Robust Quadratic Programming

Nam Ho-Nguyen

There is a summary for computing subgradients of robust quadratic constraints at the bottom if you want to skip the details.

1 Robust Portfolio Optimization

Given n assets, let:

- $r \in \mathbb{R}^n$ be a vector of returns for each asset.
- $\mu \in \mathbb{R}^n$ be a vector of mean returns for each asset.
- $f \in \mathbb{R}^m$ be factors for the market.
- $V \in \mathbb{R}^{m\times n}$ be a matrix of factor loadings.
- $\epsilon \in \mathbb{R}^m$ be a vector of residual errors for each asset.

The factor model specifies that

$$r = \mu + V^\top f + \epsilon.$$

In addition, we assume that f and ϵ are independently distributed, with

- $\mathbb{E}[f] = \mathbf{0}_m$, $\mathrm{Cov}(f) = \mathbb{E}[ff^\top] = F$ for a given positive definite $F \in \mathbb{R}^{m\times m}$.
- $\mathbb{E}[\epsilon] = \mathbf{0}_n$, $\mathrm{Cov}(\epsilon) = \mathbb{E}[\epsilon\epsilon^\top] = D$ for a given positive definite diagonal $D \in \mathbb{R}^{n\times n}$.

Then $\mathbb{E}[r] = \mu$, $\mathrm{Cov}(r) = \mathbb{E}[(r-\mu)(r-\mu)^\top] = V^\top FV + D$. Let $x \in \mathbb{R}^n$ be an allocation of investments in each asset. We allow shorting, so we need not restrict x to be non-negative, but we do need our total investments not to exceed our wealth, and after normalising by initial wealth, we need the constraint $\mathbf{1}_n^\top x = 1$. Our return from x is $r^\top x$. We can easily check that

$$\mathbb{E}[r^\top x] = \mu^\top x, \quad \mathrm{Cov}(r^\top x) = \mathbb{E}[((r-\mu)^\top x)^2] = x^\top(V^\top FV + D)x.$$

The Markowitz mean-variance portfolio is the one that minimizes the variance (or risk) while maximizing the mean. In practice, we don't know μ and V a priori, so we estimate them via linear regression from past data on r and f to obtain μ_0 and V_0. The portfolio optimization problem is

$$\begin{aligned}
\min \quad & b + c - \lambda a \\
\text{s.t.} \quad & \mu_0^\top x \geq a \\
& x^\top(V_0^\top F V_0)x \leq b \\
& x^\top D x \leq c \\
& \mathbf{1}_n^\top x = 1.
\end{aligned}$$

Here, $\lambda \geq 0$ is a fixed parameter that trades off minimizing risk and maximizing return.

Since we learn μ_0, V_0 from a regression, they are uncertain. We use robust optimization to immunize our solutions against this uncertainty. Specifically, we will define two uncertainty sets U_μ, U_V and instead solve

$$\begin{aligned} \min \quad & b + c - \lambda a \\ \text{s.t.} \quad & \min_{\tilde{\mu} \in U_\mu} \tilde{\mu}^\top x \geq a \\ & \max_{\tilde{V} \in U_V} x^\top (\tilde{V}^\top F \tilde{V}) x \leq b \\ & x^\top D x \leq c \\ & \mathbf{1}_n^\top x = 1. \end{aligned}$$

For μ_0, we define

$$U_\mu = \{\mu_0 + u : u \in \mathbb{R}^n,\ |u_i| \leq \gamma_i,\ i \in [n]\}$$

where γ_i are fixed constants defined from the regression output. Note that

$$\min_{\tilde{\mu} \in U_\mu} \tilde{\mu}^\top x \geq \alpha \iff \mu_0^\top x - \sum_{i \in [n]} \gamma_i |x_i| \geq \alpha.$$

This can be recast as a linear system

$$\begin{aligned} \mu_0^\top x \geq \alpha + \sum_{i \in [n]} \gamma_i w_i \\ w_i \geq x_i, \quad i \in [n] \\ w_i \geq -x_i, \quad i \in [n]. \end{aligned}$$

For V_0, we will define the uncertainty set

$$U_V = \left\{ V_0 + \sum_{j \in [k]} P_j u_j : u \in \mathbb{R}^k,\ \|u\|_2 \leq 1 \right\}$$

where $P_j \in \mathbb{R}^{m \times n}$ are fixed matrices.

2 Robust SVM

We are given data $\{x_i, y_i\}_{i \in [m]}$ where $x_i \in \mathbb{R}^n$ and $y_i \in \{\pm 1\}$. We wish to learn a classifier $c(x) = \text{sign}(\langle w, x \rangle + b)$. The parameters w, b will be learnt from data by minimizing the hinge loss $h(r) = \max\{1 - r, 0\}$ between points on the hyperplane $\langle w, x_i \rangle + b$ and labels y_i, specifically, we want to solve

$$\begin{aligned} \min_{a,w,b} \quad & \sum_{i \in [m]} a_i \\ \text{s.t.} \quad & a_i \geq 1 - y_i(\langle w, x_i \rangle + b), \quad i \in [m] \\ & a_i \geq 0, \quad i \in [m]. \end{aligned}$$

We also often add regularization on w to ensure well-posedness of the problem:

$$\begin{aligned}
\min_{a,w,b} \quad & C \sum_{i \in [m]} a_i + \frac{1}{2}\|w\|_2 \\
\text{s.t.} \quad & a_i \geq 1 - y_i(\langle w, x_i \rangle + b), \quad i \in [m] \\
& a_i \geq 0, \quad i \in [m],
\end{aligned}$$

where $C > 0$ is the regularization parameter. We rewrite this as as a second-order cone program:

$$\begin{aligned}
\min_{a,w,b} \quad & C \sum_{i \in [m]} a_i + \frac{1}{2}t \\
\text{s.t.} \quad & a_i \geq 1 - y_i(\langle w, x_i \rangle + b), \quad i \in [m] \\
& a_i \geq 0, \quad i \in [m] \\
& 1 + t \geq \sqrt{4\|w\|_2^2 + (1-t)^2}.
\end{aligned}$$

After some straightforward reasoning, the dual of this SOCP is equivalent to

$$\begin{aligned}
\min_{\gamma,\alpha} \quad & 2\alpha - \sum_{i \in [m]} \gamma_i \\
\text{s.t.} \quad & 0 \leq \gamma_i \leq C, \quad i \in [m] \\
& \sum_{i \in [m]} \gamma_i y_i = 0 \\
& \gamma^\top Y^\top X^\top X Y \gamma \leq \alpha,
\end{aligned}$$

where $X = \begin{bmatrix} x_1 & \dots & x_m \end{bmatrix} \in \mathbb{R}^{n \times m}$, and $Y = \mathrm{Diag}(\{y_i\}_{i \in [m]}) \in \mathbb{R}^{m \times m}$.

If the data points x_i are uncertain, then we could observe $x_{0,i}$ instead of x_i. Furthermore, our true data matrix X is now $X = X_0 + W$. Since we don't know what W will be, we use a robust constraint where we optimize for all $X \in U_X$. We define the uncertainty set similar to before:

$$U_X = \left\{ X_0 + \sum_{j \in [k]} P_j u_j : u \in \mathbb{R}^k, \ \|u\|_2 \leq 1 \right\},$$

where $P_j \in \mathbb{R}^{n \times m}$ are fixed matrices. The dual problem for SVM becomes

$$\begin{aligned}
\min_{\gamma,\alpha} \quad & 2\alpha - \sum_{i \in [m]} \gamma_i \\
\text{s.t.} \quad & 0 \leq \gamma_i \leq C, \quad i \in [m] \\
& \sum_{i \in [m]} \gamma_i y_i = 0 \\
& \max_{\|u\|_2 \leq 1} \gamma^\top Y^\top \left(X_0 + \sum_{j \in [k]} P_j u_j \right)^\top \left(X_0 + \sum_{j \in [k]} P_j u_j \right) Y \gamma \leq \alpha.
\end{aligned}$$

This robust quadratic constraint is of the same form as the ones from the robust portfolio optimization problem. We examine how to process these next.

3 Robust Convex Quadratic Constraints

We now examine a certain type of uncertainty on convex quadratic constraints of the form

$$\|Ax\|_2^2 = x^\top A^\top Ax \le c,$$

where $x \in \mathbb{R}^n$, $A \in \mathbb{R}^{m\times n}$, and $c \in \mathbb{R}$. We can also include a linear term $b^\top x$ but this will not affect things much, so for simplicity we exclude it. The uncertainty we consider is on the A matrix, and takes the form

$$U_A = \left\{ A_0 + \sum_{j\in[k]} P_j u_j : u \in \mathbb{R}^k,\ \|u\|_2 \le 1 \right\},$$

for fixed matrices $P_j \in \mathbb{R}^{m\times n}$, $j \in [k]$. For $A \in U_A$, rewrite

$$\begin{aligned}
x^\top A^\top Ax &= x^\top \left(A_0 + \sum_{j\in[k]} P_j u_j\right)^\top \left(A_0 + \sum_{j\in[k]} P_j u_j\right) x \\
&= x^\top A_0^\top A_0 x + 2\sum_{j\in[k]} (x^\top A_0^\top P_j x)u_j + x^\top \left(\sum_{j\in[k]} P_j u_j\right)^\top \left(\sum_{j\in[k]} P_j u_j\right) x \\
&= x^\top A_0^\top A_0 x + 2\sum_{j\in[k]} (x^\top A_0^\top P_j x)u_j + \sum_{j,j'\in[k]} \left(x^\top P_j^\top P_{j'} x\right) u_j u_{j'}.
\end{aligned}$$

We know that this is convex in x since $A^\top A$ is always positive semidefinite. In fact, it is also convex in u. To see this, let $Y(x) \in \mathbb{R}^{k\times k}$ be the matrix with entries $Y(x)_{jj'} = x^\top P_j^\top P_{j'} x$. Then in fact $Y(x)$ is positive semidefinite, since we can write

$$Y(x) = \begin{bmatrix} x^\top P_1^\top \\ \vdots \\ x^\top P_k^\top \end{bmatrix} \underbrace{\begin{bmatrix} P_1 x & \dots & P_k x \end{bmatrix}}_{=Y'(x)} = Y'(x)^\top Y'(x).$$

We can now write

$$\begin{aligned}
&x^\top \left(A_0 + \sum_{j\in[k]} P_j u_j\right)^\top \left(A_0 + \sum_{j\in[k]} P_j u_j\right) x \\
&= x^\top A_0^\top A_0 x + 2\sum_{j\in[k]} (x^\top A_0^\top P_j x)u_j + u^\top Y(x) u.
\end{aligned}$$

The first term is constant in u, the second term is linear, and the third term is a convex quadratic in u. To satisfy the robust constraint, we need

$$\max_{A\in U_A} x^\top A^\top Ax = \max_{\|u\|_2\le 1} \left\{ x^\top A_0^\top A_0 x + 2\sum_{j\in[k]} (x^\top A_0^\top P_j x)u_j + u^\top Y(x)u \right\} \le c.$$

Thus, a pessimization oracle for this constraint needs to maximize a convex quadratic in u, which is a trust region subproblem (TRS). We can make the TRS objective concave by shifting the $Y(x)$ matrix (letting I_k be the $k \times k$ identity matrix):

$$\begin{aligned}
&\max_{\|u\|_2 \le 1} \left\{ x^\top A_0^\top A_0 x + 2 \sum_{j \in [k]} (x^\top A_0^\top P_j x) u_j + u^\top Y(x) u \right\} \\
&= \max_{\|u\|_2 \le 1} \left\{ x^\top A_0^\top A_0 x + 2 \sum_{j \in [k]} (x^\top A_0^\top P_j x) u_j + u^\top (Y(x) - \lambda_{\max}(Y(x)) I_k) u + \lambda_{\max}(Y(x)) \right\} \\
&= \max_{\|u\|_2 \le 1} \left\{ x^\top A_0^\top A_0 x + 2 \sum_{j \in [k]} (x^\top A_0^\top P_j x) u_j + u^\top Y(x) u + \lambda_{\max}(Y(x))(1 - \|u\|_2^2) \right\}.
\end{aligned}$$

For a pessimization oracle, we can just input this into Gurobi, since the new objective is a concave quadratic. Note that this new quadratic is still convex in x, as we will explain later.

Let us now examine this new objective within a subgradient scheme. Define

$$f(x,u) = x^\top A_0^\top A_0 x + 2 \sum_{j \in [k]} (x^\top A_0^\top P_j x) u_j + u^\top Y(x) u + \lambda_{\max}(Y(x))(1 - \|u\|_2^2).$$

We are interested in $\nabla_x f(x,u)$ and $\nabla_u f(x,u)$. The gradient in u is straightforward:

$$\nabla_u f(x,u) = 2\{x^\top A_0^\top P_j x\}_{j \in [k]} + 2Y(x)u - 2\lambda_{\max}(Y(x))u.$$

The gradient in x is a bit more complicated. It helps to rewrite $f(x)$ using the original objective:

$$f(x,u) = x^\top \left(A_0 + \sum_{j \in [k]} P_j u_j \right)^\top \left(A_0 + \sum_{j \in [k]} P_j u_j \right) x + + \lambda_{\max}(Y(x))(1 - \|u\|_2^2).$$

Then

$$\nabla_x f(x,u) = 2 \left(A_0 + \sum_{j \in [k]} P_j u_j \right)^\top \left(A_0 + \sum_{j \in [k]} P_j u_j \right) x + (1 - \|u\|_2^2) \nabla_x \lambda_{\max}(Y(x)).$$

The term we need to investigate further is $\nabla_x \lambda_{\max}(Y(x))$. To do this, we rewrite

$$\begin{aligned}
\lambda_{\max}(Y(x)) = \max_{\|v\|_2 \le 1} v^\top Y(x) v &= \max_{\|v\|_2 \le 1} \sum_{j,j' \in [k]} \left(x^\top P_j^\top P_{j'} x \right) v_j v_{j'} \\
&= \max_{\|v\|_2 \le 1} x^\top \left(\sum_{j \in [k]} P_j v_j \right)^\top \left(\sum_{j \in [k]} P_j v_j \right) x
\end{aligned}$$

In fact, this shows that $f(x,u)$ is convex in x: the first part $x^\top \left(A_0 + \sum_{j \in [k]} P_j u_j \right)^\top \left(A_0 + \sum_{j \in [k]} P_j u_j \right) x$ is a convex quadratic, while the second part $\lambda_{\max}(Y(x))(1 - \|u\|_2^2)$ is a maximum over convex quadratic functions, which is convex. To get the (sub)gradient of the max term, we need to compute the maximizing v, which is simply the leading eigenvector of $Y(x)$, denote this by $\bar{v}$. Then we simply take the gradient of $\bar{v}^\top Y(x)\bar{v}$, thus:

$$\nabla_x \lambda_{\max}(Y(x)) = 2\left(\sum_{j\in[k]} P_j \bar{v}_j\right)^\top \left(\sum_{j\in[k]} P_j \bar{v}_j\right) x.$$

Putting this together, we have

$$\nabla_x f(x,u) = 2\left(A_0 + \sum_{j\in[k]} P_j u_j\right)^\top \left(A_0 + \sum_{j\in[k]} P_j u_j\right) x + 2(1-\|u\|_2^2)\left(\sum_{j\in[k]} P_j \bar{v}_j\right)^\top \left(\sum_{j\in[k]} P_j \bar{v}_j\right) x.$$

Note that if $\|u\|_2 = 1$ (which may often be the case for a subgradient scheme), we need not compute the second term.

3.1 Summary

Given x, u, and *fixed* data $A_0, P_j \in \mathbb{R}^{k\times n}$, we have a robust constraint

$$\max_{\|u\|_2\le 1}\left\{x^\top \left(A_0 + \sum_{j\in[k]} P_j u_j\right)^\top \left(A_0 + \sum_{j\in[k]} P_j u_j\right) x\right\} \le c.$$

Defining $Y(x) = \{x^\top P_j^\top P_{j'} x\}_{j,j'\in[k]} \in \mathbb{R}^{k\times k}$, the max term can be transformed into

$$\begin{aligned}
&\max_{\|u\|_2\le 1}\left\{x^\top \left(A_0 + \sum_{j\in[k]} P_j u_j\right)^\top \left(A_0 + \sum_{j\in[k]} P_j u_j\right) x\right\}\\
&= \max_{\|u\|_2\le 1}\left\{x^\top A_0^\top A_0 x + 2\sum_{j\in[k]} (x^\top A_0^\top P_j x) u_j + u^\top Y(x) u\right\}\\
&= \max_{\|u\|_2\le 1}\left\{x^\top A_0^\top A_0 x + 2\sum_{j\in[k]} (x^\top A_0^\top P_j x) u_j + u^\top Y(x) u + \lambda_{\max}(Y(x))(1-\|u\|_2^2)\right\}\\
&= \max_{\|u\|_2\le 1}\left\{x^\top \left(A_0 + \sum_{j\in[k]} P_j u_j\right)^\top \left(A_0 + \sum_{j\in[k]} P_j u_j\right) x + \lambda_{\max}(Y(x))(1-\|u\|_2^2)\right\}.
\end{aligned}$$

3.1.1 Subgradients (Ben-Tal et. al. and our approach)

Here is a recipe for computing the gradients. Given $x \in \mathbb{R}^n, u \in \mathbb{R}^k$:

- Compute $Y(x) = \{x^\top P_j^\top P_{j'} x\}_{j,j'\in[k]} \in \mathbb{R}^{k\times k}$.
- Compute $\lambda_{\max}(Y(x))$ and associated leading eigenvector $\bar{v}(x)$.
- Compute $b(x) = \{x^\top A_0^\top P_j x\}_{j\in[k]} \in \mathbb{R}^k$.

- Compute
$$\nabla_u f(x,u) = 2b(x) + 2Y(x)u - 2\lambda_{\max}(Y(x))u.$$
- Compute $A(u) = \left(A_0 + \sum_{j\in[k]} P_j u_j\right)^\top \left(A_0 + \sum_{j\in[k]} P_j u_j\right) \in \mathbb{R}^{n\times n}$.
- If $\|u\|_2 = 1$:
  - Compute
  $$\nabla_x f(x,u) = 2A(u)x.$$

  Else:
  - Compute $P(\bar{v}(x)) = \left(\sum_{j\in[k]} P_j \bar{v}(x)_j\right)^\top \left(\sum_{j\in[k]} P_j \bar{v}(x)_j\right) \in \mathbb{R}^{n\times n}$.
  - Compute
  $$\nabla_x f(x,u) = 2A(u)x + 2(1-\|u\|_2^2)P(\bar{v}(x))x.$$

3.1.2 Pessimization (Mutapcic and Boyd)

For a pessimization oracle approach, we solve

$$\max_{\|u\|_2\le 1} \left\{ x^\top A_0^\top A_0 x + 2\sum_{j\in[k]} (x^\top A_0^\top P_j x)u_j + u^\top (Y(x) - \lambda_{\max}(Y(x))I_k)u + \lambda_{\max}(Y(x)) \right\}.$$

To input this into Gurobi for fixed $\bar{x}$, we can just pass the expression

$$\max_{\|u\|_2\le 1} \left\{ 2b(\bar{x})^\top u + u^\top (Y(\bar{x}) - \lambda_{\max}(Y(\bar{x}))I_k)u \right\}.$$

with u variables. Then, once we have a solution $\bar{u}$, we need to 'send it to the boundary' by adding multiples of $\bar{v}(\bar{x})$ (the leading eigenvector of $Y(\bar{x})$) until $\|\bar{u} + \alpha\bar{v}(\bar{x})\| = 1$. This ensures that $\bar{u} + \alpha\bar{v}(\bar{x})$ is an optimal solution to the noncpncave quadratic as well as the concave relaxation. We then update $\bar{u} := \bar{u} + \alpha\bar{v}(\bar{x})$.

Then, if we want to solve a nominal program for given then $\bar{u}$, we need only solve with constraints

$$x^\top A(\bar{u})x \le c.$$

In other words, eo *don't need to worry about the* $\lambda_{\max}(Y(x))$ *part*, since we made $\|\bar{u}\|_2 = 1$.