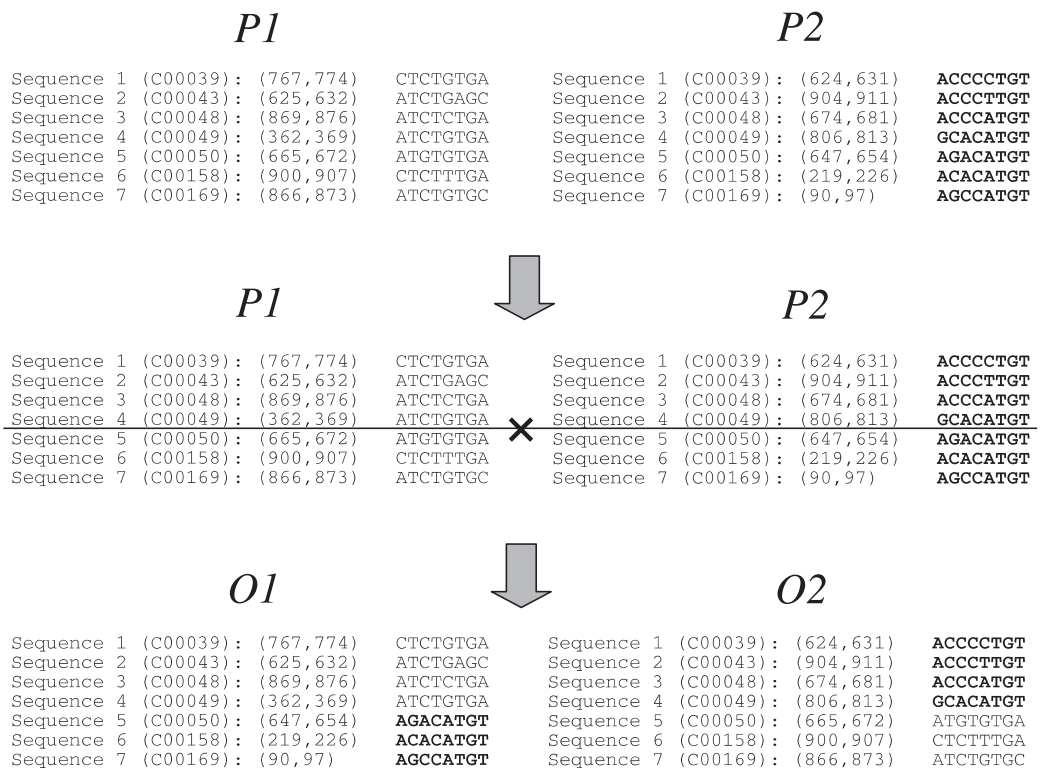

Figure 1. Window recombination. Each solution is represented by a set of sequences (depicted here as sequence 1–7) each with a COMPEL database identifier (C00###), starting and ending indices, and nucleotide sequence. Two parents (*P1* and *P2*) are selected at random from the population of solutions. A point of crossover is chosen at random (in this case between sequences 4 and 5 as denoted by an ×). At this point of crossover, information is exchanged between parent solutions to generate two new offspring solutions (*O1* and *O2*). One of these two offspring solutions is chosen at random with equal probability to be included in the resulting population as a new solution, while the other is discarded.

Variation operator #3—Window Recombination

With the Window Recombination operator (Figure 1), when generating an offspring solution from a parent solution (P_1), a second parent (P_2) was chosen at random from the population. A randomly chosen number of points of recombination r were selected from the range [1, $x - 1$], where x is the number of windows representing an individual (for Oct, $x = 7$ and for NF-κB, $x = 9$). The condition of one point of recombination was considered as 'one-point crossover'. The condition of more than one point of recombination was considered as 'multipoint crossover'. The placement of r recombination points was chosen at random with equal probability at each position in the set of sequences. In the case of multipoint crossover, the placement of r recombination points was made such that no two points of recombination shared an identical location in the set of sequences. Following exchange of sequences about the point(s) of recombination, two offspring solutions O_1 and O_2 were generated. One of these two offspring solutions was chosen at random with equal probability to be included in the resulting population. The other offspring was deleted. It should be noted that this operator only serves to shuffle the grouping of motifs rather than alter the sequences of motifs.

Fitness evaluation

Each member of the population was scored with respect to two criteria: overall similarity (S) and overall complexity (K). The objective of this process was to maximize the similarity of the sequence motifs found within each window while avoiding saturation of low complexity solutions. Initial investigations using only the similarity criteria resulted in rapid entrapment of the algorithm at solutions of very low complexity with very high similarity. The complexity term was added to counteract this process.

Calculation of similarity

Given the windows and their respective sequences contained within each individual, a nucleotide likelihood matrix was derived using the normalized frequency of A, T, G and C at each position over the range [0, 1]. Each column of the nucleotide likelihood matrix was compared with respect to the basis matrix that was closest to the nucleotide of highest frequency of occurrence within that column (Figure 2). The summed absolute value of the difference between these columns for the 4 nt represents the value of each column. The subtraction of this value from 1 provided a similarity score to be maximized. The sum of similarity scores over all columns represented the total similarity component (S) for each individual.

Calculation of complexity

Compositional complexity (K) of a window of length w can be calculated using the equation

$$K = \frac{1}{w \log_N(w!/\prod n_i!)}, \qquad \mathbf{2}$$

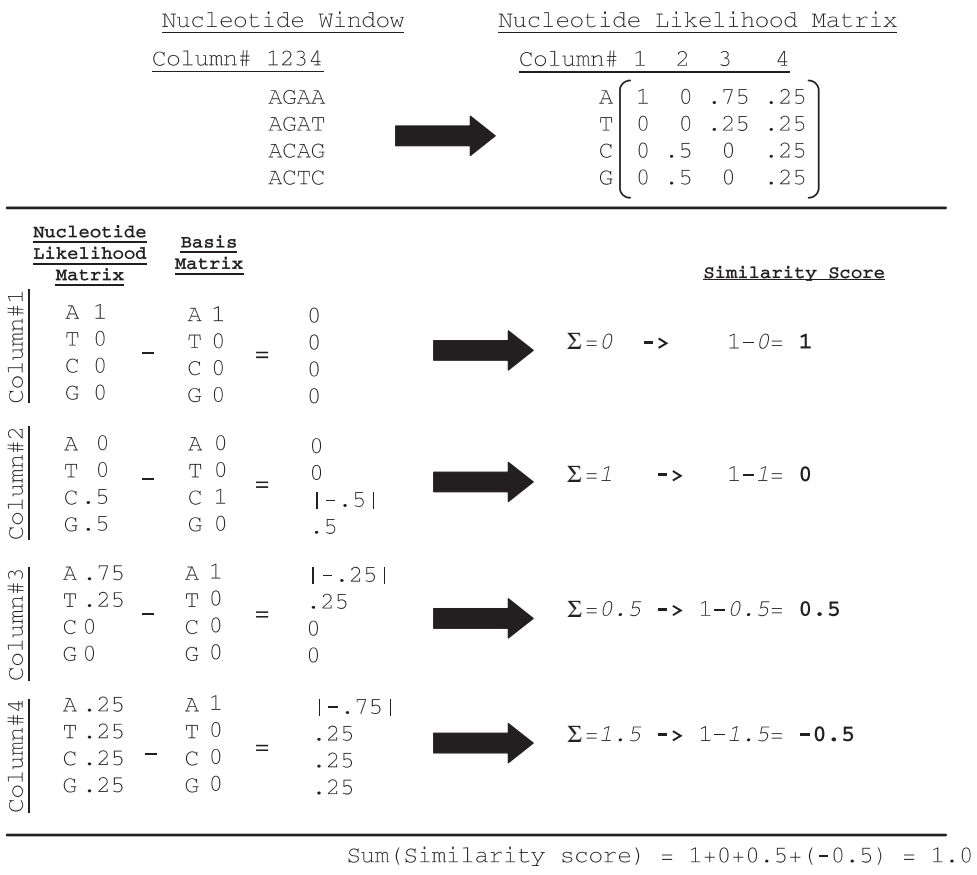

Figure 2. Calculation of similarity. Given a set of windows of length 4, a nucleotide likelihood matrix is calculated. Each column in the nucleotide likelihood matrix is then scored for similarity with respect to a basis matrix representing the complete conservation of the most prominent nucleotide. The first column of the nucleotide likelihood matrix (all As) is compared to the case of all As from the basis matrix, and the sum (Σ) of the absolute values in the subtraction matrix is calculated (in this case, $\Sigma = 0$). A difference of 1 minus this sum equals 1.0, for perfect similarity (all As). The second column of the nucleotide likelihood matrix (0.5 C, 0.5 G) is compared to either the C or the G basis matrix (in this case C was chosen at random). This process is iterated over all columns and the final summation of column scores (in this case 1.0) reflects the worth of the similarity component for this individual solution in the population.

where $n = 4$ for DNA sequences, n_i is the number of nucleotides of type i, where $i \in \{A, T, C, G\}$, for the w-mer window (38). This formula was used to compute compositional complexity for all windows in an individual solution. The average compositional complexity for all windows represented the total complexity score for each individual solution.

Total fitness

The total fitness (F) of each individual was calculated as:

$$F = w_1(S) + w_2(K), \quad \mathbf{3}$$

where the weights w_1 and w_2 were user-defined and fixed during the evolution. These weights were used to adjust the importance of these two terms in relation to fitness. The values for w_1 and w_2 were adjusted following a series of experiments on the control examples such that both control examples could be recapitulated.

Selection

Before using a method of selection, the population of solutions is interrogated for duplicate solutions by means of a direct pairwise comparison for both sequence information and positional indices. It is important to minimize the number of duplicate solutions during the evolutionary search to avoid premature stagnation on local optima and to maintain a large amount of variance in the population for best coverage of the response surface. There are several ways to increase population variance in the evolutionary optimization, and we have included one such approach with this algorithm. Upon discovery of a duplicate solution, one of the duplicates is given an artificially low fitness score. This artificial low-fitness score guarantees the elimination during elitist selection. When using tournament selection, duplicate solutions with low fitness are excluded from the tournament competition and eliminated from the population (see below).

Based on the fitness scores, a mechanism of selection (30) determines which individuals from the current population will be removed. Under an elitist selection approach, all individuals in the population are ranked with respect to fitness and the worst O individuals are discarded leaving in the remaining P individuals to serve as parents for the next generation. Under a tournament selection approach (30), each individual 'competes' with a user-defined number of randomly chosen other individuals in the same population. 'Competition' in this regard is a simple comparison of fitness. Each time the first individual's fitness is higher than (or equal to) the competitor's fitness, the first individual receives a 'win'. The number of

wins is recorded for all competitions and this process is iterated over all members of the population. All individuals are then ranked with respect to the number of wins during the tournament, rather than simply their original fitness values. Selection is used to remove the lower O individuals from this ranked list. In the case of a tie in the number of wins, those specific individuals are re-ranked by fitness before selection. After selection, the P remaining individuals are saved to serve as parents for the next generation of solutions.

Elitism guarantees that at each generation the solution with the best fitness is retained in the population. Tournament selection does not guarantee that at each generation the best solution is retained in the population, which might appear counterproductive as an optimization process. However, this selective method allows the solutions of low fitness to be retained for multiple generations. Such a strategy can avoid entrapment in the local optima by providing access to unusual solutions that may lead to more fertile portions of the fitness landscape.

Parallelization

To create an environment to test the utility of parallelization of the evolutionary procedure for this problem, an 'island model' was developed internal to each run of the EA. Within the settings file, a user-defined number of parallel 'islands' can be established as can the desired number of generations each island performs evolution in the isolation. After isolation, the best solutions from each of the islands can be pooled and the islands can be 're-populated' with either the best solutions prior to pooling or the best solutions resulting from pooling (or some combination thereof). The amount and interval of island communications were defined by the user. This sharing of information can facilitate escape from local optima and improve the overall rate of convergence but this performance improvement is not guaranteed (39).

Extinction

The methods of simulated extinction coupled with the methods of standard EC can provide increased convergence rates on highly multimodal fitness landscapes (40–42). In this method, 'extinction' events are applied to the population during evolution to remove a percentage of the population generally without respect (or with only minor respect) to fitness. A large extinction event removes virtually every individual from the population and restarts evolution with randomly derived solutions or solutions derived from the small remainder of individuals.

Given that the context of the TFBS discovery problem presents a rugged landscape with many local optima; through early experimentation, we observed that the islands were becoming entrapped in local optima despite methods to maintain diversity in the population. These solutions were quite reasonable in terms of their quality and it was to our advantage to capture as many of these locally optimal solutions and explore their relative worth at the end of the evolutionary process. To do this, we implemented a variant of the extinction method that monitors the convergence for each island. When the population stagnated for a user-defined period of generations, the best result was saved and the population was restarted at random. This provides additional opportunity for multiple sampling of the fitness landscape while at the same time saving the best solutions discovered by previous evolutionary paths. Thus, the output of each island contained a listing of the best local optima discovered as well as an output from the final generation of evolution.

Termination

Following the final generation of evolution, a user-defined number of 'best' solutions were written to file for further investigation by the user. To generate this list, the results from the final generation of each island evolution are saved to file. This list, in combination with the list of all best solutions discovered via the extinction process described above, was pooled, ranked by fitness, and any duplicates were eliminated. A user-defined number of best final solutions were reported to file for future analysis.

Program implementation

For the experiments presented here, unless otherwise stated, elitist selection, uniform distribution for the number of mutations, a population of 15 parents and 100 offspring for 500 generations of evolution were used to discover similar motifs of window length 8 nt. The evolution was performed on an Intel Pentium III, 450 MHz, 256 MB RAM computers operating either Linux O/S or Windows 2000 O/S. The approach outlined above operates in $O(n)$ computational time where n is the problem size.

DISCOVERY OF TFBS

Calculation of G+C% for Oct and NF-κB sequences

In order to use the G+C% Window Slide variation operator, G+C% were calculated over all 1 kb upstream sequences and were used on the basis of average G+C% calculation and for position of new windows. Calculation of the G+C% for the 1 kb region was made by sliding a window of length 8 nt over the sequences and reporting the window G+C% relative to the position of the first nucleotide in the window. Thus, the 1 kb sequence was transformed into a sequence of length 992 in terms of G+C%.

These values showed considerable variation within each sequence and between sequences. Some sequences, such as Oct C00043, showed a spike in high G+C% within 200 nt of TSS. Other sequences such as NF-κB C00165 appeared to have low G+C% throughout the upstream region. When averaged together, there appeared to be a slight increase in G+C% over all Oct sequences and no major difference in G+C% over all NF-κB sequences. The small window size of 8 nt provides a highly irregular representation of G+C% and hence the window size might affect the utility of the G+C% Window Slide operator.

Putative Oct TFBSs

Initial experimentation investigated the proper tuning of w_1 and w_2 associated with the fitness function terms. Weights of $w_1 = 0.7$ and $w_2 = 0.3$ generated results that placed the known Oct TFBSs as solution #1 in the resulting top 100 solutions (Table 2), with other variations of the known Oct TFBSs occurring throughout the top 100 solutions. The top 100

Table 2. Highest scoring solution using evolutionary computation for the Oct sequence set

COMPEL entry	Evolved motif
C00039	ATGCAAAT
C00043	ATGTAAAT
C00048	ATGTAAAT
C00049	ATGCAAAT
C00050	ATGCAAAG
C00158	ATGTAAAA
C00169	ATGCAAAT

This top solution contains the known TFBSs for all seven upstream regions.

Table 3. List of identical motifs discovered searching the upstream regions of genes that are known to be regulated by Oct

Cluster #	COMPEL entry	Motif
1	C00039, C00049, C00169	ATGCAAAT
	C00049, C00049*, C00158	ATGCACAT
	C00043, C00048	ATGTAAAT
2	C00039, C00050, C00169	TCCACAGA
	C00049, C00158	TGCACATA
3	C00043, C00050	TTTCCATA
4	C00049, C00050	AGACATGT
	C00043, C00158	AGACAGGT
5	C00049, C00043	CTTTTGGA
	C00050, C00169	CTTCTGGA
	C00039, C00169	CTTCTGCA
	C00158, C00158*	CATGTGCA
6	C00039, C00048	GTCAGAAG
	C00043, C00050	TTCAGAAG
	C00169, C00169*	TCCAGAAA
7	C00050, C00158, C00048	CTCCACAG

The motifs in cluster #1 correspond to those for the known Oct TFBSs. *denotes the COMPEL sequences with multiple repeats of identical motifs.

solutions were manually grouped into clusters of similar TFBS sequences rather than displayed in terms of fitness score. Seven major clusters can be defined, with cluster #1 corresponding to the known Oct TFBSs (Table 3). The other clusters contain sequence motifs that are 100% conserved in other upstream sequences or repeated within the same upstream sequence (Table 3).

The motifs in the clusters shown in Table 3 can also be arranged by COMPEL identifier to find sequences that contain multiple versions of a similar putative TFBSs with slight alteration. For instance, sequence C00049 in cluster 1 contains three motifs that match the known Oct-binding site, the first was considered as a control example (ATGCAAAT) and the other two (ATGCACAT) differ by only one nucleotide. Other sequences such as C00039 have only one representative in cluster 1 (the previously known Oct-binding site ATGCAAAT).

Within 50 generations, the evolutionary process was able to quickly converge on useful solutions. Repeated resampling with either restarted evolutionary runs or via extinction assist in capturing information from many local optima.

Putative NF-κB TFBSs

When the same settings ($w_1 = 0.7$ and $w_2 = 0.3$) were used to generate the NF-κB results, the resulting output did not contain the known NF-κB solution (data not shown), which demonstrates that the requirement for optimal tuning of these parameters relative to as many known TFBSs as possible. Additional tuning of the fitness function weights for both NF-κB and Oct provided satisfactory results on both control sets with $w_1 = 0.65$ and $w_2 = 0.35$. Using these settings, known solutions for both Oct and NF-κB were discovered in the top 100 solutions resulting from evolution; however, none of the known solution was considered 'best'.

Table 4. List of identical motifs discovered searching the upstream regions of genes that are known to be regulated by NF-κB

Cluster #	COMPEL entry	Motif
1	C00097, C00101, C00155	TTTCCCAG
2	C00097, C00099	CCTCTGTG
	C00155, C00156	CCTCAGTG
3	C00155, C00165, C00099	GAAATTCC
	C00156, C00152	GAGATTCC
4	C00165, C00156	CAGTCAGT

The motifs in cluster #3 correspond to those for the known NF-κB TFBSs.

Table 5. List of overlapping motifs coupled with known NF-κB TFBSs

Cluster #	COMPEL entry	Motif	Combined motif
1	C00152	ATTCCAAG	GAGATTCCAAG
3	C00152	GAGATTCC	
2	C00153	CCTAACTG	GAAAGTCCTAACTG
3	C00153	GAAAGTCC	
4	C00153	CAGAAAGT	CAGAAAGTCC
3	C00153	GAAAGTCC	

The top 100 solutions for NF-κB by this weighting were manually grouped into clusters in Table 4. Four major clusters can be defined, with cluster #3 corresponding to the known NF-κB TFBSs. The other clusters contain sequence motifs that are 100% conserved in other upstream sequences or repeated within the same upstream sequence.

Although not all known NF-κB TFBSs were identified using this approach, the most similar core of sequences within this TFBS family was discovered. It remains unclear if the other sequences in cluster 3 are also in some manner involved with NF-κB binding. A number of partially overlapping combinations can be found that link clusters together (Table 5). These combined windows of length 8 nt also demonstrate that the potential utility of using smaller windows to build larger combinations of motifs which might be relevant as putative TFBSs. Curiously, two of these overlapping regions occur within sequence C00153 and are at a very similar distance to the TSS relative to sequence C00152.

Using $w_1 = 0.65$ and $w_2 = 0.35$ with the Oct dataset places the known Oct TFBSs as the second best solution of the top 100 solutions reported at the end of evolution (data not shown). This demonstrates that it is possible to adjust the weights for these parameters and still return the correct TFBSs as well as other putative motifs.

DISCUSSION

The two control sets used in this research were chosen as representatives from the COMPEL/TRANSFAC databases.

The high degree of sequence conservation found in the Oct example made this a much easier task for automated TFBS discovery. Thus, Oct was used for initial code development, testing and refinement. This effort resulted in parameter settings and variation techniques that could identify the Oct sequences as the best of 100 top ranked solutions.

Initial performance using the Oct settings on the NF-κB example resulted in solutions that did not look like NF-κB. The weights on the fitness terms were adjusted such that one weight settings could be used to discover both Oct and NF-κB. It is clear that as expected, NF-κB presents a much more difficult search problem due to the general lack of sequence conservation in the binding motif. However, enough sequence conservation was observed in a subset of the NF-κB sequences to 'pull in' windows on other upstream regions with a similar sequence motif to the NF-κB TFBSs. It remains to be determined if other parameter settings, variation operators or methods of selection could be used to find all of the known NF-κB and list these as one of the top 100 solutions. However, presently, the method can result in at least the identification of a subset of known NF-κB motifs. Without such computational assistance and guidance, the researcher is forced to experimentally test for putative TFBSs at each upstream nucleotide position. The computational approach described here reduces the time and effort required for TFBS discovery by pointing the researcher to areas of highest probability of TFBSs detection for downstream experimental validation. The time required to verify the top 20 or even 100 best results from the EA experimentally is significantly lower than that of experimental validation at each position without these hints. When applying Equation 1 to the Oct 1 example, a space of 9.5×10^{20} possible solutions is defined. The evolutionary approach above required only 7.5×10^{5} function evaluations to complete. Thus, only 7.9×10^{-16} of the search space was interrogated and yet correct solution was identified in <5 h.

The motifs in Tables 3 and 4 for both Oct and NF-κB were separately concatenated and examined using the MatInspector 6.2.1 algorithm (43) and Matrix Family Library Version 3.1.2/ ALL vertebrates.lib (core: 0.70/maxtrix; sim: Optimized) to identify any previously known TFBSs that correspond to these different motifs. For the Oct motifs, six matches were discovered including the known Oct TFBSs, two Oct TFBS variants, as well as TFBSs for microphthalmia transcription factor (MIT), PAR-type chicken vitellogenin promoter-binding protein and cut-like homeodomain protein (Table 6). Only one match relative to NF-κB was discovered and that match corresponds to the known NF-κB TFBSs. These results indicate that the known truths were discovered using the approach as well as several other previously known TFBSs. In addition, experimental evidence is required to validate the remaining putative TFBSs from Table 6. The discovery of Oct and MIT TFBS is interesting in that both transcription factors are known to be involved in bone tissue. The MIT TFBS motif CATGTGCA occurs in the sequence C00158, an upstream region for Interleukin-2.

The problem of TFBS discovery with a fixed window length of 8 nt presents a rugged fitness landscape with many local optima. It is clear that the evolutionary process can rapidly progress from randomly derived solutions of low fitness to solutions of much higher fitness. However, it appears that the EA is prone to entrapment in these local optima and this reduces the speed of convergence on the global optimum. However, for this problem, identification of locally optimal solutions can present novel putative TFBSs that are yet to be discovered. Capturing the information in all local optima is perhaps just as important as capturing the globally optimal solution. We designed this EA to capture these local optima by monitoring periods of stagnation, copying the best results to file and restarting the EA multiple times within a single run to sample the space as much as possible in the allocated time. This process of simulated extinction assisted in the discovery of as many local optima as possible. Simulated parallelization of the overall process also provides *N*-fold additional samplings and an even greater probability that all locally optimal solutions of sufficient worth have been discovered.

Table 6. TFBS identified using MatInspector for both Oct and NF-κB

Family/matrix	Name	Strand	Core sim.	Matrix sim.	Sequence
OCT1.02	Octamer-binding factor 1	(+)	1.000	0.854	ATGCacat
VBP.01	PAR-type chicken vitellogenin promoter-binding protein	(−)	1.000	0.871	tTTACat
OCT1.02	Octamer-binding factor 1	(+)	0.755	0.869	ATGTaaat
OCT1.04	Octamer-binding factor 1	(−)	1.000	0.842	TATGgaaa
CDPCR3.01	Cut-like homeodomain protein	(−)	1.000	0.762	tATGGa
MIT.01	MIT and TFE3	(+)	1.000	0.843	CATGtgca
NF-κB65.01	NF-κB (p65)	(−)	0.804	0.876	ggaatTTC

Nucleotides in upper case denote the core sequence used by MatInspector.

In general, EC showed several advantages over existing methods on these two test cases. As mentioned above, published methods such as word enumeration (1,11,15) and position-specific weight matrix updating (9,18,21,22) have several serious disadvantages. For example, one disadvantage of the former method is that it does not allow mutation in the 'words', thus reducing the power to detect true TFBSs, most of which show some degree of variation. Our approach allows variation in the motif by using the measurement of a similarity score. The position-specific weight matrix method also avoids this pitfall, but relies on the accuracy of position-specific weight matrices, which are not always easy to define. Our approach does not rely on any pre-defined or estimated weight matrices. A recent paper combining these two approaches (44) was shown to be faster and more accurate than the previous methods. However, this hybrid method relies heavily on a low-order Markov model, which is more successful in recovering motifs with uniform distribution of di- or tri-nucleotides than other motifs. Our method avoids the above caveats and is capable of recovering TFBSs in these two test cases. However, a detailed investigation of performance on the additional control examples is yet to be completed and is the subject of further research.

CONCLUSION

This research effort focused on the development and examination of the utility of EC for the discovery of known and

putative TFBS motifs in upstream regions of coexpressed genes. In two examples, the EC approach was able to discover the known solutions and provide additional putative TFBSs that could be verified by computational and/or experimental means. This method does not require exhaustive search of windows and does not utilize a Gibbs sampling process. It is not organism-specific and does not require a pre-alignment. This effort demonstrates that the feasibility of the method for TFBS discovery for longer sequence motif lengths and provides a framework for additional refinement and success.

PROGRAM AVAILABILITY

Please contact Dr Chen Su and Dr Gary Fogel for information regarding program implementation and/or motif searches using the code.

ACKNOWLEDGEMENTS

The authors would like to acknowledge Dr David B. Fogel and Dr Seppo J. Karrila for their review of this manuscript and would like to thank Pavan Cheruvu for his contribution to an earlier phase of the project. The authors would also like to thank the anonymous reviewers for excellent suggestions.

REFERENCES

1. Brazma,A. and Vilo,J. (2000) Gene expression data analysis. *FEBS Lett*., **480**, 17–24.
2. Bucher,P. (1999) Regulatory elements and expression profiles. *Curr. Opin. Struct. Biol*., **9**, 400–407.
3. Chu,S., DeRisi,J., Eisen,M.B., Mulholland,J., Botstein,D., Brown,P.O. and Herskowitz,I. (1998) The transcriptional program of sporulation in budding yeast. *Science*, **282**, 699–705.
4. DeRisi,J.L., Iyer,V.R. and Brown,P.O. (1997) Exploring the metabolic and genetic control of gene expression on a genomic scale. *Science*, **278**, 680–686.
5. Spellman,P.T., Sherlock,G., Zhang,M.Q., Lyer,V.R., Anders,K., Eisen,M.B., Brown,P.O., Botstein,D. and Futcher,B. (1998) Comprehensive identification of cell cycle-regulated genes of the years *Saccharomyces cerevisiae* by mircoarray hybridization. *Mol. Biol. Cell*, **9**, 3272–3297.
6. Wolfsberg,T.G., Gabrielian,A.E., Campbell,M.J., Cho,R.J., Spouge,J.L. and Landsman,D. (1999) Candidate regulatory sequence elements for cell cycle-dependent transcription in *Saccharomyces cerevisiae*. *Genome Res*., **9**, 775–792.
7. Zhang,M.Q. (1999) Large-scale gene expression data analysis: a new challenge to computational biologist. *Genome Res*., **9**, 681–688.
8. Ohler,U. and Niemann,H. (2001) Identification and analysis of eukaryotic promoters: recent computational approaches. *Trends Genet*., **17**, 56–60.
9. Liu,X., Brutlag,D.L. and Liu,J.S. (2001) Bioprospector: discovering conserved DNA motifs in upstream regulatory regions of co-expressed genes. *Pac. Symp. Biocomput*., **6**, 127–138.
10. Atteson,K. (1998) Calculating the exact probability of language-like patterns in biomolecular sequences. *Proc. Int. Conf. Intell. Syst. Mol. Biol*., **6**, 17–24.
11. Tompa,M. (1999) An exact method for finding short motifs in sequences, with application to the ribosome binding site problem. *Proc. Int. Conf. Intell. Syst. Mol. Biol*., **7**, 262–271.
12. Brazma,A., Jonassen,I., Vilo,J. and Ukkonen,E. (1998) Predicting gene regulatory elements *in silico* on a genomic scale. *Genome Res*., **8**, 1202–1215.
13. Jensen,L.J. and Knudsen,S. (2000) Automatic discovery of regulatory patterns in promoter regions based on whole cell expression data and functional annotation. *Bioinformatics*, **16**, 326–333.
14. Vanet,A., Marsan,L., Labigne,A. and Sagot,M.F. (2000) Inferring regulatory elements from a whole genome. An analysis of *Helicobacter pylori* sigma(80) family of promoter signals. *J. Mol. Biol*., **297**, 335–353.
15. Van Helden,J., André,B. and Collado-Vides,L. (1998) Extracting regulatory sites from upstream region of yeast genes by computational analysis of oligonucleotide frequencies. *J. Mol. Biol*., **281**, 827–842.
16. Lawrence,C.E. and Reilly,A.A. (1990) An expectation maximization (EM) algorithm for the identification and characterization of common sites in unaligned biopolymer sequences. *Proteins*, **7**, 41–51.
17. Lawrence,C.E., Altschul,S.F., Boguski,M.S., Liu,J.S., Neuwald,A.F. and Wootton,J.C. (1993) Detecting subtle sequence signals: a Gibbs sampling strategy for multiple alignment. *Science*, **262**, 208–214.
18. Bailey,T. and Elkan,C. (1995) Unsupervised learning of multiple motifs in biopolymers using expectation maximization. *Machine Learning*, **21**, 51–80.
19. Hughes,J.D., Estep,P.W., Tavazoie,S. and Church,G.M. (2000) Computational identification of *cis*-regulatory elements associated with groups of functionally related genes in *Saccharomyces cerevisiae*. *J. Mol. Biol*., **296**, 1205–1214.
20. Neuwald,A.F., Liu,J.S. and Lawrence,C.E. (1995) Gibbs motif sampling: detection of bacterial outer membrane protein repeats. *Protein Sci*., **4**, 1618–1632.
21. Roth,F.P., Hughes,J.D., Estep,P.W. and Church,G.M. (1998) Finding DNA regulatory motifs within unaligned noncoding sequence clustered by whole genome mRNA quantitation. *Nat. Biotechol*., **16**, 939–945.
22. Workman,C.T. and Stormo,G.D. (2000) ANN-SPEC: a method for discoverying transcription binding sites with improved specificity. *Pac. Symp. Biocomput*., **5**, 467–478.
23. Thijs,G., Lescot,M., Marchal,K., Rombauts,S., De Moor,B., Rouzé,P. and Moreau,Y. (2001) A higher-order background model improves the detection of promoter regulatory elements by Gibbs sampling. *Bioinformatics*, **17**, 1113–1122.
24. Bremmerman,H.J. (1962) Optimization through evolution and recombination. In: Yovits,M.C., Jacobi,G.T. and Goldstein,G.D. (eds), *Self-Organizing Systems*. Spartan Press, Washington, DC.
25. Holland,J.H. (1975) *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor, MI.
26. Fogel,L.J., Owens,A. and Walsh,M.J. (1966) *Artificial Intelligence Through Simulated Evolution*. Wiley, New York, NY.
27. Rechenberg,I. (1973) *Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien der biologischen Evolution*. Fromman-Holzboog, Stuttgart, Germany.
28. Koza,J. (1992) *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT Press, Cambridge, MA.
29. Holland,J.H., Holyoak,K.J., Nisbett,R.E. and Thagard,P.R. (1986) *Induction: Processes of Inference, Learning, and Discovery*. MIT Press, Cambridge, MA.
30. Bäck,T., Fogel,D.B. and Michalewicz,Z. (eds) (1997) *Handbook on Evolutionary Computation*. Oxford University Press, NY.
31. Fogel,D.B. (ed.) (1998) *Evolutionary Computation: The Fossil Record*. IEEE Press, Piscataway, NJ.
32. Fogel,D.B. (2000) *Evolutionary Computation: Toward a New Philosophy of Machine Intelligence*, 2nd edn. IEEE Press, Piscataway, NJ.
33. Fogel,G.B. and Corne,D.W. (eds) (2002) *Evolutionary Computation in Bioinformatics*. Morgan Kaufmann, San Francisco, CA.
34. Bäck,T. (1996) *Evolutionary Algorithms in Theory and Practice*. Oxford University Press, New York, NY.
35. Michalewicz,Z. (1996) *Genetic Algorithms + Data Structures = Evolution Programs*, 3rd edn. Springer, New York, NY.
36. Matys,V., Fricke,E., Geffers,R., Gossling,E., Haubrock,M., Hehl,R., Hornischer,K., Karas,D., Kel,A.E., Kel-Margoulis,O.V. *et al*. (2003) TRANSFAC: transcriptional regulation, from patterns to profiles. *Nucleic Acids Res*., **31**, 374–378.
37. Kel-Margoulis,O.V., Romashchenko,A.G., Kolchanov,N.A., Wingender,E. and Kel,A.E. (2000) COMPEL: a database on composite regulatory elements providing combinatorial transcriptional regulation. *Nucleic Acids Res*., **28**, 311–315.
38. Wootton,J.C. and Federhen,S. (1996) Analysis of compositionally biased regions in sequence databases. *Methods Enzymol*., **266**, 554–571.
39. Cantú-Paz,E. (2000) *Efficient and Accurate Parallel Genetic Algorithms*. Kluwer Academic Publishers, Boston, MA.

40. Greenwood,G.W., Fogel,G.B. and Ciobanu,M. (1999) Emphasizing extinction in evolutionary programming. *Proceedings of the 1999 Congress on Evolutionary Computation*, pp. 666–671.
41. Fogel,G.B., Greenwood,G.W. and Chellapilla,K. (2000) Evolutionary computation with extinction: experiments and analysis. *Proceedings of the 2000 Congress on Evolutionary Computation*, pp. 1415–1420.
42. Krink,T. and Thomsen,R. (2001) Self-organized criticality and mass extinction in evolutionary algorithms. *Proceedings of the 2001 Congress on Evolutionary Computation*, pp. 1155–1161.
43. Quandt,K., Frech,K., Karas,H., Wingender,E. and Werner,T. (1995) MatInd and MatInspector: new fast and versatile tools for detection of consensus matches in nucleotide sequence data. *Nucleic Acids Res*., **23**, 4878–4884.
44. Liu,X.S., Brutlag,D.L. and Liu,J.S. (2002) An algorithm for finding protein–DNA binding sites with applications to chromatin-immunoprecipitation microarray experiments. *Nat. Biotechnol*., **20**, 835–839.