The Max-kCut Problem

Adam Rahman

February 8, 2019

Similar to the Max-Cut problem, the Max-kCut problem asks, given a graph $\mathbf{G} = (\mathbf{V}, \mathbf{E})$ and an integer k, does a cut exist of at least size k. For a given (weighted) adjacency matrix $\mathbf{B}$ and integer k, the Max-kCut problem is formulated as the following primal problem

$$\begin{array}{ll} \underset{\mathbf{X}}{\text{minimize}} & \langle \mathbf{C},\ \mathbf{X} \rangle \\ \text{subject to} & \\ & \begin{array}{rcll} diag(\mathbf{X}) & = & \mathbf{1} & \\ X_{ij} & \geq & 1/(k-1) & \forall\ i \neq j \\ \mathbf{X} & \in & \mathcal{S}_n & \end{array} \end{array}$$

Here, $\mathbf{C} = -(1 - 1/k)/2 * (diag(\mathbf{B1}) - \mathbf{B})$. The Max-kCut problem is slightly more complex than the Max-Cut problem due to the inequality constraint. In order to turn this into a standard SQLP, we must replace the inequality constraints with equality constraints, which we do by introducing a slack variable $\mathbf{x}^l$, allowing the problem to be restated as

$$\begin{array}{ll} \underset{\mathbf{X}}{\text{minimize}} & \langle \mathbf{C},\ \mathbf{X} \rangle \\ \text{subject to} & \\ & \begin{array}{rcll} diag(\mathbf{X}) & = & \mathbf{1} & \\ X_{ij} - x^l & = & 1/(k-1) & \forall\ i \neq j \\ \mathbf{X} & \in & \mathcal{S}^n & \\ \mathbf{x}^l & \in & \mathcal{L}^{n(n+1)/2} & \end{array} \end{array}$$

The function `maxkcut` takes as input an adjacency matrix `B` and an integer `k`, and returns the optimal solution using `sqlp`.

```
R> out <- maxkcut(B,k)
```

Numerical Example

To demonstrate the output provided by `sqlp`, consider the adjacency matrix

```
R> data(Bmaxkcut)
R> Bmaxcut
```

```
     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
```

Like the max-cut problem, here we are interested in the primal objective function. Like the max-cut problem, we take the negated value. We will use a value of $k = 5$ in the example.

```
R> out <- maxkcut(Bmaxkcut,5)

R> -out$pobj
[1] 19
```

Note also that the returned matrix `X` is a correlation matrix

```
      [,1]   [,2]   [,3]   [,4]   [,5]   [,6]   [,7]   [,8]   [,9]  [,10]
V1   1.000  0.381  0.503 -0.250  0.403  0.347 -0.250 -0.250  0.060  0.181
V2   0.381  1.000  0.231 -0.250  0.627  0.380 -0.250  0.160 -0.250 -0.250
V3   0.503  0.231  1.000  0.395  0.387  0.597  0.185 -0.250  0.074  0.089
V4  -0.250 -0.250  0.395  1.000  0.134  0.261  0.449 -0.250  0.163 -0.250
V5   0.403  0.627  0.387  0.134  1.000  0.348 -0.250 -0.250 -0.250 -0.250
V6   0.347  0.380  0.597  0.261  0.348  1.000  0.224  0.180 -0.250  0.239
V7  -0.250 -0.250  0.185  0.449 -0.250  0.224  1.000 -0.250 -0.250 -0.250
V8  -0.250  0.160 -0.250 -0.250 -0.250  0.180 -0.250  1.000  0.118  0.216
V9   0.060 -0.250  0.074  0.163 -0.250 -0.250 -0.250  0.118  1.000 -0.250
V10  0.181 -0.250  0.089 -0.250 -0.250  0.239 -0.250  0.216 -0.250  1.000
```