8

Machine Learning Optimization Algorithms & Portfolio Allocation

Portfolio optimization emerged with the seminal paper of Markowitz (1952). The original mean-variance framework is appealing because it is very efficient from a computational point of view. However, it also has one well-established failing since it can lead to portfolios that are not optimal from a financial point of view (Michaud 1989). Nevertheless, very few models have succeeded in providing a real alternative solution to the Markowitz model. The main reason lies in the fact that most academic portfolio optimization models are intractable in real life although they present solid theoretical properties. By intractable we mean that they can be implemented for an investment universe with a small number of assets, using a lot of computational resources and skills, but they are unable to manage a universe with dozens or hundreds of assets. However, the emergence and the rapid development of robo-advisors means that we need to rethink portfolio optimization and go beyond the traditional mean-variance optimization approach.

Another industry and branch of science has faced similar issues concerning large-scale optimization problems. Machine learning and applied statistics have long been associated with linear and logistic regression models. Again, the reason was the inability of optimization algorithms to solve high-dimensional industrial problems. Nevertheless, the end of the 1990s marked an important turning point with the development and the rediscovery of several methods that have since produced impressive results. The goal of this chapter is to show how portfolio allocation can benefit from the development of these large-scale optimization algorithms. Not all

Chapter written by Sarah PERRIN and Thierry RONCALLI.

Machine Learning for Asset Management: New Developments and Financial Applications, First Edition. Edited by Emmanuel Jurczenko.
© ISTE Ltd 2020. Published by ISTE Ltd and John Wiley & Sons, Inc.

of these algorithms are useful in our case, but four of them are essential when solving complex portfolio optimization problems. These four algorithms are the coordinate descent, the alternating direction method of multipliers, the proximal gradient method and Dykstra's algorithm. This chapter reviews them and shows how they can be implemented in portfolio allocation.

8.1. Introduction

The contribution of Markowitz to economics is considerable. The mean-variance optimization framework marks the beginning of portfolio allocation in finance. In addition to the seminal paper of 1952, Markowitz proposed an algorithm for solving quadratic programming problems in 1956. At that time, very few people were aware of this optimization framework. We can cite Mann (1943), but it is widely accepted that Markowitz is the "father of quadratic programming" (Cottle and Infanger 2010). This is not the first time that economists are participating in the development of mathematics[1], but this is certainly the first time that mathematicians will explore a field of research whose main application during the first years of research is exclusively an economic problem.

The success of mean-variance optimization (MVO) is due to the appealing properties of the quadratic utility function, but it should also be assessed in light of the success of quadratic programming (QP). Because it is easy to solve QP problems and because QP problems are available in mathematical software, solving MVO problems is straightforward and does not require a specific skill. Hence, the mean-variance optimization is a universal method which is used by all portfolio managers. However, this approach has been widely criticized by academics and professionals. Indeed, mean-variance optimization is very sensitive to input parameters and produces corner solutions. Moreover, the concept of mean-variance diversification is confused with the concept of hedging (Bourgeron *et al.* 2018). These different issues make the practice of mean-variance optimization less attractive than the theory (Michaud 1989). In fact, solving MVO allocation problems requires

[1] For example, Leonid Kantorovich made major contributions to the success of linear programming.

the right weight constraints to be specified, in order to obtain acceptable solutions. It follows that designing the constraints is the most important component of mean-variance optimization. In this case, MVO appears to be a trial-and-error process, not a systematic solution.

The success of the MVO framework is also explained by the fact that there are very few competing portfolio allocation models that can be implemented from an industrial point of view. There are generally two reasons for this. The first one is that some models use input parameters that are difficult to estimate or understand, making these models definitively unusable. The second reason is that other models use a more complex objective function than the simple quadratic utility function. In this case, the computational complexity makes these models less attractive than the standard MVO model. Among these models, some of them are based on the mean-variance objective function, but introduce regularization penalty functions, in order to improve the robustness of the portfolio allocation. Again, these models have little chance of being used if they cannot be cast into a QP problem. However, new optimization algorithms have emerged for solving large-scale machine learning problems. The purpose of this chapter is to present these new mathematical methods and show that they can be easily applied to portfolio allocation in order to go beyond the MVO/QP model.

This chapter is based on several previous research papers (Bourgeron *et al.* 2018, Griveau-Billion *et al.* 2013, Richard and Roncalli 2015 and Richard and Roncalli 2019) and extensively uses four leading references (Beck 2017; Boyd *et al.* 2010; Combettes and Pesquet 2011; Tibshirani 2017). It is organized as follows. In section 8.2, we present the mean-variance approach and how it is related to the QP framework. The third section is dedicated to large-scale optimization algorithms that have been used in machine learning: coordinate descent, alternating direction method of multipliers, proximal gradient and Dykstra's algorithm. Section 8.4 shows how these algorithms can be implemented in order to solve portfolio optimization problems and build a more robust asset allocation. Finally, section 8.5 offers some concluding remarks.

8.2. The quadratic programming world of portfolio optimization

8.2.1. *Quadratic programming*

8.2.1.1. *Primal and dual formulation*

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

$$x^\star = \arg\min_x \frac{1}{2} x^\top Q x - x^\top R \qquad \text{s.t.} \quad Sx \leq T \tag{8.1}$$

where x is an $n \times 1$ vector, Q is an $n \times n$ matrix and R is an $n \times 1$ vector. We note that the system of constraints $Sx \leq T$ allows us to specify linear equality constraints[2] $Ax = B$ or box constraints $x^- \leq x \leq x^+$. Most numerical packages then consider the following formulation:

$$\begin{aligned} x^\star &= \arg\min_x \frac{1}{2} x^\top Q x - x^\top R \\ &\text{s.t.} \left\{ \begin{array}{l} Ax = B \\ Cx \leq D \\ x^- \leq x \leq x^+ \end{array} \right. \end{aligned} \tag{8.2}$$

because the problem [8.2] is equivalent to the canonical problem [8.1] with the following system of linear inequalities: $-Ax \leq -B$, $Ax \leq B$, $Cx \leq D$, $-I_n x \leq -x^-$ and $I_n x \leq x^+$. If the space Ω defined by $Sx \leq T$ is non-empty and if Q is a symmetric positive definite matrix, the solution exists because the function $f(x) = \frac{1}{2} x^\top Q x - x^\top R$ is convex. In the general case where Q is a square matrix, the solution may not exist.

The QP problem has the interesting property that the dual formulation is another quadratic programming problem:

$$\lambda^\star = \arg\min_\lambda \frac{1}{2} \lambda^\top \bar{Q} \lambda - \lambda^\top \bar{R} \qquad \text{s.t.} \quad \lambda \geq 0 \tag{8.3}$$

where $\bar{Q} = SQ^{-1}S^\top$ and $\bar{R} = SQ^{-1}R - T$. This duality property is very important for some machine learning methods. For example, this is the case of

[2] This is equivalent to impose that $Ax \geq B$ and $Ax \leq B$.

support vector machines and kernel methods that extensively use the duality for defining the solution (Cortes and Vapnik 1995).

8.2.1.2. *Numerical algorithms*

There is a substantial literature on the methods for solving quadratic programming problems (Gould and Toint 2000). The research begins in the 1950s with different key contributions: Frank and Wolfe (1956), Markowitz (1956), Beale (1959) and Wolfe (1959). Nowadays, QP problems are generally solved using three approaches: active set methods, gradient projection methods and interior point methods. All these algorithms are implemented in standard mathematical programming languages (MATLAB, Matematica, Python, Gauss, R, etc.). This explains the success of QP problems since the 2000s, because they can be easily and rapidly solved.

8.2.2. *Mean-variance optimized portfolios*

The concept of portfolio allocation has a long history and dates back to the seminal work of Markowitz (1952). In his paper, Markowitz defined precisely what *portfolio selection* means: "the investor does (or should) consider expected return a desirable thing and variance of return an undesirable thing". Indeed, Markowitz showed that an efficient portfolio is the portfolio that maximizes the expected return for a given level of risk (corresponding to the variance of portfolio return) or a portfolio that minimizes the risk for a given level of expected return. Even if this framework has been extended to many other allocation problems (index sampling, turnover management, etc.), the mean-variance model remains the optimization approach that is most widely used in finance.

8.2.2.1. *The Markowitz framework*

We consider a universe of n assets. Let $x = (x_1, \ldots, x_n)$ be the vector of weights in the portfolio. We assume that the portfolio is fully invested meaning that $\sum_{i=1}^{n} x_i = \mathbf{1}_n^\top x = 1$. We denote $\mathfrak{R} = (\mathfrak{R}_1, \ldots, \mathfrak{R}_n)$ as the vector of asset returns where $\mathfrak{R}_i$ is the return of asset i. The return of the portfolio is then equal to $\mathfrak{R}(x) = \sum_{i=1}^{n} x_i \mathfrak{R}_i = x^\top \mathfrak{R}$. Let $\mu = \mathbb{E}[\mathfrak{R}]$ and $\Sigma = \mathbb{E}\left[(\mathfrak{R} - \mu)(\mathfrak{R} - \mu)^\top\right]$ be the vector of expected returns and the covariance matrix of asset returns, respectively. The expected return of the portfolio is equal to $\mu(x) = \mathbb{E}[\mathfrak{R}(x)] = x^\top \mu$, whereas its variance is equal to

$\sigma^2(x) = \mathbb{E}\left[(\mathfrak{R}(x) - \mu(x))(\mathfrak{R}(x) - \mu(x))^\top\right] = x^\top \Sigma x$. Markowitz (1952) formulated the investor's financial problem as follows:

1) maximizing the expected return of the portfolio under a volatility constraint:

$$\max \mu(x) \quad \text{s.t.} \quad \sigma(x) \leq \sigma^\star \qquad [8.4]$$

2) or minimizing the volatility of the portfolio under a return constraint:

$$\min \sigma(x) \quad \text{s.t.} \quad \mu(x) \geq \mu^\star \qquad [8.5]$$

Markowitz's bright idea was to consider a quadratic utility function $\mathcal{U}(x) = x^\top \mu - \frac{\phi}{2} x^\top \Sigma x$, where $\phi \geq 0$ is the risk aversion. Since maximizing $\mathcal{U}(x)$ is equivalent to minimizing $-\mathcal{U}(x)$, the Markowitz problems [8.4] and [8.5] can be cast into a QP problem[3]:

$$x^\star(\gamma) = \arg\min_x \frac{1}{2} x^\top \Sigma x - \gamma x^\top \mu \qquad \text{s.t.} \quad \mathbf{1}_n^\top x = 1 \qquad [8.6]$$

where $\gamma = \phi^{-1}$. Therefore, solving the μ-problem or the σ-problem is equivalent to finding the optimal value of γ such that $\mu(x^\star(\gamma)) = \mu^\star$ or $\sigma(x^\star(\gamma)) = \sigma^\star$. We know that the functions $\mu(x^\star(\gamma))$ and $\sigma(x^\star(\gamma))$ are increasing with respect to γ and are bounded. The optimal value of γ can then be easily computed using the bisection algorithm. It is obvious that a large part of the success of the Markowitz framework lies on the QP trick. Indeed, problem [8.6] corresponds to the QP problem [8.2], where $Q = \Sigma$, $R = \gamma\mu$, $A = \mathbf{1}_n^\top$ and $B = 1$. Moreover, it is easy to include bounds on the weights, inequalities between asset classes, etc.

8.2.2.2. *Solving complex MVO problems*

The previous framework can be extended to other portfolio allocation problems. However, from a numerical point of view, the underlying idea is to always find an equivalent QP formulation (Roncalli 2013).

8.2.2.2.1. Portfolio optimization with a benchmark

We now consider a benchmark b. We note $\mu(x \mid b) = (x - b)^\top \mu$ as the expected excess return and $\sigma(x \mid b) = \sqrt{(x - b)^\top \Sigma (x - b)}$ as the tracking

[3] This transformation is called the QP trick.

error volatility of Portfolio x with respect to Benchmark b. The objective function corresponds to a trade-off between minimizing the tracking error volatility and maximizing the expected excess return (or the alpha):

$$f(x \mid b) = \frac{1}{2}\sigma^2(x \mid b) - \gamma\mu(x \mid b)$$

We can show that the equivalent QP problem is:

$$x^\star(\gamma) = \arg\min_x \frac{1}{2}x^\top \Sigma x - \gamma x^\top \tilde{\mu}$$

where $\tilde{\mu} = \mu + \gamma^{-1}\Sigma b$ is the regularized vector of expected returns. Therefore, portfolio allocation with a benchmark can be viewed as a regularization of the MVO problem and is solved using a QP numerical algorithm.

8.2.2.2.2. Index sampling

The goal of index sampling is to replicate an index portfolio with a smaller number of assets than the index (or the benchmark) b. From a mathematical point of view, index sampling could be written as follows:

$$\begin{aligned} x^\star &= \arg\min_x \frac{1}{2}(x-b)^\top \Sigma (x-b) \\ & \text{s.t.} \begin{cases} \mathbf{1}_n^\top x = 1 \\ x \geq \mathbf{0}_n \\ \sum_{i=1}^n \quad \{x_i > 0\} \leq n_x \end{cases} \end{aligned} \qquad [8.7]$$

The idea is to minimize the volatility of the tracking error such that the number of stocks n_x in the portfolio is smaller than the number of stocks n_b in the benchmark. For example, we would like to replicate the S&P 500 index with only 50 stocks and not the entire 500 stocks that compose this index. Professionals generally solve problem [8.7] with the following heuristic algorithm:

1) We set $x^{+}_{(0)} = \mathbf{1}_n$. At the iteration $k+1$, we solve the QP problem:

$$\begin{aligned} x^{\star} &= \arg\min_{x} \frac{1}{2}\left(x-b\right)^{\top}\Sigma\left(x-b\right) \\ & \text{s.t.} \left\{ \begin{array}{l} \mathbf{1}_n^{\top} x = 1 \\ \mathbf{0}_n \leq x \leq x^{+}_{(k)} \end{array} \right. \end{aligned}$$

2) We then update the upper bound $x^{+}_{(k)}$ of the QP problem by deleting the asset $i^{\star}$ with the lowest non-zero optimized weight[4]:

$$x^{+}_{(k+1),i} \leftarrow x^{+}_{(k),i} \quad \text{if } i \neq i^{\star} \quad \text{and} \quad x^{+}_{(k+1),i^{\star}} \leftarrow 0$$

3) We iterate the two steps until $\sum_{i=1}^{n} \quad \{x_i^{\star} > 0\} = n_x$.

The purpose of the heuristic algorithm is to delete one asset at each iteration in order to obtain an invested portfolio, which is composed exactly of n_x assets and has a low tracking error volatility. Again, we note that solving the index sampling problem is equivalent to solving $(n_b - n_x)$ QP problems.

8.2.2.2.3. Turnover management

If we note $\bar{x}$ as the current portfolio and x as the new portfolio, the turnover of portfolio x with respect to portfolio $\bar{x}$ is the sum of purchases and sales:

$$\boldsymbol{\tau}\left(x \mid \bar{x}\right) = \sum_{i=1}^{n}\left(x_i - \bar{x}_i\right)^{+} + \sum_{i=1}^{n}\left(\bar{x}_i - x_i\right)^{+} = \sum_{i=1}^{n}\left|x_i - \bar{x}_i\right|$$

Adding a turnover constraint in long-only MVO portfolios leads to the following problem:

$$\begin{aligned} x^{\star} &= \arg\min_{x} \frac{1}{2} x^{\top}\Sigma x - \gamma x^{\top}\mu \\ & \text{s.t.} \left\{ \begin{array}{l} \sum_{i=1}^{n} x_i = 1 \\ \sum_{i=1}^{n} \left|x_i - \bar{x}_i\right| \leq \boldsymbol{\tau}^{+} \\ 0 \leq x_i \leq 1 \end{array} \right. \end{aligned}$$

[4] We have $i^{\star} = \{i : \arg\inf x_i^{\star} \,|\, x_i^{\star} > 0\}$.

where $\boldsymbol{\tau}^{+}$ is the maximum turnover with respect to the current portfolio $\bar{x}$. Scherer (2007) introduces the additional variables x_i^- and x_i^+ such that $x_i = \bar{x}_i + x_i^+ - x_i^-$, where $x_i^- \geq 0$ indicates a negative weight change with respect to the initial weight $\bar{x}_i$ and $x_i^+ \geq 0$ indicates a positive weight change. The expression of the turnover becomes:

$$\sum_{i=1}^{n} \left| x_i - \bar{x}_i \right| = \sum_{i=1}^{n} \left| x_i^+ - x_i^- \right| = \sum_{i=1}^{n} x_i^+ + \sum_{i=1}^{n} x_i^-$$

because one of the variables x_i^+ or x_i^- is necessarily equal to zero due to the minimization problem. The γ-problem of Markowitz becomes:

$$\begin{aligned}
x^{\star} &= \arg\min_{x} \frac{1}{2} x^{\top} \Sigma x - \gamma x^{\top} \mu \\
&\text{s.t.} \begin{cases}
\sum_{i=1}^{n} x_i = 1 \\
x_i = \bar{x}_i + x_i^+ - x_i^- \\
\sum_{i=1}^{n} x_i^+ + \sum_{i=1}^{n} x_i^- \leq \boldsymbol{\tau}^{+} \\
0 \leq x_i, x_i^-, x_i^+ \leq 1
\end{cases}
\end{aligned}$$

We obtain an augmented QP problem of dimension $3n$ (see Appendix 8.7.1.1).

8.2.2.2.4. Transaction costs

The previous analysis assumes that there is no transaction cost $\boldsymbol{c}(x \mid \bar{x})$ when we rebalance the portfolio from the current portfolio $\bar{x}$ to the new optimized portfolio x. If we note c_i^- and c_i^+ as the bid and ask transaction costs, then we have:

$$\boldsymbol{c}(x \mid \bar{x}) = \sum_{i=1}^{n} x_i^- c_i^- + \sum_{i=1}^{n} x_i^+ c_i^+$$

The net expected return of portfolio x is then equal to $\mu(x) - c(x \mid \bar{x})$. It follows that the γ-problem of Markowitz becomes[5]:

$$
\begin{aligned}
x^{\star} &= \arg\min_{x} \frac{1}{2} x^{\top} \Sigma x - \gamma \left(\sum_{i=1}^{n} x_i \mu_i - \sum_{i=1}^{n} x_i^{-} c_i^{-} - \sum_{i=1}^{n} x_i^{+} c_i^{+} \right) \\
& \text{s.t.} \left\{ \begin{array}{l} \sum_{i=1}^{n} x_i + \sum_{i=1}^{n} x_i^{-} c_i^{-} + \sum_{i=1}^{n} x_i^{+} c_i^{+} = 1 \\ x_i = \bar{x}_i + x_i^{+} - x_i^{-} \\ 0 \leq x_i, x_i^{-}, x_i^{+} \leq 1 \end{array} \right.
\end{aligned}
$$

Once again, we obtain a QP problem of dimension $3n$.

8.2.3. *Issues with QP optimization*

The concurrent model of the Markowitz framework is the risk budgeting approach (Qian 2005; Maillard *et al.* 2010; Roncalli 2013). The goal is to define a convex risk measure $\mathcal{R}(x)$ and to allocate the risk according to some specified risk budgets $\mathcal{RB} = (\mathcal{RB}_1, \ldots, \mathcal{RB}_n)$, where $\mathcal{RB}_i > 0$. This approach exploits the Euler decomposition property of the risk measure:

$$
\mathcal{R}(x) = \sum_{i=1}^{n} x_i \frac{\partial \, \mathcal{R}(x)}{\partial \, x_i}
$$

By noting $\mathcal{RC}_i(x) = x_i \cdot \partial_{x_i} \mathcal{R}(x)$ as the risk contribution of asset i with respect to portfolio x, the risk budgeting (RB) portfolio is defined by the following set of equations: $x^{\star} = \{x \in [0,1]^n : \mathcal{RC}_i(x) = \mathcal{RB}_i\}$. Roncalli (2013) showed that it is equivalent to solving the following nonlinear optimization problem[6]:

$$
x^{\star} = \arg\min_{x} \mathcal{R}(x) - \lambda \sum_{i=1}^{n} \mathcal{RB}_i \cdot \ln x_i \qquad [8.8]
$$

[5] The equality constraint $1_n^{\top} x = 1$ becomes $1_n^{\top} x + c(x \mid \bar{x}) = 1$ because the rebalancing process has to be financed.

[6] In fact, the solution $x^{\star}$ must be rescaled after the optimization step.

where $\lambda > 0$ is an arbitrary positive constant. Generally, the most frequently used risk measures are the volatility risk measure $\mathcal{R}(x) = \sqrt{x^\top \Sigma x}$ (Maillard *et al.* 2010) and the standard deviation-based risk measure $\mathcal{R}(x) = -x^\top (\mu - r) + \xi \sqrt{x^\top \Sigma x}$, where r is the risk-free rate and ξ is a positive scalar (Roncalli 2015). In particular, this last one encompasses the Gaussian value-at-risk – $\xi = \Phi^{-1}(\alpha)$ – and the Gaussian expected shortfall – $\xi = (1-\alpha)^{-1} \phi\left(\Phi^{-1}(\alpha)\right)$.

The risk budgeting approach has displaced the MVO approach in many fields of asset management, in particular in the case of factor investing and alternative risk premia. Nevertheless, we note that problem [8.8] is a not a quadratic programming problem, but a logarithmic barrier problem. Therefore, the risk budgeting framework opens a new world of portfolio optimization that is not necessarily QP! That is all the more true since MVO portfolios face robustness issues (Bourgeron *et al.* 2018). Regularization of portfolio allocation has then become the industry standard. Indeed, it is frequent to add an ℓ_1-norm or ℓ_2-norm penalty function to the MVO objective function. This type of penalty is, however, tractable in a quadratic programming setting. With the development of robo-advisors, nonlinear penalty functions have emerged, in particular the logarithmic barrier penalty function. And these regularization techniques result in a non-quadratic programming world of portfolio optimization.

The success of this non-QP financial world will depend on how quickly and easily these complex optimization problems can be solved. Griveau-Billon *et al.* (2013), Bourgeron *et al.* (2018) and Richard and Roncalli (2019) have already proposed numerical algorithms that are doing the work in some special cases. The next section reviews the candidate algorithms that may compete with QP numerical algorithms.

8.3. Machine learning optimization algorithms

The machine learning industry has experienced a similar trajectory to portfolio optimization. Before the 1990s, statistical learning focused mainly on models that were easy to solve from a numerical point of view. For instance, the linear (and the ridge) regression has an analytical solution, we can solve logistic regression with the Newton–Raphson algorithm, whereas supervised and unsupervised classification models[7] consist of performing a singular value decomposition or a generalized eigenvalue decomposition. The 1990s saw the emergence of three models that have deeply changed the machine learning approach: neural networks, support vector machines and lasso regression.

Neural networks have been extensively studied since the seminal work of Rosenblatt (1958). However, the first industrial application dates back to the publication of LeCun *et al.* (1989) on handwritten zip code recognition. At the beginning of the 1990s, a fresh craze then emerged with the writing of many handbooks that were appropriate for students, the most popular of which was Bishop (1995). With neural networks, two main issues arise concerning calibration: the large number of parameters to estimate and the absence of a global maximum. The traditional numerical optimization algorithms[8] that were popular in the 1980s cannot be applied to neural networks. New optimization approaches are then proposed. First, researchers have considered more complex learning rules than the steepest descent (Jacobs 1988), for example, the momentum method of Polyak (1964) or the Nesterov accelerated gradient approach (Nesterov 1983). Second, the descent method is generally not performed on the full sample of observations, but on a subset of observations that changes at each iteration. This is the underlying idea behind batch gradient descent (BGD), stochastic descent gradient (SGD) and mini-batch gradient descent (MGD). We note that adaptive learning methods and batch optimization techniques have marked the revival of the gradient descent method.

The development of support vector machines is another important step in the development of machine learning techniques. Like neural networks, they can be seen as an extension of the perceptron. However, they present nice theoretical properties and a strong geometrical framework. Once SVMs have been first developed for linear classification, they have been extended for nonlinear classification and regression. A support vector machine consists of separating hyperplanes and finding the optimal separation by maximizing the

[7] For example, principal component analysis (PCA), linear/quadratic discriminant analysis (LDA/QDA) and Fisher classification method.

[8] For example, we can cite the quasi-Newton BFGS (Broyden–Fletcher–Goldfarb–Shanno) and DFP (Davidon–Fletcher–Powell) methods and the Fletcher–Reeves and Polak–Ribiere conjugate gradient methods.

margin. The original problem called the hard margin classification can be formulated as a quadratic programming problem. However, the dual problem, which is also a QP problem, is generally preferred to the primal problem for solving SVM classification, because of the sparse property of the solution (Cortes and Vapnik 1995). Over the years, the original hard margin classification has been extended to other problems: soft margin classification with binary hinge loss, soft margin classification with squared hinge loss, least squares SVM regression, ε-SVM regression, kernel machines (Vapnik 1998). All these statistical problems share the same calibration framework. The primal problem can be cast into a QP problem, implying that the corresponding dual problem is also a QP problem. Again, we note that the success and the prominence of statistical methods are related to the efficiency of the optimization algorithms and it is obvious that support vector machines have substantially benefited from the QP formulation. From an industrial point of view however, support vector machines present some limitations. Indeed, if the dimension of the primal QP problem is the number p of features (or parameters), the dimension of the dual QP problem is the number n of observations. It is becoming absolutely impossible to solve the dual problem when the number of observations is larger than 100,000 and sometimes as high as several millions. This implies that new algorithms that are more appropriate for large-scale optimization problems need to be developed.

Lasso regression is the third disruptive approach that put machine learning in the spotlight in the 1990s. Like the ridge regression, lasso regression is a regularized linear regression where the ℓ_2-norm penalty is replaced by the ℓ_1-norm penalty (Tibshirani 1996). Since the ℓ_1 regularization forces the solution to be sparse, it has been largely used first for variable selection and then for pattern recognition and robust estimation of linear models. For finding the lasso solution, the technique of augmented QP problems is widely used since it is easy to implement. The extension of the lasso-ridge regularization to the other ℓ_p norms is straightforward but these approaches have never been popular. The main reason is that existing numerical algorithms are not sufficient to make these models tractable.

Therefore, the success of a quantitative model may be explained by two conditions. First, the model must be obviously appealing. Second, the model must be solved by an efficient numerical algorithm that is easy to implement or available in mathematical programming software. As shown previously, quadratic programming and gradient descent methods have been key for many statistical and financial models. In what follows, we consider four algorithms and techniques that have been popularized by their use in machine learning: coordinate descent, alternating direction method of multipliers, proximal operators and Dykstra's algorithm. In particular, we illustrate how they can be used for solving complex optimization problems.

8.3.1. *Coordinate descent*

8.3.1.1. *Definition*

We consider the following unconstrained minimization problem:

$$x^{\star} = \arg\min_{x} f\left(x\right) \qquad [8.9]$$

where $x \in \mathbb{R}^n$ and $f\left(x\right)$ is a continuous, smooth and convex function. A popular method to find the solution $x^{\star}$ is to consider the descent algorithm, which is defined by the following rule:

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

where $x^{(k)}$ is the approximated solution of problem [8.9] at the k^{th} iteration, $\eta > 0$ is a scalar that determines the step size and $D^{(k)}$ is the direction. We note that the current solution $x^{(k)}$ is updated by going in the opposite direction to $D^{(k)}$ in order to obtain $x^{(k+1)}$. In the case of the gradient descent, the direction is equal to the gradient vector of $f\left(x\right)$ at the current point: $D^{(k)} = \nabla f\left(x^{(k)}\right)$. Coordinate descent (CD) is a variant of the gradient descent and minimizes the function along one coordinate at each step:

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

where $D_i^{(k)} = \nabla_i f\left(x^{(k)}\right)$ is the i^{th} element of the gradient vector. At each iteration, a coordinate i is then chosen via a certain rule, while the other coordinates are assumed to be fixed. Coordinate descent is an appealing algorithm because it transforms a vector-valued problem into a scalar-valued problem that is easier to implement. Another formulation of the coordinate descent method is to replace the descent approximation by the exact problem.

Indeed, the objective of the descend step is to minimize the scalar-valued problem:

$$x_i^\star = \arg\min_{\varkappa} f\left(x_1^{(k)}, \dots, x_{i-1}^{(k)}, \varkappa, x_{i+1}^{(k)}, \dots, x_n^{(k)}\right) \quad [8.10]$$

Coordinate descent is efficient in large-scale optimization problems, in particular when there is a solution to the scalar-valued problem [8.10]. Furthermore, convergence is guaranteed when $f(x)$ is convex and differentiable.

REMARK 8.1.– Coordinate descent methods were introduced in several handbooks on numerical optimization in the 1980s and 1990s (Wright 2015). However, the most important step is the contribution of Tseng (2001), who studied the block-coordinate descent method and extended CD algorithms in the case of a non-differentiable and non-convex function $f(x)$.

8.3.1.2. *Cyclic or random coordinates?*

There are several options for choosing the coordinate of the k^{th} iteration. A natural choice could be to choose the coordinate, which minimizes the function:

$$i^\star = \arg\inf \left\{ f_i^\star : i \in \{1, n\}, f_i^\star = \min_{\varkappa} f\left((1 - e_i)\, x^{(k)} + e_i \varkappa\right) \right\}$$

However, it is obvious that choosing the optimal coordinate $i^\star$ would require the gradient along each coordinate to be calculated. This causes the coordinate descent to be no longer efficient, since a classic gradient descent would then be of equivalent cost at each iteration and would converge faster because it requires fewer iterations.

The simplest way to implement the CD algorithm is to consider cyclic coordinates, meaning that we cyclically iterate through the coordinates (Tseng 2001): $i = k \bmod n$. This ensures that all the coordinates are selected during one cycle $\{k - n + 1, \dots, k\}$ in the same order. This approach, called cyclical coordinate descent (CCD), is the most popular and most used method, even if it is difficult to estimate the rate of convergence.

The second way is to consider random coordinates. Let π_i be the probability of choosing the coordinate i at the iteration k. The simplest approach is to consider uniform probabilities: $\pi_i = 1/n$. A better approach consists of pre-specifying probabilities according to the Lipschitz constants[9]: $\pi_i = \mathfrak{L}_i^\alpha / \sum_{j=1}^n \mathfrak{L}_j^\alpha$. Nesterov (2012) considers three schemes: $\alpha = 0$, $\alpha = 1$ and $\alpha = \infty$ – in this last case, we have $i = \arg\max \{\mathfrak{L}_1, \ldots, \mathfrak{L}_n\}$. From a theoretical point of view, the random coordinate descent (RCD) method based on the probability distribution π leads to a faster convergence, since coordinates that have a large Lipschitz constant $\mathfrak{L}_i$ are more likely to be chosen. However, it requires additional calculus to compute the Lipschitz constants and CCD is often preferred from a practical point of view. In what follows, we only use the CCD algorithm described below. In Algorithm [8.1.], the variable k represents the number of cycles, whereas the number of iterations is equal to $k \cdot n$. For the coordinate i, the lower coordinates $j < i$ correspond to the current cycle $k + 1$, while the upper coordinates $j > i$ correspond to the previous cycle k.

Algorithm 8.1. Cyclical coordinate descent algorithm

```
The goal is to find the solution $x^\star = \arg\min f(x)$
We initialize the vector $x^{(0)}$
Set $k \leftarrow 0$
repeat
    for $i = 1 : n$ do
        $x_i^{(k+1)} = \arg\min_\varkappa f\left(x_1^{(k+1)}, \ldots, x_{i-1}^{(k+1)}, \varkappa, x_{i+1}^{(k)}, \ldots, x_n^{(k)}\right)$
    end for
    $k \leftarrow k + 1$
until convergence
return $x^\star \leftarrow x^{(k)}$
```

8.3.1.3. *Application to the λ-problem of the lasso regression*

We consider the linear regression $Y = X\beta + \varepsilon$, where Y is the $n \times 1$ vector, X is the $n \times p$ design matrix, β is the $p \times 1$ vector of coefficients and ε is the $n \times 1$ vector of residuals. In this model, n is the number of observations and p is the number of parameters (or the number of explanatory variables). The objective of the ordinary least squares is to minimize the residual sum of squares $\hat{\beta} = \arg\min_\beta \frac{1}{2} \operatorname{RSS}(\beta)$, where $\operatorname{RSS}(\beta) =$

[9] Nesterov (2012) assumes that $f(x)$ is convex, differentiable and Lipschitz smooth for each coordinate: $\|\nabla_i f(x + e_i h) - \nabla_i f(x)\| \leq \mathfrak{L}_i \|h\|$, where $h \in \mathbb{R}$.

$\sum_{i=1}^{n} \varepsilon_i^2$. Since we have $\text{RSS}(\beta) = (Y - X\beta)^\top (Y - X\beta)$, we obtain $\partial_{\beta_j} f(\beta) = -x_j^\top (Y - X\beta)$, where x_j is the $n \times 1$ design vector corresponding to the j^{th} explanatory variable. Because we can write $X\beta = X_{(-j)}\beta_{(-j)} + x_j\beta_j$, where $X_{(-j)}$ and $\beta_{(-j)}$ are the design matrix and the beta vector by excluding the j^{th} explanatory variable, respectively, it follows that $\partial_{\beta_j} f(\beta) = x_j^\top X_{(-j)}\beta_{(-j)} + x_j^\top x_j\beta_j - x_j^\top Y$. At the optimum, we have $\partial_{\beta_j} f(\beta) = 0$ or:

$$\beta_j = \frac{x_j^\top \left(Y - X_{(-j)}\beta_{(-j)}\right)}{x_j^\top x_j} \qquad [8.11]$$

The implementation of the coordinate descent algorithm is straightforward. It suffices to iterate equation [8.11] through the coordinates.

The lasso regression problem is a variant of the OLS regression by adding an ℓ_1-norm regularization (Tibshirani 1996):

$$\hat{\beta}(\lambda) = \arg\min_{\beta} \frac{1}{2} (Y - X\beta)^\top (Y - X\beta) + \lambda \|\beta\|_1 \qquad [8.12]$$

In this formulation, the residual sum of squares of the linear regression is penalized by a term that will force a sparse selection of the coordinates. Since the objective function is the sum of two convex norms, the convergence is guaranteed for the lasso problem. Because $\|\beta\|_1 = \sum_{j=1}^{n} |\beta_j|$, the first-order condition becomes $x_j^\top x_j\beta_j - x_j^\top \left(Y - X_{(-j)}\beta_{(-j)}\right) + \lambda\partial |\beta_j| = 0$. In Appendix 8.7.1.3, we show that the solution is given by:

$$\beta_j = \frac{\mathcal{S}\left(x_j^\top \left(Y - X_{(-j)}\beta_{(-j)}\right); \lambda\right)}{x_j^\top x_j} \qquad [8.13]$$

where $\mathcal{S}(v; \lambda)$ is the soft-thresholding operator $\mathcal{S}(v; \lambda) = \text{sign}(v) \cdot (|v| - \lambda)_+$. It follows that the lasso CD algorithm is a variation of the linear regression CD algorithm by applying the soft-threshold operator to the residuals $x_j^\top \left(Y - X_{(-j)}\beta_{(-j)}\right)$ at each iteration.

Let us consider an experiment with $n = 10,0000$ and $p = 50$. The design matrix X is built using the uniform distribution, while the residuals are simulated using a Gaussian distribution and a standard deviation of 20%. The beta coefficients are distributed uniformly between -3 and $+3$ except four coefficients that take a larger value. We then standardize the data of X and Y because the practice of the lasso regression is to consider comparable beta coefficients. By considering uniform numbers between -1 and $+1$ for initializing the coordinates, results of the CCD algorithm are given in Figure 8.1. We note that the CCD algorithm quickly converges after three complete cycles. In the case of a large-scale problem when $p \gg 1\,000$, it has been shown that CCD may be faster for the lasso regression than for the OLS regression because of the soft-thresholding operator. Indeed, we can initialize the algorithm with the null vector $\mathbf{0}_p$. If λ is large, a lot of optimal coordinates are equal to zero and a few cycles are needed to find the optimal values of non-zero coefficients.

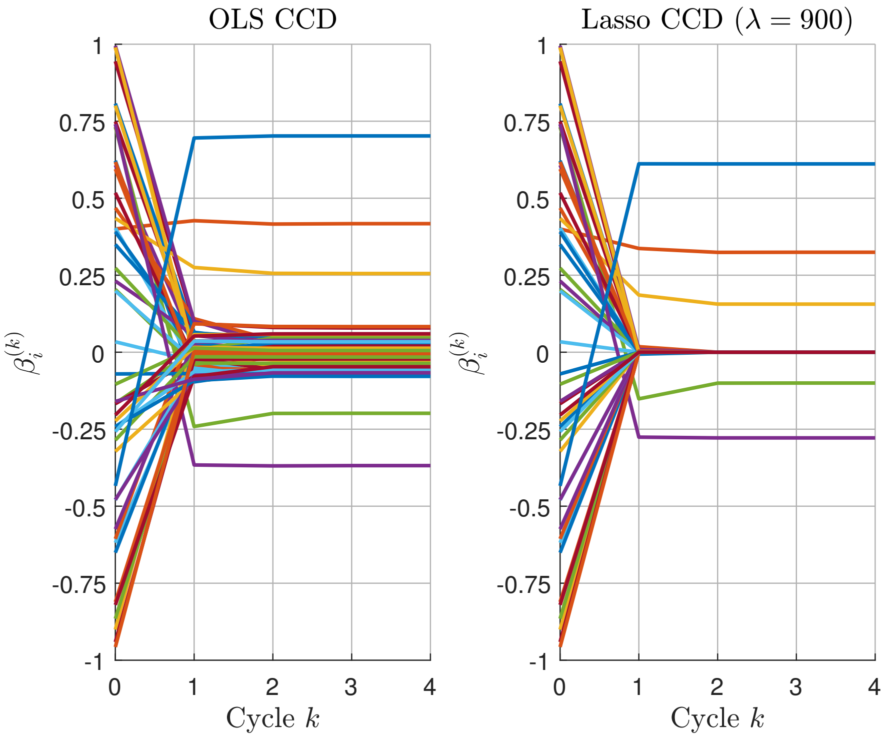

Figure 8.1. *CCD algorithm applied to the lasso optimization problem. For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

8.3.2. *Alternating direction method of multipliers*

8.3.2.1. *Definition*

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

$$\begin{aligned}\{x^\star, y^\star\} &= \arg\min_{(x,y)} f_x(x) + f_y(y) \\ &\qquad \text{s.t. } Ax + By = c\end{aligned} \qquad [8.14]$$

where $A \in \mathbb{R}^{p\times n}$, $B \in \mathbb{R}^{p\times m}$, $c \in \mathbb{R}^p$ and the functions $f_x : \mathbb{R}^n \to \mathbb{R}\cup\{+\infty\}$ and $f_y : \mathbb{R}^m \to \mathbb{R}\cup\{+\infty\}$ are proper closed convex functions. Boyd *et al.* (2011) show that the ADMM algorithm consists of the following three steps:

1) the x-update is:

$$x^{(k+1)} = \arg\min_x \left\{ f_x(x) + \frac{\varphi}{2} \left\| Ax + By^{(k)} - c + u^{(k)} \right\|_2^2 \right\}; \qquad [8.15]$$

2) the y-update is:

$$y^{(k+1)} = \arg\min_y \left\{ f_y(y) + \frac{\varphi}{2} \left\| Ax^{(k+1)} + By - c + u^{(k)} \right\|_2^2 \right\}; \qquad [8.16]$$

3) the u-update is:

$$u^{(k+1)} = u^{(k)} + \left(Ax^{(k+1)} + By^{(k+1)} - c \right). \qquad [8.17]$$

In this approach, $u^{(k)}$ is the dual variable of the primal residual $r = Ax+By-c$ and φ is the ℓ_2-norm penalty variable. The parameter φ can be constant or may change at each iteration. The ADMM algorithm benefits from the dual ascent principle and the method of multipliers. The difference with the latter is that the x- and y-updates are performed in an alternating way. Therefore, it is more flexible because the updates are equivalent to computing proximal operators for f_x and f_y independently. In practice, ADMM may be slow to converge with high accuracy but is fast to converge if we consider modest accuracy. Hence, ADMM is a good candidate for solving large-scale machine learning problems, where high accuracy does not necessarily lead to a better solution.

REMARK 8.2.– In this chapter, we use the notations $f_x^{(k+1)}(x)$ and $f_y^{(k+1)}(y)$ when referring to the objective functions that are defined in the x- and y-updates.

8.3.2.2. *ADMM tricks*

The appeal of ADMM is that it can separate a complex problem into two sub-problems that are easier to solve. However, most of the time, the optimization problem is not formulated using a separable objective function. The question is then how to formulate the initial problem as a separable problem. We now list some tricks that show how ADMM may be used in practice.

8.3.2.2.1. First trick

We consider a problem of the form $x^\star = \arg\min_x g(x)$. The idea is then to write $g(x)$ as a separable function $g(x) = g_1(x) + g_2(x)$ and to consider the following equivalent ADMM problem:

$$\begin{aligned} \{x^\star, y^\star\} &= \arg\min_{(x,y)} f_x(x) + f_y(y) \\ &\quad \text{s.t. } x = y \end{aligned} \qquad [8.18]$$

where $f_x(x) = g_1(x)$ and $f_y(y) = g_2(y)$. Usually, the smooth part of $g(x)$ will correspond to $g_1(x)$ while the non-smooth part will be included in $g_2(y)$. The underlying idea is that the x-update is straightforward, whereas the y-update deals with the tricky part of $g(x)$.

8.3.2.2.2. Second trick

If we want to minimize the function $g(x)$, where $x \in \Omega$ is a set of constraints, the optimization problem can be cast into the ADMM form [8.18], where $f_x(x) = g(x)$, $f_y(y) = {}_{\Omega}(y)$ and ${}_{\Omega}(x)$ is the convex indicator function of Ω:

$${}_{\Omega}(x) = \begin{cases} 0 & \text{if } x \in \Omega \\ +\infty & \text{if } x \notin \Omega \end{cases} \qquad [8.19]$$

For example, if we want to solve the QP problem [8.2] given on page 264, we have $f_x(x) = \frac{1}{2}x^\top Qx - x^\top R$ and $\Omega = \{x \in \mathbb{R}^n : Ax = B, Cx \leq D, x^- \leq x \leq x^+\}$.

8.3.2.2.3. Third trick

We can combine the first and second tricks. For instance, if we consider the following optimization problem:

$$\begin{aligned} x^\star &= \arg\min_x g_1(x) + g_2(x) \\ &\text{s.t. } x \in \Omega_1 \cap \Omega_2 \end{aligned}$$

the equivalent ADMM form is:

$$\begin{aligned} \{x^\star, y^\star\} &= \arg\min_{(x,y)} \underbrace{\left(g_1(x) + \ {}_{\Omega_1}(x)\right)}_{f_x(x)} + \underbrace{\left(g_2(y) + \ {}_{\Omega_2}(y)\right)}_{f_y(y)} \\ &\text{s.t. } x = y \end{aligned}$$

Let us consider a variant of the QP problem where we add a nonlinear constraint $h(x) = 0$. In this case, we can write the set of constraints as $\Omega = \Omega_1 \cap \Omega_2$, where $\Omega_1 = \{x \in \mathbb{R}^n : Ax = B, Cx \leq D, x^- \leq x \leq x^+\}$ and $\Omega_2 = \{x \in \mathbb{R}^n : h(x) = 0\}$.

8.3.2.2.4. Fourth trick

Finally, if we want to minimize the function $g(x) = g(x, Ax + b) = g_1(x) + g_2(Ax + b)$, we can write:

$$\begin{aligned} \{x^\star, y^\star\} &= \arg\min_{(x,y)} g_1(x) + g_2(y) \\ &\text{s.t. } y = Ax + b \end{aligned}$$

For instance, this trick can be used for a QP problem with a nonlinear part:

$$g(x) = \frac{1}{2} x^\top Q x - x^\top R + h(x)$$

If we assume that Q is a symmetric positive-definite matrix, we set $x = Ly$, where L is the lower Cholesky matrix such that $LL^\top = Q$. It follows that the

ADMM form is equal to[10]:

$$\begin{aligned}\{x^\star, y^\star\} &= \arg\min_{(x,y)} \underbrace{\frac{1}{2}x^\top x}_{f_x(x)} + \underbrace{h(y) - y^\top R}_{f_y(y)} \\ &\quad \text{s.t. } x - Ly = \mathbf{0}_n\end{aligned}$$

We note that the x-update is straightforward because it corresponds to a standard QP problem. If we add a set Ω of constraints, we specify $f_y(y) = h(y) - y^\top R + \ {}_{\Omega}(y)$.

REMARK 8.3.– In the previous cases, we have seen that when the function $g(x)$ may contain a QP problem, it is convenient to isolate this QP problem into the x-update:

$$x^{(k+1)} = \arg\min_x \left\{\frac{1}{2}x^\top Qx - x^\top R + \ {}_{\Omega}(x) + \frac{\varphi}{2}\left\|x - y^{(k)} + u^{(k)}\right\|_2^2\right\}$$

Since we have:

$$\begin{aligned}\frac{\varphi}{2}\left\|x - y^{(k)} + u^{(k)}\right\|_2^2 &= \frac{\varphi}{2}x^\top x - \varphi x^\top\left(y^{(k)} - u^{(k)}\right) \\ &\quad + \frac{\varphi}{2}\left(y^{(k)} - u^{(k)}\right)^\top\left(y^{(k)} - u^{(k)}\right)\end{aligned}$$

we deduce that the x-update is a standard QP problem where:

$$f_x^{(k+1)}(x) = \frac{1}{2}x^\top(Q + \varphi I_n)x - x^\top\left(R + \varphi\left(y^{(k)} - u^{(k)}\right)\right) + \ {}_{\Omega}(x) \qquad [8.20]$$

8.3.2.3. *Application to the λ-problem of the lasso regression*

The λ-problem of the lasso regression [8.12] has the following ADMM formulation:

$$\begin{aligned}\{\beta^\star, \bar{\beta}^\star\} &= \arg\min \frac{1}{2}(Y - X\beta)^\top(Y - X\beta) + \lambda\|\bar{\beta}\|_1 \\ &\quad \text{s.t. } \beta - \bar{\beta} = \mathbf{0}_p\end{aligned}$$

[10] This Cholesky trick has been used by Gonzalvez *et al.* (2019) to solve trend-following strategies using the ADMM algorithm in the context of Bayesian learning.

Since the x-step corresponds to a QP problem[11], we use the results given in Remark 8.3 to find the value of $\beta^{(k+1)}$:

$$\begin{aligned}\beta^{(k+1)} &= (Q+\varphi I_p)^{-1}\left(R+\varphi\left(\bar{\beta}^{(k)}-u^{(k)}\right)\right)\\ &= \left(X^\top X+\varphi I_p\right)^{-1}\left(X^\top Y+\varphi\left(\bar{\beta}^{(k)}-u^{(k)}\right)\right)\end{aligned}$$

The y-step is:

$$\begin{aligned}\bar{\beta}^{(k+1)} &= \arg\min_{\bar{\beta}}\left\{\lambda\|\bar{\beta}\|_1+\frac{\varphi}{2}\left\|\beta^{(k+1)}-\bar{\beta}+u^{(k)}\right\|_2^2\right\}\\ &= \arg\min\left\{\frac{1}{2}\left\|\bar{\beta}-\left(\beta^{(k+1)}+u^{(k)}\right)\right\|_2^2+\frac{\lambda}{\varphi}\|\bar{\beta}\|_1\right\}\end{aligned}$$

We recognize the soft-thresholding problem with $v=\beta^{(k+1)}+u^{(k)}$. Finally, the ADMM algorithm is made up of the following steps (Boyd *et al.* 2011):

$$\begin{cases}\beta^{(k+1)}=\left(X^\top X+\varphi I_p\right)^{-1}\left(X^\top Y+\varphi\left(\bar{\beta}^{(k)}-u^{(k)}\right)\right)\\ \bar{\beta}^{(k+1)}=\mathcal{S}\left(\beta^{(k+1)}+u^{(k)};\varphi^{-1}\lambda\right)\\ u^{(k+1)}=u^{(k)}+\left(\beta^{(k+1)}-\bar{\beta}^{(k+1)}\right)\end{cases}$$

8.3.3. *Proximal operators*

The x- and y-update steps of the ADMM algorithm require an ℓ_2-norm penalized optimization problem to be solved. Proximal operators are special cases of this type of problem when the matrices A or B correspond to the identity matrix I_n or its opposite $-I_n$.

8.3.3.1. *Definition*

Let $f:\mathbb{R}^n\rightarrow\mathbb{R}\cup\{+\infty\}$ be a proper closed convex function. The proximal operator $\mathbf{prox}_f(v):\mathbb{R}^n\rightarrow\mathbb{R}^n$ is defined by:

$$\mathbf{prox}_f(v)=x^\star=\arg\min_x\left\{f_v(x)=f(x)+\frac{1}{2}\|x-v\|_2^2\right\}\qquad[8.21]$$

[11] We have $Q=X^\top X$ and $R=X^\top Y$.

Since the function $f_v(x) = f(x) + \frac{1}{2}\|x - v\|_2^2$ is strongly convex, it has a unique minimum for every $v \in \mathbb{R}^n$ (Parikh and Boyd 2014). By construction, the proximal operator defines a point $x^\star$ which is a trade-off between minimizing $f(x)$ and being close to v.

In many situations, we need to calculate the proximal of the scaled function $\lambda f(x)$, where $\lambda > 0$. In this case, we use the notation $\mathbf{prox}_{\lambda f}(v)$ and we have:

$$\begin{aligned}\mathbf{prox}_{\lambda f}(v) &= \arg\min_x \left\{ \lambda f(x) + \frac{1}{2}\|x - v\|_2^2 \right\} \\ &= \arg\min_x \left\{ f(x) + \frac{1}{2\lambda}\|x - v\|_2^2 \right\}\end{aligned}$$

For instance, if we consider the y-update of the ADMM algorithm with $B = -I_n$, we have:

$$\begin{aligned}y^{(k+1)} &= \arg\min_y \left\{ f_y(y) + \frac{\varphi}{2}\left\| y - v_y^{(k+1)} \right\|_2^2 \right\} \\ &= \arg\min_y \left\{ \varphi^{-1} f_y(y) + \frac{1}{2}\left\| y - v_y^{(k+1)} \right\|_2^2 \right\} \\ &= \mathbf{prox}_{\varphi^{-1} f_y}\left(v_y^{(k+1)} \right)\end{aligned}$$

where $v_y^{(k)} = Ax^{(k+1)} - c + u^{(k)}$. The interest of this mathematical formulation is to write the ADMM algorithm in a convenient form such that the x-update corresponds to the tricky part of the optimization, while the y-update is reduced to an analytical formula.

8.3.3.2. *Proximal operators and generalized projections*

In the case where $f(x) = \ {}_{\Omega}(x)$ is the indicator function, the proximal operator is then the Euclidean projection onto Ω:

$$\mathbf{prox}_f(v) = \arg\min_x \left\{ \ {}_{\Omega}(x) + \frac{1}{2}\|x - v\|_2^2 \right\}$$

$$= \arg\min_{x\in\Omega} \left\{ \|x - v\|_2^2 \right\}$$
$$= \mathcal{P}_\Omega(v)$$

where $\mathcal{P}_\Omega(v)$ is the standard projection of v onto Ω. Parikh and Boyd (2014) interpret proximal operators then as a generalization of the Euclidean projection.

Let us consider the constrained optimization problem $x^\star = \arg\min f(x)$ subject to $x \in \Omega$. Using the second ADMM trick, we have $f_x(x) = f(x)$, $f_y(y) = {}_{\Omega}(y)$ and $x - y = \mathbf{0}_n$. Therefore, we can use the ADMM algorithm since the v- and y-steps become $v_y^{(k+1)} = x^{(k+1)} + u^{(k)}$ and[12] $y^{(k+1)} = \mathcal{P}_\Omega\left(v_y^{(k+1)}\right)$. Here, we give the results of Parikh and Boyd (2014) for some simple polyhedra:

Notation	Ω	$\mathcal{P}_\Omega(v)$
$\mathcal{A}_{ffineset}[A, B]$	$Ax = B$	$v - A^\dagger(Av - B)$
$\mathcal{H}_{yperplane}[a, b]$	$a^\top x = b$	$v - \dfrac{(a^\top v - b)}{\Vert a\Vert_2^2} a$
$\mathcal{H}_{alfspace}[c, d]$	$c^\top x \leq d$	$v - \dfrac{(c^\top v - d)_+}{\Vert c\Vert_2^2} c$
$\mathcal{B}_{ox}[x^-, x^+]$	$x^- \leq x \leq x^+$	$\mathcal{T}(v; x^-, x^+)$

where $A^\dagger$ is the Moore–Penrose pseudo-inverse of A and $\mathcal{T}(v; x^-, x^+)$ is the truncation operator.

8.3.3.3. *Main properties*

There are many properties that are useful for finding the analytical expression of the proximal operator. In what follows, we consider three main properties. Refer to Combettes and Pesquet (2011), Parikh and Boyd (2014) and Beck (2017) for a more exhaustive list.

[12] We note that the parameter φ has no impact on the y-update because $\varphi^{-1} f_y(y) = f_y(y) = 1_{\Omega}(y)$. We then deduce that:

$$\mathbf{prox}_{\varphi^{-1} f_y}\left(v_y^{(k+1)}\right) = \mathbf{prox}_{f_y}\left(v_y^{(k+1)}\right) = \mathcal{P}_\Omega\left(v_y^{(k+1)}\right).$$

8.3.3.3.1. Separable sum

Let us assume that $f(x) = \sum_{i=1}^{n} f_i(x_i)$ is fully separable, then the proximal of $f(v)$ is the vector of the proximal operators applied to each scalar-valued function $f_i(x_i)$:

$$\mathbf{prox}_f(v) = \begin{pmatrix} \mathbf{prox}_{f_1}(v_1) \\ \vdots \\ \mathbf{prox}_{f_n}(v_n) \end{pmatrix}$$

For example, if $f(x) = \lambda \|x\|_1$, we have $f(x) = \lambda \sum_{i=1}^{n} |x_i|$ and $f_i(x_i) = \lambda |x_i|$. We deduce that the proximal operator of $f(x)$ is the vector formulation of the soft-thresholding operator:

$$\mathbf{prox}_{\lambda\|x\|_1}(v) = \operatorname{sign}(v) \odot (|v| - \lambda \mathbf{1}_n)_+$$

This result has been used to solve the λ-problem of the lasso regression on p. 282.

If we consider the scalar-valued logarithmic barrier function $f(x) = -\lambda \ln x$, we have:

$$f_v(x) = -\lambda \ln x + \frac{1}{2}(x - v)^2 = -\lambda \ln x + \frac{1}{2}x^2 - xv + \frac{1}{2}v^2$$

The first-order condition is $-\lambda x^{-1} + x - v = 0$. We obtain two roots with opposite signs: $x^\star = \frac{1}{2}\left(v \pm \sqrt{v^2 + 4\lambda}\right)$. Since the logarithmic function is defined for $x > 0$, we deduce that the proximal operator is the positive root. In the case of the vector-valued logarithmic barrier $f(x) = -\lambda \sum_{i=1}^{n} \ln x_i$, it follows that:

$$\mathbf{prox}_f(v) = \frac{v + \sqrt{v \odot v + 4\lambda}}{2}$$

8.3.3.3.2. Moreau decomposition

An important property of the proximal operator is the Moreau decomposition theorem:

$$\mathbf{prox}_f(v) + \mathbf{prox}_{f^*}(v) = v$$

where f^* is the convex conjugate of f. This result is used extensively to find the proximal of norms, the max function, the sum-of-k-largest-values function, etc. (Beck 2017).

In the case of the pointwise maximum function $f(x) = \max x$, we can show that:

$$\mathbf{prox}_{\lambda \max x}(v) = \min(v, s^\star)$$

where $s^\star$ is the solution of the following equation:

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

If we assume that $f(x) = \|x\|_p$, we obtain:

p	$\mathbf{prox}_{\lambda f}(v)$
$p = 1$	$\mathcal{S}(v; \lambda) = \operatorname{sign}(v) \odot (\lvert v\rvert - \lambda \mathbf{1}_n)_+$
$p = 2$	$\left(1 - \dfrac{\lambda}{\max(\lambda, \|v\|_2)}\right) v$
$p = \infty$	$\operatorname{sign}(v) \odot \mathbf{prox}_{\lambda \max x}(\lvert v\rvert)$

If $f(x)$ is an ℓ_q-norm function, then $f^*(x) = {}_{\mathcal{B}_p}(x)$, where $\mathcal{B}_p$ is the ℓ_p unit ball and $p^{-1} + q^{-1} = 1$. Since we have $\mathbf{prox}_{f^*}(v) = \mathcal{P}_{\mathcal{B}_p}(v)$, we deduce that:

$$\mathbf{prox}_f(v) + \mathcal{P}_{\mathcal{B}_p}(v) = v$$

More generally, we have:

$$\mathbf{prox}_{\lambda f}(v) + \lambda \mathcal{P}_{\mathcal{B}_p}\left(\frac{v}{\lambda}\right) = v$$

It follows that the projection onto the ℓ_p ball can be deduced from the proximal operator of the ℓ_q-norm function. Let $\mathcal{B}_p(c,\lambda) = \{x \in \mathbb{R}^n : \|x - c\|_p \leq \lambda\}$ be the ℓ_p ball with center c and radius λ. We obtain:

p	$\mathcal{P}_{\mathcal{B}_p(\mathbf{0}_n,\lambda)}(v)$	q
$p = 1$	$v - \operatorname{sign}(v) \odot \mathbf{prox}_{\lambda \max x}(\lvert v \rvert)$	$q = \infty$
$p = 2$	$v - \mathbf{prox}_{\lambda \|x\|_2}(v)$	$q = 2$
$p = \infty$	$\mathcal{T}(v; -\lambda, \lambda)$	$q = 1$

8.3.3.3.3. Scaling and translation

Let us define $g(x) = f(ax + b)$, where $a \neq 0$. We have[13]:

$$\mathbf{prox}_g(v) = \frac{\mathbf{prox}_{a^2 f}(av + b) - b}{a}$$

We can use this property when the center c of the ℓ_p ball is not equal to $\mathbf{0}_n$. Since we have $\mathbf{prox}_g(v) = \mathbf{prox}_f(v - c) + c$, where $g(x) = f(x - c)$, and the equivalence $\mathcal{B}_p(\mathbf{0}_n, \lambda) = \{x \in \mathbb{R}^n : f(x) \leq \lambda\}$, where $f(x) = \|x\|_p$, we deduce that:

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

8.3.3.4. *Application to the τ-problem of the lasso regression*

We have previously presented the lasso regression problem by considering the Lagrange formulation (λ-problem). We now consider the original τ-problem:

$$\hat{\beta}(\tau) = \arg\min_{\beta} \frac{1}{2}(Y - X\beta)^\top (Y - X\beta) \qquad \text{s.t.} \quad \|\beta\|_1 \leq \tau$$

The ADMM formulation is:

$$\begin{aligned} \left\{\beta^\star, \bar{\beta}^\star\right\} &= \arg\min_{(\beta,\bar{\beta})} \frac{1}{2}(Y - X\beta)^\top (Y - X\beta) + \ \ \Omega\left(\bar{\beta}\right) \\ &\text{s.t. } \beta = \bar{\beta} \end{aligned}$$

[13] The proof can be found in Beck (2017, p. 138).

where $\Omega = \mathcal{B}_1 \left(\mathbf{0}_n, \tau\right)$ is the centered ℓ_1 ball with radius τ. We note that the x-update is:

$$\begin{aligned}\beta^{(k+1)} &= \arg\min_{\beta} \left\{ \frac{1}{2} \left(Y - X\beta\right)^\top \left(Y - X\beta\right) + \frac{\varphi}{2} \left\| \beta - \bar{\beta}^{(k)} + u^{(k)} \right\|_2^2 \right\} \\ &= \left(X^\top X + \varphi I_p\right)^{-1} \left(X^\top Y + \varphi \left(\bar{\beta}^{(k)} - u^{(k)}\right)\right)\end{aligned}$$

where $v_x^{(k+1)} = \bar{\beta}^{(k)} - u^{(k)}$. For the y-update, we deduce that:

$$\begin{aligned}\bar{\beta}^{(k+1)} &= \arg\min_{\bar{\beta}} \left\{ \ \Omega\left(\bar{\beta}\right) + \frac{\varphi}{2} \left\| \beta^{(k+1)} - \bar{\beta} + u^{(k)} \right\|_2^2 \right\} \\ &= \mathcal{P}_\Omega \left(v_y^{(k+1)}\right) \\ &= v_y^{(k+1)} - \operatorname{sign}\left(v_y^{(k+1)}\right) \odot \mathbf{prox}_{\tau \max x} \left(\left\| v_y^{(k+1)} \right\|\right)\end{aligned}$$

where $v_y^{(k+1)} = \beta^{(k+1)} + u^{(k)}$. Finally, the u-update is defined by $u^{(k+1)} = u^{(k)} + \beta^{(k+1)} - \bar{\beta}^{(k+1)}$.

REMARK 8.4.– The ADMM algorithm is similar for λ- and τ-problems since the only difference concerns the y-step. For the λ-problem, we apply the soft-thresholding operator while we use the ℓ_1 projection in the case of the τ-problem. However, our experience shows that the τ-problem is easier to solve with the ADMM algorithm from a practical point of view. The reason is that the y-update of the τ-problem is independent of the penalization parameter φ. This is not the case for the λ-problem, because the soft-thresholding depends on the value taken by $\varphi^{-1}\lambda$.

8.3.4. *Dykstra's algorithm*

We now consider the proximal optimization problem where the function $f(x)$ is the convex sum of basic functions $f_j(x)$:

$$x^\star = \arg\min_x \left\{ \sum_{j=1}^m f_j(x) + \frac{1}{2} \left\| x - v \right\|_2^2 \right\}$$

and the proximal of each basic function is known.

8.3.4.1. *The* $m = 2$ *case*

In the previous section, we listed some analytical solutions of the proximal problem when the function $f(x)$ is basic. For instance, we know the proximal solution of the ℓ_1-norm function $f_1(x) = \lambda_1 \|x\|_1$ or the proximal solution of the logarithmic barrier function $f_2(x) = \lambda_2 \sum_{i=1}^{n} \ln x_i$. However, we do not know how to compute the proximal operator of $f(x) = f_1(x) + f_2(x)$:

$$x^{\star} = \mathbf{prox}_f(v) = \arg\min_x f_1(x) + f_2(x) + \frac{1}{2}\|x - v\|_2^2$$

Nevertheless, an elegant solution is provided by Dykstra's algorithm (Dykstra 1983; Bauschke and Borwein 1994; Combettes and Pesquet 2011), which is defined by the following iterations:

$$\begin{cases} x^{(k+1)} = \mathbf{prox}_{f_1}\left(y^{(k)} + p^{(k)}\right) \\ p^{(k+1)} = y^{(k)} + p^{(k)} - x^{(k+1)} \\ y^{(k+1)} = \mathbf{prox}_{f_2}\left(x^{(k+1)} + q^{(k)}\right) \\ q^{(k+1)} = x^{(k+1)} + q^{(k)} - y^{(k+1)} \end{cases} \qquad [8.22]$$

where $x^{(0)} = y^{(0)} = v$ and $p^{(0)} = q^{(0)} = \mathbf{0}_n$. This algorithm is obviously related to the Douglas–Rachford splitting framework[14], where $x^{(k)}$ and $p^{(k)}$ are the variable and the residual associated with $f_1(x)$, and $y^{(k)}$ and $q^{(k)}$ are the variable and the residual associated with $f_2(x)$, respectively.

8.3.4.2. *The* $m > 2$ *case*

The case $m > 2$ is a generalization of the previous algorithm by considering m residuals:

1) the x-update is:

$$x^{(k+1)} = \mathbf{prox}_{f_{j(k)}}\left(x^{(k)} + z^{(k+1-m)}\right)$$

2) the z-update is:

$$z^{(k+1)} = x^{(k)} + z^{(k+1-m)} - x^{(k+1)}$$

[14] See Combettes and Pesquet (2011).

where $x^{(0)} = v$, $z^{(k)} = \mathbf{0}_n$ for $k < 0$ and $j(k) = \mathrm{mod}(k+1, m)$ denotes the modulo operator taking values in $\{1, \ldots, m\}$. The variable $x^{(k)}$ is updated at each iteration, while the residual $z^{(k)}$ is updated every m iterations. This implies that the basic function $f_j(x)$ is related to the residuals $z^{(j)}$, $z^{(j+m)}$, $z^{(j+2m)}$, etc. Following Tibshirani (2017), it is better to write Dykstra's algorithm by using two iteration indices k and j. The main index k refers to the cycle[15], whereas the sub-index j refers to the constraint number:

1) The x-update is:

$$x^{(k+1,j)} = \mathbf{prox}_{f_j}\left(x^{(k+1,j-1)} + z^{(k,j)}\right) \tag{[8.23]}$$

2) The z-update is:

$$z^{(k+1,j)} = x^{(k+1,j-1)} + z^{(k,j)} - x^{(k+1,j)} \tag{[8.24]}$$

where $x^{(1,0)} = v$, $z^{(k,j)} = \mathbf{0}_n$ for $k = 0$ and $x^{(k+1,0)} = x^{(k,m)}$.

Dykstra's algorithm is particularly efficient when we consider the projection problem $x^\star = \mathcal{P}_\Omega(v)$, where $\Omega = \Omega_1 \cap \Omega_2 \cap \cdots \cap \Omega_m$. Indeed, the solution is found by replacing equation [8.23] with:

$$x^{(k+1,j)} = \mathcal{P}_{\Omega_j}\left(x^{(k+1,j-1)} + z^{(k,j)}\right) \tag{[8.25]}$$

8.3.4.3. *Application to general linear constraints*

Let us consider the case $\Omega = \{x \in \mathbb{R}^n : Cx \leq D\}$ where the number of inequality constraints is equal to m. We can write $\Omega = \Omega_1 \cap \Omega_2 \cap \cdots \cap \Omega_m$, where $\Omega_j = \left\{x \in \mathbb{R}^n : c_{(j)}^\top x \leq d_{(j)}\right\}$, $c_{(j)}^\top$ corresponds to the j^{th} row of C and $d_{(j)}$ is the j^{th} element of D. Since the projection $\mathcal{P}_{\Omega_j}$ is known and has been given on p. 101, we can find the projection $\mathcal{P}_\Omega$ using Algorithm [8.2.]. If we consider $\Omega = \{x \in \mathbb{R}^n : Ax = B, Cx \leq D, x^- \leq x \leq x^+\}$, we decompose Ω as the intersection of three basic convex sets $\Omega = \Omega_1 \cap \Omega_2 \cap \Omega_3$, where $\Omega_1 = \{x \in \mathbb{R}^n : Ax = B\}$, $\Omega_2 = \{x \in \mathbb{R}^n : Cx \leq D\}$ and $\Omega_3 = \{x \in \mathbb{R}^n : x^- \leq x \leq x^+\}$. Using Dykstra's algorithm is equivalent to formulating Algorithm [8.3.].

[15] Exactly like the coordinate descent algorithm.

Algorithm 8.2. Dykstra's algorithm for solving the proximal problem with linear inequality constraints

```
The goal is to compute the solution $x^\star = \mathbf{prox}_f(v)$ where $f(x) = \quad {}_{\Omega}(x)$
and $\Omega = \{x \in \mathbb{R}^n : Cx \leq D\}$
We initialize $x^{(0,m)} \leftarrow v$
We set $z^{(0,1)} \leftarrow \mathbf{0}_n, \ldots, z^{(0,m)} \leftarrow \mathbf{0}_n$
$k \leftarrow 0$
repeat
  $x^{(k+1,0)} \leftarrow x^{(k,m)}$
  for $j = 1 : m$ do
    The $x$-update is:
      $$x^{(k+1,j)} = x^{(k+1,j-1)} + z^{(k,j)} - \frac{\left(c_{(j)}^\top x^{(k+1;j-1)} + c_{(j)}^\top z^{(k,j)} - d_{(j)}\right)_+}{\left\|c_{(j)}\right\|_2^2} c_{(j)}$$
    The $z$-update is:
      $$z^{(k+1,j)} = x^{(k+1,j-1)} + z^{(k,j)} - x^{(k+1,j)}$$
  end for
  $k \leftarrow k + 1$
until Convergence
return $x^\star \leftarrow x^{(k,m)}$
```

Since we have:

$$\frac{1}{2}\|x - v\|_2^2 = \frac{1}{2}x^\top x - x^\top v + \frac{1}{2}v^\top v$$

we deduce that the two previous problems can be cast into a QP problem:

$$x^\star = \arg\min_x \frac{1}{2}x^\top I_n x - x^\top v \qquad \text{s.t.} \quad x \in \Omega$$

We can then compare the efficiency of Dykstra's algorithm with the QP algorithm. Let us consider the proximal problem where the vector v is defined by the elements $v_i = \ln\left(1 + i^2\right)$ and the set of constraints is:

$$\Omega = \left\{x \in \mathbb{R}^n : \sum_{i=1}^n x_i \leq \frac{1}{2}, \sum_{i=1}^n e^{-i} x_i \geq 0\right\}$$

Algorithm 8.3. Dykstra's algorithm for solving the proximal problem with general linear constraints

```
The goal is to compute the solution $x^{\star} = \mathbf{prox}_f(v)$ where $f(x) = \;_{\Omega}(x)$
and $\Omega = \{x \in \mathbb{R}^n : Ax = B, Cx \leq D, x^- \leq x \leq x^+\}$
We initialize $x_m^{(0)} \leftarrow v$
We set $z_1^{(0)} \leftarrow \mathbf{0}_n$, $z_2^{(0)} \leftarrow \mathbf{0}_n$ and $z_3^{(0)} \leftarrow \mathbf{0}_n$
$k \leftarrow 0$
repeat
    $x_0^{(k+1)} \leftarrow x_m^{(k)}$
    $x_1^{(k+1)} \leftarrow x_0^{(k+1)} + z_1^{(k)} - A^{\dagger}\left(Ax_0^{(k+1)} + Az_1^{(k)} - B\right)$
    $z_1^{(k+1)} \leftarrow x_0^{(k+1)} + z_1^{(k)} - x_1^{(k+1)}$
    $x_2^{(k+1)} \leftarrow \mathcal{P}_{\Omega_2}\left(x_1^{(k+1)} + z_2^{(k)}\right)$          ▶ Algorithm (8.2.)
    $z_2^{(k+1)} \leftarrow x_1^{(k+1)} + z_2^{(k)} - x_2^{(k+1)}$
    $x_3^{(k+1)} \leftarrow \mathcal{T}\left(x_2^{(k+1)} + z_3^{(k)}; x^-, x^+\right)$
    $z_3^{(k+1)} \leftarrow x_2^{(k+1)} + z_3^{(k)} - x_3^{(k+1)}$
    $k \leftarrow k + 1$
until Convergence
return $x^{\star} \leftarrow x_3^{(k)}$
```

Using a MATLAB implementation[16], we find that the computational time of Dykstra's algorithm, when n is equal to 10 million, is equal to the QP algorithm when n is equal to 12,500, meaning that there is a factor of 800 between the two methods!

8.4. Applications to portfolio optimization

The development of the previous algorithms will fundamentally change the practice of portfolio optimization. Until now, we have seen that portfolio managers live in a quadratic programming world. With these new optimization algorithms, we can consider more complex portfolio optimization programs with non-quadratic objective function, regularization with penalty functions and nonlinear constraints.

[16] The QP implementation corresponds to the `quadprog` function.

Item	Portfolio	$f(x)$	Reference
(1)	MVO	$\frac{1}{2}x^\top \Sigma x - \gamma x^\top \mu$	Markowitz (1952)
(2)	GMV	$\frac{1}{2}x^\top \Sigma x$	Jagganathan and Ma (2003)
(3)	MDP	$\ln\left(\sqrt{x^\top \Sigma x}\right) - \ln\left(x^\top \sigma\right)$	Choueifaty and Coignard (2008)
(4)	KL	$\sum_{i=1}^{n} x_i \ln\left(x_i / \tilde{x}_i\right)$	Bera and Park (2008)
(5)	ERC	$\frac{1}{2}x^\top \Sigma x - \lambda \sum_{i=1}^{n} \ln x_i$	Maillard *et al.* (2010)
(6)	RB	$\mathcal{R}(x) - \lambda \sum_{i=1}^{n} \mathcal{RB}_i \cdot \ln x_i$	Roncalli (2015)
(7)	RQE	$\frac{1}{2}x^\top D x$	Carmichael *et al.* (2018)

Table 8.1. *Some objective functions used in portfolio optimization*

We consider a universe of n assets. Let x be the vector of weights in the portfolio. We denote by μ and Σ the vector of expected returns and the covariance matrix of asset returns[17]. Some models also consider a reference portfolio $\tilde{x}$. In Table 8.1, we report the main objective functions that are used by professionals[18]. Besides the mean-variance optimized portfolio (MVO) and the global minimum variance portfolio (GMV), we find the equal risk contribution portfolio (ERC), the risk budgeting portfolio (RB) and the most diversified portfolio (MDP). According to Choueifaty and Coignard (2008), the MDP is defined as the portfolio which maximizes the diversification ratio $\mathcal{DR}(x) = \frac{x^\top \sigma}{\sqrt{x^\top \Sigma x}}$. We also include in the list two "academic" portfolios, which are based on the Kullback–Leibler (KL) information criteria and the Rao's quadratic entropy (RQE) measure[19].

In a similar way, we list in Table 8.2 some popular regularization penalty functions that are used in the industry (Bruder *et al.* 2013; Bourgeron *et al.* 2018). The ridge and lasso regularization are well known in statistics and machine learning (Hastie *et al.* 2009). The log-barrier penalty function comes from the risk budgeting optimization problem, whereas Shannon's entropy is another approach for imposing a sufficient weight diversification.

[17] The vector of volatilities is defined by $\sigma = (\sigma_1, \ldots, \sigma_n)$.

[18] For each model, we write the optimization problem as a minimization problem.

[19] D is the dissimilarity matrix satisfying $D_{i,j} \geq 0$ and $D_{i,j} = D_{j,i}$.

Item	Regularization	$\mathfrak{R}(x)$	Reference
(8)	Ridge	$\lambda \left\Vert x - \tilde{x} \right\Vert_2^2$	DeMiguel *et al.* (2009)
(9)	Lasso	$\lambda \left\Vert x - \tilde{x} \right\Vert_1$	Brodie *at al.* (2009)
(10)	Log-barrier	$-\sum_{i=1}^n \lambda_i \ln x_i$	Roncalli (2013)
(11)	Shannon's entropy	$\lambda \sum_{i=1}^n x_i \ln x_i$	Yu *et al.* (2014)

Table 8.2. *Some regularization penalties used in portfolio optimization*

Concerning the constraints, the most famous are the no cash/no leverage and no short selling restrictions. Weight bounds and asset class limits are also extensively used by practitioners. Turnover and transaction cost management may be an important topic when rebalancing a current portfolio $\tilde{x}$. When managing long/short portfolios, we generally impose leverage or long/short exposure limits. In the case of a benchmarked strategy, we might also want to have a tracking error limit, with respect to the benchmark $\tilde{x}$. On the contrary, we can impose a minimum tracking error or active share in the case of active management. Finally, the Herfindahl constraint is used for some smart beta portfolios.

In what follows, we consider several portfolio optimization problems. Most of them are a combination of an objective function, one or two regularization penalty functions and some constraints that have been listed above. From an industrial point of view, it is interesting to implement the proximal operator for each item. In this approach, solving any portfolio optimization problem is equivalent to using CCD, ADMM, Dykstra and the appropriate proximal functions as Lego bricks.

8.4.1. *Minimum variance optimization*

8.4.1.1. *Managing diversification*

The global minimum variance (GMV) portfolio corresponds to the following optimization program:

$$x^\star = \arg\min_x \frac{1}{2} x^\top \Sigma x \qquad \text{s.t.} \quad \mathbf{1}_n^\top x = 1$$

Item	Constraint	Formula
(12)	No cash and leverage	$\sum_{i=1}^{n} x_i = 1$
(13)	No short selling	$x_i \geq 0$
(14)	Weight bounds	$x_i^- \leq x_i \leq x_i^+$
(15)	Asset class limits	$c_j^- \leq \sum_{i \in \mathcal{C}_j} x_i \leq c_j^+$
(16)	Turnover	$\sum_{i=1}^{n} \lvert x_i - \tilde{x}_i \rvert \leq \boldsymbol{\tau}^+$
(17)	Transaction costs	$\sum_{i=1}^{n} \left(c_i^- \left(\tilde{x}_i - x_i \right)_+ + c_i^+ \left(x_i - \tilde{x}_i \right)_+ \right) \leq \boldsymbol{c}^+$
(18)	Leverage limit	$\sum_{i=1}^{n} \lvert x_i \rvert \leq \mathcal{L}^+$
(19)	Long/short exposure	$-\mathcal{LS}^- \leq \sum_{i=1}^{n} x_i \leq \mathcal{LS}^+$
(20)	Benchmarking	$\sqrt{\left(x - \tilde{x} \right)^\top \Sigma \left(x - \tilde{x} \right)} \leq \sigma^+$
(21)	Tracking error floor	$\sqrt{\left(x - \tilde{x} \right)^\top \Sigma \left(x - \tilde{x} \right)} \geq \sigma^-$
(22)	Active share floor	$\frac{1}{2} \sum_{i=1}^{n} \lvert x_i - \tilde{x}_i \rvert \geq \mathcal{AS}^-$
(23)	Number of active bets	$\left(x^\top x \right)^{-1} \geq \mathcal{N}^-$

Table 8.3. *Some constraints used in portfolio optimization*

We know that the solution is $x^\star = \left(\mathbf{1}_n^\top \Sigma^{-1} \mathbf{1}_n \right)^{-1} \Sigma^{-1} \mathbf{1}_n$. In practice, nobody implements the GMV portfolio because it is a long/short portfolio and it is not robust. Most of the time, professionals impose weight bounds: $0 \leq x_i \leq x^+$. However, this approach generally leads to corner solutions, meaning that a large number of optimized weights are equal to zero or the upper bound and very few assets have a weight within the range. With the emergence of smart beta portfolios, the minimum variance portfolio gained popularity among institutional investors. For instance, we can find many passive indices based on this framework. In order to increase the robustness of these portfolios, the first generation of minimum variance strategies has used relative weight bounds with respect to a benchmark b: $\delta^- b_i \leq x_i \leq \delta^+ b_i$, where $0 < \delta^- < 1$ and $\delta^+ > 1$. For instance, the most popular scheme is to take $\delta^- = 0.5$ and $\delta^+ = 2$. Nevertheless, this constraint produces the illusion that the portfolio is diversified, because the optimized weights are different. In fact, portfolio weights are different because benchmark weights are different. The second generation of minimum variance strategies imposes a global diversification constraint. The most popular solution is based on the Herfindahl index $\mathcal{H}(x) = \sum_{i=1}^{n} x_i^2$. This index takes the value 1 for a pure concentrated portfolio ($\exists\, i : x_i = 1$) and $1/n$ for an equally weighted portfolio. Therefore, we can define the number of effective bets as the inverse of the Herfindahl index: $\mathcal{N}(x) = \mathcal{H}(x)^{-1}$.

The optimization program becomes:

$$x^\star = \arg\min_x \frac{1}{2} x^\top \Sigma x \qquad [8.26]$$

$$\text{s.t.} \begin{cases} \mathbf{1}_n^\top x = 1 \\ \mathbf{0}_n \leq x \leq x^+ \\ \mathcal{N}(x) \geq \mathcal{N}^- \end{cases}$$

where $\mathcal{N}^-$ is the minimum number of effective bets. The Herfindhal constraint is equivalent to:

$$\mathcal{N}(x) \geq \mathcal{N}^- \Leftrightarrow \left(x^\top x\right)^{-1} \geq \mathcal{N}^- \Leftrightarrow x^\top x \leq \frac{1}{\mathcal{N}^-}$$

Therefore, a first solution to solve [8.26] is to consider the following QP problem[20]:

$$x^\star(\lambda) = \arg\min_x \frac{1}{2} x^\top \Sigma x + \lambda x^\top x \qquad [8.27]$$

$$\text{s.t.} \begin{cases} \mathbf{1}_n^\top x = 1 \\ \mathbf{0}_n \leq x \leq x^+ \end{cases}$$

where $\lambda \geq 0$ is a scalar. Since $\mathcal{N}(x^\star(\infty))$ is equal to the number n of assets and $\mathcal{N}(x^\star(\lambda))$ is an increasing function of λ, problem [8.27] has a unique solution if $\mathcal{N}^- \in [\mathcal{N}(x^\star(0)), n]$. There is an optimal value $\lambda^\star$ such that for each $\lambda \geq \lambda^\star$, we have $\mathcal{N}(x^\star(\lambda)) \geq \mathcal{N}^-$. Computing the optimal portfolio $x^\star(\lambda^\star)$ therefore implies finding the solution $\lambda^\star$ of the nonlinear equation $\mathcal{N}(x^\star(\lambda)) = \mathcal{N}^-$.

A second method is to consider the ADMM form:

$$\begin{aligned} \{x^\star, y^\star\} &= \arg\min_{(x,y)} \frac{1}{2} x^\top \Sigma x + \quad \Omega_3(x) + \quad \Omega_4(y) \\ &\text{s.t. } x = y \end{aligned}$$

[20] The objective function can be written as:

$$\frac{1}{2} x^\top \Sigma x + \lambda x^\top x = \frac{1}{2} x^\top (\Sigma + 2I_n) x.$$

where $\Omega_3 = \mathcal{H}_{yperplane}\left[\mathbf{1}_n, 1\right]$ and $\Omega_4 = \mathcal{B}_{ox}\left[\mathbf{0}_n, x^+\right] \cap \mathcal{B}_2\left(\mathbf{0}_n, \sqrt{\frac{1}{\mathcal{N}^-}}\right)$. In this case, the x- and y-updates become[21]:

$$\begin{aligned} x^{(k+1)} &= \arg\min_x \left\{ \frac{1}{2} x^\top \left(\Sigma + \varphi I_n\right) x - \varphi x^\top \left(y^{(k)} - u^{(k)}\right) + \quad {}_{\Omega_3}\left(x\right) \right\} \\ &= \left(\Sigma + \varphi I_n\right)^{-1} \left(\varphi\left(y^{(k)} - u^{(k)}\right)\right. \\ &\qquad \left. + \frac{1 - \mathbf{1}_n^\top \left(\Sigma + \varphi I_n\right)^{-1} \varphi\left(y^{(k)} - u^{(k)}\right)}{\mathbf{1}_n^\top \left(\Sigma + \varphi I_n\right)^{-1} \mathbf{1}_n} \mathbf{1}_n \right) \end{aligned}$$

and:

$$y^{(k+1)} = \mathcal{P}_{\mathcal{B}\mathrm{ox}-\mathcal{B}\mathrm{all}}\left(x^{(k+1)} + u^{(k)}; \mathbf{0}_n, x^+, \mathbf{0}_n, \sqrt{\frac{1}{\mathcal{N}^-}}\right)$$

where $\mathcal{P}_{\mathcal{B}\mathrm{ox}-\mathcal{B}\mathrm{all}}$ corresponds to Dykstra's algorithm given in Appendix 8.7.1.4.1.

We consider the parameter set #1 given in Appendix 8.7.2.1. The investment universe is made up of eight stocks. We would like to build a diversified minimum variance long-only portfolio without imposing an upper weight bound[22]. In Table 8.4, we report the solutions found by the ADMM algorithm for several values of $\mathcal{N}^-$. When there is no Herfindahl constraint, the portfolio is fully invested in the seventh stock, meaning that the asset diversification is very poor. Then, we increase the number of effective bets. If $\mathcal{N}^-$ is equal to the number n of stocks, we verify that the solution corresponds to the equally weighted portfolio. Between these two limit cases, we see the impact of the Herfindahl constraint on the portfolio diversification. The parameter set #1 is defined with respect to a capitalization-weighted index, whose weights are equal to 23%, 19%, 17%, 9%, 8%, 6% and 5%. The number of effective bets of this benchmark is equal to 6.435. If we impose that the effective number of bets of the minimum variance portfolio is at least equal to the effective number of bets of the benchmark, we find the following solution: 14.74%, 15.45%, 1.79%, 15.49%, 6.17%, 13.83%, 23.21% and 9.31%.

[21] See Appendix 8.7.1.2 for the derivation of the x-update.

[22] This means that x^+ is set to $\mathbf{1}_n$.

$\mathcal{N}^-$	**1.00**	**2.00**	**3.00**	**4.00**	**5.00**	**6.00**	**6.50**	**7.00**	**7.50**	**8.00**
$\boldsymbol{x_1^\star}$	0.00	3.22	9.60	13.83	15.18	15.05	14.69	14.27	13.75	12.50
$\boldsymbol{x_2^\star}$	0.00	12.75	14.14	15.85	16.19	15.89	15.39	14.82	14.13	12.50
$\boldsymbol{x_3^\star}$	0.00	0.00	0.00	0.00	0.00	0.07	2.05	4.21	6.79	12.50
$\boldsymbol{x_4^\star}$	0.00	10.13	15.01	17.38	17.21	16.09	15.40	14.72	13.97	12.50
$\boldsymbol{x_5^\star}$	0.00	0.00	0.00	0.00	0.71	5.10	6.33	7.64	9.17	12.50
$\boldsymbol{x_6^\star}$	0.00	5.36	8.95	12.42	13.68	14.01	13.80	13.56	13.25	12.50
$\boldsymbol{x_7^\star}$	100.00	68.53	52.31	40.01	31.52	25.13	22.92	20.63	18.00	12.50
$\boldsymbol{x_8^\star}$	0.00	0.00	0.00	0.50	5.51	8.66	9.41	10.14	10.95	12.50

Table 8.4. *Minimum variance portfolios (in %)*

As explained previously, we can also solve the optimization problem by combining problem [8.27] and the bisection algorithm. However, this approach is no longer valid if we consider diversification constraints that are not quadratic. For instance, let us consider the generalized minimum variance problem:

$$x^\star = \arg\min_x \frac{1}{2} x^\top \Sigma x \qquad [8.28]$$

$$\text{s.t.} \begin{cases} \mathbf{1}_n^\top x = 1 \\ \mathbf{0}_n \leq x \leq x^+ \\ \mathcal{D}(x) \geq \mathcal{D}^- \end{cases}$$

where $\mathcal{D}(x)$ is a weight diversification measure and $\mathcal{D}^-$ is the minimum acceptable diversification. For example, we can use Shannon's entropy, the Gini index or the diversification ratio. In this case, it is not possible to obtain an equivalent QP problem, whereas the ADMM algorithm is exactly the same as previously, except for the y-update:

$$y^{(k+1)} = \mathcal{P}_{\mathcal{B}ox[\mathbf{0}_n, x^+] \cap \mathfrak{D}} \left(x^{(k+1)} + u^{(k)} \right)$$

where $\mathfrak{D} = \{x \in \mathbb{R}^n : \mathcal{D}(x) \geq \mathcal{D}^-\}$. The projection onto $\mathfrak{D}$ can be easily derived from the proximal operator of the dual function (see section 8.4.4).

REMARK 8.5.– If we compare the computational times, we observe that the best method is the ADMM algorithm. In our example, the computational time is divided by a factor of eight with respect to the bisection approach. If we consider a large-scale problem with n larger than 1,000, the computational time is generally divided by a factor greater than 50!

8.4.1.2. *Managing the portfolio rebalancing process*

Another big challenge of the minimum variance portfolio is the management of the turnover between two rebalancing dates. Let x_t be the current portfolio. The optimization program for calibrating the optimal solution x_{t+1} for the next rebalancing date $t+1$ may include a penalty function $\boldsymbol{c}\left(x \mid x_t\right)$ and/or a weight constraint $\mathfrak{C}\left(x \mid x_t\right)$ that are parameterized with respect to the current portfolio x_t:

$$\begin{aligned} x_{t+1} &= \arg\min_{x} \frac{1}{2} x^\top \Sigma x + \boldsymbol{c}\left(x \mid x_t\right) \\ &\text{s.t.} \left\{ \begin{array}{l} \mathbf{1}_n^\top x = 1 \\ \mathbf{0}_n \leq x \leq x^+ \\ x \in \mathfrak{C}\left(x \mid x_t\right) \end{array} \right. \end{aligned} \qquad [8.29]$$

Again, we can solve this problem using the ADMM algorithm. Thanks to Dykstra's algorithm, the only difficulty is finding the proximal operator of $\boldsymbol{c}\left(x \mid x_t\right)$ or $\mathfrak{C}\left(x \mid x_t\right)$ when performing the y-update.

Let us define the cost function as:

$$\boldsymbol{c}\left(x \mid x_t\right) = \lambda \sum_{i=1}^{n} \left(c_i^- \left(x_{i,t} - x_i\right)_+ + c_i^+ \left(x_i - x_{i,t}\right)_+\right)$$

where c_i^- and c_i^+ are the bid and ask transaction costs. In Appendix 8.7.1.6, we show that the proximal operator is:

$$\mathbf{prox}_{\boldsymbol{c}(x \mid x_t)}\left(v\right) = x_t + \mathcal{S}\left(v - x_t; \lambda c^-, \lambda c^+\right) \qquad [8.30]$$

where $\mathcal{S}\left(v; \lambda_-, \lambda_+\right) = \left(v - \lambda_+\right)_+ - \left(v + \lambda_-\right)_-$ is the two-sided soft-thresholding operator.

If we define the cost constraint $\mathfrak{C}\left(x \mid x_t\right)$ as a turnover constraint:

$$\mathfrak{C}\left(x \mid x_t\right) = \left\{x \in \mathbb{R}^n : \left\|x - x_t\right\|_1 \leq \boldsymbol{\tau}^+\right\}$$

the proximal operator is:

$$\mathcal{P}_{\mathfrak{C}}\left(v\right) = v - \operatorname{sign}\left(v - x_t\right) \odot \min\left(\left|v - x_t\right|, s^\star\right) \qquad [8.31]$$

where $s^\star = \left\{s \in \mathbb{R} : \sum_{i=1}^n \left(\left|v_i - x_{t,i}\right| - s\right)_+ = \boldsymbol{\tau}^+\right\}$.

REMARK 8.6.– These two examples are very basic and show how we can easily introduce turnover management using the ADMM framework. More sophisticated approaches are presented in section 8.4.4.

8.4.2. *Smart beta portfolios*

In this section, we consider three main models of smart beta portfolios: the equal risk contribution (ERC) portfolio, the risk budgeting (RB) portfolio and the most diversified portfolio (MDP). Specific algorithms for these portfolios based on the CCD method have already been presented in Griveau-Billion *et al.* (2013) and Richard and Roncalli (2015, 2019). We extend these results to the ADMM algorithm.

8.4.2.1. *The ERC portfolio*

The ERC portfolio uses the volatility risk measure $\sigma\left(x\right) = \sqrt{x^\top \Sigma x}$ and allocates the weights such that the risk contribution is the same for all the assets of the portfolio (Maillard *et al.* 2010):

$$\mathcal{RC}_i\left(x\right) = x_i \frac{\partial\,\sigma\left(x\right)}{\partial\,x_i} = x_j \frac{\partial\,\sigma\left(x\right)}{\partial\,x_j} = \mathcal{RC}_j\left(x\right)$$

In this case, we can show that the ERC portfolio is the scaled solution $x^\star / \left(\mathbf{1}_n^\top x^\star\right)$, where $x^\star$ is given by:

$$x^\star = \arg\min_x \frac{1}{2} x^\top \Sigma x - \lambda \sum_{i=1}^n \ln x_i$$

and λ is any positive scalar. The first-order condition is $\left(\Sigma x\right)_i - \lambda x_i^{-1} = 0$. It follows that $x_i \left(\Sigma x\right)_i - \lambda = 0$ or $x_i^2 \sigma_i^2 + x_i \sigma_i \sum_{j \neq i} x_j \rho_{i,j} \sigma_j - \lambda = 0$.

We deduce that the solution is the positive root of the second-degree equation. Finally, we obtain the following iteration for the CCD algorithm:

$$x_i^{(k+1)} = \frac{-v_i^{(k+1)} + \sqrt{\left(v_i^{(k+1)}\right)^2 + 4\lambda\sigma_i^2}}{2\sigma_i^2} \qquad [8.32]$$

where:

$$v_i^{(k+1)} = \sigma_i \sum_{j<i} x_j^{(k+1)} \rho_{i,j}\sigma_j + \sigma_i \sum_{j>i} x_j^{(k)} \rho_{i,j}\sigma_j$$

The ADMM algorithm uses the first trick where $f_x(x) = \frac{1}{2}x^\top \Sigma x$ and $f_y(y) = -\lambda \sum_{i=1}^n \ln y_i$. It follows that the x- and y-update steps are:

$$x^{(k+1)} = (\Sigma + \varphi I_n)^{-1} \varphi \left(y^{(k)} - u^{(k)}\right)$$

and:

$$y_i^{(k+1)} = \frac{1}{2}\left(\left(x_i^{(k+1)} + u_i^{(k)}\right) + \sqrt{\left(x_i^{(k+1)} + u_i^{(k)}\right)^2 + 4\lambda\varphi^{-1}}\right)$$

We apply the CCD and ADMM algorithms to the parameter set #1. We find that the ERC portfolio is equal to 11.40%, 12.29%, 5.49%, 11.91%, 6.65%, 10.81%, 33.52% and 7.93%. It appears that the CCD algorithm is much more efficient than the ADMM algorithm. For instance, if we set $\lambda = \sqrt{x^{(0)^\top} \Sigma x^{(0)}}$, $x^{(0)} = n^{-1}\mathbf{1}_n$ and $\varphi = 1$, the CCD algorithm needs six cycles to converge, whereas the ADMM algorithm needs 156 iterations if we set the convergence criterion[23] $\varepsilon = 10^{-8}$. Whatever the values of λ, $x^{(0)}$ and ε, our experience is that the CCD generally converges within less than 15 cycles even if the number of assets is greater than 1,000. The convergence of the ADMM is more of an issue because it depends on the parameters λ and φ. In Figure 8.2, we have reported the number of iterations of the ADMM with respect to φ for several values of ε when $\lambda = 1$ and $x^{(0)} = \mathbf{1}_n$. We verify that it is very sensitive to the value taken by φ. Curiously, the parameter λ has little influence, meaning that the convergence issue mainly concerns the x-update step.

[23] The termination rule is defined as $\max \left| x_i^{(k+1)} - x_i^{(k)} \right| \leq \varepsilon$.

8.4.2.2. *Risk budgeting optimization*

The ERC portfolio has been extended by Roncalli (2013) when the risk budgets are not equal and when the risk measure $\mathcal{R}(x)$ is convex and coherent:

$$\mathcal{RC}_i(x) = x_i \frac{\partial\, \mathcal{R}(x)}{\partial\, x_i} = \mathcal{RB}_i$$

where $\mathcal{RB}_i$ is the risk budget allocated to asset i. In this case, we can show that the risk budgeting portfolio is the scaled solution of the following optimization problem:

$$x^\star = \arg\min_x \mathcal{R}(x) - \lambda \sum_{i=1}^{n} \mathcal{RB}_i \cdot \ln x_i$$

where λ is any positive scalar. Depending on the risk measure, we can use the CCD or the ADMM algorithm.

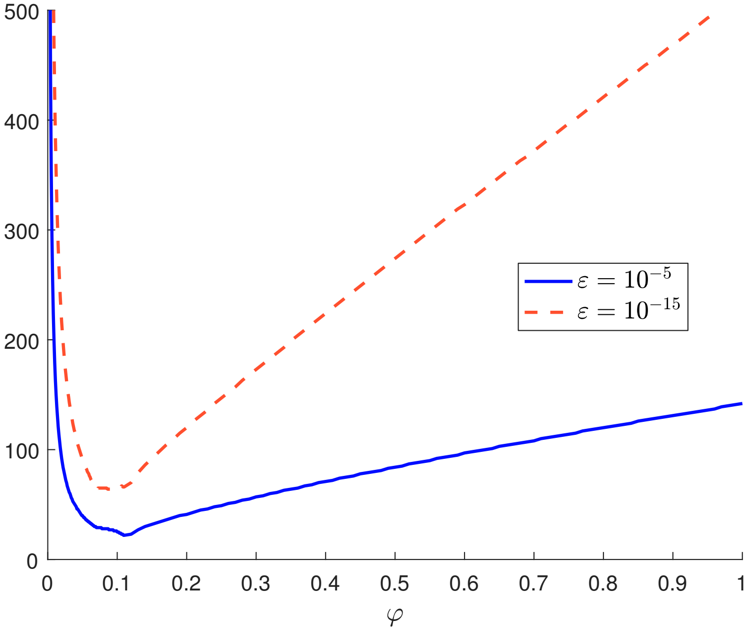

Figure 8.2. *Number of ADMM iterations for finding the ERC portfolio. For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

For example, Roncalli (2015) proposes using the standard deviation-based risk measure: $\mathcal{R}(x) = -x^\top (\mu - r) + \xi \sqrt{x^\top \Sigma x}$. In this case, the first-order condition for defining the CCD algorithm is:

$$-(\mu_i - r) + \xi \frac{(\Sigma x)_i}{\sqrt{x^\top \Sigma x}} - \lambda \frac{\mathcal{RB}_i}{x_i} = 0$$

It follows that $\xi x_i (\Sigma x)_i - (\mu_i - r) x_i \sigma(x) - \lambda \sigma(x) \cdot \mathcal{RB}_i = 0$ or equivalently $\alpha_i x_i^2 + \beta_i x_i + \gamma_i = 0$, where $\alpha_i = \xi \sigma_i^2$, $\beta_i = \xi \sigma_i \sum_{j \neq i} x_j \rho_{i,j} \sigma_j - (\mu_i - r) \sigma(x)$ and $\gamma_i = -\lambda \sigma(x) \cdot \mathcal{RB}_i$. We note that the solution x_i depends on the volatility $\sigma(x)$. Here, we face an endogenous problem, because $\sigma(x)$ depends on x_i. Griveau-Billon *et al.* (2015) note that this is not an issue because we may assume that $\sigma(x)$ is almost constant between two coordinate iterations of the CCD algorithm. They deduce that the coordinate solution is then the positive root of the second-degree equation:

$$x_i^{(k+1)} = \frac{-\beta_i^{(k+1)} + \sqrt{\left(\beta_i^{(k+1)}\right)^2 - 4\alpha_i^{(k+1)} \gamma_i^{(k+1)}}}{2\alpha_i^{(k+1)}} \qquad [8.33]$$

where:

$$\begin{cases} \alpha_i^{(k+1)} = \xi \sigma_i^2 \\ \beta_i^{(k+1)} = \xi \sigma_i \left(\sum_{j<i} x_j^{(k+1)} \rho_{i,j} \sigma_j + \sum_{j>i} x_j^{(k)} \rho_{i,j} \sigma_j \right) - (\mu_i - r) \sigma_i^{(k+1)}(x) \\ \gamma_i^{(k+1)} = -\lambda \sigma_i^{(k+1)}(x) \cdot \mathcal{RB}_i \\ \sigma_i^{(k+1)}(x) = \sqrt{\chi^\top \Sigma \chi} \\ \chi = \left(x_1^{(k+1)}, \ldots, x_{i-1}^{(k+1)}, x_i^{(k)}, x_{i+1}^{(k)} \ldots, x_n^{(k)} \right) \end{cases}$$

In the case of the volatility or the standard deviation-based risk measure, we apply the exact formulation of the CCD algorithm because we have an analytical solution of the first-order condition. This is not always the case, especially when we consider skewness-based risk measure (Bruder *et al.* 2016;

Lezmi *et al.* 2018). In this case, we can use the gradient formulation of the CCD algorithm or the ADMM algorithm, which is defined as follows:

$$\begin{cases} x^{(k+1)} = \mathbf{prox}_{\varphi^{-1}\mathcal{R}(x)}\left(y^{(k)} - u^{(k)}\right) \\ v_y^{(k+1)} = x^{(k+1)} + u^{(k)} \\ y^{(k+1)} = \frac{1}{2}\left(v_y^{(k+1)} + \sqrt{v_y^{(k+1)} \odot v_y^{(k+1)} + 4\lambda\varphi^{-1} \cdot \mathcal{RB}}\right) \\ u^{(k+1)} = u^{(k)} + x^{(k+1)} - y^{(k+1)} \end{cases}$$

8.4.2.3. *The most diversified portfolio*

Choueifaty and Coignard (2008) introduce the concept of diversification ratio, which corresponds to the following expression:

$$\mathcal{DR}(x) = \frac{\sum_{i=1}^{n} x_i \sigma_i}{\sigma(x)} = \frac{x^\top \sigma}{\sqrt{x^\top \Sigma x}}$$

By construction, the diversification ratio of a portfolio fully invested in one asset is equal to 1, whereas it is larger than 1 in the general case. The authors then propose building the most diversified portfolio as the portfolio which maximizes the diversification ratio. It is also the solution to the following minimization problem[24]:

$$\begin{aligned} x^\star &= \arg\min_x \frac{1}{2}\ln\left(x^\top \Sigma x\right) - \ln\left(x^\top \sigma\right) \\ &\text{s.t.} \begin{cases} \mathbf{1}_n^\top x = 1 \\ x \in \Omega \end{cases} \end{aligned}$$

This problem is relatively easy to solve using standard numerical algorithms if Ω corresponds to linear constraints, for example, weight constraints. However, the optimal solution may face the same problem as the minimum variance portfolio since most of the time it is concentrated on a small number of assets. Hence, it is interesting to add a weight diversification constraint $\mathcal{D}(x) \geq \mathcal{D}^-$. For example, we can assume that the number of

[24] See Choueifaty *et al.* (2013).

effective bets $\mathcal{N}(x)$ is larger than a minimum acceptable value $\mathcal{N}^-$. Contrary to the minimum variance portfolio, we do not obtain a QP problem and we observe that the optimization problem is tricky in practice. Thanks to the ADMM algorithm, we can, however, simplify the optimization problem by splitting the constraints and using the same approach that has been already described on p. 299. The x-update consists of finding the regularized standard MDP:

$$\begin{aligned} x^{(k+1)} = \arg\min_x \Bigg\{ & \frac{1}{2}\ln\left(x^\top \Sigma x\right) - \ln\left(x^\top \sigma\right) \\ & + \frac{\varphi}{2}\left\| x - y^{(k)} + u^{(k)} \right\|_2^2 \quad \text{s.t. } \mathbf{1}_n^\top x = 1 \Bigg\} \end{aligned}$$

whereas the y-update corresponds to the projection onto the intersection of Ω and $\mathfrak{D} = \{x \in \mathbb{R}^n : \mathcal{D}(x) \geq \mathcal{D}^-\}$:

$$y^{(k+1)} = \mathcal{P}_{\Omega \cap \mathfrak{D}}\left(x^{(k+1)} + u^{(k)}\right)$$

	L/S	**Long-only**					
$\mathcal{N}^-$		0.00	3.00	4.00	5.00	6.00	7.00
$\boldsymbol{x_1^\star}$	41.81	41.04	35.74	30.29	26.08	22.44	18.83
$\boldsymbol{x_2^\star}$	51.88	50.92	43.91	36.68	31.05	26.12	21.19
$\boldsymbol{x_3^\star}$	8.20	8.05	10.12	11.52	12.33	12.80	13.01
$\boldsymbol{x_4^\star}$	−0.43	0.00	2.48	5.12	7.16	8.90	10.51
$\boldsymbol{x_5^\star}$	−0.26	0.00	0.92	2.28	3.60	5.02	6.85
$\boldsymbol{x_6^\star}$	−0.38	0.00	2.03	4.36	6.28	8.02	9.79
$\boldsymbol{x_7^\star}$	−0.51	0.00	3.47	6.68	8.85	10.44	11.65
$\boldsymbol{x_8^\star}$	−0.31	0.00	1.32	3.07	4.65	6.27	8.17
$\boldsymbol{\mathcal{N}(x)}$		2.30	3.00	4.00	5.00	6.00	7.00

Table 8.5. *MDP portfolios (in %)*

We consider the parameter set #2 given in Appendix 8.7.2.2. The results are reported in Table 8.5. The second column corresponds to the long/short MDP portfolio (or $\Omega = \mathbb{R}^n$). By definition, we cannot compute the number of effective bets because it contains short positions. The other columns correspond to the long-only MDP portfolio (or $\Omega = [0,1]^n$) when we impose a sufficient number of effective bets $\mathcal{N}^{-}$. We note that the traditional long-only MDP is poorly diversified in terms of weights since we have $\mathcal{N}(x) = 2.30$. As for the minimum variance portfolio, the MDP tends to the equally weighted portfolio when $\mathcal{N}^{-}$ tends to the number of assets

8.4.3. *Robo-advisory optimization*

Today's financial industry is facing a digital revolution in all areas: payment services, online banking, asset management, etc. This is particularly true for the financial advisory industry, which has been impacted in the last few years by the emergence of digitalization and robo-advisors. The demand for robo-advisors is strong, which explains the growth of this business[25]. How does one characterize a robo-advisor? This is not simple, but the underlying idea is to build a systematic portfolio allocation in order to provide a customized advisory service. A robo-advisor has two main objectives. The first objective is to know the investor better than a traditional asset manager. Because of this better knowledge, the robo-advisor may propose a more appropriate asset allocation. The second objective is to perform the task in a systematic way and to build an automated rebalancing process. Ultimately, the goal is to offer a customized solution. In fact, the reality is very different. We generally note that many robo-advisors are more a web or a digital application, but not really a robo-advisor. The reason is that portfolio optimization is a very difficult task. In many robo-advisors, asset allocation is then rather human-based or not completely systematic, with the aim to rectify the shortcomings of mean-variance optimization. Over the next five years, the most important challenge for robo-advisors will be to reduce these discretionary decisions and improve the robustness of their systematic asset allocation process. However, this means that robo-advisors must give up the quadratic programming world of the portfolio allocation.

8.4.3.1. *Specification of the objective function*

In order to make mean-variance optimization more robust, two directions can be followed. The first one has been largely explored and adds a penalty function in order to regularize or sparsify the solution (Brodie *et al.* 2009;

[25] For instance, the growth was 60% per year in the United States over the last five years. In Europe, the growth is also impressive, even though the market is smaller. In the last two years, assets under management have increased 14-fold.

DeMiguel *et al.* 2009; Carrasco and Noumon 2010; Bruder *et al.* 2013; Bourgeron *et al.* 2018). The second one consists of changing the objective function and considering risk budgeting portfolios instead of mean-variance optimized portfolios (Maillard *et al.* 2010; Roncalli 2013). Even if this second direction has encountered great success, it presents a solution that is not sufficiently flexible in terms of active management. Nevertheless, these two directions are not so divergent. Indeed, Roncalli (2013) shows that the risk budgeting optimization can be viewed as a nonlinear shrinkage approach of the minimum variance optimization. Richard and Roncalli (2015) propose then a unified approach of smart beta portfolios by considering alternative allocation models as penalty functions of the minimum variance optimization. In particular, they use the logarithmic barrier function in order to regularize minimum variance portfolios. This idea has also been reiterated by de Jong (2018) who considers a mean-variance framework.

Therefore, we propose defining the robo-advisor optimization problem as follows:

$$\begin{aligned} x_{t+1}^{\star} &= \arg\min_{x} f_{\mathcal{R}obo}\left(x\right) \\ &\text{s.t.} \begin{cases} \mathbf{1}_n^\top x = 1 \\ \mathbf{0}_n \leq x \leq \mathbf{1}_n \\ x \in \Omega \end{cases} \end{aligned} \qquad [8.34]$$

where:

$$\begin{aligned} & f_{\mathcal{R}obo}\left(x\right) = \frac{1}{2}\left(x-b\right)^\top \Sigma_t \left(x-b\right) - \gamma\left(x-b\right)^\top \mu_t + \\ & \varrho_1 \left\|\Gamma_1\left(x-x_t\right)\right\|_1 + \frac{1}{2}\varrho_2 \left\|\Gamma_2\left(x-x_t\right)\right\|_2^2 + \\ & \tilde{\varrho}_1 \left\|\tilde{\Gamma}_1\left(x-\tilde{x}\right)\right\|_1 + \frac{1}{2}\tilde{\varrho}_2 \left\|\tilde{\Gamma}_2\left(x-\tilde{x}\right)\right\|_2^2 - \lambda \sum_{i=1}^{n} \mathcal{RB}_i \cdot \ln x_i \end{aligned} \qquad [8.35]$$

b is the benchmark portfolio, $\tilde{x}$ is the reference portfolio and x_t is the current portfolio.

This specification is sufficiently broad that it encompasses most models used by the industry. We note that the objective function is made up of three parts. The first part corresponds to the MVO objective function with a benchmark. If we set b equal to $\mathbf{0}_n$, we obtain the Markowitz utility function. The second part contains $\boldsymbol{\ell}_1$- and $\boldsymbol{\ell}_2$-norm penalty functions. The regularization can be done with respect to the current allocation x_t in order to control the rebalancing process and the smoothness of the dynamic allocation. The regularization can also be done with respect to a reference portfolio, which is generally the strategic asset allocation of the fund. The idea is to control the profile of the fund. For example, if the strategic asset allocation is an 80/20 asset mix policy, we do not want the portfolio to present a defensive or balanced risk profile. Finally, the third part of the objective function corresponds to the logarithmic barrier function, where the parameter λ controls the trade-off between MVO optimization and RB optimization. This last part is very important in order to make the dynamic asset allocation more robust. The hyperparameters of the objective function are ϱ_1, ϱ_2, $\tilde{\varrho}_1$, $\tilde{\varrho}_2$ and λ. They are all positive and can also be set to zero in order to deactivate a penalty function. For instance, ϱ_2 and $\tilde{\varrho}_2$ are equal to zero if we do not want to have a shrinkage of the covariance matrix Σ_t. The hyperparameters ϱ_1 and $\tilde{\varrho}_1$ can also be equal to zero because the $\boldsymbol{\ell}_1$ regularization is generally introduced when specifying the additional constraints Ω. The parameter γ is not really a hyperparameter because it is generally calibrated to target volatility or an expected return. We also note that this model encompasses the Black–Litterman model thanks to the specification of μ_t (Bourgeron *et al.* 2018). Another important component of this framework is the specification of the set $x \in \Omega$. It may include traditional constraints such as weight bounds and/or asset class limits, but we can also add nonlinear constraints such as a turnover limit, an active share floor or a weight diversification constraint.

8.4.3.2. *Derivation of the general algorithm*

Problem [8.34] is equivalent to solving:

$$x_{t+1}^{\star} = \arg\min_{x} f_{\mathcal{R}obo}^{+}(x)$$

where the objective function can be broken down as follows:

$$\begin{aligned} f_{\mathcal{R}obo}^{+}(x) = {} & f_{\mathrm{MVO}}(x) + f_{\boldsymbol{\ell}_1}(x) + f_{\boldsymbol{\ell}_2}(x) + f_{\mathrm{RB}}(x) + \\ & + \quad {}_{\Omega_0}(x) + \quad {}_{\Omega}(x) \end{aligned}$$

where:

$$\begin{aligned} f_{\text{MVO}}\left(x\right) &= \frac{1}{2}\left(x-b\right)^{\top}\Sigma_{t}\left(x-b\right)-\gamma\left(x-b\right)^{\top}\mu_{t} \\ f_{\ell_1}\left(x\right) &= \varrho_{1}\left\|\Gamma_{1}\left(x-x_{t}\right)\right\|_{1}+\tilde{\varrho}_{1}\left\|\tilde{\Gamma}_{1}\left(x-\tilde{x}\right)\right\|_{1} \\ f_{\ell_2}\left(x\right) &= \frac{1}{2}\varrho_{2}\left\|\Gamma_{2}\left(x-x_{t}\right)\right\|_{2}^{2}+\frac{1}{2}\tilde{\varrho}_{2}\left\|\tilde{\Gamma}_{2}\left(x-\tilde{x}\right)\right\|_{2}^{2} \\ f_{\text{RB}}\left(x\right) &= -\lambda\sum_{i=1}^{n}\mathcal{RB}_{i}\cdot\ln x_{i} \end{aligned}$$

and $\Omega_0 = \left\{x \in [0,1]^n : \mathbf{1}_n^{\top}x = 1\right\}$. The ADMM algorithm is implemented as follows:

$$\begin{aligned} \left\{x^{\star},y^{\star}\right\} &= \arg\min f_{x}\left(x\right)+f_{y}\left(y\right) \\ &\quad \text{s.t. } x-y=\mathbf{0}_{n} \end{aligned}$$

This is the general approach for solving the robo-advisor problem.

The main task is then to split the function $f_{\mathcal{R}obo}^{+}$ into f_x and f_y. However, in order to be efficient, the x- and y-update steps of the ADMM algorithm must be easy to compute. Therefore, we impose that the x-step is solved using QP or CCD methods, while the y-step is solved using Dykstra's algorithm, where each component corresponds to an analytical proximal operator. Moreover, we also split the set of constraints Ω into a set of linear constraints $\Omega_{\mathcal{L}inear}$ and a set of nonlinear constraints $\Omega_{\mathcal{N}onlinear}$. This leads to the definition of $f_x\left(x\right)$ and $f_y\left(y\right)$ as follows:

$$\begin{cases} f_{x}\left(x\right)=f_{\text{MVO}}\left(x\right)+f_{\ell_2}\left(x\right)+\mathbb{1}_{\Omega_0}\left(x\right)+\mathbb{1}_{\Omega_{\mathcal{L}inear}}\left(x\right) \\ f_{y}\left(y\right)=f_{\ell_1}\left(y\right)+f_{\text{RB}}\left(x\right)+\mathbb{1}_{\Omega_{\mathcal{N}onlinear}}\left(x\right) \end{cases} \qquad [8.36]$$

We note that $f_x\left(x\right)$ has a quadratic form, implying that the x-step may be solved using a QP algorithm. Another formulation is:

$$\begin{cases} f_{x}\left(x\right)=f_{\text{MVO}}\left(x\right)+f_{\ell_2}\left(x\right)+f_{\text{RB}}\left(x\right) \\ f_{y}\left(y\right)=f_{\ell_1}\left(y\right)+\mathbb{1}_{\Omega_0}\left(x\right)+\mathbb{1}_{\Omega_{\mathcal{L}inear}}\left(x\right)+\mathbb{1}_{\Omega_{\mathcal{N}onlinear}}\left(x\right) \end{cases} \qquad [8.37]$$

In this case, the x-step is solved using the CCD algorithm.

8.4.3.3. *Specific algorithms*

8.4.3.3.1. The ADMM-QP formulation

If we consider formulation [8.36], we have:

$$\begin{aligned}f_{\mathrm{QP}}(x) &= f_{\mathrm{MVO}}(x) + f_{\boldsymbol{\ell}_2}(x)\\ &= \frac{1}{2}(x-b)^\top \Sigma_t (x-b) - \gamma (x-b)^\top \mu_t\\ &\quad +\frac{1}{2}\varrho_2 \left\|\Gamma_2 (x - x_t)\right\|_2^2 + \frac{1}{2}\tilde{\varrho}_2 \left\|\tilde{\Gamma}_2 (x - \tilde{x})\right\|_2^2\\ &= \frac{1}{2}x^\top Q x - x^\top R + C\end{aligned}$$

where $Q = \Sigma_t + \varrho_2 \Gamma_2^\top \Gamma_2 + \tilde{\varrho}_2 \tilde{\Gamma}_2^\top \tilde{\Gamma}_2$, $R = \gamma \mu_t + \Sigma_t b + \varrho_2 \Gamma_2^\top \Gamma_2 x_t + \tilde{\varrho}_2 \tilde{\Gamma}_2^\top \tilde{\Gamma}_2 \tilde{x}$ and C is a constant. Using the fourth ADMM trick, we deduce that $x^{(k+1)}$ is the solution of the following QP problem:

$$\begin{aligned}x^{(k+1)} &= \arg\min_x \frac{1}{2}x^\top (Q + \varphi I_n) x - x^\top \left(R + \varphi \left(y^{(k)} - u^{(k)}\right)\right)\\ &\quad \text{s.t.} \begin{cases} \mathbf{1}_n^\top x = 1\\ \mathbf{0}_n \leq x \leq \mathbf{1}_n\end{cases}\end{aligned}$$

Since the proximal operators of $f_{\boldsymbol{\ell}_1}$ and f_{RB} have already been computed, finding $y^{(k+1)}$ is straightforward with Dykstra's algorithm, as long as the proximal of each nonlinear constraint is known.

8.4.3.3.2. The ADMM-CCD formulation

If we consider formulation [8.37], we have:

$$f_x(x) = f_{\mathrm{QP}}(x) - \lambda \sum_{i=1}^{n} \mathcal{RB}_i \cdot \ln x_i$$

We can show that the CCD algorithm applied to x-update is:

$$x_i^{(k+1)} = \frac{R_i - \sum_{j<i} x_j^{(k+1)} Q_{i,j} - \sum_{j>i} x_j^{(k)} Q_{i,j}}{2Q_{i,i}} + \frac{\sqrt{\left(\sum_{j<i} x_j^{(k+1)} Q_{i,j} + \sum_{j>i} x_j^{(k)} Q_{i,j} - R_i\right)^2 + 4\lambda_i Q_{i,i}}}{2Q_{i,i}}$$

where $Q = \Sigma_t + \varrho_2 \Gamma_2^\top \Gamma_2 + \tilde{\varrho}_2 \tilde{\Gamma}_2^\top \tilde{\Gamma}_2 + \varphi I_n$, $R = \gamma \mu_t + \Sigma_t b + \varrho_2 \Gamma_2^\top \Gamma_2 x_t + \tilde{\varrho}_2 \tilde{\Gamma}_2^\top \tilde{\Gamma}_2 \tilde{x} + \varphi \left(y^{(k)} - u^{(k)}\right)$ and $\lambda_i = \lambda \cdot \mathcal{RB}_i$. Like the ADMM-QP formulation, the y-update step does not pose any particular difficulties.

8.4.4. *Tips and tricks*

If we consider the different portfolio optimization approaches presented in Table 8.1, we have shown how to solve MVO (1), GMV (2), MDP (3), ERC (4) and RB (5) models. The RQE (7) model is equivalent to the GMV (2) model by replacing the covariance matrix Σ by the dissimilarity matrix D. Below, we implement the Kullback–Leibler model (4) of Bera and Park (2008) using the ADMM framework. Concerning the regularization problems in Table 8.2, ridge (8), lasso (9) and log-barrier (10) penalty functions have already been covered. Indeed, ridge and lasso penalizations correspond to the proximal operator of ℓ_1- and ℓ_2-norm functions by applying the translation $g(x) = x - \tilde{x}$. Shannon's entropy (11) penalization is discussed below. For the constraints that are considered in Table 8.3, imposing no cash and leverage (12) is done with the proximal of the hyperplane $\mathcal{H}_{yperlane}\left[\mathbf{1}_n, 1\right]$. No short selling (13) and weight bounds (14) are equivalent to considering the box projections $\mathcal{B}_{ox}\left[\mathbf{0}_n, \infty\right]$ and $\mathcal{B}_{ox}\left[x^-, x^+\right]$. Asset class limits can be implemented using the projection onto the intersection of two half-spaces $\mathcal{H}_{alfspace}\left[\mathbf{1}_{i \in C_j}, c_j^+\right]$ and $\mathcal{H}_{alfspace}\left[-\mathbf{1}_{i \in C_j}, -c_j^-\right]$. The proximal of the turnover (16) had already been given in equation [8.31]. If we want to impose an upper limit on transaction costs (17), we use the Moreau decomposition and equation [8.30]. Finally, section 8.4.1.1 dealt with the weight diversification problem of the number of active bets. Therefore, it remains to solve leverage limits (18), long/short exposure (19) restrictions and active management constraints: benchmarking (20), tracking error floor (21) and active share floor (22).

8.4.4.1. *Volatility and return targeting*

We first consider the μ-problem and the σ-problem. Targeting a minimum expected return $\mu(x) \geq \mu^\star$ can be implemented in the ADMM framework using the proximal operator of the hyperplane[26] $\mathcal{H}_{yperlane}\left[-\mu, -\mu^\star\right]$. In the case of the σ-problem $\sigma(x) \leq \sigma^\star$, we use the fourth ADMM trick. Let L be the lower Cholesky decomposition of Σ, we have:

$$\sqrt{x^\top \Sigma x} \leq \sigma^\star \Leftrightarrow \sqrt{x^\top \left(LL^\top\right) x} \leq \sigma^\star \Leftrightarrow \left\|y^\top y\right\|_2 \leq \sigma^\star$$

where $y = L^\top x$. It follows that the proximal of the y-update is the projection onto the ℓ_2 ball $\mathcal{B}_2\left(\mathbf{0}_n, \sigma^\star\right)$.

8.4.4.2. *Leverage management*

If we impose a leverage limit $\sum_{i=1}^n |x_i| \leq \mathcal{L}^+$, we have $\|x\|_1 \leq \mathcal{L}^+$ and the proximal of the y-update is the projection onto the ℓ_1 ball $\mathcal{B}_1\left(\mathbf{0}_n, \mathcal{L}^+\right)$. If the leverage constraint concerns the long/short limits $-\mathcal{LS}^- \leq \sum_{i=1}^n x_i \leq \mathcal{LS}^+$, we consider the intersection of the two half-spaces $\mathcal{H}_{alfspace}\left[\mathbf{1}_n, \mathcal{LS}^+\right]$ and $\mathcal{H}_{alfspace}\left[-\mathbf{1}_n, \mathcal{LS}^-\right]$. If we consider an absolute leverage $\left|\sum_{i=1}^n x_i\right| \leq \mathcal{L}^+$, we obtain the previous case with $\mathcal{LS}^- = \mathcal{LS}^+ = \mathcal{L}^+$. Portfolio managers can also use another constraint concerning the sum of the k largest values[27]:

$$f(x) = \sum_{i=n-k+1}^{n} x_{(i:n)} = x_{(n:n)} + \ldots + x_{(n-k+1:n)}$$

where $x_{(i:n)}$ is the order statistics of x: $x_{(1:n)} \leq x_{(2:n)} \leq \cdots \leq x_{(n:n)}$. Beck (2017) shows that:

$$\mathbf{prox}_{\lambda f(x)}(v) = v - \lambda \mathcal{P}_\Omega\left(\frac{v}{\lambda}\right)$$

[26] We have $\mu(x) \geq \mu^\star \Leftrightarrow x^\top \mu \geq \mu^\star \Leftrightarrow -\mu^\top x \leq -\mu^\star$.

[27] An example is the 5/10/40 UCITS rule: A UCITS fund may invest no more than 10% of its net assets in transferable securities or money market instruments issued by the same body, with a further aggregate limitation of 40% of net assets on exposures of greater than 5% to single issuers.

where:

$$\Omega = \left\{ x \in [0,1]^n : \mathbf{1}_n^\top x = k \right\} = \mathcal{B}_{ox}\left[\mathbf{0}_n, \mathbf{1}_n\right] \cap \mathcal{H}_{yperlane}\left[\mathbf{1}_n, k\right]$$

8.4.4.3. *Entropy portfolio and diversification measure*

Bera and Park (2008) propose using a cross-entropy measure as the objective function:

$$\begin{aligned} x^\star &= \arg\min_x \mathrm{KL}\left(x \mid \tilde{x}\right) \\ &\text{s.t.} \begin{cases} \mathbf{1}_n^\top x = 1 \\ \mathbf{0}_n \leq x \leq \mathbf{1}_n \\ \mu\left(x\right) \geq \mu^\star \\ \sigma\left(x\right) \leq \sigma^\star \end{cases} \end{aligned}$$

where $\mathrm{KL}\left(x \mid \tilde{x}\right) = \sum_{i=1}^n x_i \ln\left(x_i / \tilde{x}_i\right)$ and $\tilde{x}$ is a reference portfolio, which is well diversified (e.g. the EW[28] or ERC portfolio). In Appendix 8.7.1.4.2, we show that the proximal operator of $\lambda\,\mathrm{KL}\left(x \mid \tilde{x}\right)$ is equal to:

$$\mathbf{prox}_{\lambda\,\mathrm{KL}(v \mid \tilde{x})}\left(v\right) = \lambda \begin{pmatrix} W\left(\lambda^{-1}\tilde{x}_1 e^{\lambda^{-1}v_1 - \tilde{x}_1^{-1}}\right) \\ \vdots \\ W\left(\lambda^{-1}\tilde{x}_n e^{\lambda^{-1}v_n - \tilde{x}_n^{-1}}\right) \end{pmatrix}$$

where $W\left(x\right)$ is the Lambert W function.

REMARK 8.7.– Using the previous result and the fact that $\mathrm{SE}\left(x\right) = -\mathrm{KL}\left(x \mid \mathbf{1}_n\right)$, we can use Shannon's entropy to define the diversification measure $\mathcal{D}\left(x\right) = \mathrm{SE}\left(x\right)$. Therefore, solving problem [8.28] is straightforward when we consider the following diversification set:

$$\mathfrak{D} = \left\{ x \in [0,1]^n : -\sum_{i=1}^n x_i \ln x_i \geq \mathrm{SE}^- \right\}$$

[28] In this case, it is equivalent to maximize Shannon's entropy because $\tilde{x} = \mathbf{1}_n$.

8.4.4.4. *Passive and active management*

In the case of the active share, we use the translation property:

$$\mathcal{AS}\left(x \mid \tilde{x}\right) = \frac{1}{2}\sum_{i=1}^{n}\left|x_i - \tilde{x}_i\right| = \frac{1}{2}\left\|x - \tilde{x}\right\|_1$$

The proximal operator is given in Appendix 8.7.1.5. It is interesting to note that this type of problem cannot be solved using an augmented QP algorithm since it involves the complement of the ℓ_1 ball and not directly the ℓ_1 ball itself. In this case, we face a maximization problem and not a minimization problem and the technique of augmented variables does not work. For tracking error volatility, again we use the fourth ADMM trick:

$$\sigma\left(x \mid \tilde{x}\right) = \sqrt{\left(x - \tilde{x}\right)^\top \Sigma\left(x - \tilde{x}\right)} = \left\|y\right\|_2$$

where $y = L^\top x - L^\top \tilde{x}$. Using our ADMM notations, we have $Ax + By = c$, where $A = L^\top$, $B = -I_n$ and $c = L^\top \tilde{x}$.

8.5. Conclusion

The aim of this chapter is to propose an alternative solution to the quadratic programming algorithm in the context of portfolio allocation. In numerical analysis, the quadratic programming model is a powerful optimization tool, which is computationally very efficient. In portfolio management, the mean-variance optimization model is exactly a quadratic programming model, meaning that it benefits from its computational power. Therefore, the success of the Markowitz allocation model is explained by these two factors: the quadratic utility function and the quadratic programming setup. A lot of academics and professionals have proposed an alternative approach to the MVO framework, but very few of these models are used in practice. The main reason is that these competing models focus on the objective function and not on the numerical implementation. However, we believe that any model which is not tractable will have little success with portfolio managers. The analogy is obvious if we consider the theory of options. The success of the Black–Scholes model lies in the Black–Scholes analytical formula. Over the last 30 years, many models have been created (e.g. local volatility and stochastic volatility models), but only one can really compete with the Black–Scholes model. This is the SABR model, and the main reason is that it has an analytical formula for implied volatility.

This chapter focuses then on a general approach for numerically solving non-QP portfolio allocation models. For that, we consider some algorithms that have been successfully applied to machine learning and large-scale optimization. For instance, the coordinate descent algorithm is the fastest method for performing high-dimensional lasso regression, while Dykstra's algorithm has been created to find the solution of restricted least squares regression. Since there is a strong link between MVO and linear regression (Scherer 2007), it is not a surprise if these algorithms can help solve regularized MVO allocation models. However, these two algorithms are not sufficient for defining a general framework. For that, we need to use the alternating direction method of multipliers and proximal gradient methods. Finally, the combination of these four algorithms (CD, ADMM, PO and Dykstra) allows us to consider allocation models that cannot be cast into a QP form.

In this chapter, we have first considered allocation models with non-quadratic objective functions. For example, we have used models based on the diversification ratio, Shannon's entropy or the Kullback–Leibler divergence. Second, we have solved regularized MVO models with nonlinear penalty functions such as the ℓ_p-norm penalty or the logarithmic barrier. Third, we have discussed how to handle nonlinear constraints. For instance, we have imposed constraints on active share, volatility targeting, leverage limits, transaction costs, etc. Most importantly, these three non-QP extensions can be combined.

With the development of quantitative strategies (smart beta, factor investing, alternative risk premia, systematic strategies, robo-advisors, etc.), the asset management industry has dramatically changed over the last five years. This is just the beginning and we think that alternative data, machine learning methods and artificial intelligence will massively shape investment processes in the future. This chapter is an illustration of this trend and shows how machine learning optimization algorithms allow us to move away from the traditional QP world of portfolio management.

8.6. Acknowledgements

We would like to thank Mohammed El Mendili, Edmond Lezmi, Lina Mezghani, Jean-Charles Richard, Jules Roche and Jiali Xu for their helpful comments.

8.7. Appendix

8.7.1. *Mathematical results*

8.7.1.1. *Augmented QP formulation of the turnover management problem*

The augmented QP problem is defined by:

$$\begin{aligned} X^{\star} &= \arg\min_{X} \frac{1}{2} X^{\top} Q X - X^{\top} R \\ & \text{s.t.} \left\{ \begin{array}{l} AX = B \\ CX \leq D \\ \mathbf{0}_{3n} \leq X \leq \mathbf{1}_{3n} \end{array} \right. \end{aligned}$$

where $X = \left(x_1, \ldots, x_n, x_1^-, \ldots, x_n^-, x_1^+, \ldots, x_n^+\right)$ is a $3n \times 1$ vector, Q is a $3n \times 3n$ matrix:

$$Q = \begin{pmatrix} \Sigma & \mathbf{0}_{n \times n} & \mathbf{0}_{n \times n} \\ \mathbf{0}_{n \times n} & \mathbf{0}_{n \times n} & \mathbf{0}_{n \times n} \\ \mathbf{0}_{n \times n} & \mathbf{0}_{n \times n} & \mathbf{0}_{n \times n} \end{pmatrix}$$

$R = (\gamma \mu, \mathbf{0}_n, \mathbf{0}_n)$ is a $3n \times 1$ vector, A is a $(n+1) \times 3n$ matrix:

$$A = \begin{pmatrix} \mathbf{1}_n^{\top} & \mathbf{0}_n^{\top} & \mathbf{0}_n^{\top} \\ I_n & I_n & -I_n \end{pmatrix}$$

$B = (1, \bar{x})$ is a $(n+1) \times 1$ vector, $C = \begin{pmatrix} \mathbf{0}_n^{\top} & \mathbf{1}_n^{\top} & \mathbf{1}_n^{\top} \end{pmatrix}$ is a $1 \times 3n$ matrix and $D = \boldsymbol{\tau}^{+}$.

8.7.1.2. *QP problem with a hyperplane constraint*

We consider the following QP problem:

$$x^{\star} = \arg\min_{x} \frac{1}{2} x^{\top} Q x - x^{\top} R \qquad \text{s.t.} \quad a^{\top} x = b$$

The associated Lagrange function is $\mathcal{L}(x;\lambda) = \frac{1}{2} x^{\top} Q x - x^{\top} R + \lambda \left(a^{\top} x - b\right)$. The first-order conditions are $\partial_x \mathcal{L}(x;\lambda) = Qx - R + \lambda a = \mathbf{0}_n$ and $\partial_\lambda \mathcal{L}(x;\lambda) = a^{\top} x - b = 0$. We obtain $x = Q^{-1}(R + \lambda a)$. Because $a^{\top} x - b = 0$, we have $a^{\top} Q^{-1} R + \lambda a^{\top} Q^{-1} a = b$ and:

$$\lambda^{\star} = \frac{b - a^{\top} Q^{-1} R}{a^{\top} Q^{-1} a}$$

The optimal solution is then:

$$x^{\star} = Q^{-1} \left(R + \frac{b - a^{\top} Q^{-1} R}{a^{\top} Q^{-1} a} a \right)$$

8.7.1.3. *Derivation of the soft-thresholding operator*

We consider the equation $cx - v + \lambda \partial |x| \in 0$, where $c > 0$ and $\lambda > 0$. Since we have $\partial |x| = \operatorname{sign}(x)$, we deduce that:

$$x^{\star} = \begin{cases} c^{-1}(v + \lambda) & \text{if } x^{\star} < 0 \\ 0 & \text{if } x^{\star} = 0 \\ c^{-1}(v - \lambda) & \text{if } x^{\star} > 0 \end{cases}$$

If $x^{\star} < 0$ or $x^{\star} > 0$, then we have $v + \lambda < 0$ or $v - \lambda > 0$. This is equivalent to set $|v| > \lambda > 0$. The case $x^{\star} = 0$ implies that $|v| \leq \lambda$. We deduce that $x^{\star} = c^{-1} \cdot \mathcal{S}(v;\lambda)$, where $\mathcal{S}(v;\lambda)$ is the soft-thresholding operator:

$$\begin{aligned} \mathcal{S}(v;\lambda) &= \begin{cases} 0 & \text{if } |v| \leq \lambda \\ v - \lambda \operatorname{sign}(v) & \text{otherwise} \end{cases} \\ &= \operatorname{sign}(v) \cdot (|v| - \lambda)_{+} \end{aligned}$$

8.7.1.4. *Proximal operators*

8.7.1.4.1. Projection onto the intersection of a ℓ_2 ball and a box

We note $f_1(x) = {}_{\Omega_1}(x)$ and $f_2(x) = {}_{\Omega_2}(x)$, where $\Omega_1 = \mathcal{B}_2(c, r)$ and $\Omega_2 = \mathcal{B}_{ox}\left[x^-, x^+\right] = \left\{x \in \mathbb{R}^n : x^- \leq x \leq x^+\right\}$. Dykstra”s algorithm becomes:

$$\left\{\begin{array}{l} x^{(k+1)} = c + \dfrac{r}{\max\left(r, \left\| y^{(k)} + z_1^{(k)} - c\right\|_2\right)}\left(y^{(k)} + z_1^{(k)} - c\right) \\ z_1^{(k+1)} = y^{(k)} + z_1^{(k)} - x^{(k+1)} \\ y^{(k+1)} = \mathcal{T}\left(x^{(k+1)} + z_2^{(k)}; x^-, x^+\right) \\ z_2^{(k+1)} = x^{(k+1)} + z_2^{(k)} - y^{(k+1)} \end{array}\right.$$

This algorithm is denoted by $\mathcal{P}_{\mathcal{B}\text{ox}-\mathcal{B}\text{all}}\left(v; x^-, x^+, c, r\right)$.

8.7.1.4.2. Shannon’s entropy and Kullback–Leibler divergence

If we consider the scalar function $f(x) = \lambda x \ln(x/\tilde{x})$, where $\tilde{x}$ is a constant, we have:

$$\lambda f(x) + \frac{1}{2}\|x - v\|_2^2 = \lambda x \ln\frac{x}{\tilde{x}} + \frac{1}{2}x^2 - xv + \frac{1}{2}v^2$$

The first-order condition is:

$$\lambda\frac{1}{\tilde{x}} + \lambda\ln\frac{x}{\tilde{x}} + x - v = 0 \Leftrightarrow \left(\lambda^{-1}x\right)e^{\left(\lambda^{-1}x\right)} = \lambda^{-1}\tilde{x}e^{\lambda^{-1}v - \frac{1}{\tilde{x}}}$$

We deduce that the root is equal to $x^\star = \lambda W\left(\frac{\tilde{x}e^{\lambda^{-1}v - \frac{1}{\tilde{x}}}}{\lambda}\right)$, where $W(x)$ is the Lambert W function satisfying $W(x)\,e^{W(x)} = x$. In the case of the Kullback–Leibler divergence $\mathrm{KL}(x) = \sum_{i=1}^n x_i \ln(x_i/\tilde{x}_i)$, it follows that:

$$\mathbf{prox}_{\lambda\,\mathrm{KL}(v|\tilde{x})}(v) = \lambda\begin{pmatrix} W\left(\lambda^{-1}\tilde{x}_1 e^{\lambda^{-1}v_1 - \tilde{x}_1^{-1}}\right) \\ \vdots \\ W\left(\lambda^{-1}\tilde{x}_n e^{\lambda^{-1}v_n - \tilde{x}_n^{-1}}\right) \end{pmatrix}$$

The proximal of Shannon's entropy $\mathrm{SE}(x) = -\sum_{i=1}^{n} x_i \ln x_i$ is a special case of the previous result[29] with $\tilde{x}_i = 1$.

8.7.1.4.3. Projection onto the complement $\bar{\mathcal{B}}_2(c, r)$ of the ℓ_2 ball

We consider the proximal problem where $\Omega = \{x \in \mathbb{R}^n : \|x - c\|_2 \geq r\}$:

$$\begin{aligned} x^\star &= \arg\min_x \frac{1}{2}(x - v)^\top (x - v) \\ &\quad \text{s.t. } (x - c)^\top (x - c) - r^2 \geq 0 \end{aligned}$$

We deduce that the Lagrange function is equal to:

$$\mathcal{L}(x; \lambda) = \frac{1}{2}(x - v)^\top (x - v) - \lambda \left((x - c)^\top (x - c) - r^2 \right)$$

The first-order condition is $\partial_x \mathcal{L}(x; \lambda) = x - v - 2\lambda(x - c) = \mathbf{0}_n$, whereas the KKT condition is $\min\left(\lambda, (x - c)^\top (x - c) - r^2\right) = 0$. We distinguish two cases:

1) if $\lambda = 0$, this means that $x^\star = v$ and $(x - c)^\top (x - c) - r^2 > 0$;

2) if $\lambda > 0$, we have $(x - c)^\top (x - c) = r^2$. Then, we obtain the following system:

$$\begin{cases} x - v - 2\lambda(x - c) = \mathbf{0}_n \\ (x - c)^\top (x - c) = r^2 \end{cases}$$

We deduce that:

$$(x - c) - (v - c) - 2\lambda(x - c) = \mathbf{0}_n \qquad [8.38]$$

and:

$$\lambda^\star = \frac{r^2 - (x - c)^\top (v - c)}{2r^2}$$

We note that:

$$[8.38] \Leftrightarrow (x - c)^\top (v - c)(x - c) = r^2 (v - c)$$

[29] We use the fact that $\max \mathrm{SE}(x) = \min - \mathrm{SE}(x)$.

Because $(x-c)^\top (v-c)$ is a scalar, we deduce that $x-c$ and $v-c$ are two collinear vectors. The optimal solution is:

$$x^\star = c + r\frac{(v-c)}{\|v-c\|_2}$$

Combining the two cases gives:

$$\mathbf{prox}_{1_{\Omega(x)}}(v) = c + \frac{r}{\min(r, \|v-c\|_2)}(v-c)$$

This is the formula of the projection onto the ℓ_2 ball, but the minimum function has replaced the maximum function.

8.7.1.5. *Projection onto the complement $\bar{\mathcal{B}}_1(c,r)$ of the ℓ_1 ball*

This proximal problem associated with $\bar{\mathcal{B}}_1(\mathbf{0}_n, r)$ is:

$$x^\star = \arg\min_x \frac{1}{2}(x-v)^\top (x-v) \qquad \text{s.t.} \quad \|x\|_1 \geq r$$

We deduce that the Lagrange function is equal to:

$$\mathcal{L}(x;\lambda) = \frac{1}{2}(x-v)^\top (x-v) - \lambda(\|x\|_1 - r)$$

The first-order condition is $\partial_x \mathcal{L}(x;\lambda) = x - v - \lambda \operatorname{sign}(x) = \mathbf{0}_n$, whereas the KKT condition is $\min(\lambda, \|x\|_1 - r) = 0$. We distinguish two cases. If $\lambda = 0$, this means that $x^\star = v$ and $\|x\|_1 \geq r$. If $\lambda > 0$, we have $\|x\|_1 = r$. Then, we obtain the following system of equations: $x - v = \lambda \operatorname{sign}(x)$ and $\|x\|_1 = r$. The first condition gives that $x - v$ is a vector whose elements are $+\lambda$ and/or $-\lambda$, whereas the second condition shows that x is on the surface of the ℓ_1 ball. Unfortunately, there is no unique solution. Hence, we assume that $\operatorname{sign}(x) = \operatorname{sign}(v)$ and we modify the sign function: $\operatorname{sign}(a) = 1$ if $a \geq 0$ and $\operatorname{sign}(a) = -1$ if $a < 0$. In this case, there is a unique solution $x^\star = v + \lambda^\star \operatorname{sign}(v)$, where $\lambda^\star = n^{-1}(r - \|v\|_1)$ because $|v + \lambda \operatorname{sign}(v)| = |v| + \lambda$. Combining the two cases implies that:

$$\mathcal{P}_{\bar{\mathcal{B}}_1(\mathbf{0}_n,r)}(v) = \begin{cases} v & \text{if } \|v\|_1 \geq r \\ v + \operatorname{sign}(v) \odot n^{-1}(r - \|v\|_1) & \text{if } \|v\|_1 < r \end{cases}$$

Using the translation property, we deduce that:

$$\mathcal{P}_{\bar{\mathcal{B}}_1(c,r)}(v) = v + \operatorname{sign}(v - c) \odot \frac{\max\left(r - \|v - c\|_1, 0\right)}{n}$$

8.7.1.6. *The bid-ask linear cost function*

If we consider the scalar function $f(x) = \alpha(\gamma - x)_+ + \beta(x - \gamma)_+$, we have:

$$f_v(x) = \lambda\alpha(\gamma - x)_+ + \lambda\beta(x - \gamma)_+ + \frac{1}{2}x^2 - xv + \frac{1}{2}v^2$$

Following Beck (2017), we distinguish three cases:

1) If $f_v(x) = \lambda\alpha(\gamma - x) + \frac{1}{2}x^2 - xv + \frac{1}{2}v^2$, then $f_v'(x) = -\lambda\alpha + x - v$ and $x^\star = v + \lambda\alpha$. This implies that $\gamma - x^\star > 0$ or $v < \gamma - \lambda\alpha$.

2) If $f_v(x) = \lambda\beta(x - \gamma)_+ + \frac{1}{2}x^2 - xv + \frac{1}{2}v^2$, then $f_v'(x) = \lambda\beta + x - v$ and $x^\star = v - \lambda\beta$. This implies that $x^\star - \gamma > 0$ or $v < \gamma + \lambda\beta$.

3) If $v \in [\gamma - \lambda\alpha, \gamma + \lambda\beta]$, the minimum is not obtained at a point of differentiability. Since γ is the only point of non-differentiability, we obtain $x^\star = \gamma$.

Therefore, we can write the proximal operator in the following compact form:

$$x^\star = \gamma + (v - \gamma - \lambda\beta)_+ - (v - \gamma + \lambda\alpha)_-$$

where x_- and x_+ are the negative part and the positive part of x, respectively. If we consider the vector-value function $f(x) = \sum_{i=1}^n \alpha_i(\gamma_i - x_i)_+ + \beta_i(x_i - \gamma_i)_+$, we deduce that:

$$\mathbf{prox}_{\lambda f(x)}(v) = \gamma + \mathcal{S}(v - \gamma; \lambda\alpha, \lambda\beta)$$

where $\mathcal{S}(v; \lambda_-, \lambda_+) = (v - \lambda_+)_+ - (v + \lambda_-)_-$ is the two-sided soft-thresholding operator.

8.7.2. *Data*

8.7.2.1. *Parameter set #1*

We consider a capitalization-weighted stock index, which is composed of eight stocks. The weights of this benchmark are equal to 23%, 19%, 17%, 9%, 8%, 6% and 5%. We assume that their volatilities are 21%, 20%, 40%, 18%, 35%, 23%, 7% and 29%. The correlation matrix is defined as follows:

$$\rho = \begin{pmatrix} 100\% & & & & & & & \\ 80\% & 100\% & & & & & & \\ 70\% & 75\% & 100\% & & & & & \\ 60\% & 65\% & 90\% & 100\% & & & & \\ 70\% & 50\% & 70\% & 85\% & 100\% & & & \\ 50\% & 60\% & 70\% & 80\% & 60\% & 100\% & & \\ 70\% & 50\% & 70\% & 75\% & 80\% & 50\% & 100\% & \\ 60\% & 65\% & 70\% & 75\% & 65\% & 70\% & 80\% & 100\% \end{pmatrix}$$

8.7.2.2. *Parameter set #2*

We consider a universe of eight stocks. We assume that their volatilities are 25%, 20%, 15%, 18%, 30%, 20%, 15% and 35%. The correlation matrix is defined as follows:

$$\rho = \begin{pmatrix} 100\% & & & & & & & \\ 20\% & 100\% & & & & & & \\ 55\% & 60\% & 100\% & & & & & \\ 60\% & 60\% & 60\% & 100\% & & & & \\ 60\% & 60\% & 60\% & 60\% & 100\% & & & \\ 60\% & 60\% & 60\% & 60\% & 60\% & 100\% & & \\ 60\% & 60\% & 60\% & 60\% & 60\% & 60\% & 100\% & \\ 60\% & 60\% & 60\% & 60\% & 60\% & 60\% & 60\% & 100\% \end{pmatrix}$$

8.8. References

Bauschke, H.H. and Borwein, J.M. (1994). Dykstra's alternating projection algorithm for two sets. *Journal of Approximation Theory*, 79(3), 418–443.

Beale, E.M.L. (1959). On quadratic programming. *Naval Research Logistics Quarterly*, 6(3), 227–243.

Beck, A. (2017). *First-Order Methods in Optimization*. MOS-SIAM Series on Optimization, 25, SIAM.

Bera, A.K. and Park, S.Y. (2008). Optimal portfolio diversification using the maximum entropy principle. *Econometric Reviews*, 27(4–6), 484–512.

Bishop, C.M. (1995). *Neural Networks for Pattern Recognition*. Oxford University Press.

Bourgeron, T., Lezmi, E., and Roncalli, T. (2018). Robust asset allocation for Robo-Advisors. *arXiv*, 1902.05710.

Brodie, J., Daubechies, I., De Mol, C., Giannone, D., and Loris, I. (2009). Sparse and stable Markowitz portfolios. *Proceedings of the National Academy of Sciences*, 106(30), 12267–12272.

Bruder, B., Gaussel, N., Richard, J-C., and Roncalli, T. (2013). Regularization of portfolio allocation. *SSRN*. Available at: www.ssrn.com/abstract=2767358.

Bruder, B., Kostyuchyk, N., and Roncalli, T. (2016). Risk parity portfolios with skewness risk: An application to factor investing and alternative risk Premia. *SSRN*. Available at: www.ssrn.com/abstract=2813384.

Boyd, S., Parikh, N., Chu, E., Peleato, B., and Eckstein, J. (2010). Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning*, 3(1), 1–122.

Carmichael, B., Koumou, G.B., and Moran, K. (2018). Rao's quadratic entropy and maximum diversification indexation. *Quantitative Finance*, 18(6), 1017–1031.

Carrasco, M., and Noumon, N. (2010). Optimal portfolio selection using regularization, University of Montreal. Discussion Paper.

Choueifaty, Y. and Coignard, Y. (2008). Toward maximum diversification. *Journal of Portfolio Management*, 35(1), 40–51.

Choueifaty, Y., Froidure, T. and Reynier, J. (2013). Properties of the most diversified portfolio. *Journal of Investment Strategies*, 2(2), 49–70.

Combettes, P.L. and Pesquet, J.C. (2011). Proximal splitting methods in signal processing. In *Fixed-point Algorithms for Inverse Problems in Science and Engineering*, Bauschke, H.H., Burachik, R.S., Combettes, P.L., Elser, V., Luke, D.R., and Wolkowicz, H. (eds). Springer Optimization and Its Applications, 48, Springer, New York.

Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine Learning*, 20(3), 273–297.

Cottle, R.W. and Infanger, G. (2010). Harry Markowitz and the early history of quadratic programming. In *Handbook of Portfolio Construction*, Guerard, J.B. (ed.). Springer, New York.

de Jong, M. (2018). Portfolio optimisation in an uncertain world. *Journal of Asset Management*, 19(4), 216–221.

DeMiguel, V., Garlappi, L., Nogales, F.J., and Uppal, R. (2009). A generalized approach to portfolio optimization: Improving performance by constraining portfolio norms. *Management Science*, 55(5), 798–812.

Dykstra, R.L. (1983). An algorithm for restricted least squares regression. *Journal of the American Statistical Association*, 78(384), 837–842.

Frank, M., and Wolfe, P. (1956). An algorithm for quadratic programming. *Naval Research Logistics Quarterly*, 3, 95–110.

Gabay, D., and Mercier, B. (1976). A dual algorithm for the solution of nonlinear variational problems via finite element approximation. *Computers & Mathematics with Applications*, 2(1), 17–40.

Gonzalvez, J., Lezmi, E., Roncalli, T., and Xu, J. (2019). Financial applications of Gaussian processes and bayesian optimization. *arXiv*. Available at: arxiv.org/abs/1903.04841.

Gould, N.I.M. and Toint, P.L. (2000). A quadratic programming bibliography. *Numerical Analysis Group Internal Report*, 1, 142.

Griveau-Billion, T., Richard, J-C., and Roncalli, T. (2013). A fast algorithm for computing high-dimensional risk parity portfolios. *SSRN*. Available at: www.ssrn.com/abstract=2325255.

Hastie, T., Tibshirani, R. and Friedman, J. (2009). *The Elements of Statistical Learning*. 2nd edition, Springer, New York.

Jacobs, R.A. (1988). Increased rates of convergence through learning rate adaptation. *Neural Networks*, 1(4), 295–307.

Jagannathan, R. and Ma, T. (2003). Risk reduction in large portfolios: Why imposing the wrong constraints helps. *Journal of Finance*, 58(4), 1651–1684.

LeCun, Y., Boser, B., Denker, J.S., Henderson, D., Howard, R.E., Hubbard, W. and Jackel, L.D. (1989). Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1(4), 541–551.

Lezmi, E., Malongo, H., Roncalli, T., and Sobotka, R. (2018). Portfolio allocation with skewness risk: A practical guide. *SSRN*. Available at: www.ssrn.com/abstract=3201319.

Maillard, S., Roncalli, T. and Teïletche, J. (2010). The properties of equally weighted risk contribution portfolios. *Journal of Portfolio Management*, 36(4), 60–70.

Mann, H.B. (1943). Quadratic forms with linear constraints. *American Mathematical Monthly*, 50, 430–433.

Markowitz, H. (1952). Portfolio selection. *Journal of Finance*, 7(1), 77–91.

Markowitz, H. (1956). The optimization of a quadratic function subject to linear constraints. *Naval Research Logistics Quarterly*, 3(1–2), 111–133.

Michaud, R.O. (1989). The Markowitz optimization enigma: Is 'optimized' optimal? *Financial Analysts Journal*, 45(1), 31–42.

Nesterov, Y.E. (1983). A method for solving the convex programming problem with convergence rate $O\left(k^{-2}\right)$. *Doklady Akademii Nauk SSSR*, 269, 543–547.

Nesterov, Y.E. (2012). Efficiency of coordinate descent methods on huge-scale optimization problems. *SIAM Journal on Optimization*, 22(2), 341–362.

Parikh, N., and Boyd, S. (2014). Proximal algorithms. *Foundations and Trends® in Optimization*, 1(3), 127–239.

Polyak, B.T. (1964). Some methods of speeding up the convergence of iteration methods. *USSR Computational Mathematics and Mathematical Physics*, 4(5), 1–17.

Qian, E. (2005). Risk parity portfolios: Efficient portfolios through true diversification. *Panagora Asset Management*, September.

Richard, J-C. and Roncalli, T. (2015). Smart beta: Managing diversification of minimum variance portfolios. In *Risk-based and Factor Investing*, Jurczenko, E. (ed.), ISTE Press, London and Elsevier Oxford.

Richard, J-C. and Roncalli, T. (2019). Constrained risk budgeting portfolios: Theory, algorithms, applications & puzzles. *arXiv*, 1902.05710.

Roncalli, T. (2013). *Introduction to Risk Parity and Budgeting*, Chapman & Hall/CRC Financial Mathematics Series, Boca Raton.

Roncalli, T. (2015). Introducing expected returns into risk parity portfolios: A new framework for asset allocation. *Bankers, Markets & Investors*, 138, 18–28.

Scherer, B. (2007). *Portfolio Construction & Risk Budgeting*. 3rd edition. Risk Books, London.

Tibshirani, R. (1996). Regression shrinkage and selection via the Lasso. *Journal of the Royal Statistical Society B*, 58(1), 267–288.

Tibshirani, R.J. (2017). Dykstra's algorithm, ADMM, and coordinate descent: Connections, insights, and extensions. In *Advances in Neural Information Processing Systems*, Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S. and Garnett, R. (eds), MIT Press, Massachusetts.

Tseng, P. (2001). Convergence of a block coordinate descent method for Nondifferentiable minimization. *Journal of Optimization Theory and Applications*, 109(3), 475–494.

Vapnik, V. (1998). *Statistical Learning Theory*. John Wiley & Sons, New York.

Wolfe, P. (1959). The simplex method for quadratic programming. *Econometrica*, 27(3), 382–398.

Wright, S.J. (2015). Coordinate descent algorithms. *Mathematical Programming*, 151(1), 3–34.

Yu, J.R., Lee, W.Y., and Chiou, W.J.P. (2014). Diversified portfolios with different entropy measures. *Applied Mathematics and Computation*, 241, 47–63.