The Graph Partitioning Problem

Adam Rahman

October 17, 2018

The graph partitioning problem can be formulated as the following primal optimization problem

$$\begin{array}{ll} \underset{\mathbf{X}}{\text{minimize}} & tr(\mathbf{CX}) \\ \text{subject to} & \\ & \begin{array}{rcl} tr(\mathbf{11}^\intercal\mathbf{X}) & = & \alpha \\ diag(\mathbf{X}) & = & \mathbf{1} \end{array} \end{array}$$

Here, $\mathbf{C} = -(diag(\mathbf{B1}) - \mathbf{B})$, for an adjacency matrix $\mathbf{B}$, and α is any real number.

The function `gpp`, takes as input a weighted adjacency matrix `B` and a real number `alpha` and returns the optimal solution using `sqlp`.

```
R> out <- gpp(B,alpha)
```

Numerical Example

To demonstrate the output provided by `sqlp`, we make use of the following adjacency matrix

```
R> data(Bgpp)
R> Bgpp
```

```
      V1 V2 V3 V4 V5 V6 V7 V8 V9 V10
 [1,]  0  0  0  1  0  0  1  1  0   0
 [2,]  0  0  0  1  0  0  1  0  1   1
 [3,]  0  0  0  0  0  0  0  1  0   0
 [4,]  1  1  0  0  0  0  0  1  0   1
 [5,]  0  0  0  0  0  0  1  1  1   1
 [6,]  0  0  0  0  0  0  0  0  1   0
 [7,]  1  1  0  0  1  0  0  1  1   1
 [8,]  1  0  1  1  1  0  1  0  0   0
 [9,]  0  1  0  0  1  1  1  0  0   1
[10,]  0  1  0  1  1  0  1  0  1   0
```

Any value of α in $(0, n^2)$ can be chosen, so without loss of generality, we choose a value of n to solve the problem.

```
alpha <- nrow(Bgpp)
```

```
out <- gpp(Bgpp, alpha)
```

As with the max-cut problem, the output of interest here is the primal objective function, keeping in mind that we have swapped the sign of the objective function so that the primal problem is a minimization.

```
out$pobj
```

```
[1] -57.20785
```

Also like the maxcut problem, the set of feasible solutions are correlation matrices

```
out$X[[1]]     #Rounded to 3 decimal places
      [,1]   [,2]   [,3]   [,4]   [,5]   [,6]   [,7]   [,8]   [,9]  [,10]
V1   1.000  1.000  0.550 -0.604  0.702  0.895 -0.611  0.006 -0.920  0.741
V2   1.000  1.000  0.572 -0.583  0.721  0.907 -0.590 -0.020 -0.930  0.723
V3   0.550  0.572  1.000  0.333  0.981  0.865  0.325 -0.832 -0.834 -0.153
V4  -0.604 -0.583  0.333  1.000  0.143 -0.186  1.000 -0.800  0.243 -0.983
V5   0.702  0.721  0.981  0.143  1.000  0.946  0.135 -0.708 -0.926  0.043
V6   0.895  0.907  0.865 -0.186  0.946  1.000 -0.194 -0.440 -0.998  0.365
V7  -0.611 -0.590  0.325  1.000  0.135 -0.194  1.000 -0.796  0.251 -0.984
V8   0.006 -0.020 -0.832 -0.800 -0.708 -0.440 -0.796  1.000  0.387  0.676
V9  -0.920 -0.930 -0.834  0.243 -0.926 -0.998  0.251  0.387  1.000 -0.418
V10  0.741  0.723 -0.153 -0.983  0.043  0.365 -0.984  0.676 -0.418  1.000
```