Discrete optimization - Project
Module analysis in cancer diagnosis

Maxime Meurisse (s161278)
François Rozet (s161024)

May 2, 2021

Notations

Let $S = (s_{ij}) \in \mathbb{R}^{n \times n}$ be the pairwise co-expression matrix of n genes in a disease. By definition, $s_{ij} = s_{ji}$ and we impose $s_{ii} = \infty$ for all $i, j = 1, 2, \dots, n$. Assuming an arbitrary threshold $\tau \in \mathbb{R}$, we define the *undirected graph* $G = (V, E)$, where $V = \{1, 2, \dots, n\}$ and

$$E = \{(i, j) : s_{ij} \geq \tau \mid i, j \in V\} \tag{1}$$

are respectively the vertex and edge sets of G. We denote $A_G = (a_{ij})$ the *adjacency matrix* of G, where

$$a_{ij} = \begin{cases} 1 & \text{if } (i, j) \in E \\ 0 & \text{else} \end{cases} \tag{2}$$

for $i, j \in V$. Finally, we denote

$$k_i = \sum_{j=1}^{n} a_{ij} \tag{3}$$

the *degree* of the vertex $i \in V$, *i.e.* the number of vertices adjacent to i in G. It should be noted that all self-loops $(i, i) \in E$ and, therefore, $a_{ii} = 1$ for all $i \in V$.

Module

A module M, or a *clique* [1, 2], is a subset of V such that all vertices are *pairwise adjacent*, *i.e.* $(i, j) \in E$ for all $i, j \in M$. This is equivalently expressed as $M \times M = M^2 \subseteq E$ or $M^2 \setminus E = \varnothing$. The latter is especially interesting as $M^2 \setminus E$ can be computed efficiently if G is sparse. We denote

$$\delta(M) = \left| M^2 \setminus E \right| = \left| \left\{ (i, j) : a_{ij} = 0 \mid (i, j) \in M^2 \right\} \right|, \tag{4}$$

the number of missing edges in E for M to be a *proper* module.

In the following, a module M is sometimes represented as a binary vector $x = (x_i) \in \{0, 1\}^n$ where $x_i = 1$ if $i \in M$. This notation leads

$$|M| = \sum_{i \in V} x_i = \sum_{i \in V} |x_i| = \|x\|_1 \tag{5}$$

and, from (4),

$$\delta(M) = \frac{1}{2} \sum_{i \in V} \sum_{j \in V} x_i (1 - a_{ij}) x_j. \tag{6}$$

Questions

1. Finding the largest module comes down to maximizing the size $|M|$ while keeping $\delta(M) = 0$. This can be expressed as the following (mixed-)integer optimization problem.

$$\begin{aligned} \max_{x} \quad & \|x\|_1 \\ \text{s.t.} \quad & \sum_{i \in V} \sum_{j \in V} x_i (1 - a_{ij}) x_j = 0 \\ & x \in \{0, 1\}^n \end{aligned}$$

Unfortunately, this formulation is quadratically constrained and, therefore, cannot be solved using (linear) mixed-integer programming (MIP). However, we observe that each term of the sum is non-negative, implying that either $1 - a_{ij}$ or $x_i x_j$ has to be null. Hence, the previous formulation is equivalent to

$$\begin{aligned} \max_{x} \quad & \|x\|_1 \\ \text{s.t.} \quad & (1 - a_{ij})(x_i + x_j - 1) \leq 0, \quad \forall i, j \in V \\ & x \in \{0, 1\}^n \end{aligned}$$

which is a linearly constrained formulation for the largest module problem.

In addition, we define the Best-In and Worst-Out heuristics (*cf.* Algorithms 1 and 2).

Algorithm 1 Best-In heuristic

```
function BEST-IN($G, M$)
    while $V \neq M$ do                                      Unless the module contains all vertices
        $i \leftarrow \arg\max_{i \in V \setminus M} k_i + n \sum_{j \in M} a_{ij}$      Find the one with the best connections
        if $\delta(M \cup \{i\}) = 0$ then                   If it is adjacent to all current vertices
            $M \leftarrow M \cup \{i\}$                                          Add it in the module
        else break
    return $M$
```

Algorithm 2 Worst-Out heuristic

```
function WORST-OUT($G, M$)
    while $\delta(M) \neq 0$ do                                         Until all vertices are adjacent
        $i \leftarrow \arg\min_{i \in M} \sum_{j \in M} a_{ij}$                 Find the one with the worst connections
        $M \leftarrow M \setminus \{i\}$                                         Remove it from the module
    return $M$
```

We also implement the simulated annealing (meta-)heuristic (*cf.* Appendix C).

2. We define a *quasi*-module, or *quasi*-clique [3], as a subset $M \subseteq V$ such that the number of edges that are missing in E for M to be a proper module is smaller than a certain *tolerance*, function of $|M|$.

$$\delta(M) \leq f(|M|) \tag{7}$$

For this question, the tolerance function is a constant $C \in \mathbb{R}_+$. Therefore, x represents a quasi-module iff there exists $y = (y_i) \in \mathbb{R}^n$ such that

$$\sum_{i \in V} y_i \leq 2C \tag{8}$$

and

$$\sum_{j\in V} x_i(1-a_{ij})x_j \leq y_i, \quad \forall i \in V. \tag{9}$$

By linearizing (9), we obtain that finding the largest quasi-module is equivalent to solving

$$\begin{aligned} \max_x \quad & \|x\|_1 \\ \text{s.t.} \quad & \sum_{i\in V} y_i \leq 2C \\ & \sum_{j\in V}(1-a_{ij})(x_i+x_j-1) \leq y_i, \quad \forall i \in V \\ & x \in \{0,1\}^n, y \in \mathbb{R}_+^n \end{aligned}$$

which is a linearly constrained formulation that can be solved using MIP.

Concerning the heuristics, only the stopping conditions have to be altered to take into account the tolerance. For instance, in Algorithm 1, $\delta(M\cup\{i\}) = 0$ becomes $\delta(M\cup\{i\}) \leq C$ and, in Algorithm 2, $\delta(M) \neq 0$ becomes $\delta(M) > C$.

3. For this question, f is a strictly increasing function of the size. Additionally, unless $f(n)$ is a polynomial of n, it won't be possible to formulate the problem as a linear optimization problem. Furthermore, if the order of the polynomial is strictly greater than 2, there will be a size above which the tolerance allows arbitrarily large quasi-modules, without constraints on the edges. For the same reason, the coefficient of the second order term has to be strictly smaller than $\frac{1}{2}$.

   Therefore, our tolerance function is expressed as

   $$f(n) = \frac{1}{2}(\alpha n^2 + \beta n + \gamma) \tag{10}$$

   where $\alpha \in [0,1)$, $\beta \in \mathbb{R}$ and $\gamma \in \mathbb{R}_+$, such that $\alpha+\beta \geq 0$. Therefore, x represents a quasi-module iff there exists $y = (y_i) \in \mathbb{R}^n$ such that

   $$\sum_{i\in V} y_i \leq \beta \sum_{i\in V} x_i + \gamma \tag{11}$$

   and

   $$\sum_{j\in V} x_i(1-\alpha-a_{ij})x_j \leq y_i. \tag{12}$$

   By linearizing (12), the optimization problem becomes

   $$\begin{aligned} \max_x \quad & \|x\|_1 \\ \text{s.t.} \quad & \sum_{i\in V} y_i \leq \beta \sum_{i\in V} x_i + \gamma \\ & \sum_{j\in V}(1-\alpha-a_{ij})(x_i+x_j-1) \leq y_i, \quad \forall i \in V \\ & -\alpha k_i x_i \leq y_i, \quad \forall i \in V \\ & x \in \{0,1\}^n, y \in \mathbb{R}^n \end{aligned}$$

   which is a linearly constrained formulation. Interestingly, this formulation is strictly equivalent to the previous one when $\alpha = \beta = 0$.

Once again, it is straightforward to adapt the heuristics: $\delta(M \cup \{i\}) = 0$ becomes $\delta(M \cup \{i\}) \leq f(|M| + 1)$, in Algorithm 1, and $\delta(M) \neq 0$ becomes $\delta(M) > f(|M|)$, in Algorithm 2. Furthermore, these heuristics are not limited to polynomial tolerance functions (*cf.* Appendix B).

4. To find a covering, we take inspiration in the *iterative clique enumeration* (ICE) used by Shi et al. [4]. The principle of our procedure is to find (an approximation of) the largest (quasi-)module in the graph, remove its vertices from the graph, and restart until the graph is empty. Eventually, the sequence M_i of (quasi-)modules covers the whole set of vertices, without overlap.

   The main drawback of this procedure is that, except for M_1, the (quasi-)modules are limited to a subset of the vertices. To overcome this limitation, we perform a second step to find the largest (quasi-)module M_i' in the whole graph such that $M_i' \supseteq M_i$.

   In our implementation, M_i and M_i' are determined using our greedy heuristics (*cf.* Algorithms 1 and 2).

$$M_i = \textsc{Worst-Out}\left(G, V \setminus \bigcup_{j=1}^{i-1} M_j\right) \tag{13}$$

$$M_i' = \textsc{Best-In}\,(G, M_i) \tag{14}$$

   To limit the overlap generated by the second step, we can limit the size of M_i' with respect to $|M_i|$ (*e.g.* $|M_i'| \leq |M_i| + \log|M_i|$), which requires very few modifications to Algorithm 1.

   Finally, if G is sparse, it is likely that it is *disconnected*, *i.e.* that there are vertices that cannot be joined using the existing edges. If it is the case, G can be efficiently segmented into independent *connected* sub-graphs [5], for which it is easier (less expensive) to find the largest (quasi-)modules and a covering. An additional benefit is that, since they can be treated separately, the sub-graphs can be processed in *parallel*. Thus, in our implementation(s), the first step is always to *segment* the graph into independent connected sub-graphs.

References

[1] Immanuel M Bomze, Marco Budinich, Panos M Pardalos, and Marcello Pelillo. “The maximum clique problem”. In: *Handbook of combinatorial optimization*. Springer, 1999, pp. 1–74 (page 1).

[2] Wikipedia. “Clique problem”. URL: https://en.wikipedia.org/wiki/Clique_problem (page 1).

[3] Jeffrey Pattillo, Alexander Veremyev, Sergiy Butenko, and Vladimir Boginski. “On the maximum quasi-clique problem”. In: *Discrete Applied Mathematics* 161.1-2 (2013), pp. 244–257 (page 2).

[4] Zhiao Shi, Catherine K Derow, and Bing Zhang. “Co-expression module analysis reveals biological processes, genomic gain, and regulatory mechanisms associated with breast cancer progression”. In: *BMC systems biology* 4.1 (2010), pp. 1–14 (page 4).

[5] Wikipedia. “Connectivity”. URL: https://en.wikipedia.org/wiki/Connectivity_(graph_theory) (page 4).

[6] AT Amin and SL Hakimi. “Upper bounds on the order of a clique of a graph”. In: *SIAM Journal on Applied Mathematics* 22.4 (1972), pp. 569–573 (page 6).

A Upper bounds

By definition, if $M \subseteq V$ is a module, we know that $M^2 \subseteq E$. Therefore, we have

$$|E| - |V| \geq \left|M^2\right| - |M| = \binom{|M|}{2} = \frac{|M|^2 - |M|}{2} \tag{15}$$

which is easily transformed into

$$|M| \leq \frac{1 + \sqrt{1 + 8(|E| - |V|)}}{2}. \tag{16}$$

Hence, $\omega(G)$, the size of the largest module in G, is *bounded* by the hereabove right-hand side, which we refer to as $\sqrt{G}$. Going further, one can convince itself that $\omega(G)$ is bounded by the *pivot* of G, *i.e.* the largest number p such that p is smaller than the degree of p vertices.

$$\omega(G) \leq \text{pivot}(G) = \max\left\{p : |\{i : k_i \geq p \mid i \in V\}| \geq p \mid p \in \mathbb{N}\right\} \tag{17}$$

As it is always smaller than $\sqrt{G}$, the pivot provides a stricter bound on $\omega(G)$.

Importantly, unlike $\omega(G)$, computing the pivot is *tractable* and can be performed efficiently, as demonstrated by Algorithm 3.

Algorithm 3 Efficient computation of the pivot of G

```
function PIVOT($G$)
    $K \leftarrow [k_1, k_2, \ldots, k_n]$                Array of vertex degrees
    $K \leftarrow$ SORT($K$)                           Sort from largest to smallest
    $l, u \leftarrow 1, \left\lfloor \sqrt{G} \right\rfloor$
    while $l < u$ do                                Dichotomic search
        $p \leftarrow \left\lceil \frac{l+u}{2} \right\rceil$
        if $p > K[p]$ then
            $u \leftarrow p - 1$
        else
            $l \leftarrow p$
    return $l$
```

Note. This section was written *before* looking at the literature. Unsurprisingly, our results had already been demonstrated in the past; notably by Amin et al. [6].

B Valid tolerance

Let be any function $f : \mathbb{N} \mapsto \mathbb{R}$. We say that f is a *valid* tolerance function iff

1. the empty set is a quasi-module;

$$\delta(\varnothing) \leq f(0) \tag{18}$$

2. a super-set of a quasi-module with the same number of missing edges is a quasi-module;

$$\delta(M) = \delta(N) \leq f(|M|) \Rightarrow \delta(N) \leq f(|N|), \quad \forall M \subseteq N \subseteq V \tag{19}$$

3. a non-empty quasi-module has at least one direct subset which is itself a quasi-module;

$$\delta(M) \leq f(|M|) \Rightarrow \min_{i \in M} \delta(M \setminus \{i\}) \leq f(|M|-1), \quad \forall M \subseteq V : M \neq \varnothing \tag{20}$$

for any graph $G = (E, V)$ derived from a pairwise co-expression matrix S.

The first condition implies that $f(0) \geq 0$ since $\delta(\varnothing) = 0$, by definition. From the second condition, we derive that $f(|N|) \geq f(|M|)$ for all $M \subseteq N$. Hence, $f(n+1) \geq f(n) \geq 0$ for all $n \in \mathbb{N}$, *i.e.* f is an *increasing positive* function.

Concerning the third condition, we observe that, in any non-empty quasi-module M, there is at least one vertex for which the number of missing edges is greater than $\frac{2\delta(M)}{|M|}$, the average number of missing edges per vertex. Therefore, in the worst case,

$$\min_{i \in M} \delta(M \setminus \{i\}) = \delta(M) - \left\lceil \frac{2\delta(M)}{|M|} \right\rceil. \tag{21}$$

In consequence, for f to be valid, it has to satisfy

$$\delta(M) \leq f(|M|) \Rightarrow \delta(M) - \left\lceil \frac{2\delta(M)}{|M|} \right\rceil \leq f(|M|-1) \tag{22}$$

for any non-empty $M \subseteq V$. If $|M| \in \{1, 2\}$,

$$\delta(M) - \left\lceil \frac{2\delta(M)}{|M|} \right\rceil = \delta(M) - \frac{2\delta(M)}{|M|} \leq 0 \tag{23}$$

which always fulfills the condition, since $f(n) \geq 0$. Otherwise, the largest value $\delta(M)$ can take is

$$\min \left\{ \lfloor f(|M|) \rfloor, \binom{|M|}{2} \right\} \tag{24}$$

and the condition becomes

$$\min \left\{ \lfloor f(|M|) \rfloor - \left\lceil \frac{2 \lfloor f(|M|) \rfloor}{|M|} \right\rceil, \binom{|M|-1}{2} \right\} \leq \lfloor f(|M|-1) \rfloor \tag{25}$$

Therefore,

$$f(n) \geq \binom{n}{2} \Rightarrow f(n-1) \geq \binom{n-1}{2} \tag{26}$$

and

$$f(n) < \binom{n}{2} \Rightarrow \lfloor f(n) \rfloor - \lfloor f(n-1) \rfloor \leq \left\lceil \frac{2 \lfloor f(n) \rfloor}{n} \right\rceil \leq n - 1 \tag{27}$$

for all $n \in \mathbb{N} \setminus \{0, 1, 2\}$.

Interesting examples of valid tolerance functions are

$$\begin{aligned} f(n) &= C \in \mathbb{R}^+ \\ f(n) &= \max\{0, n-2\} \\ f(n) &= \log(n+1)(n-1) \\ f(n) &= \sqrt{n}(n-1) \\ f(n) &= \frac{n}{4}(n-1) \end{aligned}$$

C Simulated annealing

The principle of *simulated annealing* is to visit the set of all possible states/modules through small iterative transitions. These transitions are guided by the maximisation (minimisation) of an objective (cost) function; in our case $|M|$. We choose a temperature such that the distribution of visited states is

$$\pi(M) \propto \begin{cases} \alpha_t^{|M|} & \text{if } \delta(M) = 0 \\ 0 & \text{else} \end{cases} \tag{28}$$

for all $M \subseteq V$, with

$$\alpha_t = 4 + 0.5 \log_{10} t \tag{29}$$

for $t = 1, 2, \ldots, T$.

Algorithm 4 Simulated-Annealing heuristic

```
function SIMULATED-ANNEALING($G, M$)
    $Best \leftarrow M$
    for $t \in \{1, 2, \ldots, T\}$ do                          For a fixed number of steps
        $i \leftarrow$ SAMPLE($V$)                               Draw a vertex uniformly
        if $i \in M$ then
            $M' \leftarrow M \setminus \{i\}$                     If it is already in, remove it
        else
            $M' \leftarrow M \cup \{i\}$                          Otherwise, add it
        if $\delta(M') = 0$ then                                  If all vertices are adjacent
            $M \leftarrow M'$ with probability $p = \min\left\{1, \alpha_t^{|M'|-|M|}\right\}$   Apply the change
            if $|Best| < |M|$ then $Best \leftarrow M$
    return $Best$
```

An important condition for this algorithm (*cf.* Algorithm 4) to work is that the Markov chain defined by the transitions is *irreducible*, *i.e.* that there exists a walkable path between any two states. This is the case of our algorithm since 1. $\varnothing$ is a subset of all modules, 2. any subset of a module is itself a module and 3. the path between a module and any of its subsets is two-way walkable.

Additionally, we consider a more refined objective function

$$|M| + \frac{1}{\max_{i \in V} k_i} \sum_{i \in M} k_i \tag{30}$$

which leads a distribution $\pi(M)$ that is favorable to large modules M whose vertices have high degrees. This objective "guides" the algorithm towards *dense* neighborhoods in the graph.