Maximum Clique and the Hessian-Gradient Dynamics

Econometrics and Operations Research

Maastricht University
Bachelor Thesis
Name: Minh Phuong Tran
Student ID: i6133932
Supervisor: Mathias Staudigl

Abstract

In this paper, we approach the maximum clique problem via an interesting heuristics called the Hessian-Gradient dynamics, with respect to the Riemannian metrics induced by Legendre functions. In particular, three trials are developed based on three convex barrier functions, in hope of understanding how different dynamics can be used to potentially bring substantial gains to the outcome. These experiments are conducted with the motivation to make certain improvements in terms of the clique size and running time. Generally speaking, all the iterations are perfectly computable, therefore this method can avoid the need of complicated projections. The proposed algorithm performs well on small data sets, yet it appears to be lacking when dealing with graphs of a larger scale.

1 Introduction

1.1 Problem definition

A simple undirected graph $G = (V, E)$ with $|V| = n$ vertices is given. A subgraph of G is complete if there is no non-adjacent pair of vertices, and its corresponding vertex set $C \subset V$ is called a *clique*. A clique is said to be *maximal* if it is not contained in any other clique of larger size. A (maximal) clique is *maximum* if it has the highest cardinality among all cliques. As its name suggests, the *maximum clique problem* (MCP) is to find a clique C of G, such that $|C|$ is maximum.

MCP is one of the core subjects in Operations Research, with numerous practical applications in a wide range of fields such as bioinformatics [11], scheduling[4], telecommunications [9], etc. For the first example, it deals with the challenge of identifying similar folding patterns, if existed, in a pair of proteins. Secondary structural elements including helices, sheets and atoms can be represented by vertices in a graph, although in this case, edge weights are also necessary to show angular and spatial relationships. Additionally, the method of clique partitioning is effective in flights scheduling to airport gates, in order to maximize the total satisfied preferences and minimize the amount of unassigned flights. Despite being rather useful, the clique problem belongs to a group of classic NP-hard problems that are highly inapproximable. Cliques relate closely to vertex cover and independence set, but their results could not be directly translated onto MCP approximations [5].

1.2 Literature Review

Due to its popularity and NP-hardness, MCP has been the topic of several studies, resulting in a significant amount of diverse methods available as well as new algorithms being suggested continuously. A famous algorithm built by Carraghan and Pardalos in 1990 [3] has been the base for many researchers to develop strategies to derive an exact solution. There are four main methodologies typically being

used [14]. The first technique is to analyze smaller subgraphs so as to acquire information on the upper-bound of the maximum clique cardinality, followed by a recursive deepening strategy [13]. Secondly, vertex coloring is employed as a branching scheme to direct the vertex selection procedure and also a bounding scheme to approximate the boundary value. The coloring algorithm can be applied on the initial graph at once, or repeatedly on smaller subgraphs to tighten the bounds. Thirdly, we can reduce the size of the candidate set by removing the unnecessary vertices that cannot contribute to the current clique situation. Lastly, vertex coloring is combined with MaxSAT [8], a technology to improve the upper-bound based on the partition P of a graph into independent sets.

On the other hand, prominent results have been achieved on heuristic frameworks of MCP. The so-called "sequential greedy heuristics" construct an initial random clique and repeatedly append more vertices according to a certain rule, until no available ones can be further added. The most successful heuristic approach is the local search algorithm, which has plenty of adaptations over the pass decades. In addition, some notable works includes the evolutionary algorithm, the genetic algorithm in combination with other search procedures and the artificial neural networks [15]. Overall, the local search algorithm family has proven to be the most efficient.

The starting point of this paper is to use an above-mentioned popular approach for MCP, the adaptation of the replicator dynamics from evolutionary game theory, in combination with the *Hessian barrier algorithm* (HBA) [10] to yield convergence to optimization points. After successfully implementing the corresponding entropy function as the base for HBA Armijo line search, we further analyze two different geometries: the logarithmic function - a classical interior point barrier, and a regularized function at p-distance, associating with the penalty ε. By tailoring these geometries and choosing a different local norm, the modified computation will possibly reach a higher optimal value or will be sped up.

2 Problem setup and directions

2.1 Motzkin-Straus theorem

Following most other proposed methods for MCP, this paper uses the well-known formulation of Motzkin and Straus [2]

$$\begin{aligned} \max \quad & x^{\mathsf{T}} A_G x \\ \text{subject to} \quad & x \in \Delta_n \end{aligned} \quad , \tag{2.1}$$

where A_G represents the adjacency matrix of G, defined by

$$a_{ij} = \begin{cases} 1 & \text{for } (i,j) \in E \\ 0 & \text{for } (i,j) \notin E \end{cases} \quad ; \tag{2.2}$$

and the unit simplex Δ_n is the set of vectors with coordinates larger than 0 and summing up to 1

$$\Delta_n \equiv \{x \in \mathbb{R}^n : x \geq 0 \text{ and } \mathbb{1}^{\mathsf{T}} x = 1\} \quad . \tag{2.3}$$

The global solution of (2.1) is a characteristic vector x^*, which could be identified as the barycentre of set C: $x^* = \frac{1}{|C|} \sum_{i \in C} e_i$ for unit vector $e_i \in \mathbb{R}^n$. All non-zero coordinates have the value of one over the cardinality of C, highlighting the nodes in the maximum clique. The optimal value also shows the clique number $w(G)$

$$OPT = 1 - \frac{1}{w(G)} \quad . \tag{2.4}$$

Based on this theorem, we are able to overlook the combinatorist structure of MCP and focus on the maximization performance of the above quadratic form instead. Albeit easing the way to numerically solve the problem, this program is still NP-hard because A_G is generally not a positive definite matrix.

2.2 Hessian gradient dynamics

The current paper aims to approach MCP via *Hessian-Gradient dynamics*, a heuristic that considers a specific class of Hessian Riemannian metrics on convex sets.

Instead of A_G, the regularized matrix $Q = A_G + \frac{1}{2}Id$ is taken into consideration, with Id being the $n \times n$ identity matrix. This modification serves to avoid spurious solutions.

As Riemannian gradient flows is a minimization method [1], MCP should be adapted as a minimization problem of $x^{\mathrm{T}}Qx$, with $Q = -(A_G + \frac{1}{2}Id)$. The Riemannian structure induces a metric on the feasible set for which the flow dynamics can be stated as:

$$\dot{x} = -P(x)H(x)^{-1}\nabla f(x) \quad . \tag{2.5}$$

A possible interpretation for (2.5) is the direction of the steepest descent with respect to the chosen geometry.

A *Legendre-type* convex function $h(x)$ satisfying $\lim_{k\to\infty} ||\Delta h(x^k)||_2 = \infty$ is selected to compute the Hessian matrix $H(x)$. Meanwhile, $P(x)$ is the orthogonal projection map onto the linear null space $\mathtt{A}_0 \equiv \{x \in \mathbb{R}^n : Ax = 0\}$

$$P(x) = I - H(x)^{-1}A^{\mathrm{T}}(AH(x)^{-1}A^{\mathrm{T}})^{-1}A \quad . \tag{2.6}$$

The following three geometries are considered:

1) $h(x) = \sum_{i=1}^{n} x_i \log x_i$

2) $h(x) = -\sum_{i=1}^{n} \log x_i$

3) $h(x) = \sum_{i=1}^{n} (x_i + \varepsilon)^p$, $p > 2$

The basic intuition behind this research is as follows. To find optimal characteristic vector x^*, we search for a sequence of points which is behaving almost like the continuous path regarding the Riemannian gradient dynamics. After constructing the initial sequence randomly in the interior of the simplex, we intrique, without discretize, the direction $v(x) \equiv -P(x)\,H(x)^{-1}\nabla f(x)$ and recursively update until x converges.

The Armijo backtracking algorithm, a minimization search scheme to set the highest amount for movement towards a certain direction, is employed. The strategy begins with a relatively large feasible step-size α, and iteratively reduces it by multiplying a shrinking factor δ until the sufficient decrease condition is satisfied. The MCP-tailored computations regarding each of the three chosen convex functions are discussed in details in the next chapters.

3 Hessiant gradient descent and Armijo backtracking

Algorithm 1: Hessian barrier algorithm

Require: sufficient decrease factor and shrink factor $\mu, \delta \in (0,1)$
initialise $x \in \Delta_n$ // initialization
while stopping condition not satisfied **do**
  $v \leftarrow -\text{grad}_x f(x)$ // search direction
  α satisfying feasibility and descent // set step-size
  $x^+ = x + \alpha v$ // set test point
  while $f(x^+) > f(x) - \mu\alpha||v||_x^2$ **do** // check sufficient decrease
    $\alpha \leftarrow \delta\alpha$ // shrink step-size
    $x^+ \leftarrow x + \alpha v$ // update test point
  $x \leftarrow x^+$ // new state

Suppose that $f(x) = \frac{1}{2}x^{\mathrm{T}}Qx$, where Q is the negative generalized adjacency matrix induced from the given graph G, as discussed before.

3.1 Step direction

For the Hessian of the chosen geometry $H(x) = \nabla^2 h(x)$, the search direction is given by the solution of the following optimization problem:

$$\begin{aligned} &\text{minimize} \quad \Delta f(x)^{\mathrm{T}} v + \frac{1}{2}||v||_x^2 \\ &\text{subject to} \quad Av = 0 \end{aligned} \quad . \tag{3.1}$$

In the case of MCP, the constraint term A takes the values of vector $\mathbb{1}^{\mathrm{T}}$ since the feasible set is the standard simplex. $||\cdot||_x$ is prescribed by some Hessian Riemannian metric

$$||v||_x^2 = \langle v, v \rangle_x = -\nabla f(x)^{\mathrm{T}} v(x) \quad , \tag{3.2}$$

which captures the first-order change in the values of $f(x)$ corresponding to $v(x)$. Therefore, the solution of (3.1) can be seen as the direction at which $f(x)$ is decreasing the most steeply.

As mentioned before, the obtained result is given by

$$v(x) \equiv \mathrm{grad}_x f(x) = -P(x)H(x)^{-1}\Delta f(x) \quad . \tag{3.3}$$

3.2 Step size

Considering the current position (coming from the discretization), it is challenging to determine a sufficient change for the gradient descent term $v(x)$. There are two required conditions

(i) ***Feasibility***: x is feasible in every iteration. To specify, all entries need to be positive and add up to 1.
The latter condition of the updated term x^+ will be proven in details for each of the three geometries. The non-negative condition is guaranteed as follows:

(a) Case 1 - If $v_i(x) \geq 0$: $\quad x_i^+ \geq 0$

(b) Case 2 - If $v_i(x) < 0$:

$$\begin{aligned} x_i^+ \geq 0 &\Leftrightarrow x_i + \alpha v_i(x) \geq 0 \\ &\Leftrightarrow x_i \geq -\alpha v_i(x) \\ &\Leftrightarrow \alpha \leq \frac{x_i}{-v_i(x)} \end{aligned} \tag{3.4}$$

$\alpha_0(x) = min\{\frac{x_i}{-v_i(x)} \mid v_i(x) < 0\}$ will ensure non-negative x^+. Then α_0 should serve as an upper-bound for α.

(ii) ***Sufficient decrease***: $f(x^+)$ is sufficiently smaller than $f(x)$ in every iteration. To start with, we take into account the below descent inequality, which holds for all feasible x if ∇f is L-Lipschitz continuous with respect to the ordinary Euclidean norm $||\cdot||_2$

$$f(x') - f(x) \leq \Delta f(x)^{\mathrm{T}}(x' - x) + \frac{L}{2}||x' - x||_2^2 \quad , \tag{3.5}$$

with the L-Lipschitz constant being

$$||\Delta f(y) - \Delta f(x)|| \leq L||y - x|| \quad . \tag{3.6}$$

Input values of the function gradient and consider the characteristics of norm

$$||Qy - Qx|| = ||Q(y - x)|| \leq ||Q|| \cdot ||y - x|| \quad , \tag{3.7}$$

then apply the property of symmetric matrix Q, the constant L can be directly calculated as

$$L = ||Q|| = \sqrt{trace(Q^{\mathrm{T}}Q)} \quad . \tag{3.8}$$

We use this L and recall the angle relation $-\Delta f(x)^{\mathrm{T}} v(x) = ||v(x)||_x^2$ to compute the inequality

$$\begin{aligned} f\big(x + \lambda v(x)\big) - f(x) &\leq \lambda ||v(x)||_x^2 + \frac{1}{2}\lambda^2 L||v(x)||_2^2 \\ &= \beta\lambda ||v(x)||_2^2 + \frac{1}{2}\lambda^2 L||v(x)||_2^2 \\ &= -\beta\lambda \left(1 - \frac{\lambda L}{2\beta}\right)||v(x)||_2^2 \quad . \end{aligned} \tag{3.9}$$

In the second line, $\beta = ||v(x)||_x^2 / ||v(x)||_2^2$, where $||v(x)||_x^2 = v(x)^{\mathrm{T}} H(x) v(x)$ and $||v(x)||_2^2 = v(x)^{\mathrm{T}} v(x)$. Consequently, let's consider the sufficient decrease we aim to achieve

$$f(x^+) \leq f(x) - \mu \cdot \alpha(x) ||v(x)||_x^2 \quad . \tag{3.10}$$

Hence, a step-size of at most $2\beta/L$ will guarantee the descent inequality. To satisfy both feasibility and descent, the initial α should be chosen as

$$\alpha = \min\{\alpha_0, 2\beta/L\} \quad . \tag{3.11}$$

3.3 Recursion

The key in Armijo backtracking is to guarantee the sufficient decrease condition, so the iterate x^+ is accepted only if the inequality in (3.10) is satisfied. Otherwise, the step-size α will be reduced repeatedly with a shrinking factor $\delta \in (0,1)$ until (3.10) is reached. The final step-size of each iteration will has the value of the initial $\underline{\alpha}$ multiplied by δ^r, where $r \geq 0$ is the first non-negative interger to satisfy

$$f\big(x + \delta^r \underline{\alpha}(x) v(x)\big) - f(x) \leq -\mu \delta^r \underline{\alpha}(x) ||v(x)||_x^2 \quad . \tag{3.12}$$

Finally, a stopping criteria is taken into account to end the procedure after the function value has converged. At first, the idea is to stop the algorithm at iteration k if the following condition happens

$$\frac{||x^{k+1} - x^k||}{||x^k||} = \frac{\sum_{i=1}^n |\alpha^k v_i^k|}{\sum_{i=1}^n x_i^k} < \varepsilon \quad . \tag{3.13}$$

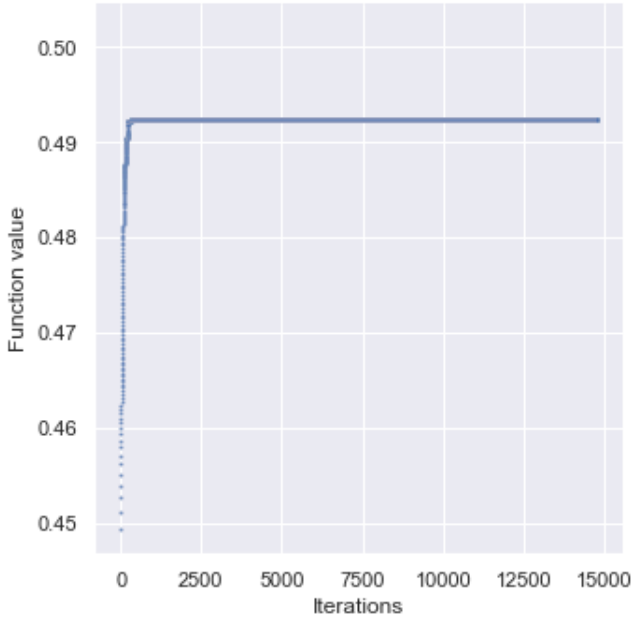

Figure 1: Convergence rate of ϕ_2 (section 6) on instance C125.9, when applying criteria (3.13)

Nevertheless, upon observing the converging behavior of the algorithm illustrated in Figure 1, it is clear that a lot of extra iterations are run even after reaching the desired clique. This is costly in terms of computational time and memory, and reduce the overall efficiency. Therefore, a more simple stopping rule is employed

$$|f(x^+) - f(x)| < \varepsilon \quad , \tag{3.14}$$

where ε is a very small number between 0 and 1 (10^{-9} is chosen in this paper). Despite its simplicity, the criteria performs well in every tested instance, reducing the number of iterations while maintaining the correctness of the result i.e convergence is achieved.

4 Negative entropy function

4.1 Barrier function

Letting $h(x) = \sum_{i=1}^{n} x_i \log x_i$, a direct calculation gives $\Delta h(x) = \begin{pmatrix} \log x_1 & \cdots & \log x_n \end{pmatrix}$, leading in turn to the Hessian form

$$\text{Hessian} = \Delta^2 h(x) = \text{diag}(1/x) = \begin{pmatrix} \frac{1}{x_1} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \frac{1}{x_n} \end{pmatrix} \quad . \tag{4.1}$$

Denote this matrix as X^{-1} and plug it in equation (2.6)

$$\begin{aligned} P(x) &= \mathtt{I} - X\,\mathbb{1}(\mathbb{1}^{\mathtt{T}} X \mathbb{1})^{-1}\mathbb{1}^{\mathtt{T}} \\ &= \mathtt{I} - X \cdot \mathbb{1}(\sum_{t=1}^{n} x_t)^{-1}\,\mathbb{1}^{\mathtt{T}} \\ &= \mathtt{I} - \begin{pmatrix} x_1 & \cdots & x_1 \\ \vdots & \ddots & \vdots \\ x_n & \cdots & x_n \end{pmatrix} \quad . \end{aligned} \tag{4.2}$$

Combining the acquired othorgonal projector with (3.3) yields

$$v(x) = \begin{pmatrix} 1-x_1 & \cdots & x_1 \\ \vdots & \ddots & \vdots \\ x_n & \cdots & 1-x_n \end{pmatrix} \begin{pmatrix} x_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & x_n \end{pmatrix} \begin{pmatrix} (Qx)_1 \\ \vdots \\ (Qx)_n \end{pmatrix} \quad , \tag{4.3}$$

from which the desired direction v_i for each associated x_i can be computed as

$$\begin{aligned} v_i(x) &= x_i(1-x_i)(Qx)_i - \sum_{j \neq i} x_i x_j (Qx)_i \\ &= x_i[(Qx)_i - \sum_{j=1}^{n} x_j (Qx)_j] = x_i[(Qx)_i - x^{\mathtt{T}} Qx] \quad . \end{aligned} \tag{4.4}$$

Then, the updated version x_i^+ in each iteration takes the form

$$x_i^+ = x_i - \alpha \cdot x_i[(Qx)_i - x^{\mathrm{T}}Qx] \quad . \tag{4.5}$$

Feasibility
In order to stay within the interior of the simplex, the total sum of all coordinates in the updated vector needs to be 1. Indeed, the updating rules used in this paper guarantee that condition, which can be shown as follows

$$\begin{aligned} \sum_{i=1}^{n} {x_i}^+ &= \sum_{i=1}^{n} x_i + \sum_{i=1}^{n} \alpha x_i[(Qx)_i - x^{\mathrm{T}}Qx] \\ &= 1 + \alpha \sum_{i=1}^{n} x_i(Qx)_i - \alpha x^{\mathrm{T}}Qx \sum_{i=1}^{n} x_i \\ &= 1 + \alpha x^{\mathrm{T}}Qx - \alpha x^{\mathrm{T}}Q_x = 1 \quad . \end{aligned} \tag{4.6}$$

Sufficient decrease
Using the Hessian matrix form in (4.1) and the corresponding descent $v(x)$, one is able to compute:

(i) $v^{\mathrm{T}}(x)H(x)v(x) = \sum_{i=1}^{n} v_i^2/x_i$

(ii) $v^{\mathrm{T}}(x)v(x) = \sum_{i=1}^{n} v_i^2$

Consequently, we are able to obtain β. Combine with the result in (3.11), we should use

$$\alpha = \min\left\{\alpha_0, \frac{2\sum_{i=1}^{n} \frac{v_i^2}{x_i}}{L \cdot \sum_{i=1}^{n} v_i^2}\right\} \tag{4.7}$$

to avoid violating the feasibility and sufficient decrease conditions.

4.2 Replicator dynamics

This section introduces an interesting theory of the replicator dynamics [2] and how it corresponds to the above-obtained form of the Hessian gradient dynamics for the negative entropy function.

The replicator dynamics is inspired by evolutionary game theory, an application of game theory to the evolution of populations in biology. The object of interest is still the quadratic form $x^{\mathrm{T}}Qx$, in which $Q = [q_{ij}]$ is a generic symmetric $n \times n$ matrix with no positive definite assumptions.

In this context, each entry q_{ij} has the interpretation of the relative fitness of phenotype i when it meets phenotype j. For each coordinate, $(Qx)_i = \sum_{j=1}^{n} q_{ij}x_j$ is the average fitness of gene i, while $f(x) = x^{\mathrm{T}}Qx$ defines the average fitness of the whole population.

Biologists came up with an idea about how genes naturally reproduce in a population. The process is based on basic fitness criteria, in which one should be better than average in the population in order to have a good chance of surviving. In other words, gene i should at least not decrease in frequency in the population if its fitness level is better than average. If, however, the fitness of phenotype i is lower than average, natural selection will drive i to extinction.

(i) $(Qx)_i < x^{\mathrm{T}}Qx$: gene i would die out

(ii) $(Qx)_i \geq x^{\mathrm{T}}Qx$: gene i would not decrease in frequency

This hypothesis leads to the definition of a dynamical system

$$\dot{x}_i(t) = x_i'(t) = x_i(t)\left[(Qx)_i - x^{\mathrm{T}}Qx\right] \quad , \tag{4.8}$$

where x can be seen as a trajectory, determining the frequency of gene i as a function of time. Consequently, $\dot{x}_i$ denotes the time derivative of the growth of this frequency, which is supposedly proportionate to its frequency value and selection

performance. A straight forward computation from (4.8) shows that for any rest points in the dynamics, surviving phenotypes have the same fitness: $(Qx)_i = (Qx)_j$ and equals to the population average.

Now we construct the LaGrange formulation for the original maximization problem, with u and λ as the two associated multipliers

$$\mathfrak{L}(x,u,\lambda) = f(x) + \sum_{i=1}^{n} u_i x_i + \lambda(\mathbb{1}^{\mathrm{T}}x - 1) \quad . \tag{4.9}$$

The Karush-Kuhn-Tucker (KKT) conditions are given by

$$\begin{aligned} &\{1\} \quad \Delta f(x) + u + \mathbb{1}^{\mathrm{T}}\lambda = 0 \\ &\{2\} \quad u_i x_i = 0 \quad ; \quad u_i \geq 0 \\ &\{3\} \quad \mathbb{1}^{\mathrm{T}}x = 1 \end{aligned}$$

Observe that in $\{2\}$, if $x_i > 0$, complementary slackness would require u_i to be equal to zero. Apply this to equation $\{1\}$ and compute for each coordinate

$$\begin{aligned} \{1\} &\Leftrightarrow 2(Qx)_i + u_i + \lambda = 0 \\ &\Rightarrow 2(Qx)_i + \lambda = 0 \Leftrightarrow (Qx)_i = -\frac{1}{2}\lambda \quad . \end{aligned} \tag{4.10}$$

The LaGrange optimization result of Qx equals to a constant $-\lambda/2$ for every coordinate i. Since the stationary points for both dynamics match, it follows that all KKT results satisfying $\{1\}$, $\{2\}$ and $\{3\}$ are fixed points of the replicator dynamics. Or to state it differently, the rest points of the dynamics identify the local solutions of the optimization problem.

Observe that the incarnation of this dynamics given in (4.8) coincides with the induced Riemannian system of function $h(x) = \sum_{i=1}^{n} x_i \log x_i$, and is employed as the search direction to find the solution of MCP.

5 Additional trials

In addition to the negative entropy, two other strictly convex functions are employed. By keeping the same problem setup and switching among different metrics $h(x)$, this paper aims to find the optimal choice of geometry tailored to solve the maximum clique problem. Specific conditions for each of these geometries will be discussed.

5.1 $h(x) = -\sum_{i=1}^{n} \log x_i$

Setting the metrics as the negative sum of $\log x_i$, some straight forward algebra gives $\Delta h(x) = \begin{pmatrix} -1/x_1 & \cdots & -1/x_n \end{pmatrix}$, and consequently

$$\text{Hessian} = \Delta^2 h(x) = \begin{pmatrix} \frac{1}{x_1^2} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \frac{1}{x_n^2} \end{pmatrix} \tag{5.1}$$

A change in the geometry format leads to a change in the projector matrix

$$P(x) = \mathtt{I} - (\sum_{t=1}^{n} x_t^2)^{-1} \begin{pmatrix} x_1^2 & \cdots & x_1^2 \\ \vdots & \ddots & \vdots \\ x_n^2 & \cdots & x_n^2 \end{pmatrix} \quad . \tag{5.2}$$

Note that this projector is almost identical to one in the previous section, except that every x_i is replaced by x_i^2. Since it can be use in formula (3.3) to obtain the dynamics $v(x)$, let's consider the step direction $v(x)_i$ for each entry x_i

$$\begin{aligned} v_i(x) &= x_i^2 \Big(1 - \frac{x_i^2}{\sum_{t=1}^{n} x_t^2}\Big)(Qx)_i - \sum_{j \neq i} \frac{x_i^2 x_j^2 (Qx)_i}{\sum_{t=1}^{n} x_t^2} \\ &= x_i^2 \Big[(Qx)_i - \frac{1}{\sum_{t=1}^{n} x_t^2} \sum_{j=1}^{n} x_j^2 (Qx)_j\Big] = x_i^2 [(Qx)_i - \frac{1}{\sum_{t=1}^{n} x_t^2} (x^2)^{\mathtt{T}} Qx] \quad . \end{aligned} \tag{5.3}$$

Hence, the updating rule is chosen as

$$x_i^+ = x_i - \alpha \cdot x_i^2[(Qx)_i - \frac{1}{\sum_{t=1}^n x_t^2}(x^2)^{\mathrm{T}}Qx] \quad . \tag{5.4}$$

Feasibility
In every iteration, the entries of updated vector x^+ always add up to 1, partly ensuring its position in the simplex. The claim is proven by

$$\begin{aligned}\sum_{i=1}^n x_i{}^+ &= \sum_{i=1}^n x_i + \sum_{i=1}^n \alpha x_i^2[(Qx)_i - \frac{1}{\sum_{t=1}^n x_t^2}(x^2)^{\mathrm{T}}Qx] \\ &= 1 + \alpha \sum_{i=1}^n x_i^2(Qx)_i - \alpha \frac{(x^2)^{\mathrm{T}}Qx}{\sum_{t=1}^n x_t^2}\sum_{i=1}^n x_i^2 \\ &= 1 + \alpha \sum_{i=1}^n x_i^2(Qx)_i - \alpha \sum_{i=1}^n x_i^2(Qx)_i = 1 \quad .\end{aligned} \tag{5.5}$$

Sufficient decrease
Similar to how α for the negative entropy function is computed, we use the corresponding Hessian matrix form and descent $v(x)$ to compute:

(i) $v^{\mathrm{T}}(x)H(x)v(x) = \sum_{i=1}^n v_i^2/x_i^2$

(ii) $v^{\mathrm{T}}(x)v(x) = \sum_{i=1}^n v_i^2$

β can then be obtained directly. Combine with the result in (3.11), we should use

$$\alpha = \min\left\{\alpha_0, \frac{2\sum_{i=1}^n \frac{v_i^2}{x_i^2}}{L \cdot \sum_{i=1}^n v_i^2}\right\} \tag{5.6}$$

to ensure that the feasibility and sufficient decrease conditions are satisfied.

5.2 $h(x) = \sum_{i=1}^{n} (x_i + \varepsilon)^p$

For this convex function, $\begin{pmatrix} p(x_1+\varepsilon)^{p-1} & \cdots & p(x_n+\varepsilon)^{p-1} \end{pmatrix}$ is the form of the gradient $\Delta h(x)$. Then we take the second derivative to obtain the Hessian matrix

$$\text{Hessian} = \Delta^2 h(x) = p(p-1) \begin{pmatrix} (x_1+\varepsilon)^{p-2} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & (x_n+\varepsilon)^{p-2} \end{pmatrix} \quad . \tag{5.7}$$

In this trials, we follow the exact steps used for the previous two geometries. By the process in (4.2), the orthorgonal projector matrix is derived as

$$P(x) = \mathtt{I} - (\sum_{i=1}^{n} (x_i+\varepsilon)^{2-p})^{-1} \begin{pmatrix} (x_1+\varepsilon)^{2-p} & \cdots & (x_1+\varepsilon)^{2-p} \\ \vdots & \ddots & \vdots \\ (x_n+\varepsilon)^{2-p} & \cdots & (x_n+\varepsilon)^{2-p} \end{pmatrix} \quad . \tag{5.8}$$

Apply the obtained projector in (3.3) to find the search direction v_i regarding the chosen convex function

$$\begin{aligned} v_i(x) &= \frac{(x_i+\varepsilon)^{2-p}}{p(p-1)} \left[\left(1 - \frac{(x_i+\varepsilon)^{2-p}}{\sum_{t=1}^{n} (x_t+\varepsilon)^{2-p}}\right)(Qx)_i - \sum_{j\neq i} \frac{(x_j+\varepsilon)^{2-p}(Qx)_i}{\sum_{t=1}^{n} (x_t+\varepsilon)^{2-p}} \right] \\ &= \frac{(x_i+\varepsilon)^{2-p}}{p(p-1)} \left[(Qx)_i - \sum_{j=1}^{n} \frac{(x_j+\varepsilon)^{2-p}(Qx)_i}{\sum_{t=1}^{n} (x_t+\varepsilon)^{2-p}} \right] \\ &= \frac{(x_i+\varepsilon)^{2-p}}{p(p-1)} \left[(Qx)_i - \frac{1}{\sum_{t=1}^{n} (x_t+\varepsilon)^{2-p}} \left((x+\varepsilon)^{2-p}\right)^T Qx \right] \quad . \end{aligned} \tag{5.9}$$

The updating rule is in turn given as

$$x_i^+ = x - \alpha \cdot \frac{(x_i+\varepsilon)^{2-p}}{p(p-1)} \left[(Qx)_i - \frac{1}{\sum_{t=1}^{n} (x_t+\varepsilon)^{2-p}} \left((x+\varepsilon)^{2-p}\right)^T Qx \right] \quad . \tag{5.10}$$

Feasibility

This updating rule does not violate the simplex feasibility condition, proven by

$$\begin{aligned}\sum_{i=1}^{n} x_i{}^{+} &= \sum_{i=1}^{n} x_i + \sum_{i=1}^{n} \alpha \frac{(x_i+\varepsilon)^{2-p}}{p(p-1)} \left[(Qx)_i - \frac{1}{\sum_{t=1}^{n} (x_t+\varepsilon)^{2-p}} \left((x+\varepsilon)^{2-p}\right)^T Qx \right] \\ &= 1 + \frac{\alpha}{p(p-1)} \left[\sum_{i=1}^{n} (x_i+\varepsilon)^{2-p} (Qx)_i - \frac{\left((x+\varepsilon)^{2-p}\right)^T Qx}{\sum_{t=1}^{n} (x_t+\varepsilon)^{2-p}} \sum_{i=1}^{n} (x_i+\varepsilon)^{2-p} \right] \\ &= 1 + \frac{\alpha}{p(p-1)} \left[\sum_{i=1}^{n} (x_i+\varepsilon)^{2-p} (Qx)_i - \left((x+\varepsilon)^{2-p}\right)^T Qx \right] = 1 \quad . \end{aligned} \tag{5.11}$$

Sufficient decrease
For this geometry, the β computations include:

(i) $v^{\mathrm{T}}(x) H(x) v(x) = \sum_{i=1}^{n} v_i^2 \cdot p(p-1)(x_i+\varepsilon)^{p-2}$

(ii) $v^{\mathrm{T}}(x) v(x) = \sum_{i=1}^{n} v_i^2$

In combination with the result in (3.11), we should use

$$\alpha = \min \left\{ \alpha_0, \frac{2p(p-1) \sum_{i=1}^{n} v_i^2 (x_i+\varepsilon)^{p-2}}{L \cdot \sum_{i=1}^{n} v_i^2} \right\} \tag{5.12}$$

to guarantee the feasibility and sufficient decrease conditions.

6 Numerical Results

We are going to conduct numerical experiments regarding the discussed updating rules, given by:

1. ϕ_1 : $\quad x_i^{+} = x_i - \alpha \cdot x_i \left[(Qx)_i - x^{\mathrm{T}} Qx \right]$
2. ϕ_2 : $\quad x_i^{+} = x_i - \alpha \cdot x_i^2 \left[(Qx)_i - \frac{1}{\sum_{t=1}^{n} x_t^2} (x^2)^{\mathrm{T}} Qx \right]$

3. $\phi_3: \quad x_i^+ = x - \alpha \cdot \frac{(x_i+\varepsilon)^{2-p}}{p(p-1)} \left[(Qx)_i - \frac{1}{\sum_{t=1}^{n} (x_t+\varepsilon)^{2-p}} \left((x+\varepsilon)^{2-p} \right)^T Qx \right]$

The results will be compared to ones obtained by a well-known algorithm ϕ [6], identified by the parameters

$$\phi(x) \equiv \alpha \sum_{i=1}^{n} (e^{-\beta x_i} - 1) \quad , \tag{6.1}$$

with $\beta > 0$ and $0 < \alpha < 2/\beta^2$. Algorithm ϕ works as an approximation of the nonsmooth function $\tilde{\phi}(x) = -\alpha ||x||_0$.

6.1 DIMACS

The testing set for experiments is the DIMACS benchmark [7]. Instances are selected from different families with the following characteristics [6]:

(i) C family: graph Cx.y with x vertices and y denoting the edge probability.

(ii) DSJC family: graph DSCJx_y with x vertices and y denoting the edge probability.

(iii) brock family: graph brockx_k with x vertices. Cliques are among vertices with low degrees.

(iv) gen family: the graphs are generated artificially with large known embedded clique.

(v) hamming family: graph hamminga-b on a-bit words, an edge occurs only if the two words are at least hamming distance b apart.

(vi) keller family: the graphs are based on the work of Keller in [12].

(vii) p_hat family: the graphs are spread flatter in terms of vertices' degrees and equipped with higher clique cardinalities, generated with the p-hat generator which is the generalization of the classical uniform random graph generator.

6.2 Experiments

For each instance, 100 trials are run with different initial sequences in the simplex interior to record four parameters: the maximum clique size achieved, the mean and standard deviation over 100 tests and the average CPU running time. To construct these starting points, we consider the Dirichlet distribution, a family of continuous random sample from a standard gamma generator of the form $x = (y_1, \cdots, y_n)/\sum_{i=1}^{n} y_i$. Besides, the variable values $\varepsilon = 10^{-9}; \delta = 0.5$ are applied.

	ϕ_1				ϕ_2				ϕ			
Instances	Max	Mean	Std	CPU	Max	Mean	Std	CPU	Max	Mean	Std	CPU
C125.9	**34**	28.46	1.75	0.08	33	28.67	1.85	2.67	**34**	33.22	0.42	0.25
C250.9	40	35.11	1.79	0.30	40	34.85	1.86	11.57	**44**	40.77	1.13	0.77
C500.9	48	42.64	2.01	9.67	47	42.23	2.09	38.88	54	50.92	1.68	3.36
C1000.9	55	50.03	1.99	60.5	53	49.07	1.61	98.70	63	59.16	1.94	15.76
DSJC500_5	11	9.46	0.86	10.53	**12**	9.23	0.83	50.48	12	10.19	0.44	2.36
brock200_2	**10**	7.95	0.79	0.34	**10**	7.6	0.76	13.2	10	9.04	0.21	0.37
brock200_4	**15**	11.99	0.97	0.38	14	11.76	1.14	11.29	15	13.4	0.64	0.43
brock400_2	23	18.49	1.47	8.19	22	17.97	1.19	38.98	24	21.80	1.23	1.60
brock400_4	22	18.36	1.44	7.66	22	18.04	1.32	64.71	24	21.46	1.05	1.51
gen200_p0.9_44	37	31.96	1.87	0.39	35	31.76	1.89	5.51	40	37.43	1.12	0.49
gen200_p0.9_55	39	34.22	1.79	0.38	37	34.12	1.62	2.34	41	38.98	1.20	0.41
hamming8−4	**16**	10.96	1.99	0.78	**16**	10.74	1.98	11.69	**16**	15.73	1.01	0.51
keller4	**11**	7.83	0.85	0.22	**11**	7.87	0.74	11.06	9	7.02	0.20	0.18
p_hat300−1	**8**	6.28	0.63	0.47	**8**	6.00	0.69	18.80	**8**	8.00	0.00	0.52
p_hat300−2	**25**	21.33	1.69	0.52	**24**	20.2	1.53	28.20	24	24.00	0.00	0.44
p_hat300−3	34	30.01	1.47	0.69	33	29.19	1.7	32.66	**36**	33.2	0.64	0.9

When the algorithm achieves an equal or better clique size, the results are shown in bold. One can observe that ϕ_1 is significantly the better algorithm compared to ϕ_2. One important point is that there are some instances where algorithm ϕ_1 performs better than the well-known algorithm ϕ, especially for instances of smaller size. For large-scale instances (graphs with higher than 300 vertices), however, the running time is notably slower in comparison to ϕ. The reason could be that matrix calculations are computationally very costly. In order to reduce the running time,

a simpler method to identify β and consequently derive α should be employed.

In total, there are 5 instances where ϕ_1 and ϕ_2 are able to compute the best-known clique size, which are recorded in pink (the results for ϕ are also bolded when reaching the best-known value). The two considered geometries have similar results and almost the same mean values. Nevertheless, ϕ_1 indeed always gives the higher maximum clique size, apart from one instance DSJC500_5, while executes in lesser time. Algorithm ϕ_2 has extremely higher CPU time, the values especially leap up when the size of the graph increases.

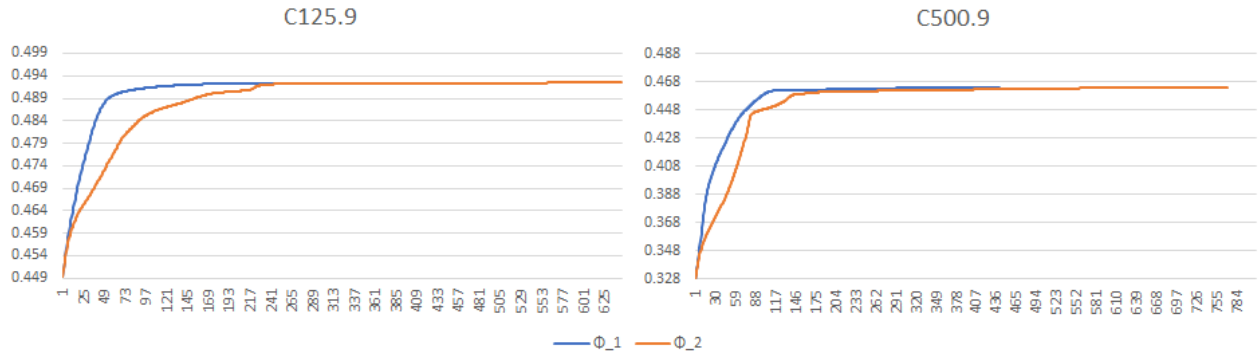

Figure 2: Convergence rates of $0.5x^{\mathrm{T}}Qx$ for ϕ_1 and ϕ_2 on two instances

Particularly for instance C125.9, the total number of iterations until reaching stopping condition is 308 for ϕ_1 and 645 for ϕ_2, which indicates that algorithm ϕ_1 converges substantially faster than ϕ_2, as can also be seen in the graph. ϕ_1 is the better performing algorithm, consistent to our analysis before.

Generally speaking, if we observe the maximum clique size acquired, ϕ_1 has the possibility to surpass algorithm ϕ in performance. The requirement is, however, that the running time must be improve for large-scale instances. On another hand, the results for algorithm ϕ_3 is regrettably not included in this paper as some implementation errors are encountered. This could serve as an interesting question left for future research.

7 Conclusion

In conclusion, we presented three experiments corresponding to three different geometries regarding the MCP. Motivated by the Hessian barrier algorithm that makes use of the Riemannian gradient dynamics and the Armijo line search, some preliminary numerical trials on the widely used DIMACS benchmark were implemented. We showed that the HBA algorithm, applied on the entropy function and logarithm barrier function, yields good outcome in solving the maximum clique problem. The results achieved so far is prominent and leave room for various improvements. To reduce the running time, data structures could be modified to better accommodate large-sized instances. One different way is to improve the rate of convergence by constructing a better technique to define the step-size α. Another intriguing question to study is which graph characteristics helps our method to perform faster or give more accurate solutions, and in turn, why certain instances trouble HBA algorithm, in the sense that the maximum cardinality acquired is considerably lower than algorithm ϕ/ the best-known answer. Finally, finding additional strictly convex Legendre functions to test could be an influential factor in attaining the best search direction to reach the optimal value.

References

[1] Felipe Alvarez, Jérôme Bolte, and Olivier Brahic. Gradient flows associated with hessian riemannian metrics induced by legendre functions in constrained optimization. *SIAM Journal on Control*, 01 2004.

[2] Immanuel M. Bomze. Evolution towards the maximum clique. *Journal of Global Optimization*, 10(2):143–164, Mar 1997.

[3] Randy Carraghan and Panos M. Pardalos. An exact algorithm for the maximum clique problem. *Operations Research Letters*, 9(6):375 – 382, 1990.

[4] Ulrich Dorndorf, Florian Jaehn, and Erwin Pesch. Modelling robust flight-gate scheduling as a clique partitioning problem. *Transportation Science*, 42:292–301, 08 2008.

[5] J. Jeffry Howbert and Jacki Roberts. The maximum clique problem. *Applied Algorithms*, 2007.

[6] James Hungerford and Francesco Rinaldi. A general regularized continuous formulation for the maximum clique problem. *Mathematics of Operations Research*, 09 2017.

[7] David S. Johnson and Michael A. Trick. Cliques, coloring, and satisfiability. 1996.

[8] Chu Min Li and Zhe Quan. An efficient branch-and-bound algorithm based on maxsat for the maximum clique problem. In *AAAI*, 2010.

[9] Immanuel M. Bomze, Marco Budinich, P Pardalos, and Marcello Pelillo. The maximum clique problem. *Handbook of Combinatorial Optimization*, 4, 05 1999.

[10] Immanuel M Bomze, Panayotis Mertikopoulos, Werner Schachinger, and Mathias Staudigl. Hessian barrier algorithms for linearly constrained optimization problems. 09 2018.

[11] Noël Malod-Dognin, Rumen Andonov, and Nicola Yanev. Maximum cliques in protein structure comparison. volume 6049, pages 106–117, 05 2010.

[12] Keller OH. uber die lückenlose erfüllung des raumes mit würfeln. In *J. Die Reine Angew Math*, 1930.

[13] Patric R.J. Östergård. A new algorithm for the maximum-weight clique problem. *Electronic Notes in Discrete Mathematics*, 3:153 – 156, 1999. 6th Twente Workshop on Graphs and Combinatorial Optimization.

[14] Qinghua Wu and Jin-Kao Hao. A review on algorithms for maximum clique problems. *European Journal of Operational Research*, 242(3):693 – 709, 2015.

[15] Gang Yang, Junyan Yi, Zhiqiang Zhang, and Zheng Tang. A tcnn filter algorithm to maximum clique problem. *Neurocomputing*, 72:1312–1318, 2009.