One of the simplest problems that can be formulated in terms of a conic linear optimization problem is finding the maximum cut of a graph. Let $G = [V, E]$ be a graph with vertices $V$ and edges $E$. A cut of the graph $G$ is a partition of the vertices of $G$ into two disjoint subsets $G_1 = [V_1, E_1]$, $G_2 = [V_2, E_2]$, with $V_1 \cap V_2 = \emptyset$. The size of the cut is defined to be the number of edges connecting the two subsets. The maximum cut is defined to be the cut of a graph $G$ whose size is at least as large as any other cut. For a weighted graph object, we can also define the maximum cut to be the cut with weight at least as large as any other cut.

Finding the maximum cut is referred to as the Max-Cut Problem, and was one of the first problems found to be NP-complete, and is also one of the 21 algorithms on Karp’s 21 NP-complete problems ([2]). The Max-Cut problem is also known to be APX hard ([3]), meaning in addition to there being no polynomial time solution, there is also no polynomial time approximation.

Using the semidefinite programming approximation formulation of [1], the Max-Cut problem can be approximated to within an approximation constant. For a weighted adjacency matrix $B$, the objective function can be stated as

$$\begin{align*}
\text{minimize } & \langle C, X \rangle \\
\text{subject to } & \text{diag}(X) = 1 \\
& X \in S^n
\end{align*}$$

where $S^n$ is the cone of symmetric positive semidefinite matrices of size $n$, and $C = -(\text{diag}(B1) - B)/4$. Here, we define $\text{diag}(a)$ for an $n \times 1$ vector $a$ to be the diagonal matrix $A = [A_{ij}]$ of size $n \times n$ with $A_{ii} = a_i$, $i = 1, \ldots, n$. For a matrix $X$, $\text{diag}(X)$ extracts the diagonal elements from $X$ and places them in a column vector.

To see that the Max-Cut problem is a conic linear optimization problem it needs to be written in the same form as the primal objective function. The objective function is already in a form identical to that of the primal objective function, with minimization occurring over $X$ of its inner product with a constant matrix $C = -(\text{diag}(B1) - B)/4$. There are $n$ equality constraints of the form $x_{kk} = 1$, $k = 1, \ldots, n$, where $x_{kk}$ is the $k^{th}$ diagonal element of $X$, and $b_k = 1$ in the primal objective function. To represent this in the form $\langle A_k, X \rangle = x_{kk}$, take $A_k$ to be

$$A_k = [a_{ij}] = \begin{cases} 1, & i = j = k \\ 0, & \text{otherwise} \end{cases}$$

Now $\langle A_k, X \rangle = \text{vec}(A_k)^T \text{vec}(X) = x_{kk}$ as required, and the Max-Cut problem is specified as a conic linear optimization problem.

To convert this to a form usable by sqlp, we begin by noting that we have one optimization variable, $X$. Therefore, with $L = 1$, and having $X$ constrained to the space of semidefinite matrices of size $n$, we specify blk as

```r
blk <- c("s" = n)
```

With the objective function in the form $\langle C, X \rangle$, we define the input $C$ as

```r
one <- matrix(1,nrow=n,ncol=1)
C <- -(diag(c(B %*% one)) - B)/4
```
where $B$ is the adjacency matrix for a graph on which we would like to find the maximum cut, such as the one in Figure 1.

$$
B = \begin{bmatrix}
0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
1 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 \\
1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 \\
0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0
\end{bmatrix}
$$

Figure 1: A graph object and associated adjacency matrix for which we would like to find the maximum cut.

The matrix $A_k$ is constructed using the upper triangular portion of the $A_k$ matrices. To do this in R, the function `svec` is made available in `sdpt3r`.

```r
R> #Construct Ak matrices
R> A <- matrix(list(), nrow=1, ncol=n)
R> for(k in 1:n){
R>   A[[k]] <- matrix(0, nrow=n, ncol=n)
R>   diag(A[[k]])[k] <- 1
R> }

R> #Combine to form At
R> At <- svec(blk[1], A, 1)
```

Having each of the diagonal elements of $X$ constrained to be 1, $b$ is a $n \times 1$ matrix of ones

```r
R> b <- matrix(1, nrow=n, ncol=1)
```

With all the input variables now defined, we can now call `sqlp` to solve the Max-Cut problem

```r
R> sqlp(blk, list(At), list(C), b)
```

The built-in function `maxcut` takes as input a (weighted) adjacency matrix $B$ and returns the maximum cut of the graph using `sqlp`. If we wish to find to the maximum cut of the graph in Figure 1, given the adjacency matrix $B$ we can compute the solution using `sqlp` using `maxcut`

```r
R> out <- maxcut(B)
R> out$pobj
```
Matrix $X$ is actually a correlation matrix by considering its eigenvalues - we can see it clearly is symmetric, with unit diagonal and all elements in [-1,1].

As an interesting aside, we can show that the matrix $X$ is indeed a correlation matrix by considering its eigenvalues.

The fact that $X$ is indeed a correlation matrix comes as no surprise.

References

