

Problem formulations

“Problem formulation is normally the most difficult part of the process. It is the selection of design variables, constraints, objectives, and models...”

– Wikipedia :: Multidisciplinary design optimization

Problem formulations?

- A problem without a solution!!
- Then specific..
 - Objectives
 - Constraints
 - Inputs
 - Outputs
 - Process (may include algorithms)

A “poor” formulation..

- ..may not have a problem to solve.
- ..may not specify objectives/constraints.
- ..may poorly define input/output.
- ..may be vague.

A “good” formulation..

- ..has a problem to solve.
- ..meets all criteria (specific objectives and constraints, well-defined input/output).
- ..may or may not suggest a process to solve problem (ideally so, but maybe not).

For this class..

- ..a formulation..
 - ..is not a bioinformatics pipeline.
 - ..does not solve a trivial problem.
 - ..is not written in “Biology language”..
 - ..but in “CS/Math language”.
 - ..but has a biological motivation.

One-page initial computational formulation

The problem: We have multiple distantly-related genomes with transcription factor (TF) binding site (TFBS) annotation. The sites are linked to the genes being regulated and to the particular TF variants that preferentially bind there. Is it possible to systematically detect very ancient links between particular TF types/families and their targets? That is, can one devise an algorithm that uses TFBS data and comparative genomics to identify pairs of TFs and their immediate targets that have persisted longer than the particular TF binding sites themselves? (For an example see [3].)

Inputs:

For each of several diverse metazoan genomes, a list of binding sites with the corresponding transcription factor binding element type, the specific sequence at that site, the regulated gene, and any putative transcription factor(s) along with a PSSM (or consensus sequence) for that binding element.

Outputs:

For each conserved ancient TF-target pair, the following outputs will be produced:

- the TF family
- a list of persistent direct targets
- a phylogenetic tree of the known motifs for each TF-target pairing

Algorithms:

The first stage will involve comparative genomics: scanning the annotation of all available genomes for any genes that have a common regulatory TF type across all (or nearly all) known metazoans. This otherwise straightforward process may be complicated by distantly-related TFs and their divergent binding elements. The results are then sorted by TF type.

Example 1 : a “poor” formulation

The second stage is phylogenetic tree construction for the each binding element. We are specifically considering linkages that persist over time-spans potentially longer than those of a given binding motif. Thus, we need to be able to construct a reliable phylogeny (with statistical significance values on the branches) from relatively little information. The problem of how to construct binding motif phylogenies is not yet solved and there is room for algorithmic innovation here [1, 2].

Taking advantage of species phylogeny information will be key to constructing the tree. Other approaches have tried to deal with the problem of assembling *de novo* phylogenies for novel motifs. Here we can take advantage of existing annotation and a link to species phylogeny. The algorithm will need to take into account additional sources of information, such as molecular clock data, motif similarity/alignment databases (e.g. JASPAR, TRANSFAC), and perhaps structural data in the form of “familial binding profiles” (FBPs) [2]. It will also need to be able to test for novel alignments using specialized dynamic programming methods (e.g. modified Smith-Waterman local alignments) with techniques for dealing with gapped half-sites, variable gaps, and low-information “edges” [2].

In order to provide confidence statistics for each tree, it is likely that a form of “bootstrap analysis” will be required. However, due to the shortness and degeneracy of the motifs as well as the relatively small number of annotated metazoan genomes, this analysis may require special statistical techniques [2].

References:

1. Kankanien M and Löytynoja A. **MATLIGN: a motif clustering, comparison and matching tool.** *BMC Bioinformatics* 2007, **8**:189.
2. Mahony S and Benos PV. **STAMP: a web tool for exploring DNA-binding motif similarities.** *Nucleic Acids Research* 2007, **35**:W253-W258.
3. Rebeiz M, Stone T and Posakony JW. **An ancient transcriptional regulatory linkage.** *Devel. Biol.* 2005, **281**:299-308.

Example 1 : a “poor” formulation

Introduction

Ultra-high throughput sequencing technology has enabled a host of studies in the field of genomics. The technology promises to improve speed and accuracy of sequencing projects while simultaneously lowering cost. While potential applications in genomics are clear, ultra-high throughput sequencing now offers an alternative to microarrays to those interested in the area of transcriptomics. Several studies now exist that have utilized sequencing technology to elucidate the expression levels of genes in the same way as microarrays while simultaneously enabling de Novo discovery of a variety of features (i.e. exon/alternative splicing boundaries, post-transcriptional modifications).

The Missing Reads

An interesting feature of some of these studies is the smaller than expected number of reads that map uniquely to the genome using commodity alignment tools. We specifically look at the results presented in two papers published in 2008. As shown in table 1, these studies mapped 40% and 21%, respectively, of their reads to a reference human genome. Each study used a separate, but publicly available, alignment tool with similar parameters (2 mismatches allowed). The goal of this project is to better understand possible explanations of this phenomenon and propose methods for testing the veracity of these explanations.

	Marioni et. al.	Li et. al.
Unique Mappings	40%	21%
Software Used	ELAND	MosaikAligner
Mismatches Allowed	2	2

Table 1: The Missing Reads

Example 2 : a “good” formulation

We consider three possible explanations for the missing reads. These are:

- Sequencing Errors. Larger than expected sequencing errors may result in reads being unmapable to the reference genome.
- Splice Junctions. If a read spans a splice junction, it will not map to the genome with any of the commodity alignment tools.
- RNA Editing. If post-transcriptional modifications occur more frequently than previously thought, they will easily confound commodity alignment tools.

Note that we have considered and excluded a fourth possibility: that reads occur within heavily repeated sections of the genome. We believe that this is unlikely to be the case as repeats are likely to be non-functional and therefore, heavily suppressed in expressed regions of the genome.

Example 2 : a “good” formulation

Motivation

The first goal of this project is to reject a null-model. That is, we desire to provide sufficient evidence to support further exploration of the subject matter. We do this by showing that our current understanding of high-throughput sequencing errors and RNA splice junctions is insufficient to explain the observed results (low percentage of mapped reads).

Sequencing Errors

We base our model of sequencing error loosely upon that demonstrated by Chaisson et. al. 2009. In this work, error in Illumina short reads is low ($< 2\%$) for the first 30nt of the read and increases rapidly thereafter. This can be seen in Figure 1. We simplify this, by considering a 2% error rate for all of the first 30nt and a 5% error rate for each of the remaining 6nt. In this model, which we believe to overstate the error, 93% of reads are expected to map to the genome with 2 mismatches or less. Clearly this is well above the 40% and 21% that mapped in the studies of interest.

Splice Junctions

Our model of splice junctions is based upon whole genome statistics produced by NCBI. According to these statistics the average exon length in the human genome is 200nt. Knowing that the read length provided by the sequencing platform is 36nt, we conclude that

$$\Pr(\text{random read contains a splice junction}) = \frac{36 \cdot (N - 1)}{200 \cdot N}$$

This is bounded by 18%. This means that more than 80% of reads should *not* map to splice junctions. We assume that sequencing error operates independently of splicing junctions. Therefore, if sequencing error and splice junctions exist in the expected proportions we expect that 70% of reads should map to the reference genome using up to 2 mismatches.

Example 2 : a “good” formulation

Input:

- A set of RNA reads $S = \{s_1, s_2, \dots, s_n\}$
Each s_i is a string matching $\{A, T, G, C\}^*$
- A reference genome G which is also a string matching $\{A, T, G, C\}^*$
- *gap* the penalty for a gap in the alignment
- *intron* the penalty for an intron length gap in the alignment
- *length* the minimum length of an intron
- *match* the premium for a match
- *mismatch* the penalty for a mismatch

Output: A set of alignments $T = \{t_1, t_2, \dots, t_n\}$

- Each t_i is a mapping from every position j in s_i to a position k in G or -1 indicating a gap.
- Each t_i is optimal according to the objective function O .
- $O(t_i)$ is defined as follows

$$O(t_i) = \sum_{k=0}^{|s_i|} \text{Score}(t_i, k) + \text{Gap}(t_i, k) \quad (1)$$

$$\text{Score}(t_i, k) = \begin{cases} \text{match} & \text{if } s_i(k) = G(t_i(k)) \\ \text{gap} & \text{if } t_i(k) = -1 \\ \text{mismatch} & \text{otherwise} \end{cases} \quad (2)$$

$$\text{Gap}(t_i, k) = \begin{cases} \text{intron} & \text{if } t_i(k) - t_i(k-1) > \text{length} \\ \text{gap} & \text{if } t_i(k) - t_i(k-1) > 1 \\ 0 & \text{otherwise} \end{cases} \quad (3)$$

Example 2 : a “good” formulation

Note that the problem formulation can be trivially modified to take into consideration consensus intron ‘donor’ and ‘acceptor’ sequences, though we do not cover it in the above description.

Algorithmics

Producing an optimal alignment is a well studied problem. The variant of the problem which takes into consideration intron gaps is also well studied. The dynamic programming recurrence presented by Mott in 1997 is shown in in Figure 2. It is well known that this problem can be solved in time $O(m \cdot n)$ where m, n are the lengths of the aligned strings. With k strings to align separately we have $O(k \cdot m \cdot n)$. For current high throughput sequencing platforms $k > 10^8$, meaning that any algorithm that operates in time multiplicative in the size of the genome is likely impossible. The dynamic programming solution to the alignment problem, while optimal, would require approximately $10^8 \cdot 10^9 \cdot 36$ computations. This is intractable. In order to address this, we propose an alternate problem formulation.

Example 2 : a “good” formulation

Fin

I will post these slides and
the example formulations
on the class website.

Questions?