BC205: Algorithms for Bioinformatics. IV. Motif Discovery

Christoforos Nikolaou

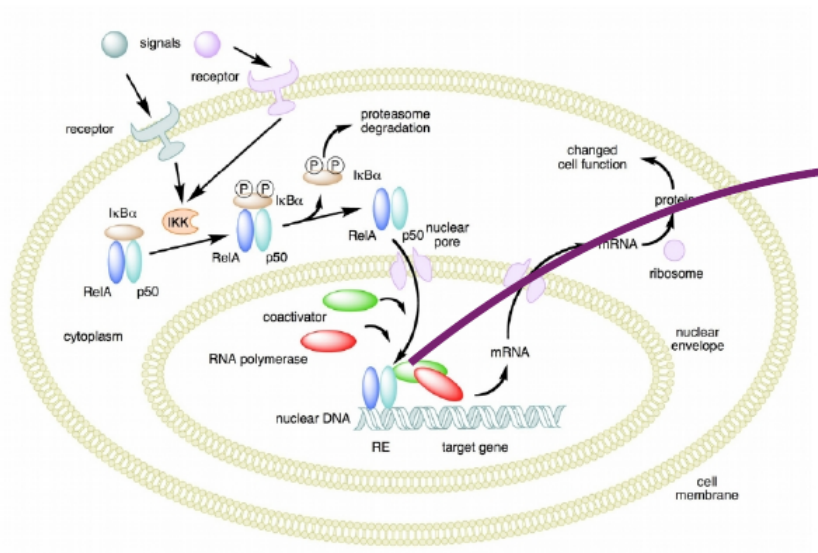
March 29th, 2018

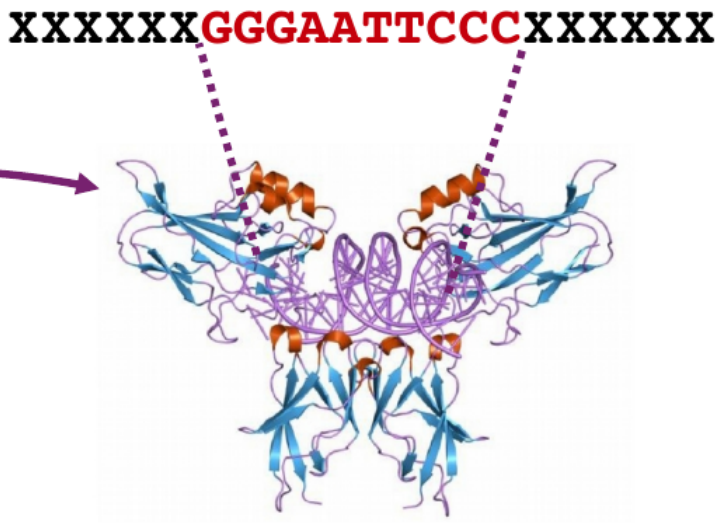
In previous chapters

- We saw how:
  - We define a sequence motif
  - We can search for known short motifs with a determined degree of ambiguity
  - We can estimate the existence of a motif in a sequence
  - We can define the strength of the motif in the sequence in an Entropy-based score

The *'hard'* motif finding problem

- Given a set of sequences, can you locate sequence instances that will represent a motif?
- In simpler words: How do we **discover** motifs in sequences?





Part #1: Formulating the problem

1. Given a set of s sequences: Find a set of k-mers (for a given length k, one from each sequence) that maximizes the score (or minimizes the distance) of each (one) k-mer with its sequence
2. Collect k-mers
3. Create a motif from them

The **Brute Force Approach**

- Assuming we have a way to calculate the distance of a k-mer *k* from a given sequence *seq*

```
for seq in sequences:
  for k in kmers:
    if distance(k, seq)<min_distance:
      min_distance<-distance(k,seq)
      motif[seq]<-k
```

Brute Force Approach: Implications

- What is the complexity of the BFA?
  1. Number of k-mers 4^k
  2. Number of k-mers in each sequence: $(n - k + 1)$
  3. Number of calculations for each k-mer given s sequences of length n: $(n - k + 1)^s$
  4. Total number of calculations $4^k(n - k + 1)^s$
- The complexity of the algorithm is at least $O(n^s)$.
- We need something faster!

Part #2: Finding the **Median String**

- Assuming we have a way to calculate the distance of a k-mer *k* from a given sequence *seq*

```
for k in kmers:
  for seq in sequences:
    if distance(k, seq)<min_distance:
      min_distance<-distance(k,seq)
      motif[seq]<-k
```

- Because each k-mer needs to pass only once through each sequence, the median string has $O(4^k)$ complexity because k is (usually) much shorter than the length of the sequence.
- However, it is still quite slow and for k>10 its implementation is still unapplicable.

Part #3: A faster heuristic approach

- Assume a greedy approach to go through all sequences updating a motif every time
- Starting from sequence i:
  1. find the most common k-mer
  2. create a profile from it (adding pseudocounts to all 0-values)
  3. go to the next sequence
  4. choose the k-mer that best fits the profile
  5. store that k-mer in the collection and update profile
  6. iterate steps 3->5.
- We' ve just described a Greedy approach for discovering a motif p of a given length k among t sequences.

Trying a **Greedy Approach** for Motif Discovery

- Assuming a set of s sequences and a given consensus k-mer k: We will construct a PWM "on the go" as we move from one sequence to the next.
  1. For i=1 :
  2. For each k in $seq[i]$:
     - 2.1 For $i = 2$ to $i = s$:
     - 2.2 Find the best (smallest distance) kmer in $seq[i]$
     - 2.3 Build a profile
     - 2.4 If the score(profile) is better than all previous update profile Repeat

Greedy Approach: Implications

1. Why Greedy: It takes kmers from the first sequence only to scan in the following. Thus it doesn't go through all combinations of sequences and k-mers. As we've seen above the trade-off is speed.
2. KEY: It assumes that all sequences contain the motif. If the first sequence doesn't contain the motif (in any variation) then we are doomed in looking for something that is non-sensical.
3. A way to go around this is to sample a small percentage of sequences randomly, which brings us to the next-to-last chapter of the motif finding problem

Part #4: A **Randomized Approach**

- In the **Greedy Approach** we take the kmers from the first sequence and scan over the rest. In this way an initial wrong choice may lead you to disastrous results.
- In a **Randomized Approach** we start, instead with a collection of s k-mers, one from each sequence, build a profile, scan the sequences with that profile, update it and repeat until the k-mer set is good enough match for the updated profile.
- Stop and think of the problems we get rid of with this approach.

A **Randomized Approach**: Pseudocode

- Starting from s sequences and a kmer length k. We set a threshold for the distance of the profile we want to assure:

```
for seq in sequences:
  profile[seq]<-random(k, seq)
while distance(profile, sequences)>threshold
  for seq in sequences:
    # choose the best k based on
    # the current version of the profile
    # and replace the corresponding ;
    profile[seq]<-max(k, profile, seq)
```

Think: Why would the randomized approach work better?

- It's all about the probability. Given that the motif **really is** somewhere in our sequences it is more likely to pick a k-mer that is close to the probability of the profile instead of the background composition.
- This is then further improved every time since the iteration is based on an optimization process.
- Also think: There is a great chance that most (even all) k are changed in every iteration of the algorithm. This radical approach can increase the time of algorithm completion. Can we devise a more "careful" strategy?

Gibbs Sampler: An improved Randomized Approach

- Based on the randomized approach it just applies the iteration to **one** sequence each time.

```
for seq in sequences:
  profile[seq]<-random(k, seq)
while distance(profile, sequences)>threshold
  # randomly sample one sequence out of the set
  seq<-random(sequences):
    # choose the best k based on
    # the current version of the profile
    # and replace the corresponding ;
  profile[seq]<-max(k, profile, seq)
```

- In this way it is more conservative than a fully randomized approach. It proceeds with greater caution than the fully randomized approach

The **Randomized Approach**: Implications

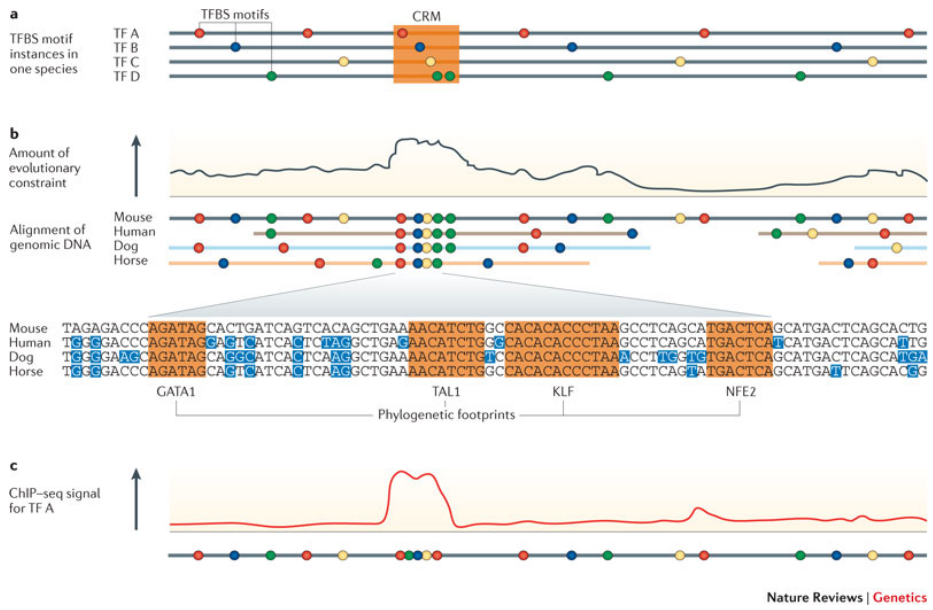
- It strikes a balance between speed and accuracy. It is fast and more robust than the Greedy Approach
- It doesn't need to have the motif in *every* sequence since sampling is performed
- Multiple runs are affordable because of the its speed and so we can increase accuracy through "voting" (take the profile that is represented in the greatest number of repeats)
- The **Gibbs Sampler** falls into a general category of *semiheuristic* methods for optimization problems that attempt to "explore" the space of solutions in the neighborhood of a given solution. They are often quite efficient in finding global instead of local solutions.

Regulatory Motif finding: Other aspects to consider

- Even the most elaborate of motif finding approaches fall short of retrieving the regulatory potential of sequences without additional information. Information that is used falls in the following main categories:
  1. Positional aspects: Clustering/clumping of motifs. Motif density
  2. Structural information: Affinity with known protein structures increases prediction accuracy
  3. Sequence Conservation: Approaches that take into account motif conservation outperform most others
  4. High throughput Experimental validation: ChIP approaches coupled with NGS offer unprecedented precision and lead to refined predictions **and** valuable insights in the regulatory process
  5. Gene Expression correlation: Powerful but imply a great number of experiments is available

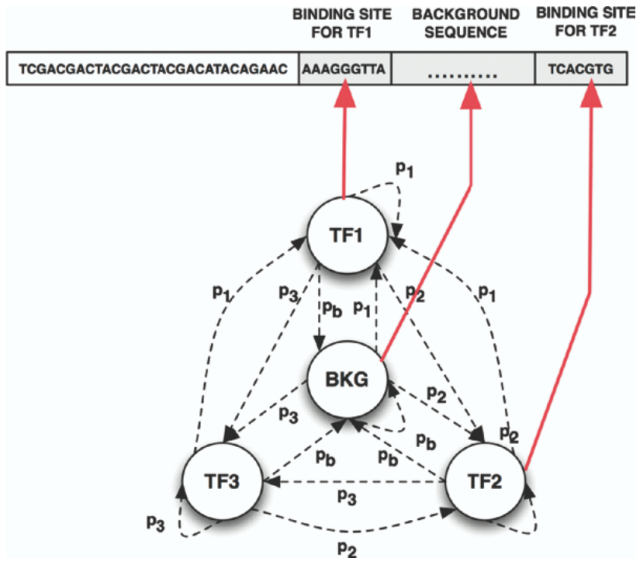
1. Positional Effects

- Clustering of TFBS is often observed, especially in regions with high regulatory potential, called CRM (cis-regulatory modules). CRMs are the "signature" of many enhancer sequences



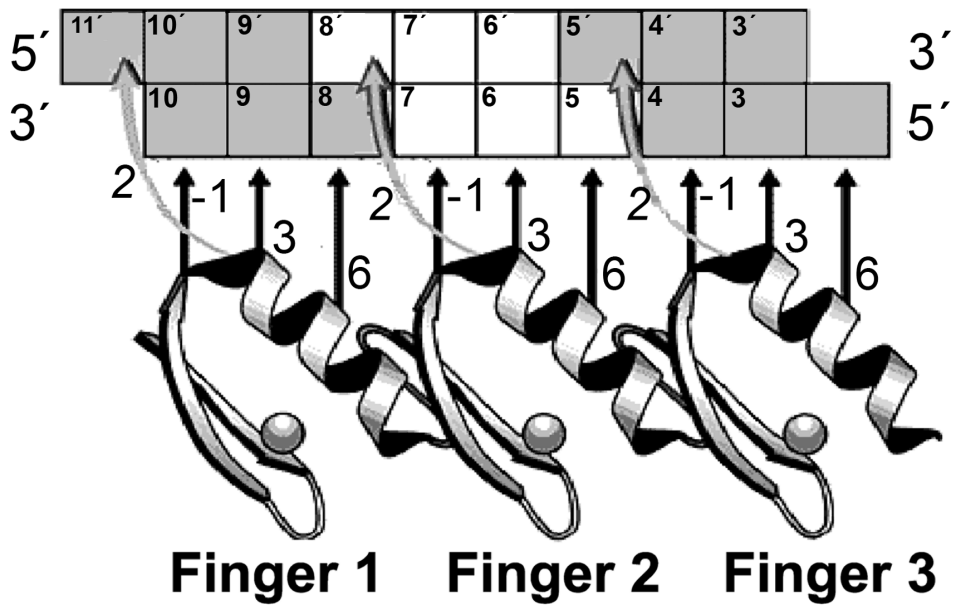
Computational Techniques for "motif clumping":

- These are based on:
  - Definition of motifs
  - Modeling of their co-occurrence based on some statistical model for their distribution (Poisson or, more often, Negative Binomial)
- Alternative approaches include HMMs for the definition of a "motif cluster"



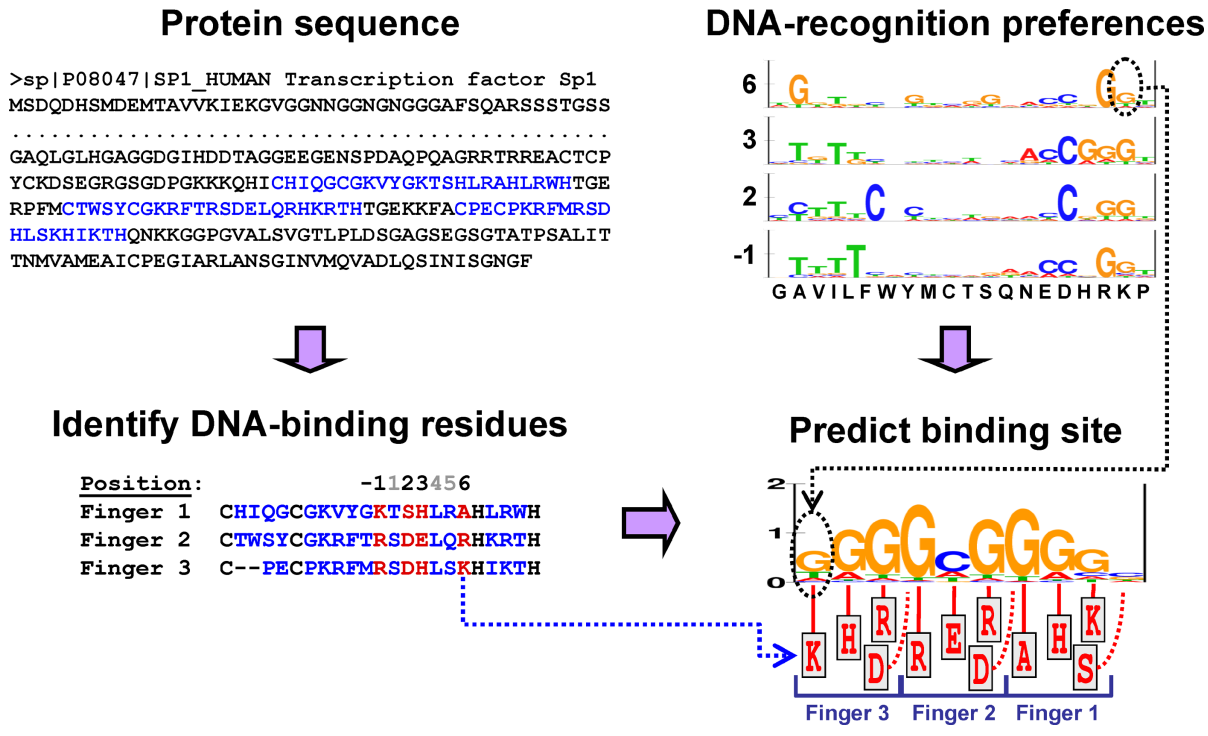
2. Structural Information

- Knowledge of the protein that bounds a specific regulatory motif may assist us in the refinement of the prediction.
- For instance, a very common protein family called “Zinc Fingers” has a specific modular architecture with similar residues repeated periodically



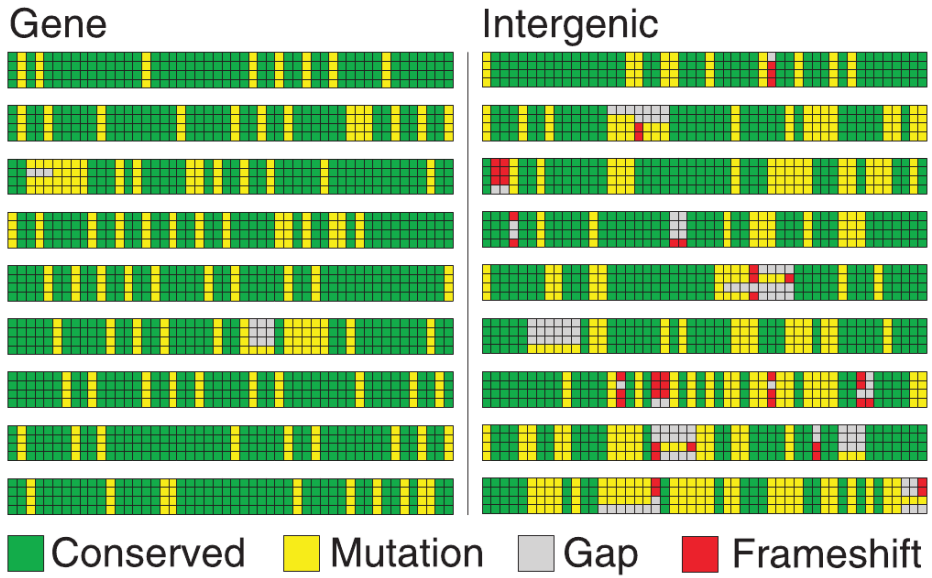
Structural Information

- This information can be used to:
  - Create aminoacid-nucleotide preference maps
  - Refine the initial predicted binding sites



3. Sequence Conservation

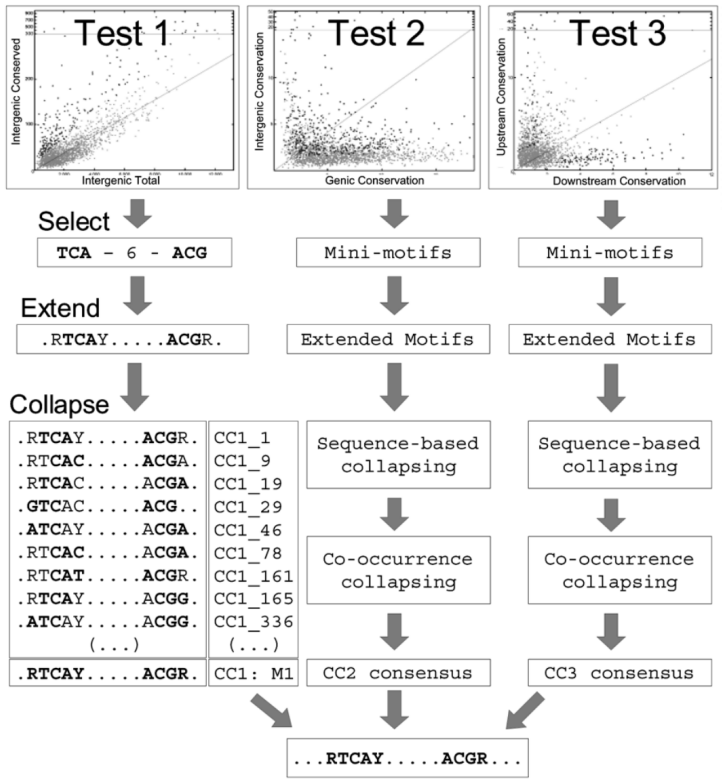
- The conservation of the genome sequence is the first and ultimate indication of function.
- Availability of sequences from related species can assist us in locating "genomic functional footprints"



- In the Figure above, sequences from genic and intergenic regions show similar patterns of conservation. In some aspects, the intergenic region is even more conserved

Sequence Conservation

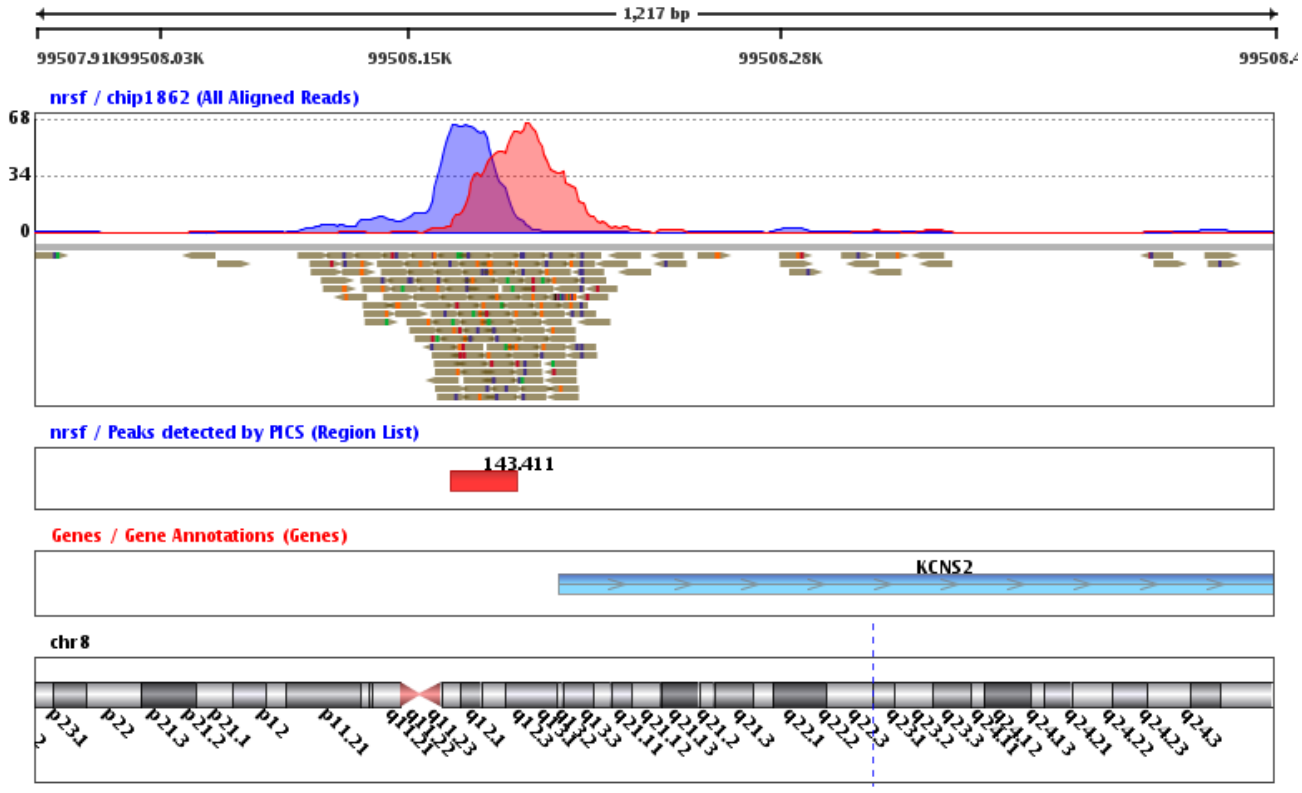
By stratifying conservation at genic and intergenic level, Kellis et al (2004) achieved to identify a number of previously unreported TFBS in yeast.



The strategy was to identify kmers that: a) are more conserved in intergenic than genic regions, b) occur in the gene upstream regions of similar genes c) the "genomic neighborhood" is conserved

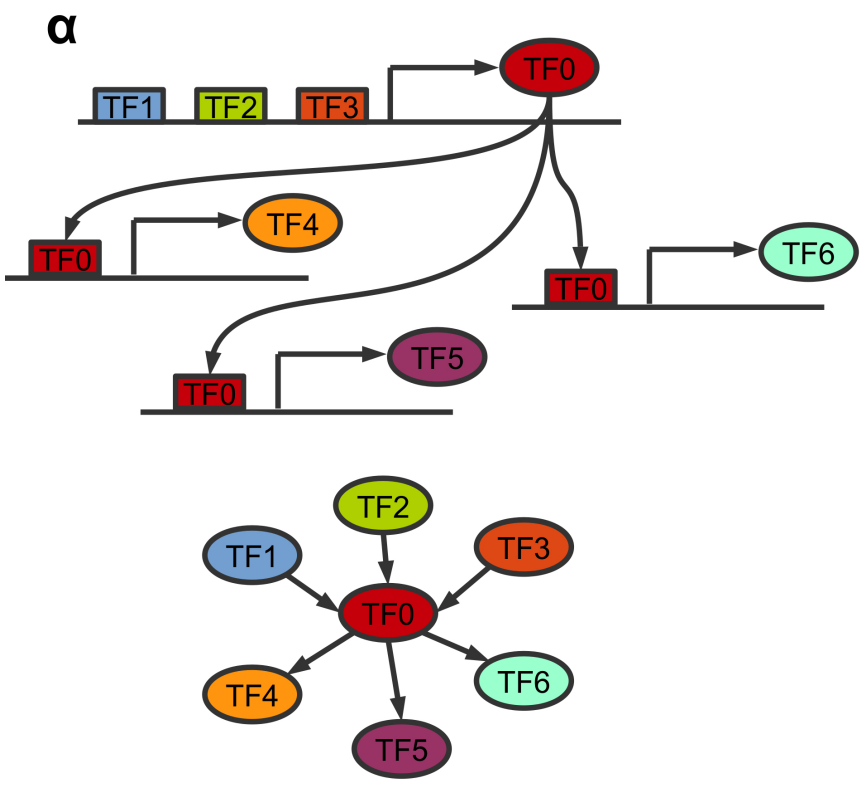
4. High-Throughput Experimental Support

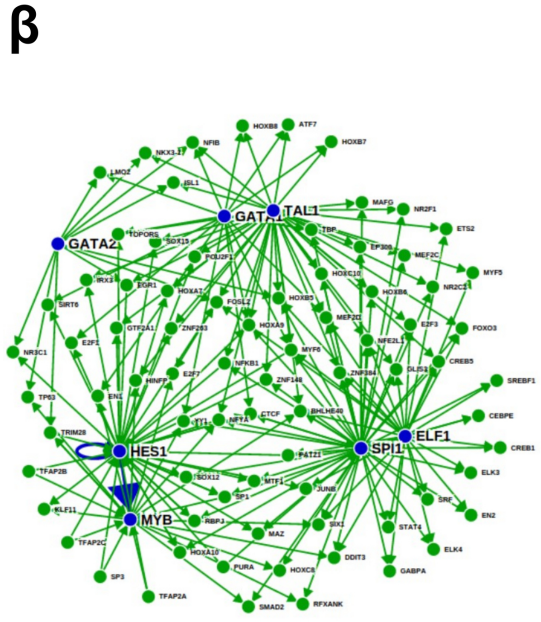
- The state of the art experimental methodologies provide us with a large number of potential sequences from which we draw the motifs. In this way:
  - We have experimental support of likely motif existence
  - We have big sequence numbers that allow us to sample motifs



High-Throughput Experiments: Regulatory Insight

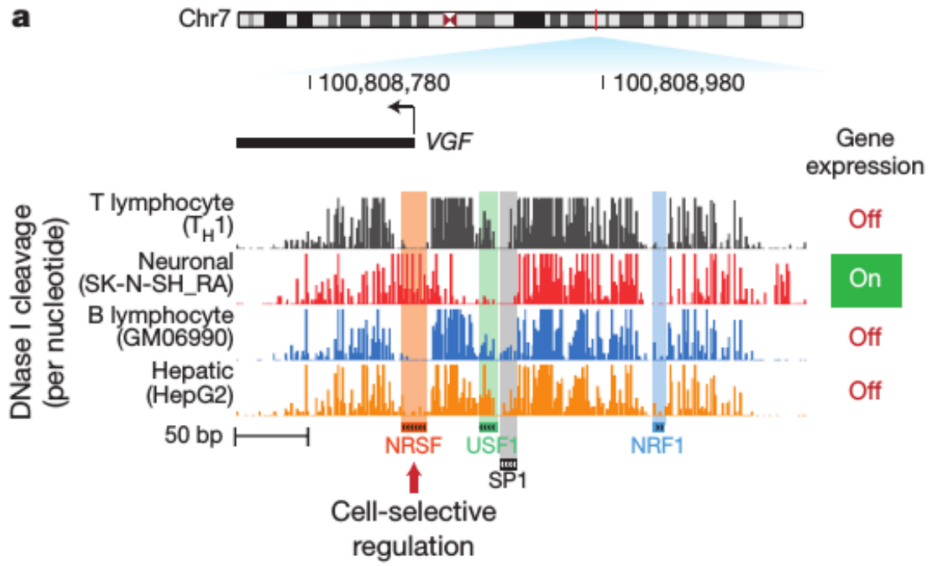
- Multiple high-throughput experiments (or other experimental techniques applied in parallel) allow us to define the compendium of TFBS in the promoters of genes and thus:
  1. Establish regulatory links between genes
  2. Reconstruct the regulatory network of a given condition





5. Gene Expression correlation

- Availability of gene expression experiments can help us deduce the regulatory background
- Algorithms may take into account gene expression levels that assist in assigning existence/function/classification of motifs



- In the example above can you say what is the function of the NRSF protein?

Conclusions

- Ab initio approaches can be fast but their accuracy is always dependent on the sequence input (junk in - junk out). If the initial sequence set does not include the motifs or is too "diluted" in terms of information, most approaches will fail
- Ensemble methods, that incorporate other types of information are preferable
- Sequence conservation and gene expression are the best options for complementary information

Exercises #2

- Write a program in which you will:

1. Implement the Gibbs Sampling Approach on the *motifs_in_sequence.fa* file to define the pattern that is implanted in the sequences. The final output should be one instance of the motif per sequence
2. Extract the Consensus Sequence of the Motif
3. Calculate the Information per Residue for the motif (to be used in a WebLogo creation).
4. Apply your program to the *motifs_in_sequence.fa* and search for the k-size that produces the motif with maximal Information