Solving Convex QCQP Mean-Variance Portfolio Optimization using CVXOPT

Ariah Klages-Mundt

October 17, 2018

Note: this builds on the following references, making some corrections and clarifying/simplifying some explanations: `https://rexyroo.github.io/Articles/2014/03/15/flipped_markowitz/` `https://pwnetics.wordpress.com/2010/12/18/second-order-cone-programming-with-cvxopt/` This also provides code for translating arbitrary standard form SOCPs into the form used in the Python CVXOPT solver.

Second-Order Cone Programs Standard form for a Second-Order Cone Program (SOCP) is

$$\begin{aligned} \min \quad & \mathbf{f}^T \mathbf{x} \\ \text{s.t.} \quad & \|A_i \mathbf{x} + \mathbf{b}_i\|_2 \leq \mathbf{c}_i^T \mathbf{x} + d_i \qquad 0 = 1, \dots, m \\ & F\mathbf{x} = \mathbf{g} \end{aligned}$$

Python's CVXOPT SOCP solver uses the alternative form

$$\begin{aligned} \min \quad & \mathbf{c}^T \mathbf{x} \\ \text{s.t.} \quad & G_k \mathbf{x} + \mathbf{s}_k \leq \mathbf{h}_k \qquad k = 0, \dots, M \\ & A\mathbf{x} = \mathbf{b} \\ & \mathbf{s_0} \geq 0 \\ & s_{k0} \geq \|\mathbf{s_{k1}}\|_2 \end{aligned}$$

The constraint $s_0 \geq 0$ is interpreted component-wise. The final constraint refers to the first entry of s_k as s_{k0} and the remaining part as s_{k1}. Notice that the s_k along with x are the decision variables.

Each constraint $\|A_i x + b_i\|_2 \leq c_i^T x + d_i$ is equivalent to

$$-\begin{bmatrix} \mathbf{c_i}^T \\ A_i \end{bmatrix} \mathbf{x} + \begin{bmatrix} s_{i0} \\ \mathbf{s_{i1}} \end{bmatrix} = \begin{bmatrix} d_i \\ \mathbf{b_i} \end{bmatrix}, \qquad s_{i0} \geq \|\mathbf{s_{i1}}\|_2.$$

Notice that s_i is like slack variables that make this equation true. We can transform the standard SOCP form into the CVXOPT form by defining

$$G_i = -\begin{bmatrix} \mathbf{c_i}^T \\ A_i \end{bmatrix}, \qquad \mathbf{h_i} = \begin{bmatrix} d_i \\ \mathbf{b_i} \end{bmatrix}.$$

Functions in the .py code file in this repo perform this transformation.

QCQP Mean-Variance Optimization The following mean-variance optimization on n assets is a quadratically constrained quadratic program (QCQP)

$$\begin{aligned} \max \quad & \mathbf{r}^T\mathbf{x} \\ \text{s.t.} \quad & \mathbf{x}^T\Sigma\mathbf{x} \leq \sigma^2_{\text{target}} \\ & \mathbf{e}^T\mathbf{x} = 1 \\ & \mathbf{x} \geq 0, \end{aligned}$$

where r is the expected returns vector, Σ is the covariance matrix, σ^2_{target} is the target maximum variance of the optimized portfolio, e is the all ones vector, and x is the vector of optimal asset weights that we want to find.

Since the covariance matrix Σ is convex, this is a convex QCQP, which can be translated into a SOCP and solved efficiently.

To translate the mean-variance optimization into the standard SOCP form, we can take

1. $\mathbf{f} = -\mathbf{r}$ (because we want to maximize)
2. $F = \mathbf{e}^T$ and $g = 1$
3. $A_0 = L^T$ where $\Sigma = LL^T$ is a Cholesky factorization, $\mathbf{b_0} = \mathbf{c_0} = \mathbf{0}$, and $d_0 = \sigma_{\text{target}}$
4. For $i = 1, \ldots, n$, $A_i = \mathbf{0}^T$, $b_i = 0$, $c_i = \mathbf{e_i}$ indicating the ith component, and $d_i = 0$.

Item (4) captures the maximum variance constraint. In particular, we have

$$\|A_0\mathbf{x}\|_2^2 = (A_0\mathbf{x})^T A_0\mathbf{x} = \mathbf{x}^T\Sigma\mathbf{x}$$

if $A_0^T A_0 = \Sigma$. Item (5) captures the $\mathbf{x} \geq 0$ constraint.

Another function in the .py file in this repo transforms the mean-variance QCQP into a SOCP that is solved with CVXOPT.