Building a Dictionary for Genomes: Statistical Analysis of Transcriptional Regulation

Harmen J. Bussemaker, Hao Li, and and Eric D. Siggia[‡]
Center for Studies in Physics and Biology, The Rockefeller University,
Box 25, 1230 York Avenue, New York, N.Y. 10021, and
[‡]*Department of Physics, Cornell University, Ithaca, N.Y. 14853-2501*

The availability of complete genomes and mRNA expression data for all genes creates new opportunities for deciphering the extent to which gene expression is controlled by sequence motifs in the genome. A new algorithm is presented which decomposes DNA sequence into the most probable 'dictionary' of motifs or words. The significance of longer words is based on the frequency of shorter ones, eliminating the need for a separate set of reference data to define probabilities. Thus genome-wide applications are possible. For the upstream control regions in yeast, the 500 strongest motifs from a dictionary of size 1200 match at a significance level of 15 standard deviations to a database of cis-regulatory elements. For the cell cycle and sporulation gene sets, defined by microarray experiments, many new putative motifs are found.

The development of computational tools for extracting cis-regulatory elements has lagged behind those for sequence comparison and gene discovery. One approach has been to delineate, as sharply as possible, a group of 10-100 genes by means of shared biochemistry [1] or similarity in expression profiles [2] and then find a pattern common to most of the upstream regions. The analysis tools employed range from general multi-alignment algorithms yielding a weight matrix [3] to comparison of the frequency counts of substrings with some reference set [1]. These approaches are very conditioned by the available experiments, and could give the impression that genes are arranged into largely disjoint groups. Analysis of an entire genome is independent of context, and uses just data whose acquistion is increasingly easy.

Although a single sequence motif may confer expression of a reporter gene with a minimal core promoter, real genomes are not designed this way [4]. For instance, the microarray experiments for yeast [5,6] have shown quantitatively that most occurrences of proven binding motifs are in non-responding genes [7]. These experiments indicate that genome-wide transcriptional regulation is achived by combinatorial control and there are multiple regulatory elements per gene. A number of other explanations were suggested but cannot account for the bulk of the data [8]. Thus there are addtional sequence signals to be found. To optimize statistics, all the regulatory regions in the genome should be analyzed together. Alternatively, to pull out regulatory elements responsive to a specific genetic or environmental change, all genes that respond in a chip experiment should be analyzed as a group. Conventional sequence alignment algorithms are not adequate for upstream regions since the relevant motifs constitute at most 10% of the sequence, and are variously positioned in a random background.

Our approach formalizes how one would proceed to decipher a 'text' (genome) consisting of a long string S of L letters written in an unknown language that did not delineate words (e.g. cis regulatory elements). Our language is defined by a dictionary, D: a list of words indexed by α with an attached probability p_α. The text is modeled as the concatenation of words drawn at random with frequency p_α; there is no syntax. Intuitively, to recover the dictionary from the text, one would start from the frequency of individual letters, find overrepresented pairs, determine their probabilities, and continue to build larger fragments in this way. The algorithm loops over a fitting step to compute p_α, followed by a prediction step which adds new words. This intrinsic decomposition of the text should be contrasted with approaches that make use of a test set or fixed 'background' probabilities. A language could be build from words that are all long independent random strings which would frustrate a discovery process based on common short fragments. Natural languages, fortunately, are not of this type nor is biology to a certain degree, as we will show.

Let $Z(S, p_\alpha)$ be the likelihood of obtaining the seqeunce S for a given set of normalized dictionary probabilities p_α,

$$Z = \sum_P \prod_\alpha (p_\alpha)^{N_\alpha(P)}. \tag{1}$$

Here the sum is over all possible segmentations P of S, and $N_\alpha(P)$ denotes the number of times word α is used in a given segmentation. E.g., if $D = \{\mathrm{A}, \mathrm{T}, \mathrm{AT}\}$ then $Z(\mathrm{TATA}) = (p_\mathrm{A}^2 p_\mathrm{T}^2 + p_\mathrm{A} p_\mathrm{T} p_\mathrm{AT})$.

For a given set of dictionary words α, we fit the p_α to the sequence S by maximizing the likelihood of reproducing S with the constraint that $p_\alpha \geq 0$ and $\sum_\alpha p_\alpha = 1$. This condition is equivalent to solving for p_α from the equation

$$p_\alpha = \langle N_\alpha \rangle / \sum_\beta \langle N_\beta \rangle, \tag{2}$$

where $\langle N_\alpha \rangle = p_\alpha (\partial / \partial p_\alpha) \ln Z$ is the average number of words α in the ensemble defined by Z. The right hand side of (2) can of course be defined for an arbitrary dictionary and (2) can be solved by iteration. However this is quite slow, and via suitable preprocessing we have implemented Newton's method for (2) which in a few iterations gets p_α to machine accuracy [9]. Dictionaries up to 10^3 words can be handled routinely, and we have verified that the solution is unique by randomizing the starting p_α and re-converging. If the defining words are known, the statistical errors on p_α are set by the Gaussian fluctuations of the background, e.g. if $p_{A,C,G,T} = 1/4$ and there is only one other word in the text of length ℓ, its probability is uncertain by $(L4^\ell)^{-1/2}$.

Recovering the dictionary which generated the data still does not permit one to 'read' the text in a unique way, since the decomposition into words is probabilistic, and there are many ways to segment the data into words; yet key words do emerge. Let Ξ_α be the number of places that the string α begins in the data (e.g., $\Xi_{AA} = 2$ for $S = AAA$). When S is generated from a dictionary, $\langle N_\alpha \rangle / \Xi_\alpha < 1$, that is, the number of occurrences of the string α in S is at least as many as are put down as complete words, but can be many more if the appropriate words juxtapose. This ratio then serves as a quality factor and when it is close to one, almost all occurrences of the string α are as delimited words. The dictionary does not distinguish between signal and background, though shorter words typically have lower quality. Methods based on frequency counts Ξ will register as significant (i.e., overrepresented) any substring containing a dictionary word, whereas our fit will assign $p_\alpha > 0$ only to the dictionary word itself.

The prediction step is necessarily heuristic, as there is no way short of enumeration to check that the frequency of all possible patterns are correctly predicted by a dictionary. In addition to checking for overrepresented pairs of juxtaposed dictionary words, we monitor the completeness of a dictionary by computing Ξ_α for all n-mers up to a certain length, and converting it to a z-score by subtracting the mean and dividing the variance of Ξ_α as predicted by the current dictionary (for 1-2Mb of data, $n \leq 8$ is the limit imposed by statistics; this length should however capture the core of most motifs). Words whose score lies outside of the expected unit-variance Gaussian are then added to the dictionary. To increase the sensitivity of our method, we also considered clusters of n-mers defined by IUPAC symbols, which can be overrepresented even if the distribution of z-scores for individual n-mers is Gaussian. In addition, a screening was performed for all dimer patterns composed of two n-mers separated by a fixed number of N symbols (matching any base).

The convergence step tests the relevance of all words in parallel, and will zero probabilities of less specific words, when a more specific one will account for the overrepresented string. Once a piece of a motif has been found, it can be grown or contracted again in parallel for all motifs. In this way the optimal word will emerge.

It is instructive to validate the performance of our algorithm with English text. We took a portion of a novel comprising $\sim 10^5$ characters and reduced it to a lower case string containing no spaces or other punctuation characters. The starting dictionary was just the English alphabet; words were added by juxtaposition, and required to have at least two matches in the data. The final dictionary had 3600 words and among the most significant 1800 were 800 English words, 800 concatenations of English words, and only 200 fragments. Virtually all the 1600 text words with two or more copies occurred somewhere in our dictionary [10]. In a sense English is too simple, the probability of obtaining longer words by chance is very small, so simply checking for all strings with two or more copies would hit many words. Biological data is different: virtually all 8-mers have 2 or more copies in the combined upstream regions in yeast, yet less than 1% occur in the dictionary.

A dictionary was prepared for all the upstream regions in the yeast genome [11]. Starting from a single base dictionary, word addition via juxtaposition missed statistically significant motifs, so a more incremental detailed search was used [12]. The final dictionary had 1200 words, of which 100 were clusters of n-mers. Two thirds of the sequence data was segmented into single letter words, and an additional 15% into words of length 4 or less. About 500 dictionary entries fell above a plausible significance level [13]. These included good matches to about half of the motifs found in Refs. [5,6,1], including the MCB, SCB, and MCM1 cell-cycle motifs and URS1 for sporulation, Table 1. We also found ~ 400 pairs of length 4-5 words, the most significant of which clustered together to generate the dimer motifs in Table 2.

Table 3 gives a more comprehensive measure of the extent to which our dictionary is predicting functional biological sites by comparing with an database of experimentally determined sites. [13]. We hit 25-35% of the sites (depending on the threshold) with a combination of dictionary motifs and dimers, which is 15 standard deviations beyond expectation. Also tabulated, following the same criterion, are matches to the dictionary with the letters in all words scrambled, and the set of all strings of length 8 with largest z-score based on single base frequencies. In contrast to less systematic assignments of probability, our dictionary fits the copy number of all strings of length 8 or less, plus certain classes of clusters composed of them. Thus we can say that most of the experimental sites are *not* distinguishable by their frequency. It is difficult to assess our false positive rate, since the database covers such a small fraction of the genome, but we note that 61 out of 500 dictionary entries (52 out of 400 dimers) were responsible for the site hits in Table 2.

Smaller and higher quality dictionaries that can be more readily correlated with the existing knowledge base, are obtained when we fit just the genes which are active in a process such as sporulation with 500 such genes [6]. We used only the most stringent criterion for word addition; the sporulation dictionary had only 250 unique words of which 150 were significant; 66% of the data was segmented into single letter words and an additional 22% by words of length 4 or less.

The two principle sporulation elements, URS1 and MSE, have consensus motifs, 5'-DSGGCGGC and 5'-CRCAAAW [6]. Dictionary words which strongly overlap with these patterns are TCGGCGGC, GGCGGCAAA, TGGGCGGCTA, CTTCGGCGGC, GGCGGCTAAA, and CACAAAA, CGTCACAAA, GTCACAAAA, plus several words which match to the reverse complements. Rarely will dictionary words precisely match a IUPAC consensus pattern, which is derived under a different set of assumptions. We have verified in this instance that the Ξ_α of all 8-mers containing GGCGGC or the MSE consensus are fit by the dictionary to within expected statistical fluctuations, e.g., there are 35 copies of CGCAAAW in our data set vs. 28 predicted from mostly words of length 3 or less. In this sense our dictionary description is consistent, eventhough CGCAAAW is not a word.

The frequency of occurrence of CGCAAAW in either the sporulation cluster or the middle temporal class [6] is close to that expected by chance based on their genome-wide population. The chance probability for the number of copies of CACAAAA in the mid-temporal class is $< 10^{-5}$. The sporulation dictionary of course 'knows' nothing about the frequency of words elsewhere in the genome.

There is much more information in the dictionary than the conventional motifs. There are about 20 clusters of dictionary words, with 10 or more copies among the sporulation genes; and 16 specific words, some belonging to clusters, whose probability of occurrence in some temporal class [6] is less than 10^{-4}, based on their genome-wide frequency. These are all potential transcription factor binding sites. A subset are shown in Table 4. The TGTG cluster is similar to one summarized by Mitchell [14] and the Abf1 site we obtained genome-wide also plays a role in sporulation [15]. We also matched the significant sporulation words against the yeast promoter database [13] and hit 56 sites, vs. 14 by chance, and 94 using the full genome dictionary. (Comparison with words selected with the single base probabilities in parallel with the full genome fit [13] gave 30 hits, vs. 18 by chance). Thus many words in the sporulation dictionary are biologically meaningful, yet not particular to sporulation. A parallel dictionary fit to the 3' UTR gave much less signal.

Problems of statistical inference and learning are ubiquitous but our model and its implementation, which are a natural first step in extracting meaning from large sets of upstream regulatory regions, seem new. Ref. [16] models English spellings, but as a Gibbs distribution, not a sum of partitions. Segmenting the character strings of Asian languages into words, has some similarity to the biological problem [17] but training sets are available and the algorithms differ. Data compression algorithms, despite a similarity in terminology (eg 'adaptive dictionary' for the Ziv-Lempel code family), satisfy very different design criteria (single pass, invertible coding) than the data model we fit to. They would certainly not recover our dictionary given a large sample of data synthesized by our model, e.g. they would try to 'compress' a random string. Codes that enumerate regular expressions without reference to a probability model could record as significant a long fragment of a dictionary word if by chance it has a few more copies than the word itself [19,18]. (Assigning probabilities based on a random sample of the genome for yeast, will be dominated by the differences between coding and non-coding [18].) Markov models are a common way to segment biological data, but they are generally employed with relatively few segment types (e.g. promoter, exon, intron) each described by many parameters; we work with many segments, most described by a single parameter.

Our algorithm admits many elaborations, such as identifying reverse complements, and an additional degree of freedom can be incorporated into our maximum likelihood method to segment the data into regions each with a different dictionary. Even in its present form, our algorithm if allowed to fit an entire yeast chromosome with all words length 4 or less, will put 99% of the weight into length 3 words and place almost all of the other length words into noncoding regions.

We can not treat as yet long, fuzzy patterns such as protein motifs over the 20 letter alphabet or promoters in *E. coli*, for which weight matrix methods [3] were designed that however require that most data segments contain one copy of the motif. There are many interesting motifs (or dimers thereof) in yeast however which are less than 10 bp long and have only a few variable sites. Any algorithm is optimal for some type of data, in our case strings prepared from a dictionary; to what model biological sequence conforms, can only be inferred from the success or failure of a particular algorithm. Once a set of words has been isolated, the challenge of deducing a 'grammar' to explain the context in which canonical motifs are functional, can be taken up.

Acknowledgements: We thank M. Zhang for making the data files from his yeast promoter database available. The NSF supported (eds) and (hjb); hl was a W. M. Keck fellow at Rockefeller University.

---

[1] J. van Helden, B. Andre, and J. Collado-Vides, J. Mol. Biol. **281**, 827-42 (1998).

[2] M. B. Eisen, P. T. Spellman, P. O. Brown, and D. Botstein, Proc. Natl. Acad. Sci. U.S.A. **95**, 14863-8 (1998); F. R. Roth, J. D. Hughes, P. W. Estep, and G. M. Church, Nature Biotechnology **16**, 939-945 (1998).

[3] G. D. Stormo and G. W. Hartzell, Proc. Natl. Acad. Sci. U.S.A. **86** 1183-1187 (1989); C. E. Lawrence, S. F. Altschul, M. S. Boguski et al., Science **262**, 208-214 (1993).

[4] C.H. Chu, H. Bolouri, and E.H. Davidson Science **279**,

1896-902 (1998).

[5] P. T. Spellman, G. Sherlock, M. Q. Zhang et al., Mol. Biol. Cell. **9**, 3273-97 (1998).

[6] S. Chu, J. DeRisi, M. Eisen, J. Mulholland et al., Science **282**, 699-705 (1998).

[7] Within the 500 bp upstream of transcription start (cf. Fig. 2C in Ref. [5]) the number of genes with multiple copies of motifs in the format (total genes; 1 copy, 2 copies, 3 or more) is for the MCB box (ACGCGT) in G1 (298; 80, 26, 1), cell cycle (794; 98, 26, 1), and genome-wide (6203; 295, 32, 1). For the sporulation MSE element (CRCAAAW) one finds within the middle group [6] (157; 64, 8, 3), all sporulation (480; 134, 18, 3), and genome-wide (6203; 966, 95, 7).

[8] Other variables such as copy number and motif position relative to translation start cannot single out the active genes [7]. Regulation by long range changes in chromosome structure are implausible in yeast, since in several experiments [5,6] the 500-1000 genes that respond are uniformly distributed over the 16 chromosomes. Chromatin structure within the regulatory region can influence transcription, but for the few cases analyzed in detail, this structure is itself under the control of cis-acting elements and therefore amenable to the type of analysis we propose, (cf. K. Struhl, D. Kadosh, M. Keaveney, L. Kuras et al., Cold Spring Harbor Symposium LXIII, 413-421 (1998)).

[9] Using transfer matrices, the derivative of Eq. (2) with respect to all p_α (i.e., all word-word correlations) can be computed to machine accuracy in a time of order $\mathcal{O}(LD\ell)$, where L is the total sequence length, D the dictionary size, and ℓ the maximum word length. Monte Carlo calculations would be hopelessly inaccurate in our context.

[10] The original text had 4214 unique words of which 2630 were a single copy which our algorithm was forced to break up. To mimic the situation in biology, we also introduced random letters (whose frequency matched the text) between each word to increase L by a factor of 3. The final dictionary had 2450 words and among the most significant 1050 were 700 English words and 40 composite words.

[11] To maximize the homogeneity of our data, we define control regions to extend upstream from the translation start site to the next coding region on either strand but not more than 600 bp (resp. 500 bp) for the genome-wide dictionary (resp. sporulation and cell cycle data) All exact repeats longer than 16 bases were removed with the program 'reputer' (http://bibiserv.techfak.uni-bielefeld.de/reputer/) and fragments less than 100 (resp. 50) bases were ignored. This removed 14%, 3% and, 5% of the data for the three cases mentioned.

[12] We experimented with many ways for adding words to the dictionaries. For the data presented here, we added overrepresented n-mers for lengths increasing from $n = 1$ to $n = 8$; subsequently, words in the dictionary were allowed to grow one base at a time if the extended word was overrepresented. All significant motifs with at most two IUPAC symbols and a total of 8 characters were then added. We imposed a minimum of two copies for any dictionary entry.

[13] J. Zhu and M. Q. Zhang, Bioinformatics **15**, 607-611 (1999). This database of experimentally confirmed regulatory sites in yeast has entries covering $\sim$ 200 genes and $\sim$ 110 factors. Almost all sites occur upstream of the translation start. For comparison with this database, we filtered our dictionaries based on a significance score ($\langle N_\alpha \rangle / \Xi_\alpha > 0.2$ and $\Xi_\alpha \leq 100$, to filter out poly-A and similar strings), and matched (by inclusion) against the 443 non-repetitive sites of length 5-30. The number of sites hit by chance is calculated by marking the dictionary words on the sequence data (to account for correlations) and then treating the database entries as intervals to be placed at random.

[14] A. P. Mitchell, Microbiological Reviews **58**, 56-70 (1994).

[15] V. Gailus-Durner, J. Xie, C. Chintamaneni, and A. Vershon, Mol. Cell. Biol. **16**, 2777-2786 (1996); N. Orzarac, M. Straffon, H.E. Dalton, and I. W. Dawes, Mol. Cell. Biol. **17**, 1152-1159 (1997).

[16] S. D. Peitra, V. D. Pietra, and J. Lafferty, IEEE Transactions Pattern Analysis and Machine Intelligence **19**, 1-13 (1997).

[17] J. M. Ponte and W. B. Croft, UMass Computer Science Tech Rep. 1996-002 (1996), available at ftp://ftp.cs.umass.edu/pub/techrept/; R. Ando and L. Lee, Cornell CS Report TR99-1756 (1999), available at http://www.cs.cornell.edu/home/llee/papers.html.

[18] A. Brazma, I. Johnassen, J. Vilo, E. Ukkonen, Genome Res. **8**, 1202-15 (1998).

[19] I. Rigoutsos and A. Floratos, Bioinformatics **14**, 55-67 (1998).

TABLE I. Known cell cycle sites (lines 1-6; from Ref. [5]), and some metabolic sites (lines 7-12; from Ref. [1]) that are hit by words from our genome-wide dictionary.

Name	Consensus sequence	Dictionary Words
MCB	ACGCGT	AAACGCGT ACGCGTCGCGT CGCGACGCGT TGACGCGT
SCB	CRCGAAA	ACGCGAAA
SCB′	ACRMSAAA	ACGCGAAA ACGCCAAA AACGCCAA
Swi5	RRCCAGCR	GCCAGCG GCAGCCAG
SIC1	GCSCRGC	GCCCAGCC CCGCGCGG
MCM1	TTWCCYAAWNNGGWAA	TTTCCNNNNNNGGAAA
NIT	GATAAT	TGATAATG
MET	TCACGTG	RTCACGTG TCACGTGM CACGTGAC CACGTGCT
PDR	TCCGCGGA	TCCGCGG
HAP	CCAAY	AACCCAAC
TUP	KANWWWWATSYGGGGW	TATATGTG CATATATG GTGGGGAG
GAL4	$CGGN_{11}CCG$	$CGGN_{10}CCG$ $CGGN_{11}CCG$

TABLE II. A sample of the dimer motifs assembled by clustering the most statistically significant words pairs of length 4-5 with a spacing of 3-30 from a genome wide dictionary. The clusters were sharp enough that all the word pairs in a cluster aligned. The probability cutoff is $\log_{10}$ of $(533)^2$ for the heterodimers and of 30×533 for the homodimers. The probability score for the most significant dimer in the cluster is tabulated.

Motif	Factor [13]	$-\log_{10}(\mathrm{P})$
KHCGTHTNNNNTGAYW	ABF1	28
ATRTCACTNNNNACGD	RAP1,ABF1	52
TTTCCNNNNNNGGAAA	MCM1	6
ATACANNNNTACAT	?	10
CCGTNNNNNGCGAT	?	9
GGGCNNNNNNACCCG	?	7
CACGTGNNNNNCACGTG	?	7

TABLE III. Statistics of matches to the yeast promoter database of Ref. [13]. For the last two entries we cut off the list of motifs ordered by significance when the total number of Ξ counts in the data equaled that for the dictionary set (approx. 21,000). All other filters were identical.

Data Set	Observed	Expected $\pm$ SD
Dictionary	94	23 ± 4.8
Dimers	26	2.7 ± 1.8
Scrambled Dictionary	33	14 ± 3.3
Ref. [18]	30	9 ± 2.9
Length 1 Dictionary	66	19 ± 4.5

TABLE IV. Several of the most significant word clusters from our sporulation dictionary that are also overrepresented in one or more of the temporal classes of Ref. [6]. A motif is defined by a cluster of similar dictionary words; the remaining columns represent number of copies in marked classes, number expected based on all promoters, and relevant temporal class(es). The last two entries are elaborations of the URS1 and MSE motifs that better discriminate responding from nonresponding genes.

Motif	Observed	Expected	Classes
TGTCACCT|TGTGTCA|TGTGTCAC|TTGTGTC	35	7	4
GTCAGTAA|TCAGTAAT	14	3	4
GACACA|GCCACA	52	2	4
CGCGACGC|GACGCGA|GACGCGAA|GAGGCGA|GAGGCGAC|GCGACGCG	18	3	1,2,5
GCGGCTAA|GGCGGCAAA|GGCGGCTAAA	10	1	1
CGTCACAAA|GTCACAAAA	17	1	4