The Soft Vertex Classification for Active Module Identification Problem

Nikita Alexeev[1], Javlon Isomurodov[1,2], Gennady Korotkevich[1], and Alexey Sergushichev[1,2]

[1]Computer Technologies Department, ITMO University, Saint-Petersburg, Russia
[2]JetBrains Research, Saint-Petersburg, Russia

September 3, 2018

Abstract

Motivation: Integrative network methods are commonly used for interpretation of high-throughput experimental biological data: transcriptomics, proteomics, metabolomics and others. One of the common approaches consists in finding a connected subnetwork of a global interaction network that best encompasses significant individual changes in the data and represents a so-called active module. Usually methods implementing this approach find a single subnetwork and thus solve a hard classification problem for vertices. This subnetwork inherently contains erroneous vertices, while no instrument is provided to estimate the confidence level of any particular vertex inclusion. To address this issue, in the current study we consider the active module problem as a soft classification problem. We propose a method to estimate probabilities of each vertex to belong to the active module based on Markov chain Monte Carlo subnetwork sampling.

Results: The proposed method allows to estimate the probability that an individual vertex belongs to the active module as well as the false discovery rate (FDR) for a given set of vertices. Given the estimated probabilities, it becomes possible to provide a connected subgraph in a consistent manner for any given FDR level: no vertex can disappear when the FDR level is relaxed. We show on simulated dataset that the proposed method has good computational performance and high classification accuracy. As an example of the performance of our method on real data, we run it on a protein-protein interaction network together with a gene expression DLBCL dataset. The results are consistent with the previous studies while, at the same time, the proposed approach is more flexible.

Source code is available at **https://github.com/ctlab/mcmcRanking** under MIT licence.

1 Introduction

Integrative network approaches are commonly used for interpretation of high-throughput data [13]. Such methods are applied in many different contexts: in genome-wide association studies [15], for elucidating mechanisms of metabolic regulation [10], for analysis of somatic mutations in cancer [12], etc. The main idea of these methods is that considering internal connections (for example, between proteins, metabolites or other entities) can lead to deeper understanding of the data and the corresponding biological processes.

The connectivity information could be used in multiple ways. The simplest analysis could involve manual exploring of connections between the input signals [11]. More sophisticated methods include using connections for gene set enrichment analysis [1], comparing networks [7] and many others.

One of the most well-developed and used approaches consists in selecting a connected subnetwork that best represents an *active* or *functional* module. This concept was initially suggested by Ideker et al [8]. The authors proposed a metric to score subnetworks based on gene expression data with a heuristic method to find top-scoring networks. Since then, multiple methods for solving this *active module identification problem* were developed. One of the most notable methods, called BioNet, was proposed by Dittrich *et al.* [4]. They suggested to use a maximum-likelihood-inspired subnetwork scoring scheme such that finding the best scoring subnetwork corresponds to solving the Maximum-Weight Connected Subgraph (MWCS) problem. While the problem is NP-hard, in the same paper a practical exact solver was proposed. Maximum-likelihood inspired formulation combined with an exact solver for the corresponding problem allowed to achieve great performance on both simulated and real data.

A question that is not usually addressed in the existing methods for solving the active module identification problem is the level of confidence of individual vertex inclusion. By design, in the resulting networks vertices with high individual significance are connected via less significant vertices. This raises the question whether vertices included in the module are more important compared to vertices with similar individual significance that are not included in the module. This is particularly important when an individually non-significant vertex is included in the module. Uncertainty in this aspect can lead to misinterpretation of the data either by attributing importance to false vertices or missing key vertices.

Previously Beisser et al [2] suggested a jackknife resampling approach where the active module problem is solved multiple times for resampled input data. It allows to introduce *support* values: how many times a particular vertex or edge was a part of the solution for the resampled data. The calculated support values can be then used to distinguish robust signals from noise in the resulting module. However, this method is limited to experiments with a large number of replicates and can not be applied for small-scale experiments.

The way different module confidence thresholds are handled is another problem of binary methods like BioNet. For example, a module of higher confidence could contain vertices not present in a module of lower confidence. Since in a real-case scenario several confidence thresholds are considered, such inconsistencies impede the interpretation of the results. To address this issue, previously [9] we considered a problem of connectivity-preserving vertex ranking – a ranking with the constraints that: 1) each prefix of the ranking should induce a connected subgraph, and 2) smaller induced subgraphs correspond to more confident modules. In that paper we proposed a semi-heuristic ranking method that was better (as measured by the area under the ROC curve, AUC ROC) compared to both baseline vertex ranking by individual input significance and ranking from multiple BioNet runs with different thresholds.

In the current study we consider the active module identification problem as a soft vertex classification problem. In this case, instead of providing a hard classification of vertices into either being in the module or not, we estimate probabilities of each vertex to belong to the active module. To estimate these probabilities we propose a method that is based on Markov chain Monte Carlo (MCMC) module sampling from a posteriori distribution. First, by producing the vertex probabilities, this approach directly resolves the question of individual vertices confidence. Moreover, we show that the estimated probabilities can be used to calculate expected AUC ROC of any ranking and thus the problem of finding the best connectivity-preserving ranking can be defined constructively. We prove that this problem is NP-complete, but show that in practice it can be solved accurately and efficiently using a heuristic algorithm. Finally, we show that the method can achieve high classification accuracy on simulated data and works well on real data.

2 Approach

While the active module problem appears in many different contexts, for the sake of clarity we focus here on the example of protein-protein interaction networks together with gene expression data. We formalize the protein-protein interaction network as a graph G, where vertices correspond to protein-encoding genes, and two vertices are linked by an edge if the corresponding proteins interact. Gene expression data are given for sets of samples for two biological conditions of interest (e.g. control and treatment), so that differential expression p-values can be calculated for each gene. While for some genes the null hypothesis of having zero expression change between conditions is true, and so the corresponding p-values are uniformly distributed on $[0, 1]$, p-values for "interesting" genes that exhibit a difference in expression would tend to be closer to zero. According to [8], those interesting genes form a connected subgraph in G, which is called an active (or functional) module.

2.1 Beta-uniform mixture model

As shown in [14, 4], the distribution of all p-values can be approximated by the so-called beta-uniform mixture (BUM) distribution, where the beta component corresponds to a signal in the data, and the uniform component corresponds to noise. Let us recall that the beta distribution has support $[0, 1]$ and is defined by its density

$$\beta(a, b)(x) = \frac{1}{B(a, b)} x^{a-1}(1 - x)^{b-1} ,$$

where $B(a, b)$ is the Beta function. The BUM distribution is a mixture of uniform and beta $\beta(a, 1)$ distributions and is defined by its density

$$\lambda + (1 - \lambda)ax^{a-1} ,$$

where λ is the mixture weight of the uniform component and a is the shape parameter of the beta distribution.

We assign a weight $w(v)$ to each vertex v of the graph G equal to the p-value assigned to the corresponding gene. Thus, in our model we have: a connected graph G on $n = |G|$ vertices, its connected subgraph M, a family of independent random variables W_v, $v \in V(G) \backslash V(M)$ with uniform distribution on $[0, 1]$, and a family of independent random variables W_v, $v \in V(M)$ with $\beta(a, 1)$ distribution.

2.2 MCMC approach

Our goal is to find out which vertices are likely to belong to the active module M:

Problem 1. *Given a connected graph G and vertex weights $w_v \in [0, 1]$ find the probability $P(v \in M \mid W = w)$ for each vertex v to belong to the module M.*

We solve this problem in the following way. We generate a large sample $\mathbb{S}$ of random subgraphs S from conditional distribution $P(S = M \mid W = w)$ using the Metropolis–Hastings algorithm (see Section 3). While all the probabilities $P(S = M \mid W = w)$ are small and the multiplicity of each subgraph in $\mathbb{S}$ is small, the probabilities of each vertex to belong to the module are cumulative statistics and show robust behavior. The same holds for the probabilities $P(V \subset M \mid W = w)$, where V are relatively small sets of vertices.

The benefits of the soft classification approach are:

1. it allows to estimate the level of confidence that a particular gene is expressed differently, which is the probability that a corresponding vertex belongs to the active module;

2. it allows to analyze alternative complement pathways by studying probabilities of the form $P((V_1 \subset M) \vee (V_2 \subset M) \mid W = w)$;

3. for any set of genes V reported as differently expressed, our method allows to compute the false discovery rate (FDR) as $\frac{1}{|V|} \sum_{v \in V} P(v \notin M \mid W = w)$;

4. it provides a vertex ranking that maximizes the area under the ROC curve (AUC ROC) (see Section 3.3);

5. it allows to heuristically find a vertex *connectivity preserving* ranking that maximizes the AUC ROC (see Section 3.4).

3 Methods

3.1 Markov chain Monte Carlo based method

We solve Problem 1 in the following way: we sample a set $\mathbb{S}$ of random subgraphs S from conditional distribution $P(S = M \mid W = w)$ using the Markov chain Monte Carlo (MCMC) approach, and estimate $P(v \in M \mid W = w)$ as

$$P(v \in M \mid W = w) \approx \frac{|\{S \in \mathbb{S} : v \in S\}|}{|\mathbb{S}|}.$$

First, we estimate the beta-uniform mixture parameters a and $\lambda = 1 - \frac{|M|}{|G|}$ with a maximum-likelihood estimator [3].

For MCMC sampling we implement the Metropolis-Hastings algorithm [6]. It starts at a random subgraph S_0 of order $k = |M| = (1 - \lambda)|G|$. On each step i we choose a candidate subgraph S' by removing a vertex v_- from S_i and adding another vertex v_+ from the neighborhood of S_i (by neighborhood of a subgraph S

$$\text{nei}(S) = \{v \in G : \min_{u \in S} dist(v,u) = 1\}$$

we mean the set of all vertices of G at distance 1 from S). We note that the subgraph S' always has the same number of vertices k as S_0. The proposal probability $Q(S'|S_i)$ is

$$Q(S'|S_i) = \frac{1}{|\,\text{nei}(S_i)| \cdot |S_i|}\,.$$

We set the acceptance probability in the Metropolis-Hastings algorithm as

$$\rho(S_i, S') = \min\left\{1, \frac{P(S' = M \mid W = w)}{P(S_i = M \mid W = w)} \frac{Q(S_i|S')}{Q(S'|S_i)}\right\}\,,$$

where $P(S' = M \mid W = w) = 0$ as soon as S' is not connected.

For any connected subgraph S of G the value $P(S' = M \mid W = w)$ can be expressed in terms of probability density p of an absolutely continuous random vector variable W:

$$P(S = M \mid W = w) = \frac{p(w \mid S = M)P(S = M)}{p(w)}\,. \tag{1}$$

So,

$$\frac{P(S' = M \mid W = w)}{P(S_i = M \mid W = w)} = \frac{p(w \mid S' = M)P(S' = M)}{p(w)} \frac{p(w)}{p(w \mid S_i = M)P(S_i = M)}\,.$$

The fraction $\frac{p(w|S'=M)}{p(w|S_i=M)}$ is equal to

$$\frac{p(w \mid S' = M)}{p(w \mid S_i = M)} = \frac{L(S')}{L(S_i)}\,, \tag{2}$$

where $L(S)$ is the likelihood of the subgraph S. In our case

$$L(S) = \prod_{v \in S} a w_v^{a-1}\,,$$

so, we have:

$$\frac{p(w \mid S' = M)}{p(w \mid S_i = M)} = \frac{\prod_{v \in S'} a w_v^{a-1}}{\prod_{u \in S_i} a w_u^{a-1}} = \frac{w_{v_+}^{a-1}}{w_{v_-}^{a-1}}\,. \tag{3}$$

Assuming that the prior distribution of choosing the module $P(S = M)$ is uniform[1] on the set of connected subgraphs S of the same order $|M|$, we get

[1] Since we do not make any additional assumptions about the module, the uniform distribution on subgraphs of equal order is the best choice of the prior distribution.

$$\rho(S_i, S') = \min\left\{1, \frac{w_{v_+}^{a-1}}{w_{v_-}^{a-1}} \frac{|\operatorname{nei}(S_i)|}{|\operatorname{nei}(S')|}\right\}.$$

The proposed MCMC algorithm can visit all possible subgraphs S, and so it converges to the distribution $P(S = M \mid W = w)$.

Algorithm 1: Metropolis-Hastings algorithm

```
**Input**: a connected graph $G$ and vertex weights $w_v \in \mathbb{R}_+$; $k$ – the
    number of vertices in the module $M$
**Output**: A random subgraph $S_N$ sampled from conditional distribution
    $P(S = M \mid W = w)$
$S_0 \leftarrow$ a random connected subgraph on $k = |M|$ vertices;
for $i = 0, 1, 2, \ldots, N$ do
    Choose $v_-$ from $V(S_i)$ and $v_+$ from $nei(S_i)$ uniformly;
    $S' \leftarrow$ induced subgraph on $V(S_i) \setminus \{v_-\} \cup \{v_+\}$;
    if $S'$ is connected then
        Acceptance Probability: $\rho(S_i, S') = \min\left\{1, \frac{w_{v_+}^{a-1}}{w_{v_-}^{a-1}} \frac{|\operatorname{nei}(S_i)|}{|\operatorname{nei}(S')|}\right\}$;
        $S_{i+1} \leftarrow \begin{cases} S' \text{ with probability } \rho(S_i, S') \\ S_i \text{ with probability } 1 - \rho(S_i, S') \end{cases}$;
    else
        $S_{i+1} \leftarrow S_i$
    end
end
return $S_N$
```

3.2 Heuristic approach to arbitrary module order

Since on the real data the estimated order of an active module has a tendency to be too large ($1 - \lambda$ is up to 0.5), we adjust our method in order to allow to change the number of vertices in the subgraph during the MCMC process. In this approach on each step of the process one can either add one vertex to the subgraph or remove one vertex from it. In order to provide subgraphs of biologically relevant size, we penalize each additional vertex by a factor $a\tau^{a-1}$, where τ is some confidence threshold as described in [4]. More formally, it means that we consider such a prior distribution on subgraphs that $\frac{P(S=M)}{P(S_+=M)} = a\tau^{a-1}$, where S_+ is an induced subgraph on the vertices $V(S) \cup \{v_+\}$. Thus our heuristic approach is very similar to Algorithm 1, but

$$\rho(S_i, S') = \min\left\{1, f(S_i, S')\right\},$$

where

$$f(S_i, S') = \begin{cases} \frac{w_{v_+}^{a-1}}{\tau^{a-1}} \frac{|\operatorname{nei}(S_i) \cup V(S_i)|}{|\operatorname{nei}(S') \cup V(S')|} & \text{if } V(S') = V(S_i) \cup \{v_+\}, \\ \\ \frac{\tau^{a-1}}{w_{v_-}^{a-1}} \frac{|\operatorname{nei}(S_i) \cup V(S_i)|}{|\operatorname{nei}(S') \cup V(S')|} & \text{if } V(S') = V(S_i) \setminus \{v_-\}. \end{cases}$$

Algorithm 2: Metropolis-Hastings algorithm for subgraphs with an arbitrary number of vertices

```
**Input**: a connected graph $G$ and vertex weights $w_v \in \mathbb{R}_+$; threshold $\tau$
**Output**: A random subgraph $S_N$ sampled from conditional distribution
        $P(S = M \mid W = w)$
$S_0 \leftarrow$ a random connected subgraph;
**for** $i = 0, 1, 2, \ldots, N$ **do**
    Choose $v$ from $V(S_i) \cup nei(S_i)$ uniformly;
    **if** $v \in V(S_i)$ **then**
        $v_- \leftarrow v$;
        $S' \leftarrow$ induced subgraph on $V(S_i) \setminus \{v_-\}$;
        $\rho(S_i, S') = \min\left\{1, \frac{\tau^{a-1}}{w_{v_-}^{a-1}} \frac{|\operatorname{nei}(S_i) \cup V(S_i)|}{|\operatorname{nei}(S') \cup V(S')|}\right\}$;
    **else**
        $v_+ \leftarrow v$;
        $S' \leftarrow$ induced subgraph on $V(S_i) \cup \{v_+\}$;
        $\rho(S_i, S') = \min\left\{1, \frac{w_{v_+}^{a-1}}{\tau^{a-1}} \frac{|\operatorname{nei}(S_i) \cup V(S_i)|}{|\operatorname{nei}(S') \cup V(S')|}\right\}$;
    **end**
    **if** $S'$ *is connected* **then**
        $S_{i+1} \leftarrow \begin{cases} S' \text{ with probability } \rho(S_i, S') \\ S_i \text{ with probability } 1 - \rho(S_i, S') \end{cases}$;
    **else**
        $S_{i+1} \leftarrow S_i$
    **end**
**end**
**return** $S_N$
```

For estimating probabilities $P(v \in M \mid W = w)$ we need to choose a set of subgraph samples $\mathbb{S}$. Here we consider two ways of doing this. For both ways we need to estimate the mixing time T of the algorithm: the number of Markov chain iterations such that the distribution of S_T approximates the target distribution well. We note that theoretically the Markov chain converges to the desired distribution since it is ergodic: it is recurrent (since the number of states is finite), it is aperiodic (since there is a positive probability that the process stays at the same state), and it is irreducible since by construction it can reach any subgraph from any other subgraph. In practice, the mixing time depends on multiple parameters including the graph G order. The first way to choose $\mathbb{S}$ is to do a number of independent MCMC runs of T iterations and add each S_T to $\mathbb{S}$. Here, all samples in $\mathbb{S}$ are independent, which can be used to calculate the accuracy of the vertex probabilities estimation. Another way consists in doing one long run of MCMC and putting all S_i for $i > T$ into $\mathbb{S}$. Although consecutive samples are not independent, the probabilities that are estimated this way converge to the true probabilities given sufficiently long series (see Section 4.1 for discussion of the converging time on the simulated data).

3.3 Features of soft classification solution

Having the set $\mathbb{S}$ of subgraphs sampled from the conditional distribution $P(S = M \mid W = w)$, one can easily answer the following questions. First, one can estimate for each vertex v_i the probability p_i that it belongs to the active module. In the case when the module size is not fixed, we can also estimate the conditional probabilities

$$p_i^{(m)} = P\big(v_i \in M \bigm| W = w, |M| = m\big)\,.$$

Second, for any boolean-valued function $A(M)$ (for example, $A(M) = (v_1 \in M) \vee (v_2 \in M) \wedge \neg(v_3 \in M)$) one can estimate the probability $P(A(M) \mid W = w)$.

Note, that if each S from $\mathbb{S}$ was generated with an independent MCMC run, then each $A(S)$ can be considered to be a result of a Bernoulli trial. Thus, all such probabilities can be estimated with a mean square error of the order $\frac{p(1-p)}{|\mathbb{S}|}$, where $p = P(A(M)|W = w)$.

We also show how the probabilities of each vertex to belong to the active module are related to the expected AUC ROC for some vertex ranking. For a vertex ranking $v_1, v_2, \ldots, v_n$, where $n = |G|$, and a module M of order m the ROC curve is a step-curve in a square $[0,1] \times [0,1]$, which starts at $(0,0)$, and on the i-th step it goes either up $\left(\frac{1}{m}, 0\right)$ (if $v_i \in M$) or right $\left(0, \frac{1}{n-m}\right)$ (if $v_i \notin M$). The AUC ROC shows how accurate the classification is.

We prove the following lemma:

Lemma 1. *Let n be the number of vertices in G, m be the number of vertices in the module M, $v_1, v_2, \ldots, v_n$ be some vertex ranking and p_i be the probabilities $p_i = P(v_i \in M \mid W = w)$. Then the expected value of the AUC ROC is equal to*

$$1 - \frac{1}{m(n-m)} \left(\sum_{i=1}^{n} i p_i - \frac{m(m+1)}{2} \right)\,.$$

Proof. See Section S1 in Supplementary Materials. □

Lemma 3 implies that the vertex ranking with the largest expected value of the AUC ROC is the ranking according to a descending order on p_i (for any particular module order m). If the number m of the vertices in the module is not known, the expected AUC ROC is equal to

$$1 - \frac{1}{\mathbb{E}\,|M| \cdot |G \setminus M|} \left(\sum_{i=1}^{n} i p'_i - \frac{\mathbb{E}\,|M|(|M|+1)}{2} \right)\,,$$

where

$$p'_i = \mathbb{E}\,\left(|M| \cdot |G \setminus M|\right) \cdot \sum_{m=1}^{n} P\big(|M| = m \bigm| W = w\big) \frac{p_i^{(m)}}{m(n-m)}\,,$$

and $\mathbb{E}$ stands for the expected value.

We note that the probabilities p'_i can be estimated based on the sample $\mathbb{S}$ or approximated by p_i.

3.4 Connectivity preserving ranking

While we provide the best solution for the soft classification problem in terms of the expected AUC ROC, one can be interested in getting the solution for the hard classification problem as well. Our method is easily transformable into a hard classification method. Namely, our goal is to define a module $M(q)$ for any FDR level q in a consistent manner. That is, we don't allow the situation when a vertex is included in a module with a small FDR level, but excluded from a module with a larger FDR (formally speaking, we demand that $M(q_1) \subset M(q_2)$ as soon as $q_1 < q_2$). Any such family of modules $M(q)$ corresponds to a connectivity preserving ranking.

Definition 1. *For a graph G with vertices v_i the* connectivity preserving *ranking is such a ranking of v_i that for any k-prefix $v_1, v_2, \dots, v_k$ the induced graph on this set of vertices is connected.*

We want to choose the best connectivity preserving ranking in terms of the expected value of the AUC ROC. Since the expected AUC ROC depends only on p_i (or their modified versions p'_i), we can formulate the following problem:

Problem 2 (Optimal Connectivity Preserving Ranking, OCPR)**.** *Given a graph G and the list of its vertices v_i, each equipped with the probability p_i, find the* connectivity preserving *ranking maximizing the expected AUC ROC.*

Lemma 2. *OCPR problem is NP-hard.*

Proof. See Section S2 in Supplementary Materials. □

As Problem 2 is NP-hard, here we provide a heuristic algorithm to solve it. On the k-th step of the algorithm we have an integer number r_k and G_k – a connected subgraph of G (where $r_1 = n$ and $G_1 = G$), and we define a connected subgraph G_{k+1} in the following way. For each vertex v we define a subgraph $G_{k+1}^{(v)}$ as the largest connected component of $G_k \setminus \{v\}$ and $H_{k+1}^{(v)}$ as $G_k \setminus G_{k+1}^{(v)}$. Then we choose such a vertex v that maximizes average FDR in $H = H_{k+1}^{(v)}$, which is equal to $\frac{1}{|H|}\sum_{u \in H} P(u \notin M \mid W = w)$. Then we assign to all vertices in the subgraph H the rank r_k and assign to $r_{k+1} = r_k - |H|$.

Each step requires time $O(n^2)$, so the performance time is $O(n^3)$. On our real data example, the running time was 13 seconds on Intel(R) Core(TM) i5-7200U CPU @ 2.50GHz (implemented in C++).

3.5 Baseline ranking methods

We compared our approach with three other ranking methods on simulated data in Section 4.1. The first method ranks vertices by the ascending order of their input weights w_v. In the second method vertices are ranked by the descending order of the number of occurrences in the modules as found by running the BioNet method with 20 different significance thresholds. The thresholds are selected to be distributed at equal intervals between maximum and minimum vertex log-likelihoods. The third method is a semi-heuristic ranking method from [9] developed for finding a good connectivity preserving ranking. To describe it briefly, it first finds the maximum-likelihood connected subgraph and recursively ranks a set of vertices inside and outside of the subgraph. The recursion step also involves finding a maximum-likelihood connected subgraph but with constraint on consistent connectivity with already defined ranking.

Note that we do not compare our method with the jackknife resampling method [2] on simulated data as it requires simulating gene expression data, which can induce artificial biases in the evaluation. Instead, we compare our results with the results of this method on real data (see Section 4.2).

4 Discussion

4.1 Experiments on simulated data

First, we consider a toy example. We generated a random graph G on 30 vertices and 65 edges. Then we chose the active module M on 9 vertices randomly and generated the weights from the beta distribution $\beta(0.2, 1)$ for the vertices in M and from the uniform distribution for all other vertices. For such a toy example we can both compute the probabilities $P(v \in M|W = w)$ directly from (1) and estimate them as $\hat{P}(v \in M|W = w)$ using our MCMC approach. We run 10^6 iterations of the Metropolis-Hastings algorithm. The estimations approximated the actual probabilities very accurately with root-mean-square error equal to $2 \cdot 10^{-3}$. We also note that the ranking of vertices based on estimations $\hat{P}(v \in M|W = w)$ was the same as the ranking based on true probabilities. For both actual and estimated probabilities the AUC ROC is equal to 0.92.

Second, we compared our MCMC-based connectivity preserving ranking method with three other ranking methods (see Section 3.5) on simulated data. Here we considered 50 random instances. For each instance a random scale-free graph on 100 vertices was generated. Then we generated active modules in two steps: 1) the number of the vertices in a module was uniformly selected from 5 to 25 and 2) a module was chosen uniformly at random from all connected subgraphs with the selected order. Vertex weights were generated from the beta-uniform mixture distribution with values of the a parameter selected from the $[0.01, 0.5]$ interval. The AUC measures for rankings produced by four tested methods are shown on Fig. 2. For all values of a our approach shows significantly better performance compared to all three baseline methods.

Next we considered MCMC performance on a real-world protein-protein interaction graph. We used a graph with 2,034 vertices and 8,399 edges as constructed in [4] for a diffuse large B-cell lymphoma dataset. We chose an active module uniformly at random from connected subgraphs on 200 vertices. Weights of the vertices in the active module were generated from the beta distribution $\beta(0.25, 1)$.

First, we considered the behavior of log-likelihood values for samples during one MCMC run (Fig. 1, green line). This plot shows that the log-likelihood value stabilizes after about 25,000 iterations. Thus, we can estimate the MCMC mixing time T as 25,000 for this case. Then we checked that 25,000 iterations is sufficient for estimating vertex probabilities. For different values of T' we calculated AUC values for rankings based on 1,000 independent runs of MCMC for T' iterations (Fig. 1, red line). The results show that indeed 25,000 iterations is enough to achieve high AUC values, and the saturation phase begins

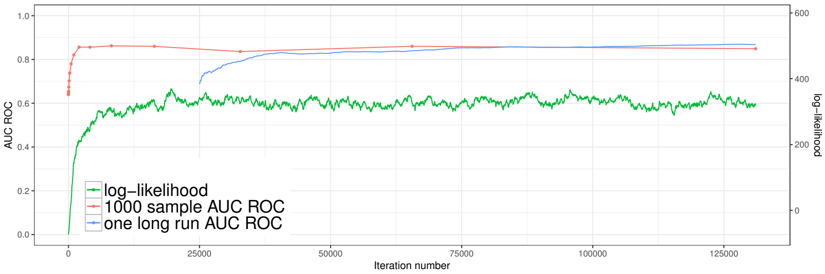

Figure 1: Behavior of subgraph log-likelihood values and ranking AUCs depending on the number of MCMC iterations. Real protein-protein interaction graph of 2,034 vertices is used as G, module of 200 vertices is chosen uniformly at random. Green line: log-likelihood values for subgraphs S_i generated during one MCMC run. Red line: AUC values for rankings based on 1,000 independent MCMC samples depending on the chosen mixing time estimate. Blue line: AUC values for rankings based on one MCMC run calculated on all samples S_i for $i >$ 25,000.

even earlier. Finally, we compared ranking for one long MCMC run and for 1,000 independent samples (Fig. 1, blue line). For one long run we estimated probabilities using all generated MCMC samples except the first 25,000. It can be seen that AUC values saturate after about 50,000 total MCMC iterations which justifies the usage of one long MCMC run. Practically, this means that good probability estimates can be achieved very fast, given that 100,000 MCMC iterations took about one minute on a laptop.

4.2 Experimental results on real data

We apply our method to the diffuse large B-cell lymphoma dataset and the protein-protein interaction graph constructed in [4]. The p-values in the DLBCL dataset are the result of a differential expression t-test between the two tumor subgroups: germinal center B-cell-like (GCB) DLBCL and activated B-cell-like (ABC) DLBCL. The estimated BUM distribution parameters are $\lambda = 0.48$ and $a = 0.18$. As described in Section 3.2, we penalized the addition of a new vertex to the module using the confidence threshold $\tau = 10^{-7}$.

The obtained module is shown on Fig. 3. The module shows the prefix of the ranking with FDR = 0.25. Additionally, we highlighted submodules of more strict FDR values: 0.15 and 0.05. Note that, by construction, a more strict module is a subgraph of a less strict one. The genes in the modules are mostly known to be associated with cancer. For example, genes BCL2, BMF, CASP3, PTK2 and WEE1 are involved in the cell apoptosis pathway. Some other genes, like LYN or KCNA3, are associated with proliferation and signalling in hematopoietic cells.

Additionally, we compared our result to one obtained by the jackknife resampling procedure as described in [2] with 100 resamples and the same threshold of $\tau = 10^{-7}$. Resampling support values are consistent with MCMC-based probabilities (see Supplementary material). However, compared to the resampling method [2] our approach has two major advantages. First, out method starts

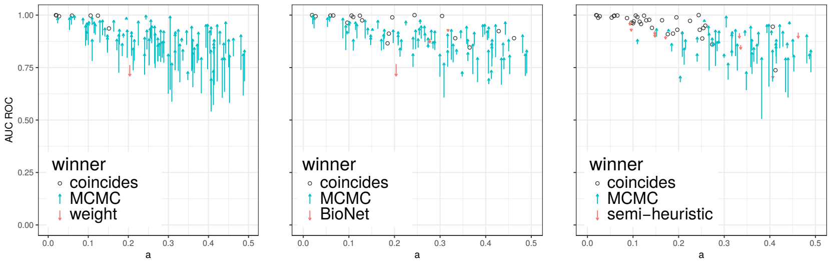

Figure 2: Ranking AUC values for simulated instances for graphs on 100 vertices. The proposed MCMC method is compared with the following methods: ranking by input weights (left), BioNet-based ranking (middle) and semi-heuristic ranking from [9] (right). One arrow corresponds to one experiment. The head of each arrow points to the AUC score of the MCMC method, while the tail indicates the AUC score of the corresponding alternative method. Thus, upward arrows indicate instances in which the MCMC method has a higher AUC score. Dots indicate the instances for which the results are equal. Color depends on which method has better AUC.

with p-values and can be used even for experiments with a small number of replicates, where resampling is not feasible. Second, calculating individual vertex probabilities does not involve solving NP-hard problems and can be easily done in practice without dependency on external solver libraries like IBM ILOG CPLEX. Even the NP-hard problem of finding the AUC ROC optimal connectivity preserving ranking can be solved well on real data with the heuristic algorithm.

5 Conclusion

While the active module identification problem was intensively studied in recent years, most of the approaches solve it as a hard classification problem. Here we study the question of assigning confidence values for individual vertices and thus consider a soft classification problem. All the hard classification methods could only provide a blackbox which tells us whether a particular gene belongs to the active module or not, and so, by design, they all have a flaw: they don't provide any way to distinguish between more and less confident inclusions. The soft classification approach addresses this issue.

We propose a method to estimate probabilities of each vertex to belong to the active module based on Markov chain Monte Carlo sampling. Based on these probabilities, for any given FDR level we can provide a solution to the hard classification problem in a consistent manner: a module for a more strict FDR level is a subgraph of any module for a more relaxed FDR level. Overall, the proposed approach is very flexible: it starts with p-values, and thus can be used in many different contexts, and does not depend on external libraries for solving NP-hard problems.

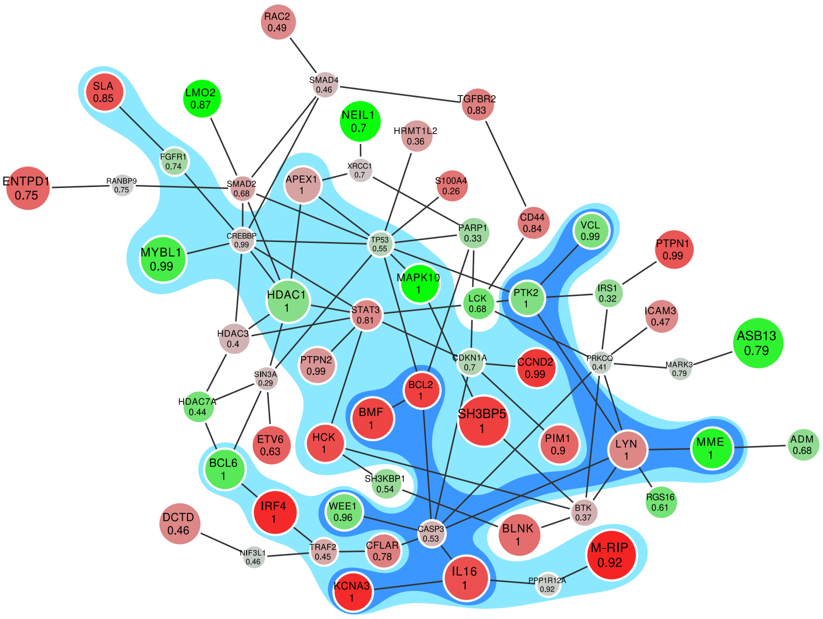

Figure 3: The module for comparison of GCB and ABC types of diffuse large B-cell lymphoma (red vertices are up-regulated in ABC type, green ones are up-regulated in GCB). The vertices of the blue subgraph belong to the active module with very high confidence (FDR is 0.05). The vertices of the light blue subgraph belong to the active module very likely (FDR is 0.15). The number in each vertex means the frequency of the vertex presence in a sampled subgraph.

Acknowledgements

The authors thank Artem Vasilyev for fruitful discussions.

Funding

The work of Javlon Isomurodov and Alexey Sergushichev was supported by the Ministry of Education and Science of the Russian Federation (agreement 2.3300.2017). The work of Nikita Alexeev was financially supported by the Government of the Russian Federation through the ITMO Fellowship and Professorship Program.

Supplementary Materials

S1

In this section we prove Lemma 3.

Lemma 3. *Let n be the number of vertices in G, m be the number of vertices in the module M, $v_1, v_2, \ldots, v_n$ be some vertex ranking and p_i be the probabilities $p_i = P(v_i \in M \mid W = w)$. Then the expected value of the AUC ROC is equal to*

$$1 - \frac{1}{m(n-m)}\left(\sum_{i=1}^{n} ip_i - \frac{m(m+1)}{2}\right).$$

Proof. First of all, we introduce a rotated not-normalized ROC curve (Fig.4). On the i-th step this curve goes either up $(\frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2})$ (if $v_i \in M$) or down $(-\frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2})$ (if $v_i \notin M$).

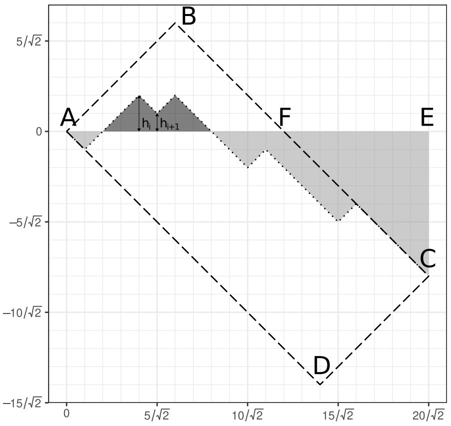

Figure 4: Rotated ROC curve

This linear transformation of the ROC curve is a function, so one can find not only the expected value of the area under it, but its pointwise expected value as well. Let its height at the point $\frac{i\sqrt{2}}{2}$ be h_i. Then the expected value of

$h_{i+1} - h_i$ is $\frac{2p_i-1}{\sqrt{2}}$. The area under the (original) ROC curve (up to the factor $m(n-m)$) is equal to the area under the rotated ROC curve plus the area of the quadrilateral $AECD$. The area of $AECD$ is equal to the area of $ABCD$ (which is equal to $m(n-m)$) minus the area of ABF (which is equal to $\frac{m^2}{2}$) plus the area of CEF (which is equal to $\frac{(n-2m)^2}{4}$). Thus, we obtian:

$$\begin{aligned}
& m(n-m) - \frac{m^2}{2} + \frac{(n-2m)^2}{4} + \sum_{i=1}^{n} \frac{h_{i-1}+h_i}{2}\frac{1}{\sqrt{2}} = \\
&= \frac{n^2}{4} - \frac{m^2}{2} + \sum_{i=1}^{n} \frac{h_i}{\sqrt{2}} + \frac{h_0 - h_n}{2\sqrt{2}} = \\
&= \frac{n^2}{4} - \frac{m^2}{2} + \sum_{i=1}^{n} \frac{(n+1-i)\frac{2p_i-1}{\sqrt{2}}}{\sqrt{2}} - \frac{\sum_{i=1}^{n}\frac{2p_i-1}{\sqrt{2}}}{2\sqrt{2}} = \\
&= \frac{n^2}{4} - \frac{m^2}{2} + \sum_{i=1}^{n} \frac{(n+1-i)(2p_i-1)}{2} - \frac{\sum_{i=1}^{n} 2p_i - 1}{4} = \\
&= \frac{n^2}{4} - \frac{m^2}{2} + \sum_{i=1}^{n} \left((n+1)p_i - \frac{n+1-i}{2} - ip_i \right) - \frac{2m-n}{4} = \\
&= \frac{n^2}{4} - \frac{m^2}{2} + (n+1)m - \frac{n(n+1)}{4} - \sum_{i=1}^{n} ip_i - \frac{2m-n}{4} = \\
&= \frac{n^2}{4} - \frac{m^2}{2} + nm + m - \frac{n^2}{4} - \frac{n}{4} - \sum_{i=1}^{n} ip_i - \frac{m}{2} + \frac{n}{4} = \\
&= -\frac{m^2}{2} + nm + \frac{m}{2} - \sum_{i=1}^{n} ip_i = \\
&= m(n-m) + \frac{m(m+1)}{2} - \sum_{i=1}^{n} ip_i.
\end{aligned}$$

After normalization we obtain

$$1 - \frac{1}{m(n-m)}\left(\sum_{i=1}^{n} ip_i - \frac{m(m+1)}{2}\right).$$

□

S2

In this section we prove Lemma 4.

Lemma 4. *The OCPR problem is NP-hard.*

Proof. To prove that OCPR is NP-complete, we will use a modification of the method for proving that the Steiner tree problem in graphs is NP-complete. The referenced method is described in [16].

In decisional form, OCPR is equivalent to:

OCPR (decisional). *Given a graph G, the list of its vertices v_i, each equipped with the probability p_i, and a real number k, determine if there is a* connectivity preserving *ranking $v_1, v_2, \dots, v_n$ such that $\sum_{i=1}^{n} ip_i \leq k$.*

First, we need to show that OCPR is in NP. Given an hypothetic positive solution $v_1, v_2, \dots, v_n$, it is trivial to check in polynomial time that this ranking is *connectivity preserving* and $\sum_{i=1}^{n} ip_i \leq k$ holds.

The Exact Cover by 3-Sets problem (X3C) is a well-known NP-complete problem mentioned in [5]:

Problem X3C. *Given a finite set X with $|X| = 3q$ and a collection C of 3-element subsets of X ($C = \{C_1, \dots, C_n\}$, $C_i \subseteq X$, $|C_i| = 3$ for $1 \leq i \leq n$), determine if C contains an* exact cover *for X, that is, a subcollection $C' \subseteq C$ such that every element of X occurs in exactly one member of C'?*

Let us propose a reduction from X3C to OCPR giving a set of rules to build an instance of OCPR starting from a generic instance of X3C. If we prove that this transformation is executable in polynomial time, we will also prove that OCPR is NP-complete.

Given an instance of X3C, defined by the set $X = \{x_1, \dots, x_{3q}\}$ and a collection of 3-element sets $C = \{C_1, \dots, C_n\}$, we have to build an OCPR instance specifying the graph $G = (V, E)$, the probabilities p_i, and the upper bound on the required sum k.

The vertices of G are defined as:

$$V(G) = \{v\} \cup \{c_1, \dots, c_n\} \cup \{x_1, \dots, x_{3q}\}.$$

That is, we add a new node v, a node for each member of C, and a node for each element of X.

The edges of G are defined as:

$$E(G) = \{vc_1, \dots, vc_n\} \cup \left(\bigcup_{x_j \in C_i} \{c_i x_j\} \right).$$

That is, there is an edge from v to each node c_i, and an edge $c_i x_j$ if the element x_j belongs to the set C_i of the X3C instance.

Let $\varepsilon = \frac{1}{12q(q+1)}$. The probabilities p_i are defined as:

$$p_v = 1, \quad p_{c_i} = 0, \quad p_{x_j} = \varepsilon,$$

and the upper bound on the required sum is defined as $k = \frac{3}{2}$.

The reduction from X3C to OCPR is easy to do in polynomial time.

Proposition 1. *There exists a connectivity preserving ranking with $\sum_{i=1}^{n} ip_i \leq k$ if and only if there is an exact cover in the corresponding instance of X3C.*

Proof. We split the proof into two parts, one for each implication.

- X3C ⇒ OCPR

  Suppose there is an exact cover C' for the X3C problem. Clearly, C' uses exactly q subsets. Without loss of generality suppose they are $C_1, \dots, C_q$ (if it is not the case, we just have to relabel them). Then a valid connectivity preserving ranking looks as follows:

$$\begin{gathered} v, \\ c_1, x_{t_1}, x_{t_2}, x_{t_3}, \\ c_2, x_{t_4}, x_{t_5}, x_{t_6}, \\ \ldots, \\ c_q, x_{t_{3q-2}}, x_{t_{3q-1}}, x_{t_{3q}}, \\ c_{q+1}, c_{q+2}, \ldots, c_n, \end{gathered}$$

where every C_i consists of elements $x_{t_{3i-2}}, x_{t_{3i-1}}, x_{t_{3i}}$. Then:

$$\sum_{i=1}^{n} ip_i = 1 + (3+4+5+7+8+9+\ldots+(4q-1)+4q+(4q+1))\varepsilon =$$
$$= 1 + 6q(q+1)\varepsilon = \tfrac{3}{2} = k.$$

- X3C $\Leftarrow$ OCPR

  Suppose there is a connectivity preserving ranking with $\sum_{i=1}^{n} ip_i \leq k$. Since $k < 2$ and $p_v = 1$, the first vertex in this ranking must be v. Let $x_{t_1}, x_{t_2}, \ldots, x_{t_{3q}}$ be the order the vertices corresponding to the X3C instance elements appear in this ranking. For every j, vertex v and at least $\lceil \frac{j}{3} \rceil$ vertices corresponding to 3-element sets of the X3C instance must appear before vertex x_{t_j} in the ranking, since every vertex c_i appearing in the ranking opens the way for at most three new vertices x_j to appear afterwards. Hence, the earliest position vertex x_{t_j} might appear at in the ranking is $j + 1 + \lceil \frac{j}{3} \rceil$. But if every x_{t_j} does appear at its earliest possible position, then, like in the former case,

$$\sum_{i=1}^{n} ip_i = 1 + (3+4+5+7+8+9+\ldots+(4q-1)+4q+(4q+1))\varepsilon =$$
$$= 1 + 6q(q+1)\varepsilon = \tfrac{3}{2} = k.$$

  If any vertex x_{t_j} appears at a later position, its index in the ranking will increase, the sum of ip_i will increase and exceed k (since the sum of ip_i is already equal to k and $p_{x_{t_j}} > 0$). Therefore, every vertex x_{t_j} must appear at position $j + 1 + \lceil \frac{j}{3} \rceil$, and the ranking looks like in the former case, too:

$$\begin{gathered} v, \\ c_1, x_{t_1}, x_{t_2}, x_{t_3}, \\ c_2, x_{t_4}, x_{t_5}, x_{t_6}, \\ \ldots, \\ c_q, x_{t_{3q-2}}, x_{t_{3q-1}}, x_{t_{3q}}, \\ c_{q+1}, c_{q+2}, \ldots, c_n. \end{gathered}$$

  It's easy to prove that $x_{t_{3i-2}}$, $x_{t_{3i-1}}$ and $x_{t_{3i}}$ belong to C_i using induction on i from 1 to q. Hence, $C' = \{C_1, C_2, \ldots, C_q\}$ is an exact cover for the X3C instance.

This concludes the proof. □

S3

In this section we compare our method and the jackknife resampling method described in [3]. We apply our method and the jackknife resampling method to the diffuse large B-cell lymphoma dataset and the protein-protein interaction graph constructed in [4]. The p-values in the DLBCL dataset are the result of a differential expression t-test between the two tumor subgroups: germinal center B-cell-like (GCB) DLBCL and activated B-cell-like (ABC) DLBCL. In our method we penalized adding a new vertex to the module using the confidence threshold $\tau = 10^{-7}$. For the jackknife method the p-values are recalculated for each of 100 data resamples and then for each collection of the p-values the active module identification problem is solved with BioNet (with the same threshold $\tau = 10^{-7}$). As the result of this procedure, for each vertex the support value (the occurrence frequency of the vertex in the solution) is computed. As one can see (Fig. 5), the results of these two methods are consistent.

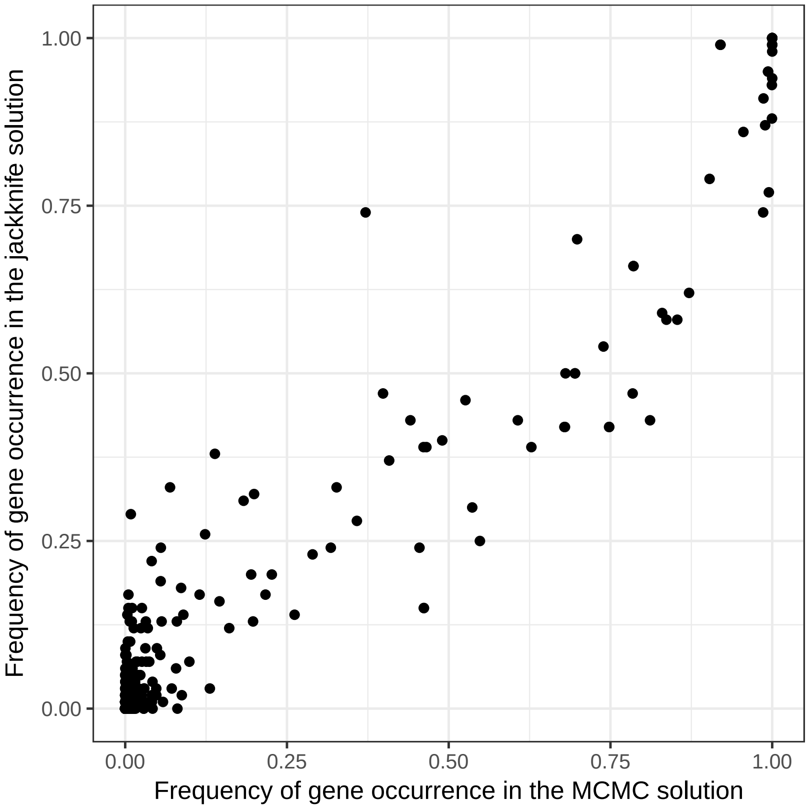

Figure 5: The support values and the probabilities of each vertex to belong to the active module are highly correlated.

References

[1] A. Alexeyenko, W. Lee, M. Pernemalm, J. Guegan, P. Dessen, V. Lazar, J. Lehtio, and Y. Pawitan. Network enrichment analysis: extension of gene-set enrichment analysis to gene networks. *BMC Bioinformatics*, 13:226, Sep 2012.

[2] D. Beisser, S. Brunkhorst, T. Dandekar, G. W. Klau, M. T. Dittrich, and T. Muller. Robustness and accuracy of functional modules in integrated network analysis. *Bioinformatics*, 28(14):1887–1894, Jul 2012.

[3] D. Beisser, G. W. Klau, T. Dandekar, T. Muller, and M. T. Dittrich. BioNet: an R-Package for the functional analysis of biological networks. *Bioinformatics*, 26(8):1129–1130, Apr 2010.

[4] Marcus T Dittrich, Gunnar W Klau, Andreas Rosenwald, Thomas Dandekar, and Tobias Müller. Identifying functional modules in protein-protein interaction networks: an integrated exact approach. *Bioinformatics (Oxford, England)*, 24(13):i223–31, 2008.

[5] Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W.H. Freeman and Company, New York, 1979.

[6] W Keith Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1):97–109, 1970.

[7] T. Ideker and N. J. Krogan. Differential network biology. *Mol. Syst. Biol.*, 8:565, Jan 2012.

[8] Trey Ideker, Owen Ozier, Benno Schwikowski, and Andrew F Siegel. Discovering regulatory and signalling circuits in molecular interaction networks. *Bioinformatics (Oxford, England)*, 18 Suppl 1:S233–S240, 2002.

[9] Javlon E Isomurodov, Alexander A Loboda, and Alexey A Sergushichev. Ranking vertices for active module recovery problem. In *International Conference on Algorithms for Computational Biology*, pp. 75–84. Springer, 2017.

[10] A. K. Jha, S. C. Huang, A. Sergushichev, V. Lampropoulou, Y. Ivanova, E. Loginicheva, K. Chmielewski, K. M. Stewart, J. Ashall, B. Everts, E. J. Pearce, E. M. Driggers, and M. N. Artyomov. Network integration of parallel metabolic and transcriptional data reveals metabolic modules that regulate macrophage polarization. *Immunity*, 42(3):419–430, 2015.

[11] A. Karnovsky, T. Weymouth, T. Hull, V. G. Tarcea, G. Scardoni, C. Laudanna, M. A. Sartor, K. A. Stringer, H. V. Jagadish, C. Burant, B. Athey, and G. S. Omenn. Metscape 2 bioinformatics tool for the analysis and visualization of metabolomics and gene expression data. *Bioinformatics*, 28(3):373–380, Feb 2012.

[12] M.D. Leiserson and others. Pan-cancer network analysis identifies combinations of rare somatic mutations across pathways and protein complexes. *Nat. Genet.*, 47(2):106–114, Feb 2015.

[13] K. Mitra, A. R. Carvunis, S. K. Ramesh, and T. Ideker. Integrative approaches for finding modular structure in biological networks. *Nat. Rev. Genet.*, 14(10):719–732, Oct 2013.

[14] S. Pounds and S. W. Morris. Estimating the occurrence of false positives and false negatives in microarray studies by approximating and partitioning the empirical distribution of p-values. *Bioinformatics*, 19(10):1236–1242, Jul 2003.

[15] E. J. Rossin and others. Proteins encoded in genomic regions associated with immune-mediated disease physically interact and suggest underlying biology. *PLoS Genet.*, 7(1):e1001273, Jan 2011.

[16] Alessandro Santuari. Steiner tree np-completeness proof. Technical report, University of Trento, 2003.