Portfolio Allocation
From QP to ML Optimization Algorithms

Thierry Roncalli*

*Amundi Asset Management¹, France

This version: February 25th, 2019

¹The opinions expressed in this presentation are those of the authors and are not meant to represent the opinions or official positions of Amundi Asset Management. This presentation is based on joint works with Edmond Lezmi, Thibault Bourgeron, Joan Gonzalvez, Jean-Charles Richard and Jiali Xu.
The Markowitz optimization problem

- $x = (x_1, \ldots, x_n)$ is the vector of weights in the portfolio
- $\mu = \mathbb{E}[R]$ and $\Sigma = \mathbb{E} \left[(R - \mu)(R - \mu)^\top\right]$ are the vector of expected returns and the covariance matrix of asset returns
- We note $\mu(x) = x^\top \mu$ the expected return of the portfolio and $\sigma(x) = \sqrt{x^\top \Sigma x}$ the portfolio volatility

Asset allocation problems (Markowitz, 1952)

1. $\sigma$-problem:
   $\max \mu(x) \quad \text{s.t.} \quad \sigma(x) \leq \sigma^*$

2. $\mu$-problem:
   $\min \sigma(x) \quad \text{s.t.} \quad \mu(x) \geq \mu^*$
The Markowitz solution problem

**QP trick (Markowitz, 1952 and 1956)**

Transform the previous problems into a QP problem:

\[
x^*(\gamma) = \arg\min_x \frac{1}{2} x^T \Sigma x - \gamma x^T \mu
\]

s.t. \(1_n^T x = 1\)

Solving \(\sigma\)- and \(\mu\)-problems are equivalent to QP + bisection algorithm

Primal QP problem

Definition

A quadratic programming (QP) problem is an optimization problem with a quadratic objective function and linear inequality constraints:

\[
x^* = \arg\min_{x} \frac{1}{2} x^T Q x - x^T R
\]

s.t. \( S x \leq T \)

where \( x \) is a \( n \times 1 \) vector, \( Q \) is a \( n \times n \) matrix and \( R \) is a \( n \times 1 \) vector.

We have

\[
S x \leq T \iff \begin{cases}
A x = B \\
C x \leq D \\
x_{\min} \leq x \leq x_{\max}
\end{cases}
\]

because:

\[
A x = B \iff \begin{cases}
A x \geq B \\
A x \leq B
\end{cases}
\]
Constrained ordinary least squares

\[ \hat{\beta}^{\text{ols}} = \arg \min_\beta \frac{1}{2} \text{RSS} (\beta) \]

where:

\[
\text{RSS} (\beta) = \left( Y - X \beta \right)^T \left( Y - X \beta \right) \\
= Y^T Y + \beta^T \left( X^T X \right) \beta - 2 \beta^T \left( X^T Y \right)
\]

We deduce that:

\[ \hat{\beta}^{\text{ols}} = \arg \min_\beta \frac{1}{2} \beta^T Q \beta - \beta^T R \]

s.t. \[
\begin{cases}
A \beta = B \\
C \beta \leq D \\
\beta_{\text{min}} \leq \beta \leq \beta_{\text{max}}
\end{cases}
\]

where \( Q = X^T X \) and \( R = X^T Y \)
Linear regression:

\[ Y = X\beta + \varepsilon \]

The solution is equal to:

\[ \hat{\beta}_{\text{ols}} = \left( X^\top X \right)^{-1} X^\top Y \]

Markowitz optimization with empirical covariance matrix \( \hat{\Sigma} \) and empirical expected returns \( \hat{\mu} \):

\[ \gamma_{1n} = Rx + \varepsilon \]

where \( R \) is the matrix of (centered) asset returns (number of observations \( \times \) number of assets). The solution is equal to:

\[ \hat{x}^{\text{mvo}} = \left( R^\top R \right)^{-1} R^\top \gamma_{1n} \]

\[ = \gamma \hat{\Sigma}^{-1} \hat{\mu} \]
Portfolio optimization with a benchmark

Let \( \mu(x | b) = (x - b)^\top \mu \) be the expected excess return and
\( \sigma(x | b) = \sqrt{(x - b)^\top \Sigma (x - b)} \) be the tracking error volatility, where \( b \) is the benchmark.

The objective function is:

\[
\begin{align*}
    f(x | b) &= \frac{1}{2} (x - b)^\top \Sigma (x - b) - \gamma (x - b)^\top \mu \\
    &\propto \frac{1}{2} x^\top \Sigma x - \gamma x^\top \left( \mu + \frac{1}{\gamma} \Sigma b \right)
\end{align*}
\]

\( \Rightarrow \) QP problem with \( Q = \Sigma \) and \( R = \gamma \tilde{\mu} \) where \( \tilde{\mu} = \mu + \frac{1}{\gamma} \Sigma b \) is the regularized vector of expected returns.

- Tracking error constraints \( \Leftrightarrow \) regularization of the QP problem
- If \( b \) is the risk-free asset, the regularized QP solution is the capital market line (Roncalli, 2013)
The portfolio sampling problem

We have:

\[ x^* = \arg \min_{x} \frac{1}{2} (x - b)^\top \Sigma (x - b) \]

u.c. \( \begin{align*}
\mathbf{1}_n^\top x &= 1 \\
x &\geq 0_n \\
\sum_{i=1}^n 1 \{ x_i > 0 \} &\leq n_x
\end{align*} \)

where \( b \) is the vector of index weights
We set \( x_{(0)}^{\text{max}} = 1_n \). At the iteration \( k \), we solve the QP problem by taking into account the upper bounds \( x_{(k)}^{\text{max}} \):

\[
 x_{(k)}^{*} = \arg \min x_{(k)} \quad \text{s.t.} \quad 1_n x_{(k)} = 1, \quad 0_n \leq x_{(k)} \leq x_{(k)}^{\text{max}}
\]

We then update the upper bounds \( x_{(k)}^{\text{max}} \) by deleting the stock with the lowest non-zero optimized weight.

We iterate the two steps until \( \sum_{i=1}^{n} \mathbb{1} \{ x_{(k),i}^{*} > 0 \} \leq n \).

The heuristic algorithm is the fastest method (vs backward elimination, forward selection, MIQP, etc.)
The Lagrange function is equal to:

\[ \mathcal{L}(x; \lambda) = \frac{1}{2} x^T Q x - x^T R + \lambda^T (S x - T) \]

We deduce that the dual problem problem is defined by:

\[
\lambda^* = \arg \max \left\{ \inf_x \mathcal{L}(x; \lambda) \right\}
\]

\[ \text{s.t. } \lambda \geq 0 \]

Duality theorem

We can show that the dual program is another quadratic program:

\[
\lambda^* = \arg \min \frac{1}{2} \lambda^T \bar{Q} \lambda - \lambda^T \bar{R}
\]

\[ \text{s.t. } \lambda \geq 0 \]

with \( \bar{Q} = SQ^{-1} S^T \) and \( \bar{R} = SQ^{-1} R - T \)
Support vector machines

Figure: Separating hyperplane picking

Support vector machines

Figure: Margins of separation

Support vector machines

Figure: Optimal hyperplane

Support vector machines

### Hard margin classification

Let \( y_i = \beta_0 + x_i^\top \beta \). The maximization problem is:

\[
\{ \hat{\beta}_0, \hat{\beta} \} = \arg \max M
\]

\[
\text{s.t. } \begin{cases} f(x_i) \geq M & \text{if } y_i = +1 \\ f(x_i) \leq -M & \text{if } y_i = -1 \end{cases}
\]

### Primal QP

We can show that:

\[
\{ \hat{\beta}_0, \hat{\beta} \} = \arg \min \frac{1}{2} \| \beta \|_2^2
\]

\[
\text{s.t. } y_i \left( \beta_0 + x_i^\top \beta \right) \geq 1 \quad \text{for } i = 1, \ldots, n
\]

and \( \hat{M} = 1 / \| \beta \|_2 \)
Dual QP (Chervonenkis-Cortes-Vapnik)

Let $\alpha$ be the vector of Lagrange multipliers. We have:

$$
\hat{\alpha} = \arg \min \frac{1}{2} \alpha^\top \Gamma \alpha - \alpha^\top \mathbf{1}_n
$$

s.t. \quad \begin{cases} 
    y^\top \alpha = 0 \\
    \alpha \geq \mathbf{0}_n 
\end{cases}

where $\Gamma_{i,j} = y_i y_j x_i^\top x_j$. It follows that $\hat{\beta} = \sum_{i=1}^n \hat{\alpha}_i y_i x_i$ and:

$$
\hat{\beta}_0 = \frac{\sum_{i=1}^n \mathbb{1} \{ \hat{\alpha}_i > 0 \} \cdot (y_i - x_i^\top \hat{\beta})}{\sum_{i=1}^n \mathbb{1} \{ \hat{\alpha}_i > 0 \}}
$$

We can classify new observations by considering the following rule:

$$
\hat{y} = \text{sign} \left( \hat{\beta}_0 + x^\top \hat{\beta} \right)
$$
Support vector machines

**Dimension of the problem**

- Primal QP \( \Rightarrow (m+1, n) \)
- Dual QP \( \Rightarrow (n, n+1) \)

Extension to:

- Soft margin classification (binary hinge loss, squared hinge loss, ramp loss, etc.)
- LS-SVM regression
- \( \varepsilon \)-SVM regression
- Non-linear SVM and kernel functions

**Dual QP everywhere!**
The Lasso revolution

Least absolute shrinkage and selection operator (lasso)

The lasso method consists in adding a $L_1$ penalty function to the least square problem:

$$\hat{\beta}^{\text{lasso}}(\tau) = \arg \min \frac{1}{2} (Y - X\beta)^\top (Y - X\beta)$$

s.t. \[ \|\beta\|_1 \leq \tau \]

Alternatively, we have:

$$\hat{\beta}^{\text{lasso}}(\lambda) = \arg \min \frac{1}{2} (Y - X\beta)^\top (Y - X\beta) + \lambda \|\beta\|_1$$
Lasso regression

We have:

$$RSS(\beta) = RSS(\hat{\beta}_{ols}) + (\beta - \hat{\beta}_{ols})^T X^T X (\beta - \hat{\beta}_{ols})$$

If we consider the equation $RSS(\beta) = c$, we distinguish three cases:

<table>
<thead>
<tr>
<th>$c &lt; RSS(\hat{\beta}_{ols})$</th>
<th>$c = RSS(\hat{\beta}_{ols})$</th>
<th>$c &gt; RSS(\hat{\beta}_{ols})$</th>
</tr>
</thead>
<tbody>
<tr>
<td>No solution</td>
<td>One solution $\hat{\beta}_{ols}$</td>
<td>An ellipsoid</td>
</tr>
</tbody>
</table>

What does this result become when imposing the lasso constraint $\|\beta\|_1 \leq \tau$?

**Sparsity theorem**

$$\exists \eta > 0: \forall \tau < \eta, \min \left( \left| \hat{\beta}_{j \text{lasso}}(\tau) \right| \right) = 0$$
The Lasso regression

\[ |\beta_1| + |\beta_2| \leq \tau \]

\[ |\beta_1| + |\beta_2| \leq \eta \]

\[ \text{RSS} (\beta_1, \beta_2) = \text{constant} \]

Figure: Interpretation of the lasso regression

Figure: Variable selection with the lasso regression


Lasso ordering: $x_3 \succ x_1 \succ x_2 \succ x_4 \succ x_5$
Factor selection in the stock market

Figure: Lasso selection (North America, 2014 – 2017)

- Quality $\succ$ ESG $\succ$ Momentum $\succ$ Value $\succ$ Low-volatility

- The ESG-Value correlation puzzle!

Source: Bennani et al. (2018).
Factor selection in the stock market

Figure: Lasso selection (Eurozone, 2014 – 2017)

- ESG $\succ$ Value $\succ$ Momentum $\succ$ Quality $\succ$ Low-volatility
- The ESG-Quality correlation puzzle!

Source: Bennani et al. (2018).
Solving the lasso regression problem

We introduce the parametrization:

$$\beta = \beta^+ - \beta^-$$

under the constraints $\beta^+ \geq 0_n$ and $\beta^- \geq 0_n$. We deduce that:

$$\|\beta\|_1 = \sum_{j=1}^{m} |\beta_j^+ - \beta_j^-| = \sum_{j=1}^{m} |\beta_j^+| + \sum_{j=1}^{m} |\beta_j^-| = 1^T \beta^+ + 1^T \beta^-$$

Since we have:

$$\beta = \begin{pmatrix} l_m & -l_m \end{pmatrix} \begin{pmatrix} \beta^+ \\ \beta^- \end{pmatrix}$$

the augmented QP program is specified as follows:

$$\hat{\theta} = \arg \min \frac{1}{2} \theta^T Q \theta - \theta^T R$$

s.t. $\theta \geq 0_{2m}$

where $\theta = (\beta^+, \beta^-)$, $\tilde{X} = \begin{pmatrix} X & -X \end{pmatrix}$, $Q = \tilde{X}^T \tilde{X}$ and $R = \tilde{X}^T Y + \lambda 1_{2m}$. If we denote $A = \begin{pmatrix} l_m & -l_m \end{pmatrix}$, we obtain $\hat{\beta}^{\text{lasso}} (\lambda) = A \hat{\theta}$
If we consider the $\tau$-problem, we obtain another augmented QP program:

$$
\hat{\theta} = \arg\min \frac{1}{2} \theta^\top Q \theta - \theta^\top R \\
\text{s.t.} \begin{cases} 
C \theta \geq D \\
\theta \geq 0_{2m}
\end{cases}
$$

where $Q = \tilde{X}^\top \tilde{X}$, $R = \tilde{X}^\top Y$, $C = -1_{2m}^\top$ and $D = -\tau$. Again, we have $\hat{\beta}(\tau) = A\hat{\theta}$.
Long-only MVO portfolios with a turnover constraint

The optimization problem becomes:

\[ x^* = \arg \min_{x} \frac{1}{2} x^\top \Sigma x - \gamma x^\top \mu \]

subject to:

\[ \sum_{i=1}^{n} x_i = 1 \]

\[ \sum_{i=1}^{n} |x_i - x_0^i| \leq \tau^+ \]

\[ 0 \leq x_i \leq 1 \]

where \( \tau^+ \) is the maximum turnover with respect to Portfolio \( x^0 \).
Scherer (2007) introduces the additional variables $x_i^-$ and $x_i^+$ such that:

$$x_i = x_i^0 + x_i^+ - x_i^-$$

with $x_i^- \geq 0$ and $x_i^+ \geq 0$. $x_i^+$ indicates then a positive weight change with respect to the initial weight $x_i^0$ whereas $x_i^-$ indicates a negative weight change. The expression of the turnover becomes:

$$\sum_{i=1}^{n} |x_i - x_i^0| = \sum_{i=1}^{n} |x_i^+ - x_i^-| = \sum_{i=1}^{n} x_i^+ + \sum_{i=1}^{n} x_i^-$$

because one of the variables $x_i^+$ or $x_i^-$ is necessarily equal to zero.
The $\gamma$-problem of Markowitz becomes

$$x^* = \arg\min \frac{1}{2} x^\top \Sigma x - \gamma x^\top \mu$$

subject to

$$\sum_{i=1}^n x_i = 1$$
$$x_i = x_i^0 + x_i^+ - x_i^-$$
$$\sum_{i=1}^n x_i^+ + \sum_{i=1}^n x_i^- \leq \tau^+$$
$$0 \leq x_i \leq 1$$
$$0 \leq x_i^- \leq 1$$
$$0 \leq x_i^+ \leq 1$$
We obtain an augmented QP problem of dimension 3n:

\[
X^* = \arg \min \frac{1}{2} X^T Q X - X^T R \\
\text{s.t.} \begin{cases} 
AX = B \\
CX \geq D \\
0_{3n} \leq X \leq 1_{3n}
\end{cases}
\]

where:

\[
X = (x_1, \ldots, x_n, x_1^-, \ldots, x_n^-, x_1^+, \ldots, x_n^+)
\]

\[
Q = \begin{pmatrix}
\Sigma & 0_{n \times n} & 0_{n \times n} \\
0_{n \times n} & 0_{n \times n} & 0_{n \times n} \\
0_{n \times n} & 0_{n \times n} & 0_{n \times n}
\end{pmatrix}, \quad R = \begin{pmatrix}
\mu \\
0_n \\
0_n
\end{pmatrix}, \quad A = \begin{pmatrix}
1_n^T & 0_n^T & 0_n^T \\
0_n & l_n & -l_n
\end{pmatrix}
\]

\[
B = \begin{pmatrix}
1 \\
x^0
\end{pmatrix}, \quad C = \begin{pmatrix}
0_n^T & -1_n^T & -1_n^T
\end{pmatrix} \text{ and } D = -\tau^+
\]
Let $c_i^-$ and $c_i^+$ be the bid and ask transactions costs. The $\gamma$-problem of Markowitz becomes:

\[
x^* = \arg \min \frac{1}{2} x^\top \Sigma x - \gamma (\sum x_i \mu_i - \sum x_i^- c_i^- - \sum x_i^+ c_i^+ )
\]

\[
\begin{align*}
\sum x_i + \sum x_i^- c_i^- + \sum x_i^+ c_i^+ &= 1 \\
x_i = x_i^0 + x_i^+ - x_i^- \\
0 \leq x_i \leq 1 \\
0 \leq x_i^- \leq 1 \\
0 \leq x_i^+ \leq 1
\end{align*}
\]

u.c.
We obtain an augmented QP problem of dimension $3n$:

$$X^* = \arg\min X^\top QX - X^\top R$$

subject to:

$$AX = B$$
$$CX \geq D$$
$$0_{3n} \leq X \leq 1_{3n}$$

where:

$$X = (x_1, \ldots, x_n, x_{1}^-, \ldots, x_n^-, x_{1}^+, \ldots, x_n^+)$$

$$Q = \begin{pmatrix}
\Sigma & 0_{n \times n} & 0_{n \times n} \\
0_{n \times n} & 0_{n \times n} & 0_{n \times n} \\
0_{n \times n} & 0_{n \times n} & 0_{n \times n}
\end{pmatrix}, \quad R = \begin{pmatrix}
\mu \\
-c^- \\
-c^+
\end{pmatrix},$$

$$A = \begin{pmatrix}
1_n^\top \\
\mathbf{1}_n \\
\mathbf{1}_n \\
-\mathbf{1}_n
\end{pmatrix} (c^-)^\top \quad (c^+)^\top$$

and $B = \begin{pmatrix}1 \\ x^0 \end{pmatrix}$.
The fall and the rise of the steepest-descent method

In the 1980s:
- Conjugate gradient methods (Fletcher–Reeves, Polak–Ribiere, etc.)
- Quasi-Newton methods (NR, BFGS, DFP, etc.)

In the 1990s:
- Neural networks
- Learning rules: Descent, Momentum/Nesterov and Adaptive learning methods

In the 2000s:
- Gradient descent: Batch gradient descent (BGD), Stochastic gradient descent (SGD), Mini-batch gradient descent (MGD)
- Coordinate descent: Cyclical coordinate descent (CCD), Random coordinate descent (RCD)
Numerical optimization

Machine learning problems

- Non-smooth objective function
- Non-unique solution
- Large-scale dimension

Optimization in machine learning requires to reinvent numerical optimization
Descent method

The descent algorithm is defined by the following rule:

\[
x^{(k+1)} = x^{(k)} + \Delta x^{(k)} = x^{(k)} - \eta D^{(k)}
\]

At the \(k\)th iteration, the current solution \(x^{(k)}\) is updated by going in the opposite direction to \(D^{(k)}\) (generally, we set \(D^{(k)} = \partial_x f(x^{(k)})\))

Coordinate descent method

Coordinate descent is a modification of the descent algorithm by minimizing the function along one coordinate at each step:

\[
x_i^{(k+1)} = x_i^{(k)} + \Delta x_i^{(k)} = x_i^{(k)} - \eta D_i^{(k)}
\]

\(\Rightarrow\) The coordinate descent algorithm becomes a scalar problem
Cyclical coordinate descent (CCD)

Choice of the variable $i$

- Random coordinate descent (RCD)
  We assign a random number between 1 and $n$ to the index $i$ (Nesterov, 2012)

- Cyclical coordinate descent (CCD)
  We cyclically iterate through the coordinates (Tseng, 2001):

$$x_i^{(k+1)} = \arg \min_x \left( x_1^{(k+1)}, \ldots, x_{i-1}^{(k+1)}, x, x_{i+1}^{(k)}, \ldots, x_n^{(k)} \right)$$
If we consider the following function:

\[
f(x_1, x_2, x_3) = (x_1 - 1)^2 + x_2^2 - x_2 + (x_3 - 2)^4 e^{x_1-x_2+3}
\]

the CCD algorithm is defined by the following iterations:

\[
\begin{align*}
x_1^{(k+1)} &= x_1^{(k)} - \eta \left( 2 (x_1^{(k)} - 1) + (x_3^{(k)} - 2)^4 e^{x_1^{(k)}-x_2^{(k)}+3} \right) \\
x_2^{(k+1)} &= x_2^{(k)} - \eta \left( 2x_2^{(k)} - 1 - (x_3^{(k)} - 2)^4 e^{x_1^{(k+1)}-x_2^{(k)}+3} \right) \\
x_3^{(k+1)} &= x_3^{(k)} - \eta \left( 4 (x_3^{(k)} - 2)^3 e^{x_1^{(k+1)}-x_2^{(k+1)}+3} \right)
\end{align*}
\]
Table: CCD algorithm ($\eta = 0.25$)

<table>
<thead>
<tr>
<th>$k$</th>
<th>$x_1^{(k)}$</th>
<th>$x_2^{(k)}$</th>
<th>$x_3^{(k)}$</th>
<th>$D_1^{(k)}$</th>
<th>$D_2^{(k)}$</th>
<th>$D_3^{(k)}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1.0000</td>
<td>1.0000</td>
<td>1.0000</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>-4.0214</td>
<td>0.7831</td>
<td>1.1646</td>
<td>20.0855</td>
<td>0.8675</td>
<td>-0.6582</td>
</tr>
<tr>
<td>2</td>
<td>-1.5307</td>
<td>0.8834</td>
<td>2.2121</td>
<td>-9.9626</td>
<td>-0.4013</td>
<td>-4.1902</td>
</tr>
<tr>
<td>3</td>
<td>-0.2663</td>
<td>0.6949</td>
<td>2.1388</td>
<td>-5.0578</td>
<td>0.7540</td>
<td>0.2932</td>
</tr>
<tr>
<td>4</td>
<td>0.3661</td>
<td>0.5988</td>
<td>2.0962</td>
<td>-2.5297</td>
<td>0.3845</td>
<td>0.1703</td>
</tr>
<tr>
<td>5</td>
<td>0.6827</td>
<td>0.5499</td>
<td>2.0758</td>
<td>-1.2663</td>
<td>0.1957</td>
<td>0.0818</td>
</tr>
<tr>
<td>6</td>
<td>0.8412</td>
<td>0.5252</td>
<td>2.0638</td>
<td>-0.6338</td>
<td>0.0989</td>
<td>0.0480</td>
</tr>
<tr>
<td>7</td>
<td>0.9205</td>
<td>0.5127</td>
<td>2.0560</td>
<td>-0.3172</td>
<td>0.0498</td>
<td>0.0314</td>
</tr>
<tr>
<td>8</td>
<td>0.9602</td>
<td>0.5064</td>
<td>2.0504</td>
<td>-0.1588</td>
<td>0.0251</td>
<td>0.0222</td>
</tr>
<tr>
<td>9</td>
<td>0.9800</td>
<td>0.5033</td>
<td>2.0463</td>
<td>-0.0795</td>
<td>0.0126</td>
<td>0.0166</td>
</tr>
<tr>
<td>$\infty$</td>
<td>1.0000</td>
<td>0.5000</td>
<td>2.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
</tbody>
</table>
Linear regression

We consider the linear regression:

$$Y = X\beta + \varepsilon$$

where $Y$ is a $n \times 1$ vector, $X$ is a $n \times m$ matrix and $\beta$ is a $m \times 1$ vector. The optimization problem is:

$$\hat{\beta} = \arg \min_{\beta} f(\beta) = \frac{1}{2} (Y - X\beta)^\top (Y - X\beta)$$

Since we have $\frac{\partial f(\beta)}{\partial \beta} = -X^\top (Y - X\beta)$, we deduce that:

$$\frac{\partial f(\beta)}{\partial \beta_j} = x_j^\top (X\beta - Y)$$

$$= x_j^\top (x_j\beta_j + X_{(-j)}\beta_{(-j)} - Y)$$

$$= x_j^\top x_j\beta_j + x_j^\top X_{(-j)}\beta_{(-j)} - x_j^\top Y$$

where $x_j$ is the $n \times 1$ vector corresponding to the $j^{\text{th}}$ variable and $X_{(-j)}$ is the $n \times (m - 1)$ matrix (without the $j^{\text{th}}$ variable)
At the optimum, we have $\partial_{\beta_j} f(\beta) = 0$ or:

$$
\beta_j = \frac{x_j^\top Y - x_j^\top X_{(-j)} \beta_{(-j)}}{x_j^\top x_j} = \frac{x_j^\top (Y - X_{(-j)} \beta_{(-j)})}{x_j^\top x_j}
$$

**CCD algorithm for the linear regression**

We have:

$$
\beta_j^{(k+1)} = \frac{x_j^\top \left( Y - \sum_{j'=1}^{j-1} x_{j'} \beta_j^{(k+1)} - \sum_{j'=j+1}^{m} x_{j'} \beta_j^{(k)} \right)}{x_j^\top x_j}
$$

$\Rightarrow$ Introducing pointwise constraints is straightforward
Lasso regression

The objective function becomes:

\[ f(\beta) = \frac{1}{2} (Y - X\beta)^\top (Y - X\beta) + \lambda \|\beta\|_1 \]

Since the norm is separable – \( \|\beta\|_1 = \sum_{j=1}^{m} |\beta_j| \), the first-order condition is:

\[ x_j^\top (X\beta - Y) + \lambda \partial |\beta_j| = 0 \]

**CCD algorithm for the lasso regression**

We have:

\[ \beta_j^{(k+1)} = \frac{1}{x_j^\top x_j} S_{\lambda} \left( x_j^\top \left( Y - \sum_{j'=1}^{j-1} x_{j'} \beta_j^{(k+1)} - \sum_{j'=j+1}^{m} x_{j'} \beta_{j'}^{(k)} \right) \right) \]

where \( S_{\lambda}(v) \) is the soft-thresholding operator:

\[ S_{\lambda}(v) = \text{sign}(v) \cdot (|v| - \lambda)_+ \]
Lasso regression

Table: Matlab code

for k = 1:nIters
    for j = 1:m
        x_j = X(:,j);
        X_j = X;
        X_j(:,j) = zeros(n,1);
        if lambda > 0
            v = x_j'* (Y - X_j*beta);
            beta(j) = max(abs(v) - lambda,0) * sign(v) / (x_j'*x_j);
        else
            beta(j) = x_j'* (Y - X_j*beta) / (x_j'*x_j);
        end
    end
end
The dimension problem is \((2m, 2m)\) for QP and \((1, 0)\) for CCD!

CCD is faster for lasso regression than for linear regression (because of the soft-thresholding operator)!

Suppose \(n = 50000\) and \(m = 1000000\) (DNA problem)
Alternative direction method of multipliers

**Definition**

The alternating direction method of multipliers (ADMM) is an algorithm introduced by Gabay and Mercier (1976) to solve problems which can be expressed as:

\[
\{x^*, z^*\} = \arg \min_{x, z} f(x) + g(z)
\]

s.t. \(Ax + Bz = c\)

The algorithm is:

\[
x^{(k)} = \arg \min \left\{ f(x) + \frac{\phi}{2} \| Ax + Bz^{(k-1)} - c + u^{(k-1)} \|^2 \right\}
\]

\[
z^{(k)} = \arg \min \left\{ g(z) + \frac{\phi}{2} \| Ax^{(k)} + Bz - c + u^{(k-1)} \|^2 \right\}
\]

\[
u^{(k)} = u^{(k-1)} + (Ax^{(k)} + Bz^{(k)} - c)
\]
An example

We consider the following optimization problem:

\[
    x^* = \arg\min_x f(x) \quad \text{s.t.} \quad x^- \leq x \leq x^+
\]

It can be written as:

\[
    \{x^*, z^*\} = \arg\min_x f(x) + g(z) \quad \text{s.t.} \quad x - z = 0_n
\]

where \( g(z) = \mathbb{1}_\Omega(x) \) and \( \Omega = \{x : x^- \leq x \leq x^+\} \). By setting \( \varphi = \frac{1}{2} \), the z-step becomes:

\[
    z^{(k)} = \arg\min \left\{ g(z) + \frac{1}{2} \| x^{(k)} - z + u^{(k-1)} \|_2^2 \right\}
\]

\[
    = \prox_g \left( x^{(k)} + u^{(k-1)} \right)
\]

where the proximal operator is the box projection:

\[
    \prox_g(v) = x^- \odot \mathbb{1}\{v < x^-\} + v \odot \mathbb{1}\{x^- \leq v \leq x^+\} + x^+ \odot \mathbb{1}\{v > x^+\}
\]
The ADMM algorithm is then:

\[
\begin{align*}
    x^{(k)} &= \arg\min \left\{ f(x) + \frac{1}{2} \left\| x - z^{(k-1)} + u^{(k-1)} \right\|^2 \right\} \\
    z^{(k)} &= \text{prox}_g \left( x^{(k)} + u^{(k-1)} \right) \\
    u^{(k)} &= u^{(k-1)} + \left( x^{(k)} - z^{(k)} \right)
\end{align*}
\]

⇒ Solving the constrained optimization problem consists in solving the unconstrained optimization problem, applying the box projection and iterating these steps until convergence.
The Cholesky trick

We consider the following problem:

\[
    x^* = \arg \max U(x)
\]

subject to

\[
    \left\{ \begin{array}{l}
        x \in \Omega \\
        \sqrt{x^\top \Sigma x} \leq \bar{\sigma}
    \end{array} \right.
\]

We have:

\[
    \{x^*, z^*\} = \arg \min f(x) + g(z)
\]

subject to

\[-Lx + z = 0_n\]

where

\[
f(x) = -U(x) + 1_\Omega(x), \quad g(z) = 1_\mathcal{E}(z), \quad \mathcal{E} = \left\{ z \in \mathbb{R}^n : \|z\|^2_2 \leq \bar{\sigma}^2 \right\}
\]

and \(L\) is the upper Cholesky decomposition matrix of \(\Sigma\):

\[
\|z\|^2_2 = z^\top z = x^\top L^\top Lx = x^\top \Sigma x = \sigma^2(x)
\]

⇒ The cholesky trick has been used by Gonzalvez et al. (2019) for solving trend-following strategies using the ADMM algorithm in the context of Bayesian learning.
The proximal operator \( \text{prox}_f(v) \) of the function \( f(x) \) is defined by:

\[
\text{prox}_f(v) = x^* = \arg \min_x \left\{ f(x) + \frac{1}{2} \|x - v\|_2^2 \right\}
\]

If \( f(x) = -\ln x \), we have:

\[
f(x) + \frac{1}{2} \|x - v\|_2^2 = -\ln x + \frac{1}{2} (x - v)^2 = -\ln x + \frac{1}{2} x^2 - xv + \frac{1}{2} v^2
\]

The first-order condition is \(-x^{-1} + x - v = 0\). It follows that:

\[
\text{prox}_f(v) = \frac{v + \sqrt{v^2 + 4}}{2}
\]

If \( f(x) = -\lambda \sum_{i=1}^{n} \ln x_i \), we have \((\text{prox}_f(v))_i = \frac{v_i + \sqrt{v_i^2 + 4\lambda}}{2}\).
An example

We consider the following optimization problem:

\[ x^* = \arg\min f(x) - \lambda \sum_{i=1}^{n} \ln x_i \]

We set \( z = x \) and \( g(z) = -\lambda \sum_{i=1}^{n} \ln x_i \). The ADMM algorithm becomes

\[
\begin{align*}
    x^{(k)} &= \arg\min \left\{ f(x) + \frac{\varphi}{2} \left\| x - z^{(k-1)} + u^{(k-1)} \right\|^2 \right\} \\
    v^{(k)} &= x^{(k)} + u^{(k-1)} \\
    z^{(k)} &= \frac{v^{(k)} + \sqrt{v^{(k)} \odot v^{(k)}} + 4\lambda}{2} \\
    u^{(k)} &= u^{(k-1)} + \left( x^{(k)} - z^{(k)} \right)
\end{align*}
\]

If \( f(x) \) is a quadratic function, the \( x \)-step is straightforward
If we assume that \( f(x) = 1_\Omega(x) \) where \( \Omega \) is a convex set, we have:

\[
\operatorname{prox}_f(v) = \arg \min_x \left\{ 1_\Omega(x) + \frac{1}{2} \|x - v\|^2 \right\} = \mathcal{P}_\Omega(v)
\]

where \( \mathcal{P}_\Omega(v) \) is the standard projection. Parikh and Boyd (2014) show that:

<table>
<thead>
<tr>
<th>( \Omega )</th>
<th>( \mathcal{P}_\Omega(v) )</th>
<th>( \Omega )</th>
<th>( \mathcal{P}_\Omega(v) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( Ax = B ) ( v - A^\dagger (Av - B) )</td>
<td>( c^\top x \leq d ) ( v - \frac{\langle c, v - d \rangle}{|c|^2} + c )</td>
<td></td>
<td></td>
</tr>
<tr>
<td>( a^\top x = b ) ( v - \frac{(a^\top v - b)}{|a|^2} a )</td>
<td>( x^- \leq x \leq x^+ ) ( \mathcal{T}(v; x^-, x^+) )</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

where \( \mathcal{T}(v; x^-, x^+) \) is the truncation operator.
Norm constraints

We have $\text{prox}_{\lambda \max} (v) = \min(v, s^*)$ where $s^*$ is given by:

$$s^* = \left\{ s \in \mathbb{R} : \sum_{i=1}^{n} (v_i - s)_+ = \lambda \right\}$$

If $f(x)$ is a $L_p$-norm function and $\mathcal{B}_p(c, \lambda)$ is the $L_p$-ball with center $c$ and radius $\lambda$, we have:

<table>
<thead>
<tr>
<th>$p$</th>
<th>$\text{prox}_{\lambda f}(v)$</th>
<th>$\mathcal{P}_{\mathcal{B}_p(0_n, \lambda)}(v)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$p = 1$</td>
<td>$S_\lambda (v) = (</td>
<td>v</td>
</tr>
<tr>
<td>$p = 2$</td>
<td>$\left(1 - \frac{1}{\max(\lambda, |v|_2)}\right)v$</td>
<td>$v - \text{prox}_{\lambda |\cdot|_2} (</td>
</tr>
<tr>
<td>$p = \infty$</td>
<td>$\text{prox}_{\lambda \max} (</td>
<td>v</td>
</tr>
</tbody>
</table>

In the case where the center $c$ is not equal to $0_n$, we have:

$$\mathcal{P}_{\mathcal{B}_p(c, \lambda)}(v) = \mathcal{P}_{\mathcal{B}_p(0_n, \lambda)}(v - c) + c$$
ADMM and constraints

We consider the following optimization problem:

\[ x^* = \arg \min_{x} f(x) \]

\[ \text{s.t. } x \in \Omega \]

where \( \Omega \) is a complex set of constraints:

\[ \Omega = \Omega_1 \cap \Omega_2 \cap \cdots \Omega_m \]

We set \( z = x \) and \( g(z) = 1_{\Omega}(z) \). The ADMM algorithm becomes

\[ x^{(k)} = \arg \min \left\{ f(x) + \frac{\phi}{2} \left\| x - z^{(k-1)} + u^{(k-1)} \right\|_2^2 \right\} \]

\[ v^{(k)} = x^{(k)} + u^{(k-1)} \]

\[ z^{(k)} = \mathcal{P}_\Omega \left( v^{(k)} \right) \]

\[ u^{(k)} = u^{(k-1)} + \left( x^{(k)} - z^{(k)} \right) \]

The question is how to compute \( \mathcal{P}_\Omega (v) \).
We consider the proximal problem \( x^* = \text{prox}_f (v) \) where \( f(x) = 1_\Omega (x) \) and:

\[
\Omega = \Omega_1 \cap \Omega_2 \cap \cdots \cap \Omega_m
\]

The Dykstra’s algorithm is:

- The \( x \)-update is:

\[
x^{(k)} = \mathcal{P}_{\Omega_{\text{mod}(k,m)}} \left( x^{(k-1)} + z^{(k-m)} \right)
\]

- The \( z \)-update is:

\[
z^{(k)} = x^{(k-1)} + z^{(k-m)} - x^{(k)}
\]

where \( x^{(0)} = v, \ z^{(k)} = 0_n \) for \( k < 0 \) and \( \text{mod}(k, m) \) denotes the modulo operator taking values in \( \{1, \ldots, m\} \)
Dykstra’s algorithm

Successive projections of $\mathcal{P}_{\Omega_k}(x^{(k-1)})$ does not work!

Successive projections of $\mathcal{P}_{\Omega_k}(x^{(k-1)} + z^{(k-m)})$ does work!
The Markowitz portfolio optimization problem becomes:

\[
x^* = \arg \min \frac{1}{2} x^\top \Sigma x - \gamma x^\top \mu + \frac{1}{2} \rho_2 \| \Gamma_2 (x - x_0) \|_2^2 + \rho_p \| \Gamma_p (x - x_0) \|_p^p
\]

s.t. \( x \in \Omega \)

where \( p > 0 \).

We have the following properties:

- The penalties \( L_p \) for \( p \geq 1 \) are used for regularization
- The penalties \( L_p \) for \( p \leq 1 \) are used for sparsity
- The case \( p = 1 \) corresponds to the lasso regression
Mixed penalties

Figure: Lasso regularization with a target portfolio (relative sparsity)
Mixed penalties

**Figure:** Lasso regularization without a target portfolio (absolute sparsity)
Solving the mixed penalty problem

If $\Omega$ is a set of linear constraints ($Ax = B$, $Cx \geq D$, $x^- \leq x \leq x^+$), the mixed penalty problem can be written as:

$$\{x^*, z^*\} = \arg\min f(x) + g(z)$$

s.t. $x - z = 0$

where:

$$f(x) = \frac{1}{2} x^T \Sigma x - \gamma x^T \mu + \frac{1}{2} \rho_2 \|\Gamma_2 (x - x_0)\|^2_2 + 1\Omega(x)$$

and:

$$g(z) = \rho_p \|\Gamma_p (z - x_0)\|^p_p$$

The ADMM algorithm is implemented as follows:

- the $x$-step is a QP problem
- the $z$-step is the $L_p$ projection
Solving the mixed penalty problem

If $\Omega$ is more complex, the mixed penalty problem can be written as:

$$\{x^*, z^*\} = \arg \min f(x) + g(z)$$

s.t. $x - z = 0_n$

where:

$$f(x) = \frac{1}{2} x^\top \Sigma x - \gamma x^\top \mu + \frac{1}{2} \rho_2 \| \Gamma_2 (x - x_0) \|^2 \propto \frac{1}{2} x^\top (\Sigma + \Lambda) x - x^\top (\gamma \mu + \Lambda x_0)$$

$$\Lambda = \rho_2 \Gamma_2^\top \Gamma_2$$ and:

$$g(z) = 1_{\Omega} (z) + \rho_p \| \Gamma_p (z - x_0) \|_p^p$$

The ADMM algorithm is implemented as follows:

1. the $x$-step is:

   $$x^{(k)} = \left( \Sigma + \Lambda + \frac{\phi}{2} I_n \right)^{-1} \left( \gamma \mu + \Lambda x_0 + \phi \left( z^{(k-1)} - u^{(k-1)} \right) \right)$$

2. the $z$-step is given by the Dykstra’s algorithm
We consider the following risk measure:

\[ R(x) = -x^\top (\mu - r) + c \cdot \sqrt{x^\top \Sigma x} \]

The risk contribution of Asset \( i \) is given by:

\[ RC_i(x) = x_i \cdot \left( - (\mu_i - r) + c \frac{(\Sigma x)_i}{\sqrt{x^\top \Sigma x}} \right) \]

Roncalli (2013) defines the risk budgeting (RB) portfolio as:

\[
\begin{cases}
RC_i(x) = b_i R(x) \\
b_i > 0, \ x_i \geq 0 \\
\sum_{i=1}^{n} b_i = 1, \ \sum_{i=1}^{n} x_i = 1
\end{cases}
\]

where \( b_i \) is the risk budget of Asset \( i \)
Wrong formulation of the optimization problem

Since we have:

\[
\frac{1}{b_i} \mathcal{RC}_i(x) = \frac{1}{b_j} \mathcal{RC}_j(x) \quad \text{for all } i,j
\]

the RB portfolio is the solution of the optimization problem:

\[
x_{RB} = \arg\min \sum_{i=1}^{n} \sum_{j=1}^{n} \left( \frac{1}{b_i} \mathcal{RC}_i(x) - \frac{1}{b_j} \mathcal{RC}_j(x) \right)^2
\]

s.t. \[
\begin{cases}
1^\top x = 1 \\
x \geq 0
\end{cases}
\]
Roncalli (2013) shows that:

\[ x_{RB} = \frac{x^*(\lambda)}{1^\top x^*(\lambda)} \]

where \( x^*(\lambda) \) is the solution of the Lagrange problem

\[
x^*(\lambda) = \arg\min_{x} \mathcal{R}(x) - \lambda \sum_{i=1}^{n} b_i \ln x_i \]

s.t. \( x \geq 0 \)

where \( \lambda \) is an arbitrary positive scalar
Griveau-Billion et al. (2013) propose applying the CCD algorithm to find the solution of the objective function:

\[ f(x) = -x^\top \pi + c \sqrt{x^\top \Sigma x} - \lambda \sum_{i=1}^{n} b_i \ln x_i \]

where \( \pi = \mu - r \). For the cycle \( k+1 \) and the \( i \)th coordinate of the CCD algorithm, we have:

\[
x_i = \frac{-c \left( \sigma_i \sum_{j \neq i} x_j \rho_{i,j} \sigma_j \right) + \pi_i \sigma(x) + \sqrt{\left(c \left( \sigma_i \sum_{j \neq i} x_j \rho_{i,j} \sigma_j \right) - \pi_i \sigma(x) \right)^2 + 4 \lambda c b_i \sigma_i^2 \sigma(x)}}{2c \sigma_i^2}
\]

In this equation, we have the following CCD correspondence:

- \( x_i \rightarrow x_i^{(k+1)} \)
- \( x_j \rightarrow x_j^{(k+1)} \) if \( j < i \)
- \( x_j \rightarrow x_j^{(k)} \) if \( j > i \)
- \( x \rightarrow \left(x_1^{(k+1)}, \ldots, x_{i-1}^{(k+1)}, x_i^{(k)}, x_{i+1}^{(k)}, \ldots, x_n^{(k)} \right) \)
We have

\[
\begin{cases}
RC_i(x) = b_i R(x) \\
x \in \mathcal{I} \\
x \in \Omega
\end{cases}
\]

where $\mathcal{I}$ is the standard simplex and $x \in \Omega$ is the set of additional constraints.
Bai et al. (2016) propose to solve the following optimization program:

\[
\{ x^* (\mathcal{I}, \Omega), \theta^* \} = \arg \min \sum_{i=1}^{n} \left( \frac{1}{b_i} \mathcal{R} \mathcal{C}_i (x) - \theta \right)^2 \\
\text{s.t. } x \in \mathcal{I} \cap \Omega
\]
The Richard-Roncalli solution

Richard and Roncalli (2019) argue that the right optimization problem is:

$$x^*(\mathcal{I}, \Omega) = \arg\min_{x} R(x)$$

s.t. \[
\begin{aligned}
\sum_{i=1}^{n} b_i \ln x_i &\geq \kappa^* \\
x &\in \mathcal{I} \cap \Omega
\end{aligned}
\]

where \(\kappa^*\) is a constant to be determined. They consider the Lagrange formulation:

$$x^*(\Omega, \lambda) = \arg\min_{x} R(x) - \lambda \sum_{i=1}^{n} b_i \ln x_i$$

s.t. \(x \in \Omega\)

The constrained risk budgeting portfolio is defined by:

$$x^*(\mathcal{I}, \Omega) = \left\{ x^*(\Omega, \lambda^*) : \sum_{i=1}^{n} x_i^*(\Omega, \lambda^*) = 1 \right\}$$

Richard and Roncalli (2019) argue that the right optimization problem is:
We note:

\[ \mathcal{L}(x; \lambda) = \mathcal{R}(x) - \lambda \sum_{i=1}^{n} b_i \ln x_i + 1_{\Omega}(x) \]

The risk budgeting portfolio is computed by:

1. Solving \( x^*(\Omega, \lambda) = \arg \min \mathcal{L}(x; \lambda) \) for a given value of \( \lambda \) (\( x \)-step)
2. Finding the optimal value \( \lambda^* \) such that \( \sum_{i=1}^{n} x_i^*(\Omega, \lambda^*) = 1 \) (\( \lambda \)-step)
Bisection algorithm for the $\lambda$-step

We consider two scalars $a_\lambda$ and $b_\lambda$ such that $a_\lambda < b_\lambda$ and $\lambda^* \in [a_\lambda, b_\lambda]$. We note $\varepsilon_\lambda$ the convergence criterion of the bisection algorithm.

```
repeat
    We calculate $\lambda = \frac{a_\lambda + b_\lambda}{2}$
    We compute $x^*(\lambda)$ the solution of the minimization problem:
    $x^*(\lambda) = \arg \min \mathcal{L}(x; \lambda)$

    if $\sum_{i=1}^{n} x^*_i(\lambda) < 1$ then
        $a_\lambda \leftarrow \lambda$
    else
        $b_\lambda \leftarrow \lambda$
    end if

until $\left| \sum_{i=1}^{n} x^*_i(\lambda) - 1 \right| \leq \varepsilon_\lambda$

return $\lambda^* \leftarrow \lambda$ and $x^*(\mathcal{S}, \Omega) \leftarrow x^*(\lambda^*)$
```
Thanks to Tseng (2001), CCD algorithm can solve:

$$\arg \min f(x) = f_0(x) + \sum_{i=1}^{n} f_i(x_i)$$

where $f_0$ is strictly convex and differentiable and the functions $f_i$ are non-differentiable. We have:

$$\mathcal{L}(x; \lambda) = -x^\top \pi + c \sqrt{x^\top \Sigma x} - \lambda \sum_{i=1}^{n} b_i \ln x_i + 1_{\Omega}(x)$$

For separable constraints $\Omega = \bigcap_{i=1}^{n} \Omega_i$, the CCD algorithm consists in adding the projection $x_i = P_{\Omega_i}(x_i)$ at each iteration.

For non-separable constraints, CCD cannot be used.
### ADMM algorithm for the $x$-step

We exploit the separability of $\mathcal{L}(x; \lambda)$:

\[
\{x^*(\lambda), z^*(\lambda)\} = \arg \min f(x) + g(z)
\]

s.t. $x - z = 0$

where:

\[
\mathcal{L}(x; \lambda) = R(x) - \lambda \sum_{i=1}^{n} b_i \ln x_i + \underbrace{\mathbb{1}_\Omega(x)}_{f(x)} + \underbrace{\mathbb{1}_\Omega(x)}_{g(x)}
\]  

(#1)

or:

\[
\mathcal{L}(x; \lambda) = R(x) + \mathbb{1}_\Omega(x) - \lambda \sum_{i=1}^{n} b_i \ln x_i + \underbrace{\mathbb{1}_\Omega(x)}_{f(x)} + \underbrace{-\lambda \sum_{i=1}^{n} b_i \ln x_i}_{g(x)}
\]  

(#2)

<table>
<thead>
<tr>
<th>Formulation</th>
<th>(#1)</th>
<th>(#2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\arg \min f^{(k)}(x)$</td>
<td>NR/BFGS/CCD</td>
<td>QP/SQP</td>
</tr>
<tr>
<td>$\arg \min g^{(k)}(z)$</td>
<td>Projection/Dykstra</td>
<td>Proximal (logarithmic barrier)</td>
</tr>
</tbody>
</table>
Table: Computational time using our Matlab implementation (relative value)

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>x-update</th>
<th>(1)</th>
<th>(2)</th>
<th>(3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADMM</td>
<td>Newton</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>ADMM</td>
<td>BFGS</td>
<td>380</td>
<td>280</td>
<td>25</td>
</tr>
<tr>
<td>ADMM</td>
<td>QP</td>
<td>220</td>
<td>120</td>
<td>110</td>
</tr>
<tr>
<td>ADMM</td>
<td>CCD</td>
<td>10</td>
<td>9</td>
<td>8</td>
</tr>
<tr>
<td>CCD</td>
<td></td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

(1) $\phi = 1 + \text{classical bisection}$
(2) $\phi = 1 + \text{accelerated bisection}$
(3) *Adaptive method* $\phi^{(k)} + \text{accelerated bisection}$

Python implementation: CCD and ADMM-QP are the best algorithms!
How does the ERC property hold?

We consider a universe of five assets. Their volatilities are equal to 15%, 20%, 25%, 30% and 10%. The correlation matrix of asset returns is given by the following matrix:

$$\rho = \begin{pmatrix}
1.00 & 0.10 & 0.40 & 0.50 & 0.50 \\
0.10 & 1.00 & 0.70 & 0.40 & 0.40 \\
0.40 & 0.70 & 1.00 & 0.80 & 0.05 \\
0.50 & 0.40 & 0.80 & 1.00 & 0.10 \\
0.50 & 0.40 & 0.05 & 0.10 & 1.00 \\
\end{pmatrix}$$

We assume that the current portfolio is $x_0 = (25\%, 25\%, 10\%, 15\%, 30\%)$

We would like to obtain an ERC portfolio with the following constraints:

$$x_0 - 5\% \leq x \leq x_0 + 5\%$$
How does the ERC property hold?

**Table:** Volatility breakdown (in %) of current and ERC portfolios

<table>
<thead>
<tr>
<th>Asset</th>
<th>Current portfolio</th>
<th>ERC portfolio</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$x_i$</td>
<td>$MR_i$</td>
</tr>
<tr>
<td>1</td>
<td>25.00</td>
<td>10.00</td>
</tr>
<tr>
<td>2</td>
<td><strong>25.00</strong></td>
<td>15.40</td>
</tr>
<tr>
<td>3</td>
<td>10.00</td>
<td>20.30</td>
</tr>
<tr>
<td>4</td>
<td>10.00</td>
<td>22.24</td>
</tr>
<tr>
<td>5</td>
<td><strong>30.00</strong></td>
<td>5.90</td>
</tr>
<tr>
<td>$\sigma(x)$</td>
<td>12.37</td>
<td></td>
</tr>
</tbody>
</table>
How does the ERC property hold?

**Table:** Volatility breakdown (in %) of naive and least squares solutions

<table>
<thead>
<tr>
<th>Asset</th>
<th>Naive solution</th>
<th>Least squares solution</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$x_i$</td>
<td>$MR_i$</td>
</tr>
<tr>
<td>1</td>
<td>22.84</td>
<td>10.25</td>
</tr>
<tr>
<td>2</td>
<td>20.00</td>
<td>14.98</td>
</tr>
<tr>
<td>3</td>
<td>12.34</td>
<td>20.18</td>
</tr>
<tr>
<td>4</td>
<td>9.83</td>
<td>22.46</td>
</tr>
<tr>
<td>5</td>
<td>35.00</td>
<td>5.99</td>
</tr>
<tr>
<td>$\sigma(x)$</td>
<td>12.13</td>
<td></td>
</tr>
</tbody>
</table>
How does the ERC property hold?

Table: Volatility breakdown (in %) of the constrained ERC portfolio

<table>
<thead>
<tr>
<th>Asset</th>
<th>( x_i )</th>
<th>( M ) ( R_i )</th>
<th>( R C_i )</th>
<th>( R C_i^* )</th>
<th>( \lambda_i^- )</th>
<th>( \lambda_i^+ )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>22.89</td>
<td>10.28</td>
<td>2.35</td>
<td><strong>19.39</strong></td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>2</td>
<td><strong>20.00</strong></td>
<td>14.90</td>
<td>2.98</td>
<td>24.55</td>
<td>3.13</td>
<td>0.00</td>
</tr>
<tr>
<td>3</td>
<td>11.69</td>
<td>20.13</td>
<td>2.35</td>
<td><strong>19.39</strong></td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>4</td>
<td>10.42</td>
<td>22.57</td>
<td>2.35</td>
<td><strong>19.39</strong></td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>5</td>
<td><strong>35.00</strong></td>
<td>6.00</td>
<td>2.10</td>
<td>17.29</td>
<td>0.00</td>
<td>0.73</td>
</tr>
</tbody>
</table>

\[ \sigma(x) = 12.14 \quad \lambda = 11.76 \]
We consider a CW index composed of seven stocks. The weights are equal to 34%, 25%, 20%, 15%, 3%, 2% and 1%. We assume that the volatilities of these stocks are equal to 15%, 16%, 17%, 18%, 19%, 20% and 21%, whereas the correlation matrix of stock returns is given by:

\[
\rho = \begin{pmatrix}
1.00 & 0.75 & 0.73 & 0.70 & 0.65 & 0.62 & 0.60 \\
0.75 & 1.00 & 0.75 & 0.70 & 0.68 & 0.65 & 0.60 \\
0.73 & 0.75 & 1.00 & 0.75 & 0.69 & 0.63 & 0.65 \\
0.70 & 0.70 & 0.75 & 1.00 & 0.75 & 0.67 & 0.68 \\
0.65 & 0.68 & 0.69 & 0.75 & 1.00 & 0.70 & 0.75 \\
0.62 & 0.65 & 0.63 & 0.67 & 0.70 & 1.00 & 0.80 \\
0.60 & 0.60 & 0.65 & 0.68 & 0.75 & 0.80 & 1.00 \\
\end{pmatrix}
\]
Smart beta portfolios without small cap bias

- LC-ERC (large cap ERC): Apply the ERC on the large cap universe
- LS-ERC (least squares ERC): Solve the RB portfolio by adding small cap constraints on the LS problem
- C-ERC (Constrained ERC): Solve the RB portfolio by imposing the weight constraints:

\[
\begin{align*}
0 & \leq x_i & \quad & \text{if } i \notin \Omega_{lc} \\
-x_{cw,i} & \leq x_i & \leq x_{cw,i} & \quad & \text{if } i \in \Omega_{lc}
\end{align*}
\]
Smart beta portfolios without small cap bias

Table: Volatility breakdown (in %) of constrained ERC portfolios

<table>
<thead>
<tr>
<th>Asset</th>
<th>CW</th>
<th>ERC</th>
<th>LC-ERC</th>
<th>LS-ERC</th>
<th>C-ERC</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$x_i$</td>
<td>$\mathcal{RC}^*_i$</td>
<td>$x_i$</td>
<td>$\mathcal{RC}^*_i$</td>
<td>$x_i$</td>
</tr>
<tr>
<td>1</td>
<td>34.00</td>
<td>32.08</td>
<td>17.22</td>
<td>14.29</td>
<td>25.81</td>
</tr>
<tr>
<td>2</td>
<td>25.00</td>
<td>24.82</td>
<td>15.90</td>
<td>14.29</td>
<td>24.06</td>
</tr>
<tr>
<td>3</td>
<td>20.00</td>
<td>20.92</td>
<td>14.78</td>
<td>14.29</td>
<td>22.44</td>
</tr>
<tr>
<td>4</td>
<td>15.00</td>
<td>16.01</td>
<td>13.83</td>
<td>14.29</td>
<td>21.69</td>
</tr>
<tr>
<td>5</td>
<td>3.00</td>
<td>3.10</td>
<td>13.17</td>
<td>14.29</td>
<td>3.00</td>
</tr>
<tr>
<td>6</td>
<td>2.00</td>
<td>2.03</td>
<td>12.86</td>
<td>14.29</td>
<td>2.00</td>
</tr>
<tr>
<td>7</td>
<td>1.00</td>
<td>1.05</td>
<td>12.23</td>
<td>14.29</td>
<td>1.00</td>
</tr>
<tr>
<td>$\sigma(x)$</td>
<td>14.50</td>
<td>15.23</td>
<td>14.68</td>
<td>14.66</td>
<td>14.68</td>
</tr>
</tbody>
</table>
Managing the portfolio turnover

The turnover of Portfolio \( x \) with respect to Portfolio \( x_0 \) is equal to:

\[
\tau(x \mid x_0) = \sum_{i=1}^{n} |x_i - x_{0,i}| = \|x - x_0\|_1
\]

Therefore, the corresponding Lagrange function is:

\[
\mathcal{L}(x; \lambda) = \mathcal{R}(x) - \lambda \sum_{i=1}^{n} b_i \ln x_i + \mathbb{1}_\Omega(x)
\]

where \( \Omega = \{ x \in \mathbb{R} : \tau(x \mid x_0) \leq \tau^* \} \) and \( \tau^* \) is the turnover limit. If we use the previous algorithms, the only difficulty is calculating the proximal operator of \( g(x) = \mathbb{1}_\Omega(x) \):

\[
\text{prox}_{g}(x) = \text{prox}_{f}(x - x_0) + x_0
\]

where \( f(x) = \mathbb{1}_{\Omega'}(x) \) and \( \Omega' = \{ x \in \mathbb{R} : \|x\|_1 \leq \tau^* \} \). We deduce that:

\[
\text{prox}_{g}(x) = x - \text{prox}_{\tau^* \max}(|x - x_0|) \odot \text{sign}(x - x_0)
\]

where \( \text{prox}_{\lambda \max}(v) \) is the proximal operator of the pointwise maximum function (see Slide 49)
Managing the portfolio turnover

We consider a universe of eight asset classes: (1) US 10Y Bonds, (2) Euro 10Y Bonds, (3) Investment Grade Bonds, (4) High Yield Bonds, (5) US Equities, (6) Euro Equities, (7) Japan Equities and (8) EM Equities

Table: Volatility and correlation matrix of asset returns (in %)

<table>
<thead>
<tr>
<th>$\sigma_i$</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>100</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>80</td>
<td>100</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>60</td>
<td>40</td>
<td>100</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>-20</td>
<td>-20</td>
<td>50</td>
<td>100</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>-10</td>
<td>-20</td>
<td>30</td>
<td>60</td>
<td>100</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>-20</td>
<td>-10</td>
<td>20</td>
<td>60</td>
<td>90</td>
<td>100</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>-20</td>
<td>-20</td>
<td>20</td>
<td>50</td>
<td>70</td>
<td>60</td>
<td>100</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>-20</td>
<td>-20</td>
<td>30</td>
<td>60</td>
<td>70</td>
<td>70</td>
<td>70</td>
<td>100</td>
</tr>
</tbody>
</table>
Managing the portfolio turnover

We assume that the current allocation is a 50/50 asset mix policy, where the weight of each asset class is 12.5%.

**Table:** Constrained RB portfolios (in %) with turnover control

<table>
<thead>
<tr>
<th>Asset</th>
<th>0.00</th>
<th>10.00</th>
<th>20.00</th>
<th>30.00</th>
<th>40.00</th>
<th>50.00</th>
<th>60.00</th>
<th>70.00</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>12.50</td>
<td>14.86</td>
<td>17.28</td>
<td>19.68</td>
<td>22.01</td>
<td>24.28</td>
<td>26.58</td>
<td>26.83</td>
</tr>
<tr>
<td>2</td>
<td>12.50</td>
<td>15.14</td>
<td>17.72</td>
<td>20.32</td>
<td>22.99</td>
<td>25.72</td>
<td>28.42</td>
<td>28.68</td>
</tr>
<tr>
<td>3</td>
<td>12.50</td>
<td>12.50</td>
<td>12.50</td>
<td>12.50</td>
<td>12.50</td>
<td>12.50</td>
<td>11.65</td>
<td>11.41</td>
</tr>
<tr>
<td>4</td>
<td>12.50</td>
<td>12.50</td>
<td>12.50</td>
<td>12.50</td>
<td>12.50</td>
<td>11.50</td>
<td>9.90</td>
<td>9.80</td>
</tr>
<tr>
<td>5</td>
<td>12.50</td>
<td>11.20</td>
<td>9.70</td>
<td>8.49</td>
<td>7.27</td>
<td>6.28</td>
<td>5.66</td>
<td>5.61</td>
</tr>
<tr>
<td>6</td>
<td>12.50</td>
<td>12.02</td>
<td>10.36</td>
<td>9.02</td>
<td>7.69</td>
<td>6.63</td>
<td>5.95</td>
<td>5.90</td>
</tr>
<tr>
<td>7</td>
<td>12.50</td>
<td>12.50</td>
<td>11.72</td>
<td>10.16</td>
<td>8.66</td>
<td>7.47</td>
<td>6.71</td>
<td>6.66</td>
</tr>
<tr>
<td>8</td>
<td>12.50</td>
<td>9.28</td>
<td>8.22</td>
<td>7.33</td>
<td>6.39</td>
<td>5.62</td>
<td>5.14</td>
<td>5.11</td>
</tr>
<tr>
<td>(\tau(x^*</td>
<td>x_0))</td>
<td>0.00</td>
<td>10.00</td>
<td>20.00</td>
<td>30.00</td>
<td>40.00</td>
<td>50.00</td>
<td>60.00</td>
</tr>
</tbody>
</table>

The last column corresponds to the risk parity portfolio (75% of bonds)
Unsolved problems

- Cardinality constraints:

<table>
<thead>
<tr>
<th>Strategy</th>
<th>Constraints</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sampling</td>
<td>$\text{card} (x_i \neq 0) = m$</td>
</tr>
<tr>
<td>Short</td>
<td>$\text{card} (x_i &lt; 0) = m$</td>
</tr>
<tr>
<td>Long-/short</td>
<td>$\text{card} (x_i &lt; 0) = \text{card} (x_i &gt; 0)$</td>
</tr>
<tr>
<td>Stock picking</td>
<td>$\text{card} (x_i &gt; \epsilon) = m$</td>
</tr>
</tbody>
</table>

- Scaling puzzle and the homogeneity property of the risk measure
Conclusion

- QP algorithm = universal algorithm in MVO-type asset allocation problems
- Machine learning ⇒ new optimization algorithms
  - Non-smooth objective function
  - Large-scale dimension
- Ridge/Lasso regularization ⇒ basic of modern portfolio optimization
- The 4 pillars are:
  1. CCD
  2. ADMM
  3. Proximal operators
  4. Dykstra's algorithm
- Applications: Robo-advisors, Smart beta portfolios, Dynamic risk parity strategies, Turnover management, etc.
How ESG Investing has Impacted the Asset Pricing in the Equity Market?, Amundi
Discussion Paper, 36.


An Algorithm for Restricted Least Squares Regression, Journal of the American Statistical
Association, 78(384), pp. 837-842.

Distributed Optimization and Statistical Learning via the Alternating Direction Method of

References II

Friedman, J., Hastie, T., and Tibshirani, R. (2010)

Gabay, D., and Mercier, B. (1976)

Gonzalvez, J., Lezmi, E., Roncalli, T., and Xu, J. (2019)


Markowitz H. (1952)
References III

- Markowitz H. (1956)  
The Optimization of a Quadratic Function Subject to Linear Constraints, *Naval Research Logistics Quarterly*, 3(1-2), pp. 111-133.

- Nesterov, Y. (2012)  


- Roncalli, T. (2013)  


- Scherer B. (2007)  

Tibshirani, R. (1996)

Tibshirani, R.J. (2017)

Tseng, P. (2001)
Disclaimer

This material is provided for information purposes only and does not constitute a recommendation, a solicitation, an offer, an advice or an invitation to purchase or sell any fund, SICAV, sub-fund, (“the Funds”) described herein and should in no case be interpreted as such.

This material, which is not a contract, is based on sources that Amundi considers to be reliable. Data, opinions and estimates may be changed without notice.

Amundi accepts no liability whatsoever, whether direct or indirect, that may arise from the use of information contained in this material. Amundi can in no way be held responsible for any decision or investment made on the basis of information contained in this material.

The information contained in this document is disclosed to you on a confidential basis and shall not be copied, reproduced, modified, translated or distributed without the prior written approval of Amundi, to any third person or entity in any country or jurisdiction which would subject Amundi or any of "the Funds", to any registration requirements within these jurisdictions or where it might be considered as unlawful. Accordingly, this material is for distribution solely in jurisdictions where permitted and to persons who may receive it without breaching applicable legal or regulatory requirements.

Not all funds, or sub-funds will be necessarily be registered or authorized in all jurisdictions or be available to all investors.

Investment involves risk. Past performances and simulations based on these, do not guarantee future results, nor are they reliable indicators of futures performances.

The value of an investment in the Funds, in any security or financial product may fluctuate according to market conditions and cause the value of an investment to go up or down. As a result, you may lose, as the case may be, the amount originally invested.

All investors should seek the advice of their legal and/or tax counsel or their financial advisor prior to any investment decision in order to determine its suitability.

It is your responsibility to read the legal documents in force in particular the current French prospectus for each fund, as approved by the AMF, and each investment should be made on the basis of such prospectus, a copy of which can be obtained upon request free of charge at the registered office of the management company.

This material is solely for the attention of institutional, professional, qualified or sophisticated investors and distributors. It is not to be distributed to the general public, private customers or retail investors in any jurisdiction whatsoever nor to “US Persons”.

Moreover, any such investor should be, in the European Union, a “Professional” investor as defined in Directive 2004/39/EC dated 21 May 2004 on markets in financial instruments (“MiFID”) or as the case may be in each local regulations and, as far as the offering in Switzerland is concerned, a “Qualified Investor” within the meaning of the provisions of the Swiss Collective Investment Schemes Ordinance of 23 June 2006 (CISA), the Swiss Collective Investment Schemes Ordinance of 22 November 2006 (CISO) and the FINMA’s Circular 08/8 on Public Offering within the meaning of the legislation on Collective Investment Schemes of 20 November 2008. In no event may this material be distributed in the European Union to non “Professional” investors as defined in the MiFID or in each local regulation, or in Switzerland to investors who do not comply with the definition of “qualified investors” as defined in the applicable legislation and regulation.

Amundi, French joint stock company (“Société Anonyme”) with a registered capital of € 1 086 262 605 and approved by the French Securities Regulator (Autorité des Marchés Financiers-AMF) under number GP 04000036 as a portfolio management company,

90 boulevard Pasteur, 75015 Paris-France
437 574 452 RCS Paris.
www.amundi.com