ISEN 621: Homework 4

Feyi Olalotiti-Lawal

April 19, 2015

Problem 1: Graph Bisection Problem with the Neural Network Approach

From any graph $G = (V, E)$ such that $|V| = 2n$, we intend to from 2 disjoint subgraphs V_1 and V_2 such that $V_1 \cup V_2 = V$, $V_1 \cap V_2 = \emptyset$ and $|V_1| = |V_2| = n$ while minimizing the total number of edges between the subgraphs.
In other words, we wish to minimize $f(x) = -\frac{1}{2}x^T A_G x$ where each element in x is defined as $x_i = -1$ if $i \in V_1$ and $x_i = 1$ if $i \in V_2$; A_G represents the adjacency matrix for graph G. However this needs to be penalized to ensure equal number of vertices in both subgraphs. This results in a final Energy function given by:

$$E(x) = -\frac{1}{2}x^T A_G x + \frac{1}{2}\alpha\left(\Big(\sum_{i=1}^{2n} x_i\Big)^2 - \Big(\sum_{i=1}^{2n} x_i^2\Big)\right) \tag{1}$$

This also reduces to:

$$E(x) = -\frac{1}{2}x^T A_G x + \frac{1}{2}\alpha\Big(2\sum_{i \neq j} x_i x_j\Big) \tag{2}$$

Thus the Neural Network transfer function can be constructed as:

$$x_i(k+1) = \tanh\left(\frac{1}{T}\sum_{i \neq j=1}^{2n}(a_{ij} - 2\alpha)x_j\right) = \tanh\left(\frac{1}{T}\sum_{i \neq j=1}^{2n} w_{ij}x_j\right) \;\forall i = 1, \ldots, 2n \tag{3}$$

Where α represents the LaGrange multiplier. The first term in the penalty terms of Equation **??** calculates difference in the sizes of the subgraphs while the second term which has a constant value of $2n$ only regularrizes the augumented matrix W. A function value $f(x)$ which is maximized in the Minimum Bisection algorithm can also constructed as:

$$f(x)) = \frac{1}{2}\sum_{i<j} a_{ij}(1 - x_i x_j) = \frac{1}{4}\Big(\sum_{i,j} a_{ij} - x^T A_G x\Big) \tag{4}$$

A MATLAB code shown in **??** was written to implement this algorithm an apply to the graph shown in **??**. Figure **??** shows the behavior of the energy function $E(x)$ and the objective $f(x)$. As expected, $E(x)$ is minimuzed while $f(x)$ is maximized. For this run, the following optimal bisection subgraphs were obtained: $V_1 = \{1, 2, 4, 6, 9\}$ and $V_2 = \{3, 5, 7, 8, 10\}$; with the corresponding energy and objective function values of $E(x) = -93$ and $f(x) = 17$ respectively.

```
load adjMat;
x0 = 2*rand(size(A,1),1)-1;
numT = 10; maxit = 10; alpha = 10; T0 = 10;
[xbest, fvec, Evec] = MinBisection(A, x0, numT, maxit, alpha, T0);

function [xbest, fvec, Evec] = MinBisection(A, x0, numT, maxit, alpha, T0)
graphSize = size(A,1);
if rem(graphSize,2)~=0; error('Graph size must be even for minimum
bisection'); end
n = graphSize/2;

%Initialize & parameters
T=T0; Tfac = 0.9;
x = x0; xbest = x0;
W = A - 2*alpha*(1-eye(2*n));
Ebest = -0.5*xbest'*W*xbest;

Evec = Ebest;
fvec = 0.25*(sum(sum(A)) - xbest'*A*xbest);

%Calculate
for i = 1:maxit
    x = xbest;
    for j = 1:numT
        for k = 1:2*n
            x(k) = tanh(W(k,:)*x/T);
            E = -0.5*x'*W*x;
            if E<Ebest; xbest = x; Ebest = E; end
        end
        fvec = [fvec; 0.25*(sum(sum(A)) - xbest'*A*xbest)];
        Evec = [Evec; Ebest];
        figure(1)
        subplot(2,1,1), plot(Evec,'o-'); hold on
        subplot(2,1,2), plot(fvec,'ro-'); hold on
    end
    T = Tfac*T;
end
```

Figure 1: MATLAB code for Problem 1

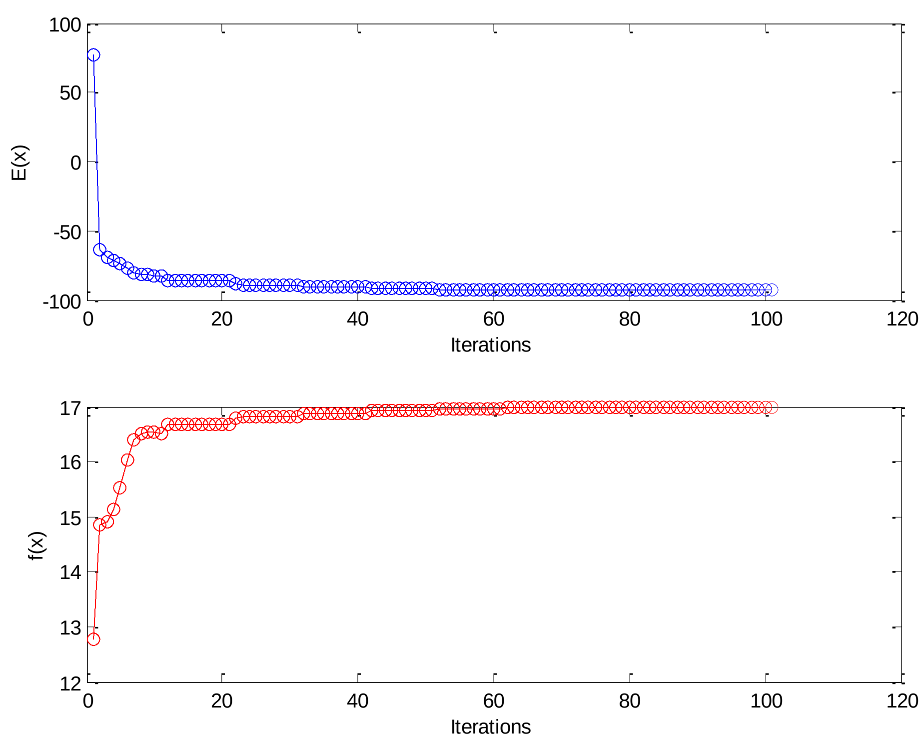

Figure 2: Results for the Neural Network Algorithm

Problem 2: Maximum Clique Problem with the Replicator Dynamics Approach

(a) We consider a maximum clique problem of some graph $G = (V, E)$ of size n and adjacency matrix A_G. This is reduced to a form for which we perform the Motzkin-Straus quadratic program regularization:

$$\begin{aligned} \underset{x}{\text{maximize}} \quad & f(x) = x^T(A_G + \frac{1}{2}I_n)x \\ \text{subject to} \quad & e^T x = 1; x \geq 0 \end{aligned}$$

Defining the characteristic vector of a set of vertices $C \in V$ as a feasible point $x = x^C$ given as:

$$x_i^C = \begin{cases} 1/|C|, & \text{if } i \in C \\ 0, & \text{otherwise} \end{cases}$$

Bomze(1997) showed that x^* maximizes $f(x)$ over the feasible region if and only if x^* is the characteristic vector of the maximum clique in G. It is required to prove that the size of this clique is given by $\frac{1}{2(1-f(x^*))}$.

Proof. Let $m = |C^*|$ represent the maximum clique size so that each element of the corresponding characteristic vector is defiend as:

$$x_i^* = \begin{cases} 1/|C^*|, & \text{if } i \in C^* \\ 0, & \text{otherwise} \end{cases}$$

Writing the augumeneted adjacency matrix in the regularized QP formulation as

$$\widehat{A_G} = A_G + \frac{1}{2}I_n \tag{5}$$

where I_n represents a $n \times n$ identity matrix, the objective function becomes $f(x) = x^T \widehat{A_G} x$ which can be written as:

$$x^T \widehat{A_G} x = \sum_{i,j}^{n} a_{ij}x_i x_j = \sum_{i=j}^{n} a_{ij}x_i x_j + \sum_{i \neq j}^{n} a_{ij}x_i x_j \tag{6}$$

Knowing that $[a]_{ij} = 1/2 \;\; \forall i = j \in [1, n]$; going by the deinition for x^* given above and for a clique size of $|C^*| = m$, the first term of v becomes:

$$\sum_{i=j}^{n} a_{ij}x_i x_j = \sum_{i \in C^*} a_{ii}x_i^2 = \sum_{i \in C^*} \frac{1}{2}\frac{1}{m^2} = \frac{1}{m^2}\frac{m}{2} \tag{7}$$

Also, knowing that $[a]_{ij} = 1 \;\; \forall i = j \in [1, m]$ the second term of equation **??** can be expressed as:

$$\sum_{i \neq j}^{n} a_{ij}x_i x_j = \sum_{i=1; i \in C^*}^{m-1} \sum_{j=i+1}^{m} 2a_{ij}x_i x_j = 2\sum_{i=1}^{m-1} x_i \sum_{j=i+1}^{m} x_j = \frac{2}{m^2}\sum_{i=1}^{m-1}(m-i)$$

But

$$2\sum_{i=1}^{m-1}(m-i) = 2\Big(\sum_{i=1}^{m-1} m - \sum_{i=1}^{m-1} i\Big) = 2\Big(m(m-1) - \frac{m(m-1)}{2}\Big) = m(m-1) \tag{8}$$

Thus The objective function at the maximum clique, $m = |C^*|$ becomes:

$$f(x^*) = x^T \widehat{A_G} x = \frac{1}{m^2}\Big(\frac{m}{2} + m(m-1)\Big) = 1 - \frac{1}{2m} \tag{9}$$

Therefore

$$m = |C^*| = \frac{1}{2(1 - f(x^*))} \quad \square \tag{10}$$

(b) Solving the Motzkin-Straus QP results in an approximate solution of the maximum clique problem for some graph $G = (V, E)$. Bomze (1997) proposed a recurvsive algorithm, called the replicator dynamics, to solve the problem. In this algorithm, starting from a random vector of size $|V|$ for x, the $(k+1)^{th}$ value of x_i for vertex i is calculated from the k^{th} value as follows:

$$x_i(k+1) = x_i^k \frac{(A_G x(k))_i}{x(k)^T A_G x(k)}; \ \ \forall i \in V \tag{11}$$

A MATLAB code shown in Figure **??** was written to implement the replicator dynamics algorithm. This was applied to a graph of size 10 shown in Figure **??**. The stopping criteria was set for some small $\epsilon > 0$ as follows:

$$\max_{i \in 1,\dots,n} |x_i(k+1) - x_i(k)| < 2\epsilon \tag{12}$$

The augumented adjacency matrix corresponding the graph in Figure **??** is given as:

$$A_G = \begin{bmatrix} \frac{1}{2} & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & \frac{1}{2} & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & \frac{1}{2} & 1 & 0 & 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & \frac{1}{2} & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & \frac{1}{2} & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & \frac{1}{2} & 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 & 1 & 1 & \frac{1}{2} & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & \frac{1}{2} & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & \frac{1}{2} & 1 \\ 1 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & \frac{1}{2} \end{bmatrix}$$

The algorithm was run 10 times, each time with a completely random intial starting vector for x such that $x_i \in [0, 1] \ \ \forall i \in V$. Figure **??** shows the objective function values with iteration for the 10 trials. As can be noticed, the maximum objective function, $f(x^*) = 0.900$, was obtained for all but one trails. The maximum clique obtained was $C^* = \{1, 3, 4, 9, 10\}$, that is $|C^*| = 5$. The deviant trail, on the other hand gave a $C^* = \{2, 3, 4, 9\}$, that is $|C^*| = 4$.

```
function [Results, Obj, maxClique, maxObj] = ReplicatorDyn(AG,N)

%Initialize & parameters
n = size(AG,1);
Results = zeros(n,N);
Initial = zeros(n,N);
myeps = 1e-6;
Obj = zeros(1000,N);

% Begin Algorithm
for i = 1:N
    x0 = rand(n,1); x0 = x0/sum(x0);
    Initial(:,i) = x0;
    itr = 1; notConverged = 1;
    while notConverged
        Obj(itr,i) = (x0'*AG*x0);
        x = x0.*(AG*x0)/Obj(itr,i);
        notConverged = max(abs(x-x0)) > myeps;
        x0 = x;
        itr = itr+1;
    end
    Results(:,i) = x0;
end
Results(Results<myeps) = 0;

% Visualize
maxObj = 0; imax = 0;
figure(1); hold on
for i = 1:N
    stopind = find(Obj(:,i)<eps,1,'first')-1;
    plot(Obj(1:stopind,i),'linewidth',2);
    if Obj(stopind) > maxObj; maxObj = Obj(stopind); imax = i; end
end
maxClique = find(Results(:,imax)~=0);

grid on;
xlabel('Iterations'); ylabel('Objective Function')
```

Figure 3: MATLAB code for Problem 2

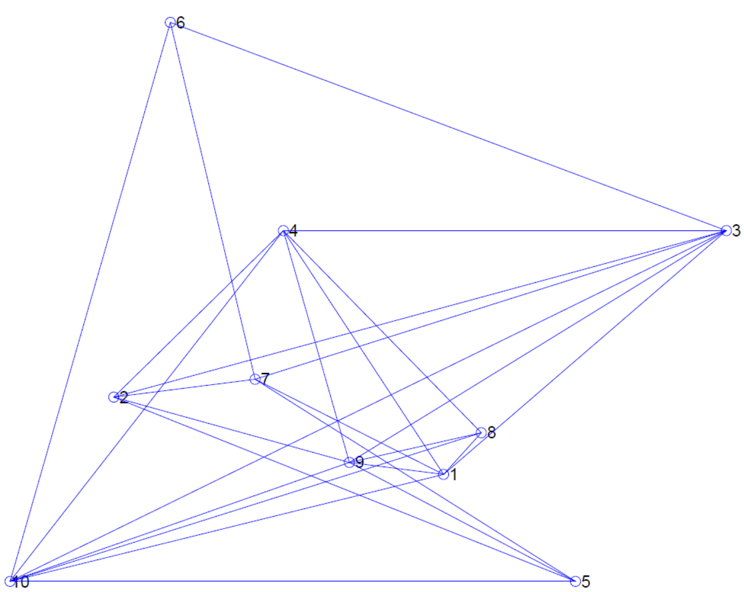

Figure 4: Graph Example

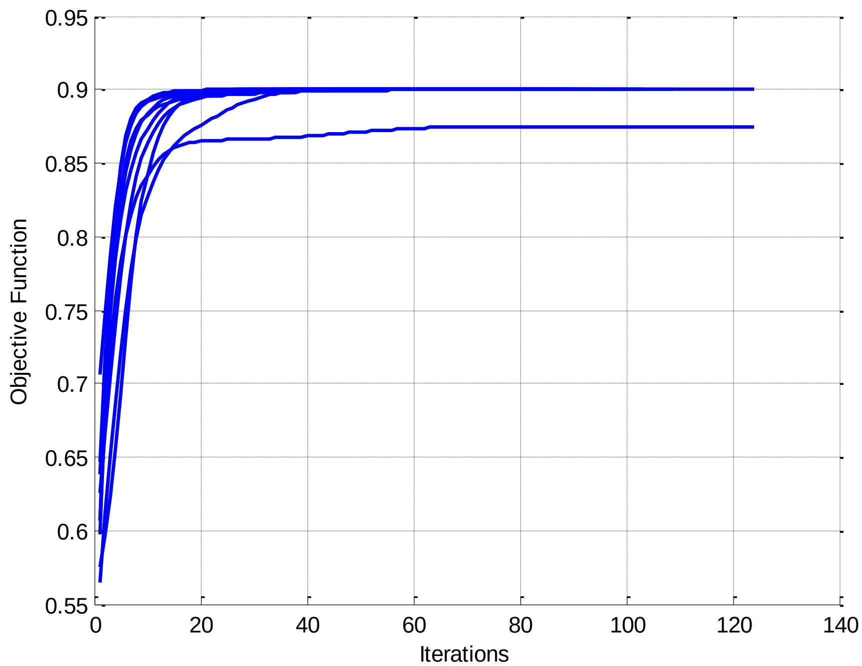

Figure 5: Plot of Objective Function for 10 Trials