CMF: a combinatorial tool to find composite motifs

Mauro Leoncini[1,3], Manuela Montangero[1,3], Marco Pellegrini[3], and Karina Panucia Tillán[2]

[1] Dip. di Scienze Fisiche, Informatiche e Matematiche
[2] Dip. di Scienze e Metodi dell'Ingegneria
Univ. di Modena e Reggio Emilia, Italy
{manuela.montangero, 83672, leoncini}@unimore.it
[3] Istituto di Informatica e Telematica, CNR - Pisa, Italy
marco.pellegrini@iit.cnr.it

Abstract. Controlling the differential expression of many thousands genes at any given time is a fundamental task of metazoan organisms and this complex orchestration is controlled by the so-called *regulatory genome* encoding complex regulatory networks. *Cis-Regulatory Modules* are fundamental units of such networks. To detect Cis-Regulatory Modules "in-silico" a key step is the discovery of recurrent clusters of DNA binding sites for sets of cooperating Transcription Factors. *Composite motif* is the term often adopted to refer to these clusters of sites. In this paper we describe CMF, a new efficient combinatorial method for the problem of detecting composite motifs, given in input a description of the binding affinities for a set of transcription factors. Testing with known benchmark data, we attain statistically significant better performance against nine state-of-the-art competing methods.

1 Introduction

Transcription Factors (or simply *factor*) are particular proteins that bind to short specific stretches of DNA (called *TFBS - Transcription Factor Binding Sites*) in the proximity of genes and participate in regulating the expression of those genes [1]. The "language" of gene regulation is a complex one since a single factor regulates multiple genes, and a gene is usually regulated over time by a cohort of cooperating factors. This network of interactions is still far from being completely uncovered and understood even for well studied model species. Groups of factors that concur in regulating the expression of groups of genes form functional elements of such complex network and are likely to have TFBS in the proximity of the regulated genes. TFBSs are often described by means of *Position Weight Matrices* (*PWMs*) (see Section 2 for a quick recap).

Over the last two decades more than a hundred computational methods have been proposed for the de-novo prediction "in silico" of single functional TFBSs (often called *single motifs*, or simply motifs) [2–5]. Moreover, several hundreds of validated PWMs for identifying TFBS are available in databases such as TRANSFAC [6] and JASPAR [7]. Observe that, although these PWM have been subject to validation in some form, the highly degenerate nature of the TFBS implies that, when scanning sequences for PWM matches, false positive non-functional matches are quite likely.

In this paper we address the problem of discovering groups of TFBSs that are functional for a set of cooperating factors, given in input a set of PWMs that describe the binding affinities of the single factors. This is known as the *Composite Motif Discovery* problem in the literature [8]. For us, a composite motif will be simply a set of TFBSs that are close by in a stretch of DNA, i.e., we do not pose any constraints on the order or the spacing between the participating TFBSs (but see [9] for other possible models).

The composite motif discovery problem has been the subject of a number of studies, and we refer to [10] for a survey. In addition we observe that the phenomenon of *clustering* of TFBS is used also by tools that try to predict the location and composition of *Cis-Regulatory Modules* (see, e.g., [11]), which then address composite motif discovery problems of some sort.

In this paper we present a new tool (*CMF*) for composite motifs discovery that adopts a two stage approach: first it looks for candidate single TFBSs in the given sequences, and then uses them to devise the prospective composite motifs by using mainly combinatoric techniques. CMF borrows the idea of the two stage approach from a previous tool we developed for the related problem of structured motif detection [12]. Using the data set and the published results in [8, 13] we can readily compare CMF's performance against the eight state of the art methods listed in [8] and other three more recent methods [13–15], showing that our tool is highly competitive with the others.

The detection of TFBS and composite motifs is a complex challenging problem (witness the wide spectrum of approaches) which is far from having a satisfactory solution [16], thus there is ample scope for improvements from both the modeling and the algorithmic one points of view. *CMF* introduces several new key ideas within a combinatorial approach, which, on one hand, have been shown empirically to be valid on challeng-

ing benchmarks, and on the other hand may prove useful in developing future more advanced solutions.

The rest of the paper is organized as follows: Section 2 introduces preliminary notions and definitions, Section 3 describes the algorithm adopted by CMF and, finally, Section 4 reports experimental results.

2 Preliminary notions

In this Section we introduce the fundamental notions used in the description of the algorithm that forms the computational core of CMF.

Given the DNA alphabet $\mathcal{D} = \{A, C, G, T\}$, a short word $w \in \mathcal{D}^*$ is called an *oligonucleotide*, or simply *oligo* (typically $|w| \leq 20$), and we say that w *occurs* in $S \in \mathcal{D}^*$ if and only if w is a substring of S.

From a computational point of view, a *DNA motif* (or simply *motif*) is a representation of a set of oligos, that are meant to describe possible factor binding loci. The representation can be made according to one of a number of models presented in the literature. Here we adopt the well-known *Position Weight Matrices* (*PWMs*). A PWM $M = (m_{b,j})$, $b \in \mathcal{D}, j = 1, \ldots, k$, is a $4 \times k$ real matrix. The element $m_{b,j}$ gives a score for nucleotide b being found at position j in the subset of length-k oligos that M is intended to represent. Scores are typically computed from frequency values.

Among the different ways in which oligos can be associated to PWMs (*e.g.*, [17–19]), here we adopt perhaps the simplest one. Consider a word $w = w_1 w_2 \ldots w_k$ over $\mathcal{D}^k$, and define the *score* of w, according to M, simply as the sum of the scores of all nucleotides: $S_M(w) = \sum_{j=1}^{k} m_{w_j,j}$. The maximum possible score given by M to any word in $\mathcal{D}^k$ is clearly $S_M = \sum_{j=1}^{k} \max_{b \in D} m_{b,j}$. Then we say the M *represents* word w iff $\frac{S_M(w)}{S_M} \geq \tau$, for some threshold value $\tau \in (0, 1]$. In the following, we will identify motifs with their matrix representation.

Let $\mathcal{S} \subseteq \mathcal{D}^*$ denote the set of N input sequences. A motif M has a *match* in $S \in \mathcal{S}$ if and only if there is a substring of S that is represented by M. As in [13], we call *discretization* the process of determining the matches of a motif in a set of DNA sequences.

A *motif class* is a set of motifs. Ideally, in CMF all the motifs in a class describe potential binding sites for the same factor. For this reason, we often freely speak of *factors* to refer to motif classes. A *factor match* in a DNA sequence is thus a match of any of the motifs in the class associated to that factor. Note that motif classes have the ability to represent oligos of different lengths, since different matrices usually exist for the same

factor that have a different number of columns. Let $\mathcal{F}$ be the set of factors having matches in $\mathcal{S}$. We consider a one-to-one mapping between $\mathcal{F}$ and an arbitrary alphabet $\mathcal{R}$ of $|\mathcal{F}|$ symbols, which we refer to $\mathcal{R}$ as the *mapping alphabet*.

A *combinatorial group* (or just *group*) is a collection of not necessarily distinct factors that have close-by matches in a sufficiently large fraction of the input sequences[4]. The minimum fraction allowed for a collection of factors to be considered a combinatorial group is termed *quorum*. The *width* or *span* of a collection of factor matches in a sequence S is the "distance" (measured in bps) between the first bps of first and last factor match of the group in S.

Finally, a *Composite Motif* is simply a collection of close-by factor matches in some input sequence, representing CMF's *best guess* for functional factor binding regions. Note that no quorum constraint is imposed to composite motifs. Indeed, as collection of factor matches, composite motifs are clearly unique objects. As we shall see in Section 3, CMF builds composite motifs by extending the matchings of some combinatorial group.

In set-theoretic terms, groups are multisets. In CMF they are represented as character sorted strings over the mapping alphabet $\mathcal{R}$. In the algorithm of Section 3.3 we will make use of some operations than involve multisets. We first recall, using two simple examples, the customary definitions of intersection and symmetric difference:

$$xxyyyz \cap xyyw = xyy$$
$$xxyyyz \setminus xyyw = xyz$$

Note that we have adopted the string representation for multisets. We next consider pairs $\langle M, n\rangle$, where M is a multiset and n is a positive integer, and sets P of such pairs which only include maximal pairs. That is, if $\langle M, n\rangle \in P$ then there is no other pair $\langle \bar{M}, \bar{n}\rangle$ in P such that $\bar{M} \supseteq M$ and $\bar{n} \geq n$, where inclusion takes multiplicity into account.

We define special union and intersection operations, denoted by $\vee$ and $\wedge$, over sets of maximal pairs. The definition of $\vee$ is easy:

$$P \vee Q = \{p : p \text{ is maximal in } P \cup Q\}$$

We first define $\wedge$ for singleton sets:

[4] Assuming the number N of sequences is clearly understood, we silently equate the fraction $q \in (0, 1]$ and the absolute number of sequences $\lceil q \cdot N \rceil$.

$$\{\langle M_1, n_1\rangle\} \wedge \{\langle M_2, n_2\rangle\} = \{\langle M_1 \cap M_2, n_1 + n_2\rangle\}_{\text{if } M_1 \cap M_2 \neq \emptyset} \cup \{\langle M_1, n_1\rangle\}_{\text{if } M_1 \setminus M_2 \neq \emptyset} \cup \{\langle M_2, n_2\rangle\}_{\text{if } M_2 \setminus M_1 \neq \emptyset}$$

Then, for arbitrary sets $P_1 = \{p_i^{(1)}\}_{i=1,\ldots,h}$ and $P_2 = \{p_j^{(2)}\}_{j=1,\ldots,k}$:

$$P_1 \wedge P_2 = \bigvee_{i,j} \left(\left\{ p_i^{(1)} \right\} \wedge \left\{ p_j^{(2)} \right\} \right).$$

3 Algorithm

CMF main operation mode is composite motifs discovery in a set $\mathcal{S} = \{S_1, \ldots, S_N\}$ of DNA sequences, using a collection of PWMs.[5]

PWMs can be passed to CMF in either a single or multiple files. In the latter case, CMF assumes that each file contains PWMs for only one given factor. Actually, when the input set is prepared using matrices taken from an annotated repository (*e.g.*, the TRANSFAC database [20]), assuming the knowledge of the corresponding factors is not an artificial scenario. However, here we describe the main steps implementing CMF's operation mode on input a single PWM file, namely:

1. (Optional) *PWM clustering*, to organize the matrices in classes believed to belong to different factors;
2. *Discretization*, to detect PWM matches in the input sequences;
3. *Group and composite motif finding.*

3.1 PWM clustering

By default, CMF assumes that the PWMs in the input file correspond to different factors, and hence it does not perform any clustering. However, in many cases the number of matrices available, which describe the binding affinities of the factors involved in the upstream experimental protocol, is much larger than the number of such factors. If the latter information is available to the user, then clustering may be highly useful both to improve the accuracy and to reduce the group finding complexity. Another circumstance in which clustering is advisable (not discussed here) is when the input matrices are produced by third-party motif discovery tools.

To perform the clustering, CMF first builds a weighted adjacency graph whose nodes are the matrices and edges the pairs (M_1, M_2) such

[5] Even if not taken into consideration in this paper, CMF is also able to run a number of third-party motif discovery tools to "synthesize" PWMs.

that the similarity[6] between M_1 and M_2 is above a given threshold. Then, CMF executes a single-linkage partitioning step of the graph vertices; finally, it identifies the dense cores in each set of the partition, via pseudo-cliques enumeration [22], returning them as the computed clusters.

The experiments described in Section 4 suggest that, when the PWM file mainly includes good matrices corresponding to possibly different factors, then even a simple clustering algorithm like the one mentioned above is able to correctly separate them into the "right" groups (factors). In general, however, performing a good partitioning of the input matrices when the fraction of "noisy" PWM increases (as is the case when CMF is used downstream de-novo motif discovery software tools) is one of the major issues left open for further work.

3.2 Discretization

Even with the most accurate PWM description of a motif, the problem of determining the "true" motif matches in the input sequences is all but a trivial task. Whatever the algorithm adopted, there is always the problem of setting some thresholds τ to distinguish matches from non-matches, a choice that may have a dramatic impact on the tool's performance.

In general, low thresholds improve sensitivity while high thresholds may improve the rate of positive predicted values (PPVs). A reasonable strategy is to moderately privilege sensitivity during discretization, with the hope to increase the positive predicted rate thanks to the combinatorial effect of close-by matches. Indeed, keeping initial low thresholds may give the benefit of not filtering out low-score matches[7]. On the other hand, complexity issues demand that the number of possible combinations of motif matches, which the composite motifs should emerge from, will not explode. Now, for factors with many matrices, low thresholds may incur in a very high number of matches and these in turn affect the number of potential composite motifs.

In light of the above arguments, we formulate the following general and simple qualitative criterion: assign factors (motif classes) with many/few matrices a high/low threshold. All the experiments of Section 4 were performed with fixed threshold values. Although these can be varied (in the configuration file, hence in a completely transparent way to the typical user), the overall good results suggests that the above criterion may have some merits, to be further investigated.

[6] Currently, CMF invokes RSAT's utility `compare-matrices` for this purpose [21], which uses pairwise normalized correlation

[7] Sometimes referred to as *weak signals* in the literature.

3.3 Composite Motif finding

The previous two steps result in a set of factors (motif classes) and a set of factors matches, which are the "input" to the Composite Motif Finding step. This is in turn divided into two main sub-processes:

(a) Finding combinatorial groups. CMF uses a simple search strategy, with the aim of trading computation time for accuracy. Let $\{W_1, \ldots, W_r\}$ be a set of (internal) window sizes and let $\{q_1, \ldots, q_s\}$ be a set of (internal) quorum values, with $W_1 < W_2 < \ldots < W_r$ and $1 \geq q_1 > q_2 > \ldots > q_s > 0$. For a given window size value W and sequence S_i, we say that a multiset m over $\mathcal{R}$ is *feasible* iff each letter/factor of m corresponds to a match in S_i and the span of all the matches in S_i is bounded by W.

The algorithm that computes the combinatorial groups can be described as follows.

1. Set $W = W_1$ and $q = q_1$.
2. For $i = 1, \ldots, N$, compute the maximal multisets $M_1^{(i)}, \ldots, M_{n_i}^{(i)}$ that are feasible for W and S_i, and form the set of pairs
$$P_i = \{\langle M_1^{(i)}, 1\rangle, \ldots, \langle M_{n_i}^{(i)}, 1\rangle\}$$
3. Set $G_1 = P_1$
4. For $i = 2, \ldots, N$ compute
$$G_i = G_{i-1} \wedge P_i$$
5. Discard from G_N all the pairs $\langle M, n\rangle$ such that $n < \lceil q \cdot N \rceil$.
6. If the (remaining) multisets in G_N include all the letters of $\mathcal{R}$ or $W = W_r$ and $q = q_s$, then set $G = \{M : \langle M, n\rangle \in G_N\}$ and return G.
7. In alternate order (whenever possible) advance W or q to the next value and jump to step 2.

The above general description has only explanatory purposes, since a direct implementation would be highly inefficient. For instance, when relaxing the quorum value, step 2 can be avoided, since the multisets $M_j^{(i)}$ have already been computed. On the other hand, the pairwise intersections of step 4 can be performed quite efficiently thanks to the character sorted string representation of multisets of factors.

By the properties of the $\vee$ and $\wedge$ operators, the pairs $\langle M, n\rangle$ included in G_N are maximal, with n satisfying the last fixed quorum value. Note, however, that even with the weakest parameter values (i.e., widest window and smallest quorum), some factors may not be represented in G. This is not necessarily a problem, since the user may have provided PWMs for irrelevant factors.

(b) Computing the composite motifs. For any combinatorial group g in G, CMF first retrieves its actual matches from the input sequences; then tries to extend each group of matches by possibly including other strong factors matches that do not make the extended group unfeasible with respect to the window constraint. This is done independently for each sequence. All these extended group matches form the composite motifs that CMF gives in output under the ANR (Any NumbeR) model. Under the ZOOPS (Zero Or One Per Sequence) model, groups and composite motifs are further filtered basing on the most recurrent span width (details not reported here for lack of space).

3.4 Computational cost

The cost of the bare CMF algorithm is dominated by the Composite Motif finding step or, more precisely, by the combinatorial group finding subprocess. This is easily seen to be exponential in the length of the longest group g (regarded as a string over $\mathcal{R}$) in any of the initial sets M_i's, simply because g may have an exponential number of maximal subgroups that satisfy also the quorum constraint. In turn, the length of g may be of the order of composite motif width and hence of sequence length. At the other extreme, there is the situation where we only have two (of few) factors and look for sites where both factors bind (as for the TRANSCompel datasets of Section 4). In this case the cost of the subprocess is linear in the number of sequences.

When combinatorial group finding is fast (as in all the experiments we have performed) the computational bottlenecks move to other parts of the code, *i.e.*, outside of the software module that implements the core CMF algorithm. In particular, the computation of PWM pairwise similarities takes quadratic time in the number of PWMs, which can be pretty high in a number of scenarios.

4 Experiments

In this section we present the results obtained from a number of experiments performed on the twelve benchmark datasets presented in [8][8].

We compare CMF against the eight tools considered in the assessment paper (CisModule [23], Cister [24], Cluster-Buster (CB) [25], Composite Module Analyst (CMA) [26], MCAST [27], ModuleSearcher (MS) [28], MSCAN [29] and Stubb [30]). We also consider two other (more recent)

[8] In the following, we refer to [8] as to the *assessment paper*.

tools, named COMPO, developed by the same research group that performed the assessment [13], and MOPAT [14]. We based our choice on tools whose code was available or for which we could find reported results for all datasets taken into consideration in this paper. We also compare CMF against CORECLUST [15] on just one dataset, the only one for which data are available.

4.1 Datasets

We use the TRANSCompel as well as the liver and muscle datasets presented in [8]. The TRANSCompel benchmark includes ten datasets corresponding to as many composite motifs, each consisting of two binding sites for different factors.

In [8], all the matrices corresponding to a same factor were grouped to form an "equivalence set", and treated as they were one. These matrices form what is called, in the assessment paper, the *noise_0 benchmark*. To simulate conditions in which input data are fuzzier, we also consider the so-called *noise_50 benchmark* presented in [8], in which each dataset is composed of an equal number of good and random (i.e., taken at random from TRANSFAC) matrices.

Two additional benchmarks are discussed in [8], namely *liver* and *muscle*, having very different characteristics from the previous ones. Liver includes sequences with up to nine binding sites from four different factors, while muscle includes sequences with up to eight sites from five factors.

Statistics for tools evaluated in [8] were downloaded from the site `http://tare.medisin.ntnu.no/composite/composite.php`. Regarding COMPO, we computed the statistics for liver and muscle datasets starting from the prediction files made available by the authors at the address `http://tare.medisin.ntnu.no/compo/`. For the TRANSCompel datasets (noise_0 and noise_50), we directly used the statistic results provided at the same address.

4.2 Scoring predictions

We compare CMF against all the other eleven tools using the *correlation coefficient* (CC). We also compare CMF and COMPO (the best performing tool among CMF's competitors) using other popular statistics, namely: *Sensitivity* (Sn), *Positive Predicted Values* (PPV), *Performance Coefficient* (PC), and *Average Site Performance* (ASP) (see [31] for definitions). All the mentioned statistics are computed at the *nucleotide-level*. CMF and COMPO are also compared using *motif level* statistics.

Dataset/Tool	CMF	COMPO	CB	Cister	MSCAN	MS	MCAST	Stubb	CMA	CM	MOPAT	C
AP1-Ets	**0.52**	0.19	0.24	0.00	0.11	0.30	0.20	0.15	0.22	-0.0	0.37	
AP1-NFAT	0.11	0.06	0.04	0.00	0.00	0.05	0.14	-0.01	**0.15**	-0.02	0.14	
AP1-NFkB	**0.76**	0.59	0.49	0.19	0.36	0.29	0.26	0.35	0.55	0.05	0.18	
CEBP-NFkB	**0.74**	0.70	0.72	0.45	0.56	0.56	0.60	0.36	0.60	-0.03	0.38	
Ebox-Ets	**0.59**	0.55	0.16	0.26	0.44	0.20	0.23	0.14	0.18	0.05	0.15	
Ets-AML	**0.49**	0.42	0.30	0.07	0.31	0.38	0.26	0.23	0.33	0.03	0.27	
IRF-NFkB	**0.92**	0.73	0.77	0.62	0.91	0.85	0.41	0.41	0.69	0.04	0.57	
NFkB-HMGIY	0.26	0.31	0.35	0.10	0.30	**0.40**	0.23	0.07	0.15	-0.03	0.13	
PU1-IRF	**0.92**	0.28	0.16	0.27	0.00	0.43	0.16	0.17	0.24	-0.01	0.21	
Sp1-Ets	**0.20**	0.05	0.09	**0.20**	0.00	0.00	0.13	0.19	0.15	0.02	0.09	
Liver	0.49	0.57	**0.59**	0.31	0.51	0.42	0.50	0.48	0.36	-0.01	0.33	
Muscle	**0.56**	0.52	0.41	0.36	0.50	0.46	0.30	0.24	0.46	0.29	0.37	**0.56**

Table 1. CC results for noise_0, liver, and muscle data, with best figures in bold-face. CB = Cluster-Buster, MS = ModuleSearcher, CMA = Composite Module Analyst, CM = CisModule, C = CORECLUST.

4.3 Results

In all the experiments, CMF was run with fixed configuration file, with $W = \{50, 75, 100, 125, 150\}$, $q = \{0.9, 0.8, 0.7, 0.6., 0.5., 0.4, 0.3, 0.2, 0.1\}$ (see Section 3.3).

Nucleotide level analysis. Table 1 shows the results obtained by CMF compared to eleven competitor algorithms on the whole collection of twelve datasets (noise_0, liver, and muscle). The results suggest that CMF is indeed competitive with other state of the art tools. In the attempt to assess the significance of the results of Table 1, we first performed a Friedman non-parametric test (see, e.g., [32]) that involved eleven tools (all but CORECLUST, because of the limited availability of homogeneous data with which to perform the comparisons). As it can be easily argued, here the null hypothesis (i.e., that all the considered algorithms behave similarly, and hence that the average ranks over the all datasets are essentially the same), can be safely rejected, with a P-value around $2.2 \cdot 10^{-9}$.

We then performed the post hoc tests associated to the Friedman statistics, by considering CMF as the new proposed methods to be compared against the other ten tools. Table 2 shows the P-values of the ten comparisons, adjusted (according to the Hochberg step-up procedure [32]) to take into account possible type-I errors in the whole set of comparisons.

COMPO	ClusterBuster	MS	CMA	MSCAN	MCAST	MOPAT	Cister	Stubb	CM
0.1961	0.0455	0.0455	0.0455	0.0308	0.0102	0.0012	$3.4e^{-4}$	$1.9e^{-5}$	$6.9e^{-10}$

Table 2. Adjusted P-values for post hoc comparisons of CFM against other 10 tools: MS = ModuleSearcher, CMA = Composite Module Analyst, CM = CisModule

For nine competing algorithms we obtained figures below the critical 0.05 threshold; only in case of COMPO we cannot reject with high confidence the null hypothesis, namely that the observed average ranks of the two algorithms (CMF and COMPO) are different by chance only.

We next concentrate on the comparison between CMF and COMPO, which is the best performing tool among the CMF competitors considered here. Table 3 compare CMF and COMPO on a wider sets of statistics. For the noise_0 benchmark, the results shown combine the results of the corresponding ten datasets (i.e., counting the total numbers of positive, positive predicted, negative, and negative predicted nucleotides). For the noise_50 benchmark, the combined results are the average of the figures obtained on ten runs on each datasets. In each run, the "good" matrices were mixed with different sets of decoy PWMs (see [8]).

Statistics	Noise_0	Noise_50	Liver	Muscle	Tool
PPV	**0.67**	**0.45**	0.67	**0.60**	CMF
	0.40	0.37	**0.85**	0.52	COMPO
Sn	**0.54**	**0.49**	**0.429**	0.65	CMF
	0.47	0.48	0.425	**0.69**	COMPO
PC	**0.42**	**0.31**	0.35	**0.45**	CMF
	0.28	0.26	**0.40**	0.42	COMPO
ASP	**0.60**	**0.47**	0.55	**0.62**	CMF
	0.44	0.42	**0.64**	0.60	COMPO
CC	**0.58**	**0.45**	0.49	**0.56**	CMF
	0.41	0.39	**0.57**	0.52	COMPO

Table 3. Further nucleotide level comparisons between CMF and COMPO. We report here the results most favorable to COMPO, as the authors provide three different files with predictions for each datasets.

Motif level analysis. We compared CMF and COMPO to understand their ability to correctly tell the matrices (possibly within an equivalence set) whose matches belong to a predicted composite motif. In contrast

to [8], we do not consider here true negative predictions, as we regard the concept of *true negative* not well defined at the motif level[9]. Hence we only computed Sensitivity, Positive Predicted Value, and Performance Coefficient as motif level statistics.

Table 4 reports the performances at motif level obtained using our computed CMF predictions and the predictions made by COMPO on the TRANSCompel datasets. The results are essentially similar, with a slightly better Performance Coefficient (the sole comprehensive measure computed at motif level) exhibited by CMF. The results suggest once more that our software is competitive with current state of the art tools.

Statistics	AP1-Ets	AP1-NFAT	AP1-NFkB	CEBP-NFkB	Ebox-Ets	Ets-AML	Tool
Sn	**0.647**	0.045	**0.75**	0.625	0.417	**0.9**	CMF
	0.47	**0.364**	0.625	**0.75**	**0.5**	0.8	COMPO
PPV	0.733	0.5	**1**	**1**	0.833	0.9	CMF
	1	**1**	**1**	**1**	**1**	**1**	COMPO
PC	**0.524**	0.043	**0.75**	0.625	0.385	**0.818**	CMF
	0.47	**0.364**	0.625	**0.75**	**0.5**	0.8	COMPO

Statistics	IRF-NFkB	NFkB-HMGIY	PU1-IRF	Sp1-Ets	Combined	Tool
Sn	**1**	0.357	**1**	**0.438**	**0.574**	CMF
	0.833	**0.429**	0.6	0	0.506	COMPO
PPV	**1**	0.625	**1**	**0.875**	0.861	CMF
	1	**1**	**1**	0	**0.932**	COMPO
PC	**1**	0.294	**1**	**0.412**	**0.525**	CMF
	0.833	**0.429**	0.6	0	0.488	COMPO

Table 4. Motif level results for CMF and COMPO on the noise_0 dataset

Finally, Table 5 reports CMF statistics on the muscle and liver datasets. We do not include a comparison against COMPO here since the way to correctly and fairly interpret COMPO's prediction is not completely clearly to us. First of all, the authors present three different prediction sets, obtained under different configuration runs. Secondly, all the prediction files contain multiple identical predictions, which negatively influences the PPV counts.

[9] Note that in the already cited paper by Tompa et al. [31], true negative predictions at the motif level are not considered.

Liver Dataset			Muscle Dataset		
Sn	PVV	PC	Sn	PVV	PC
0.5	0.728	0.42	0.74	0.66	0.54

Table 5. CMF motif level statistics for the muscle and liver datasets

5 Conclusions

In this paper we have presented CMF, a novel tool for Composite Motif detection, a computational problem which is well-known to be very difficult. Indeed, to date, no available software for (simple or composite) motif discovery can be clearly identified as the "best one" under all application settings. Knowing this, we are also aware that more comparisons are required, in different experimental frameworks, for general conclusions to be drawn about the competitiveness of CMF.

However, we think that some interesting findings have emerged from this work, all related to the power of simple motif combinations. First of all, that the good results exhibited by CMF have been obtained without using any sophisticated statistic filtering criteria; the combination of "right" simple sites were often strong enough to emerge from a huge set of potentially active motif clusters. Secondly, that the conceptually simple CMF architecture, based on a two-stage approach to composite motif finding (i.e., first detect simple motifs, then combine them to form clusters of prospective functional motifs) proved to be competitive against other, more sophisticated approaches (see also [12]). In the third place, that lowering the thresholds that "define" (in silico) the DNA occupancy by a transcription factor, can be appropriate a strategy that can be kept hidden to the user.

On the other hand, the same issues outlined in the preceding paragraph suggest possible directions to improve CMF performance. For instance, incorporating a statistical filtering may enhance the PPV rate of the prospective composite motifs devised by simple site combinations. However, we think that the most delicate aspect has to do with thresholding and discretization. It is a growing popular belief among biologists that the DNA occupancy is determined mostly by chromatin accessibility (rather than DNA-factor affinities), with the occupancy scale being a continuum of thermodynamics levels. Turning this knowledge into a computable property of the potential binding site seems indeed a hard challenge.

6 Acknowledgments

The present work is partially supported by the Flagship project *InterOmics* (PB.P05), funded by the Italian MIUR and CNR organizations, and by the joint IIT-IFC Lab for Integrative System Medicine (LISM).

References

1. Davidson, E.H.: The Regulatory Genome: Gene Regulatory Networks In Development And Evolution. 1 edn. Academic Press (2006)
2. Pavesi, G., Mauri, G., Pesole, G.: In silico representation and discovery of transcription factor binding sites. Briefings in Bioinf. **5** (2004) 217–236
3. Sandve, G.K., Drabløs, F.: A survey of motif discovery methods in an integrated framework. Biol. direct **1** (2006) 11+
4. Häußler, M., Nicolas, J.: Motif Discovery on Promotor Sequences. Research Report RR-5714, INRIA (2005)
5. Zambelli, F., Pesole, G., Pavesi, G.: Motif discovery and transcription factor binding sites before and after the next-generation sequencing era. Brief. in Bioinf. (2012)
6. Wingender, E., et al.: Transfac: a database on transcription factors and their dna binding sites. Nucl. Acids Res. **24** (1996) 238–241
7. Sandelin, A., Alkema, W., Engström, P.G., Wasserman, W.W., Lenhard, B.: Jaspar: an open-access database for eukaryotic transcription factor binding profiles. Nucl. Acids Res. **32** (2004) 91–94
8. Klepper, K., Sandve, G., Abul, O., Johansen, J., Drabløs, F.: Assessment of composite motif discovery methods. BMC Bioinformatics **9** (2008) 123
9. Sinha, S.: Finding Regulatory Elements in Genomic Sequences. PhD thesis, University of Washington (2002)
10. Van Loo, P., Marynen, P.: Computational methods for the detection of cis-regulatory modules. Briefings in Bioinf. **10** (2009) 509–524
11. Ivan, A., Halfon, M., Sinha, S.: Computational discovery of cis-regulatory modules in drosophila without prior knowledge of motifs. Genome Biology **9** (2008) R22
12. Federico, M., Leoncini, M., Montangero, M., Valente, P.: Direct vs 2-stage approaches to structured motif finding. Algorithms for Molecular Biology **7** (2012) 20
13. Sandve, G., Abul, O., Drablos, F.: Compo: composite motif discovery using discrete models. BMC Bioinf. **9** (2008) 527
14. Hu, J., Hu, H., Li, X.: Mopat: a graph-based method to predict recurrent cis-regulatory modules from known motifs. Nucleic Acids Research **36** (2008) 4488–4497
15. Nikulova, A.A., Favorov, A.V., Sutormin, R.A., Makeev, V.J., Mironov, A.A.: Coreclust: identification of the conserved crm grammar together with prediction of gene regulation. Nucl. Acids Res. (2012) doi: 10.1093/nar/gks235.
16. Vavouri, T., Elgar, G.: Prediction of cis-regulatory elements using binding site matrices - the successes, the failures and the reasons for both. Curr. Opinion in Genetics & Develop. **15** (2005) 395 – 402
17. Kel, A., Göß ling, E., Reuter, I., Cheremushkin, E., Kel-Margoulis, O., Wingender, E.: Matchtm: a tool for searching transcription factor binding sites in dna sequences. Nucleic Acids Research **31** (2003) 3576–3579

18. Chen, Q.K., Hertz, G.Z., Stormo, G.D.: Matrix search 1.0: a computer program that scans dna sequences for transcriptional elements using a database of weight matrices. Comp. appl. in the biosciences : CABIOS **11** (1995) 563–566
19. Prestridge, D.S.: Signal scan: a computer program that scans dna sequences for eukaryotic transcriptional elements. Comp. appl. in the biosc.: CABIOS **7** (1991) 203–206
20. Matys, V., et al.: TRANSFAC and its module TRANSCompel: transcriptional gene regulation in eukaryotes. Nucl. acids Res. **34** (2006) D108–D110
21. Thomas-Chollier, M., et al.: RSAT: regulatory sequence analysis tools. Nucleic Acids Research **36** (2008) W119–W127
22. Uno, T.: Pce: Pseudo clique enumerator, ver. 1.0 (2006)
23. Zhou, Q., Wong, W.H.: Cismodule: De novo discovery of cis-regulatory modules by hierarchical mixture modeling. Proc. of Nat. Ac. of Sci. **101** (2004) 12114–12119
24. Frith, M.C., Hansen, U., Weng, Z.: Detection of cis -element clusters in higher eukaryotic dna. Bioinf. **17** (2001) 878–889
25. Frith, M.C., Li, M.C., Weng, Z.: Cluster-Buster: Finding dense clusters of motifs in DNA sequences. Nucl. Acids Res **31** (2003) 3666–8
26. Kel, A., Konovalova, T., Waleev, T., Cheremushkin, E., Kel-Margoulis, O., Wingender, E.: Composite module analyst: a fitness-based tool for identification of transcription factor binding site combinations. Bioinf. **22** (2006) 1190–1197
27. Bailey, T.L., Noble, W.S.: Searching for statistically significant regulatory modules. Bioinformatics **19** (2003) ii16–ii25
28. Aerts, S., Van Loo, P., Thijs, G., Moreau, Y., De Moor, B.: Computational detection of cis -regulatory modules. Bioinf. **19** (2003) ii5–ii14
29. Johansson, Ö., Alkema, W., Wasserman, W.W., Lagergren, J.: Identification of functional clusters of transcription factor binding motifs in genome sequences: the mscan algorithm. Bioinf. **19** (2003) i169–i176
30. Sinha, S., van Nimwegen, E., Siggia, E.D.: A probabilistic method to detect regulatory modules. Bioinf. **19** (2003) i292–i301
31. Tompa, M., et al.: Assessing computational tools for the discovery of transcription factor binding sites. Nature Biotechnology **23** (2005) 137–144
32. García, S., Fernández, A., Luengo, J., Herrera, F.: Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power. Information Sciences **180** (2010) 2044–2064