The Nearest Correlation Matrix

Adam Rahman

October 17, 2018

First addressed by [1] in dealing with correlations between stock prices, difficulty arises when data is not available for all stocks on each day, which is unfortunately a common occurrence. To help address this situation, correlations are calculated for pairs of stocks only when data is available for both stocks on any given day. The resulting correlation matrix is only approximate in that it is not necessarily positive semidefinite.

This problem was cast by [1] as

$$\begin{array}{ll} \underset{\mathbf{X}}{\text{minimize}} & ||\mathbf{R}-\mathbf{X}||_F \\ \text{subject to} & \\ & \begin{array}{rcl} diag(\mathbf{X}) & = & \mathbf{1} \\ \mathbf{X} & \in & \mathcal{S}^n \end{array} \end{array}$$

where $\mathbf{R}$ is the approximate correlation matrix and $||\cdot||_F$ denotes the Frobenius norm. Unfortunately, the Frobenius norm in the objective function prevents the problem being formatted as a conic linear optimization problem.

Since the matrix $\mathbf{X}$ is constrained to have unit diagonal and be symmetric, and the matrix $\mathbf{R}$ is an approximate correlation matrix, meaning it will also have unit diagonal and be symmetric, we can re-write the objective function as

$$||\mathbf{R}-\mathbf{X}||_F = 2 * ||svec(\mathbf{R}) - svec(\mathbf{X})|| = 2 * ||\mathbf{e}||$$

Now, introduce a variable e_0 such that $e_0 \geq ||\mathbf{e}||$, and define $\mathbf{e}^* = [e_0;\ \mathbf{e}]$. The vector $\mathbf{e}^*$ is now restricted to be in the quadratic cone $\mathcal{Q}^{n(n+1)/2+1}$. This work leads to the formulation of [3]

$$\begin{array}{ll} \underset{\mathbf{e}^*,\ \mathbf{X}}{\text{minimize}} & e_0 \\ \text{subject to} & \\ & \begin{array}{rcl} svec(\mathbf{R}) - svec(\mathbf{X}) & = & [\mathbf{0}, \mathbf{I}_{n(n+1)/2}]\ \mathbf{e}^* \\ diag(\mathbf{X}) & = & \mathbf{1} \\ \mathbf{X} & \in & \mathcal{S}^n \\ \mathbf{e}^* & \in & \mathcal{Q}^{n(n+1)/2+1} \end{array} \end{array}$$

Here, $[\mathbf{X}, \mathbf{Y}]$ denotes column binding of the two matrices $\mathbf{X}_{n\times p}$ and $\mathbf{Y}_{n\times m}$ to form a matrix of size $n \times (p+m)$. By minimizing e_0, we indirectly minimize $\mathbf{e} = svec(\mathbf{R}) - svec(\mathbf{X})$, since recall we have $e_0 \geq ||\mathbf{e}||$, which is the goal of the original objective function.

To see this as a conic linear optimization problem, notice that e_0 can be written as $\langle \mathbf{C}^q,\ \mathbf{X}^q \rangle$ by letting $\mathbf{C}^q = [1; \mathbf{0}_{n(n+1)/2}]$ and $\mathbf{X}^q = \mathbf{e}^*$. Since the matrix $\mathbf{X}$ (i.e. $\mathbf{X}^s$) does not appear in the objective function, the matrix $\mathbf{C}^s$ is an $n \times n$ matrix of zeros.

Re-writing the first constraint as

$$svec(\mathbf{X}) + [\mathbf{0}, \mathbf{I}_{n(n+1)/2}]\ \mathbf{e}^* = \quad svec(\mathbf{R})$$

we can easily define the constraint matrices and right hand side of the first constraint as

$$\begin{aligned}\mathbf{A}_1^s &= \mathbf{I}_{n(n+1)/2} \\ \mathbf{A}_1^q &= [\mathbf{0}, \mathbf{I}_{n(n+1)/2}] \\ \mathbf{b}_1 &= svec(\mathbf{R})\end{aligned}$$

The second constraint is identical to the constraint from the Max-Cut problem, where each diagonal element of $\mathbf{X}$ is constrained to be equal to 1. Define $\mathbf{b}_2 = \mathbf{1}$, and for the k^{th} diagonal element of $\mathbf{X}$, define the matrix $\mathbf{A}_k$ as

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

yielding $\langle \mathbf{A}_k, \mathbf{X} \rangle = x_{kk}$. To write this as $(\mathbf{A}_2^s)^\mathsf{T} \mathbf{X}^s$, define

$$\mathbf{A}_2^s = [svec(\mathbf{A}_1), ..., svec(\mathbf{A}_n)]$$

Since $\mathbf{e}^*$ does not appear in the second constraint, $\mathbf{A}_2^q = \mathbf{0}_{n(n+1)/2+1}$.

The final step is to combine the individual constraint matrices from each constraint to form one constraint matrix for each variable, which is done by defining $\mathbf{A}^s = [\mathbf{A}_1^s,\ \mathbf{A}_2^s]$, $\mathbf{A}^q = [\mathbf{A}_1^q,\ \mathbf{A}_2^q]$. We also concatenate both right hand side vectors to form a single vector by defining $\mathbf{b} = [\mathbf{b}_1;\ \mathbf{b}_2]$. Here, the notation $[\mathbf{X}; \mathbf{Y}]$ is used to denote two matrices $\mathbf{X}_{p \times m}$ and $\mathbf{Y}_{q \times m}$ bound vertically to form a matrix of size $(p + q) \times m$. With this, the nearest correlation matrix problem is written as a conic linear optimization.

To solve this problem using `sqlp`, we first define `blk`. There are two optimization variables in the formulation of the nearest correlation matrix - $\mathbf{X}$ is an $n \times n$ matrix constrained to be in a semidefinite cone, and $\mathbf{y}$ is an $n(n+1)/2+1$ length vector constrained to be in a quadratic cone, so

```
R> blk <- c("s" = n, "q" = n*(n+1)/2+1)
```

Note that $\mathbf{X}$ does not appear in the objective function, so the `C` entry corresponding to the block variable $\mathbf{X}$ is an $n \times n$ matrix of zeros, which defines `C` as

```
R> C1 <- matrix(0,nrow=n,ncol=n)
R> C2 <- rbind(1, matrix(0,nrow=n2,ncol=1))
R> C = list(C1,C2)
```

Next comes the constraint matrix for $\mathbf{X}$

```
R> At <- matrix(list(),nrow=2,ncol=1)
R>
R> #Constraint Matrix for Upper Triangular Elements of X
R> A1s <- diag(1,nrow=n2,ncol=n2)
R>
R> #Construct Ak matrices
R> Aks <- matrix(list(),nrow=1,ncol=n)
R> for(k in 1:n){
R>   Aks[[k]] <- matrix(0,nrow=n,ncol=n)
R>   diag(Aks[[k]])[k] <- 1
R> }
R>
R> A2s <- svec(blk[1,],Aks)
R>
R> #Combined Constraint Matrix for X
R> At1 <- cbind(A1s,A2s)
```

then the constraint matrix for $\mathbf{e}^*$.

```
R> A1q<- matrix(0,nrow=n,ncol=n2+1)
R>
R> A2q1 <- matrix(0,nrow=n2,ncol=1)
R> A2q2 <- diag(1,nrow=n2,ncol=n2)
R> A2q <- cbind(A211, A212)
R>
R> At2 <- rbind(A1q, A2q)
```

and the right hand side vector `b`

```
R> b <- rbind(svec(blk[1],R,1),matrix(1,n,1))
```

The nearest correlation matrix problem is now solved by

```
R> sqlp(blk, list(At1,At2), C, b)
```

To demonstrate the nearest correlation matrix problem, we will modify an existing correlation matrix by exploring the effect of changing the sign of just one of the pairwise correlations. In the context of stock correlations, we make use of tools available in the `R` package `quantmod` ([2]) to access stock data from five tech firms (Microsoft, Apple, Amazon, Alphabet/Google, and IBM) beginning in 2007.

```
R> library(quantmod)

R> getSymbols(c("MSFT", "AAPL", "AMZN", "GOOGL", "IBM"))
R> stock.close <- as.xts(merge(MSFT, AAPL, AMZN, GOOGL,IBM))[,c(4,10,16,22,28)]
```

The correlation matrix for these five stocks is

```
R> stock.corr <- cor(stock.close)
R> stock.corr


            MSFT.Close AAPL.Close AMZN.Close GOOGL.Close IBM.Close
MSFT.Close   1.0000000 -0.2990463  0.9301085   0.5480033 0.2825698
AAPL.Close  -0.2990463  1.0000000 -0.1514348   0.3908624 0.6887127
AMZN.Close   0.9301085 -0.1514348  1.0000000   0.6228299 0.3870390
GOOGL.Close  0.5480033  0.3908624  0.6228299   1.0000000 0.5885146
IBM.Close    0.2825698  0.6887127  0.3870390   0.5885146 1.0000000
```

Next, consider the effect of having a positive correlation between Microsoft and Apple

```
R> stock.corr[1,2] <- -1 * stock.corr[1,2]
R> stock.corr[2,1] <- stock.corr[1,2]
R> stock.corr

            MSFT.Close AAPL.Close AMZN.Close GOOGL.Close IBM.Close
MSFT.Close   1.0000000  0.2990463  0.9301085   0.5480033 0.2825698
AAPL.Close   0.2990463  1.0000000 -0.1514348   0.3908624 0.6887127
AMZN.Close   0.9301085 -0.1514348  1.0000000   0.6228299 0.3870390
GOOGL.Close  0.5480033  0.3908624  0.6228299   1.0000000 0.5885146
IBM.Close    0.2825698  0.6887127  0.3870390   0.5885146 1.0000000
```

Unfortunately, this correlation matrix is not positive semidefinite

```
R> eigen(stock.corr)$values

[1]  2.8850790  1.4306393  0.4902211  0.3294150 -0.1353544
```

Given the approximate correlation matrix `stock.corr`, the built-in function `nearcorr` provides the optimal solution using `sqlp`

```
R> out <- nearcorr(stock.corr)
```

Since this is a minimization problem, and thus a primal formulation of the SQLP, the output `X` from `sqlp` will provide the optimal solution to the problem - that is, `X` will be the nearest correlation matrix to `stock.corr`.

```
foo$X

          [,1]        [,2]        [,3]      [,4]      [,5]
[1,] 1.0000000  0.25388359  0.86150833 0.5600734 0.3126420
[2,] 0.2538836  1.00000000 -0.09611382 0.3808981 0.6643566
[3,] 0.8615083 -0.09611382  1.00000000 0.6115212 0.3480430
[4,] 0.5600734  0.38089811  0.61152116 1.0000000 0.5935021
[5,] 0.3126420  0.66435657  0.34804303 0.5935021 1.0000000
```

The matrix above is symmetric with unit diagonal and all entries in $[-1, 1]$. By checking the eigenvalues,

```
eigen(X)

$values

[1] 2.846016e+00 1.384062e+00 4.570408e-01 3.128807e-01 9.680507e-11
```

we can see that `X` is indeed a correlation matrix.

References

[1] Nicholas J Higham. Computing the nearest correlation matrix-a problem from finance. *IMA Journal of Numerical Analysis*, 22(3):329–343, 2002.

[2] Jeffrey A. Ryan and Joshua M. Ulrich. *quantmod: Quantitative Financial Modelling Framework*, 2017. R package version 0.4-9.

[3] Kim-Chuan Toh, Michael J Todd, and Reha H Tütüncü. Sdpt3 - a matlab software package for semidefinite programming, version 1.3. *Optimization methods and software*, 11(1-4):545–581, 1999.