Quantum Machine Intelligence (2025) 7:43
https://doi.org/10.1007/s42484-025-00268-2

RESEARCH ARTICLE

A real-world test of portfolio optimization with quantum annealing

Wolfgang Sakuler[1] · Johannes M. Oberreuter[2] · Riccardo Aiolfi[3] · Luca Asproni[3] · Branislav Roman[1] · Jürgen Schiefer[1]

Received: 19 April 2024 / Accepted: 1 March 2025

Abstract
In this note, we describe an experiment on portfolio optimization using the Quadratic Unconstrained Binary Optimization (QUBO) formulation. The dataset we use is taken from a real-world problem for which a classical solution is currently deployed and used in production. In this work, carried out in a collaboration between the Raiffeisen Bank International (RBI) and Reply, we derive a QUBO formulation, which we solve using various methods: two D-Wave hybrid solvers, that combine the employment of a quantum annealer together with classical methods, and a purely classical algorithm. Particular focus is given to the implementation of the constraint that requires the resulting portfolio's variance to be below a specified threshold, whose representation in an Ising model is not straightforward. We find satisfactory results, consistent with the global optimum obtained by the exact classical strategy. However, since the tuning of QUBO parameters is crucial for the optimization, we investigate a hybrid method that allows for automatic tuning.

Keywords QUBO · Quantum annealing · Hybrid quantum algorithms · Portfolio optimization · Financial services

1 Introduction

Portfolio optimization (PO) is a standard problem in the financial industry (Markowitz 1952). A monetary budget needs to be completely invested on a given set of financial assets with known historical returns and volatilities. The whole investment amount must be equal to 100% of the initial budget and needs to be split, possibly in different percentages, among the assets. The aim is to maximize the expected return of the resulting portfolio while keeping the risk profile, which is measured by the volatility computed from the covariance matrix of the assets, below a specified limit. Additional constraints may be added to help the diversification of the portfolio over multiple sectors or asset classes. A similar problem has been solved by Grant et al. (2021).

The computational complexity of combinatorial optimization problems tends to increase exponentially with the number of variables—here, the number of assets—which at large scale can make solvers incapable of providing only optimal solutions. Instead, the results are likely suboptimal. Currently, it is being investigated in various circumstances whether quantum computers can help cope with this complexity. In particular, a strategy called *quantum annealing* has proven to be a particularly useful approach to optimization problems (Farhi et al. 2000; Morita and Nishimori 2008).

The data used in our work consists of a portfolio structured into three main asset classes: equity (EQ), fixed-income (FI), and money market (MM). A client portfolio typically ranges from 9 to 11 assets. We have chosen this type of dataset because of the following:

1. It represents a setup that is actually used in a real-world bank's production environment.
2. It makes it possible to run the optimization in a short amount of time on quantum computers.

Various constraints have to be imposed on the composition of the portfolio, which we describe in greater detail in Section 2. When imposing constraints in a Quadratic Unconstrained Binary Optimization (QUBO) formulation (Grant et al. 2021; Lucas 2014; Glover and Kochenberger 2018), which is by definition unconstrained, each of these terms must be properly weighted in the objective function such

✉ Wolfgang Sakuler
wolfgang.sakuler@rbinternational.com

[1] Raiffeisen Bank International AG, Am Stadtpark 9, 1030 Vienna, Austria

[2] Machine Learning Reply GmbH, Reply SE, Luise-Ullrich-Str. 14, 80636 Munich, Germany

[3] Data Reply S.r.l., Corso Francia 110, 10143 Turin, Italy

Published online: 21 March 2025

that the resulting solution not only satisfies the constraints, but also maximizes the returns.

Our contribution and the novelty of our work revolve around 2 main points. On the one hand, we conducted a comparison of various approaches to solving the portfolio optimization problem modeled as a QUBO, namely D-Wave's QBSolv, hybrid binary quadratic model (BQM), and hybrid constrained quadratic model (CQM). This comparative analysis allows both to elucidate the strengths and weaknesses of existing methods, as well as improve the understanding of how these techniques perform. A substantial portion of our research is dedicated to exploring the use of CQM as a new and alternative method for addressing QUBO problems. We acknowledge that identifying suitable penalty coefficients for the different Hamiltonian terms can be particularly challenging in solving QUBO problems. This is why we investigate an approach, which is expected to make this procedure easier. By comparing this technique with traditional methods, we have demonstrated that our approach can simplify the solution of QUBO problems and identify the effective strategies for optimization.

On the other hand, our research represents a synthesis of multiple relevant methodologies and considerations that directly impact the practical application of portfolio optimization. The data, constraints, and problem objectives come from industry-specific scenarios. Focusing on the concrete applicability of these techniques, we elaborate on the use of inequality constraints, additional external constraints, and other considerations that are required in industrial use cases, providing a comprehensive framework for portfolio optimization. To this end, we had to develop a number of technical improvements which allow us to treat the problem completely as a QUBO. In particular, we developed a representation of the weights as rational numbers with multiple qubits, which leverage specific conditions of the problem at hand to improve the use of resources, and we devise a strategy to handle inequality constraints which would involve higher than quadratic order terms in the objective function. The added value of our work therefore lies in the effective combination of all of these developments.

By articulating these contributions, we believe we can better convey the advancements our research offers to the field and its potential implications for real-world financial applications.

In this study, we focus deliberately on testing the capabilities of quantum computing to serve as a solution method for standard portfolio optimization and contrast it with classical solutions. Deeper questions on the robustness and stability of the approach to, e.g., different market conditions and its general validity from the point of view of financial mathematics are undoubtedly valuable but beyond the scope of our work. We see no a priori reason why a different solution method like quantum computing would change the validity of this method in general.

We structure this paper as follows: in Section 2, we explain the structure of the problem in detail, and we describe the mathematical formulation used by classical algorithms and we introduce the QUBO approach. In Section 3, we dive deeper into how we cast our problem as a QUBO and put our work in the context of current research. The results of the various approaches that we use to solve the QUBO are presented in Section 4, where the different solutions are compared and benchmarked against the exact global optimum.

2 Problem formulation

We consider the Markowitz portfolio optimization as a quadratic programming problem (Markowitz 1952) that determines the fraction ω_i of available budget B to be allocated on the purchase of the i^{th} asset out of potentially N assets with the goal of maximizing returns, while keeping the risk below a target volatility σ^2_{target}. For simplicity, we set $B = 1$, and we consider weights ω_i as normalized weights.

The optimization problem is formulated as

$$\max_{\omega} \{r^T \cdot \omega\} \tag{1}$$

subject to

$$\omega^T \, \Sigma \, \omega \leq \sigma^2_{\text{target}} \qquad \text{(Volatility constraint)} \tag{2}$$

$$1^T \cdot \omega = 1 \,, \;\; \omega_i \geq 0 \,, \qquad \forall i = 1, \ldots, N \qquad \text{(Weights constraint)} \tag{3}$$

$$A \cdot \omega \, \langle \text{op} \rangle \, b \,, \qquad \langle \text{op} \rangle \in \{=, \leq, \geq\} \qquad \text{(Linear constraints)} \tag{4}$$

where

- r is the vector of (mean historical) asset returns.
- ω is the vector of asset weights.
- Σ is the covariance matrix of the returns.
- σ^2_{target} is the target volatility, i.e., the maximum allowed risk.
- A is a matrix of coefficients specifying further linear constraints.
- b is a vector of constants.

2.1 Classical formulation

The PO problem investigated in this work is based on a calculation that is performed in production at Raiffeisen Bank International AG (RBI) as a service for RBI clients. The main objective of the PO is to maximize the expected return while fulfilling several constraints. The risk constraint limiting the

portfolio volatility is written, using the notation from Eq. 2, as follows:

$$\omega^T \Sigma \omega = \sum_{i=1}^{N} \sum_{j=1}^{N} \sigma_{ij} \omega_i \omega_j \leq \sigma^2_{\text{target}} . \tag{5}$$

The corresponding risk term is hence a quadratic form that is bounded from above.

We impose several additional linear constraints, either defined globally or for a specific client:

1. Normalization constraint: sum of weights ω_i is normalized to 1, i.e., all budget needs to be invested:

$$\sum_{i=1}^{N} \omega_i = 1 . \tag{6}$$

2. Single asset constraints: defining lower and upper bound of the weight of a single asset:

$$\omega_i \geq \omega_{i,\min}, \tag{7}$$
$$\omega_i \leq \omega_{i,\max}. \tag{8}$$

3. Multi asset constraints: conditions involving a set of assets, e.g., constraints for a specific asset-class group (EQ, FI, or IR):

$$\sum_{i=1}^{N} a_{j,i} \omega_i \ \langle \text{op} \rangle \ b_j , \quad j \in \{1, ..., M\}, \quad \langle \text{op} \rangle \in \{=, \leq, \geq\} , \tag{9}$$

where $a_{j,i}$ are the elements of matrix A, b_j are the constants of the constraints j, and M is the number of multi asset constraints.

In the classical calculation, the strategic asset allocation is accomplished via Markowitz optimization (Markowitz 1952), and the tactical asset allocation is based on the so-called Black-Litterman model (1991a; 1991b). The Black-Litterman approach takes (i) market expectations derived from market data and (ii) the objective and independent forecasts provided by the bank's internal research group (Raiffeisen Research) as input parameters, and then produces the posterior asset returns and covariance through an optimization process. The outputs from the Black-Litterman process are then used as input for the strategic Markowitz optimization.

For the classical calculations the statistical software programming environment R is employed. The optimization algorithm is run via IBM's CPLEX optimization software package, which provides solvers for linear and quadratic programming problems (IBM 2022a). These CPLEX solvers can be called from the R environment via the R interface module "Rcplex" (IBM 2022b).

Since the mathematical optimization model represents a convex problem, and furthermore, the volume of the data sets currently faced in RBI's production environment is rather small, the classical optimization procedure is able to quickly find the exact solution, which is the global optimum. Therefore, it clearly cannot be the goal of the study to achieve a more accurate result. This solution rather serves as the ultimate target goal to be ideally achieved by the optimization procedure executed on a quantum computer. The study serves as a starting point to apply the working quantum algorithms to improve results, where classical solutions are not satisfactory.

2.2 QUBO formulation

The Quadratic Unconstrained Binary Optimization (QUBO) model represents a wide range of combinatorial optimization problems (Lucas 2014; Grant et al. 2021; Glover and Kochenberger 2018). It is currently the most applied model in the quantum computing area for these kinds of problems. The QUBO model is expressed by the following optimization problem:

$$\text{minimize } f_Q(x) , \tag{10}$$

where $f_Q : \{0, 1\}^n \to \mathbb{R}$,

$$f_Q(x) = \sum_{i=1}^{n} \sum_{j=i}^{n} q_{ij} x_i x_j , \tag{11}$$

is a quadratic polynomial over binary variables $x_i \in \{0, 1\}$ and coefficients $q_{ij} \in \mathbb{R}$ for $1 \leq i \leq j \leq n$. The QUBO problem consists of finding a binary vector x^* that is minimal with respect to f among all other binary vectors, namely

$$x^* = \arg\min_{x \in \{0,1\}^n} f_Q(x) . \tag{12}$$

In order to maximize $f_Q(x)$, one simply minimizes $f_{-Q}(x) = -f_Q$.

Another, more compact way to formulate $f_Q(x)$ is using matrix notation,

$$f_Q(x) = x^T Q x , \tag{13}$$

where $Q \in \mathbb{R}^{n \times n}$ is a square matrix containing the coefficients q_{ij}. It is common to assume an upper triangular form for Q since it is a symmetric matrix; thus, the transformation can always be achieved without loss of generality with simple tricks. Many problems can be effectively reformulated as a QUBO model by introducing quadratic penalties into the objective function as an alternative to explicitly imposing

constraints in the classical sense (Glover and Kochenberger 2018). The penalties introduced are chosen so that the influence of the constraints on the solution process can alternatively be achieved by the natural functioning of the optimizer as it looks for solutions that avoid incurring the penalties. For a minimization problem, these penalties are used to create an augmented objective function to be minimized.

3 Methodology

In this work, we tackle the PO problem by modeling it as a QUBO (Grant et al. 2021; Mugel et al. 2020, 2022). This formulation enables the use of special-purpose quantum computers, quantum annealers, to find the minimum of a given objective function.

In recent years, along with the developments in the quantum computing field, increasing attention has been drawn to the formulation of well-known combinatorial optimization problems as a QUBO model or, equivalently, the Ising model (Lucas 2014; Glover and Kochenberger 2018). The equivalence consists in the solution of one of the two models also being the solution of the second one, up to a linear change of variables: this allows to adhere to common formulations in operations research that exploit binary variables taking values in $\{0, 1\}$, while being able to exploit quantum annealers to find the minimum of the optimization problem at hand. Therefore, a current focal point in the quantum optimization literature is to examine the capabilities of quantum annealers in application to combinatorial optimization problems (Kadowaki and Nishimori 1998; Farhi et al. 2001; Kochenberger et al. 2014; Katzgraber et al. 2015; Heim et al. 2015, 2017; Asproni et al. 2020).

The PO problem is a key activity in the financial services industry (Mugel et al. 2022). The classical Markowitz model is a convex quadratic programming problem, which in its simplest form, the Mean-Variance model, has a polynomial worst-case complexity bound (Nesterov and Nemirovski 1994; Kerenidis et al. 2019), where the algorithm's running time t behaves as follows:

$$t \sim O(N^k), \quad 2 \leq k \leq 4\,. \tag{14}$$

Numerical calculations using state-of-the-art classical optimization algorithms indicate that the classical Markowitz model shows at best a quadratic time complexity with respect to the number of assets (Brown 2011; Pedersen 2021). However, the complexity of enhanced PO problems, e.g., the so-called Limited Asset Markowitz (LAM) model (also called cardinality constrained Markowitz model), depends on the specific constraints that are additionally imposed on the basic objective (Maringer 2008; Cesarone et al. 2009, 2011). Additional constraints increase the level of complexity, which can result at worst in an NP-hard problem whose complexity scales exponentially as the number of assets grows (Bienstock 1995; Jin et al. 2016):

$$t \sim O(\mathrm{e}^N)\,. \tag{15}$$

This, combined with the nonlinear nature of the problem that particularly fits the QUBO formulation, has led to the use of a quantum computing approach to tackle the problem (Rosenberg et al. 2016). In this work, we build a QUBO model similar to Grant et al. (2021). We follow the approach by including a risk measure constraint on the assets' covariances, given by Eq. 2, and include the left-hand side term of the inequality in the QUBO formulation, fine-tuning the model parameters such that the overall risk does not exceed σ^2_{target} (cf. Sect. 3.2.4 for details). Finally, we discretize the continuous variables ω into a set of binary variables, each of which is weighted in the QUBO by a coefficient. Differently from the approach proposed in Grant et al. (2021), for each asset i and the corresponding variable ω_i, our discretization uses a fixed number of binary variables, representing a given interval $[\omega_{i,\min}, \omega_{i,\max}]$ which may differ from asset to asset. This entails the possibility to use a reduced number of variables to represent assets' weights in the portfolio, while on the other hand potentially providing a different granularity for different assets. Further mathematical details are explained in Section 3.1.

3.1 Discretization of variables

In order to cast the problem into the QUBO formulation, one needs to choose a binary encoding of the weights. As the weights are fractions, the exponents in the binary expansion of the weights are going to be negative. Using discrete rather than continuous variables inevitably limits the accuracy of the solution. While the accuracy increases if more binary variables are being used for each weight, so do the resource needs. Thus, one has to carefully find an optimal number of binary variables that represents a trade-off between target accuracy and acceptable resource usage.

If we allow K variables for the discretization of each weight ω_i, $i = 1, \ldots, N$, the upper bound of the number of QUBO variables for the discretization without considering any additional (slack) variable is $N \cdot K$.

However, the number of variables needed can be reduced after carefully analyzing the linear constraints that restrict the weights for single assets, the so-called single-min and single-max constraints (see Eqs. 4, 7, and 8 of Section 2, respectively). For example, if the weight of an asset is limited within a specific range, fewer binary variables are needed to achieve the same granularity covering only that range

$$\omega_i = \omega_{i,\min} + (\omega_{i,\max} - \omega_{i,\min}) \cdot \omega_i'\,, \tag{16}$$

where the normalized weight ω_i' is restricted to

$$0 \leq \omega_i' \leq 1\,. \tag{17}$$

With the definition

$$\Delta\omega_i = \omega_{i,\max} - \omega_{i,\min}\,, \tag{18}$$

one gets

$$\omega_i = \omega_{i,\min} + \Delta\omega_i \cdot \omega_i'\,. \tag{19}$$

In a binary expansion using K bits, i.e., having a granularity $p_K = 1/2^K$, ω_i' is given as

$$\omega_i' = \sum_{k=1}^{K} 2^{k-1} x_{i,k}\, p_K\,, \tag{20}$$

where $x_{i,k} \in \{0, 1\}, i = 1, \ldots, N, k = 1, \ldots, K$, are binary variables.

Naturally, the granularity would be chosen to be $p_K = 1/2^K$. Then, the normalized weight ω_i' in Eq. 16 is effectively restricted to

$$0 \leq \omega_i' \leq (1 - p_K)\,, \tag{21}$$

and the effective granularity $p_{K,\text{eff}\,i}$ is given by

$$p_{K,\text{eff}\,i} = \Delta\omega_i \cdot p_K\,. \tag{22}$$

While this choice for the granularity technically does not allow to reach exactly $\omega_{\max}$ for each asset, we can reach a number close to it by summing up all the terms, incidentally ensuring by design to have $\omega_i < \omega_{i,\max}$ such that the max constraint is automatically fulfilled. An alternative choice of $\tilde{p}_K = 1/2^{K-1}$ for the granularity would not have these advantages but would allow to reach the maximum amount exactly. This would also come at the cost of using one binary variable more for each asset.

For example, if $K = 10$, the granularity of the normalized weight ω_i' is $p_K = 1/2^{10}$, and the effective granularity for the weight ω_i of asset i is $1/2^{10} \cdot \Delta\omega_i = 9.765625 \cdot 10^{-4} \cdot \Delta\omega_i$ of the budget. The maximum fraction of the normalized weight is $\sum_{i=1}^{K} 2^{i-1-K} = 1 - 1/2^K \approx 0.999023$, thus giving for asset i an effective maximum weight

$$\omega_{i,\text{max\_eff}} = \omega_{i,\max} - 9.765625 \cdot 10^{-4} \cdot \Delta\omega_i\,. \tag{23}$$

However, if we use $p_{K'} = 1/2^{20}$, the effective granularity for asset i is approx. $9.5367 \cdot 10^{-7} \cdot \Delta\omega_i$ of the budget, while the maximum fraction is $1 - 1/2^{K'} \approx 0.999999046$.

The choice of K relies on the effective granularity needed, the level of approximation manageable, and the number of variables implementable.

Finally, we use the same number of variables for all assets, although the discretized ranges vary among different assets, since the effective granularity $p_{K,\text{eff}\,i}$ for asset i is given by $p_K \cdot \Delta\omega_i$. That means that the effective granularity for each asset is in fact finer than p_K, because in our real-world setup $\Delta\omega_i$ is always smaller than 1 for all assets (in the current setup, one has $\omega_{i,\max} < 1$, and $\Delta\omega_i = 0.1$ for all i; thus, the effective granularity is the same for all assets, i.e., $p_{K,\text{eff}\,i} = p_{K,\text{eff}}$). A possible improvement when having the same granularity for each asset would be to reduce the number of binary variables for each asset.

The error in representing the individual weights due to the finite granularity also leads to an error in the total budget invested. This is because every weight is in principle a random rational number between 0 and 1. When approximating the ω_i' in any weight in a binary expansion, this number will be represented with an error depending on p_K. Treating these errors as a distribution, we are calculating the expected value and variance of the error in the Appendix C. The resulting expectation value of the error ϵ is

$$\mathrm{E}[\epsilon] = \frac{p_K^2}{2}\,, \tag{24}$$

and standard deviation is

$$\mathrm{Var}[\epsilon] = \frac{p_K^2}{12} + \frac{p_K^3}{4} - \frac{p_K^4}{4}\,. \tag{25}$$

However, given our construction, we are sampling the interval between the minimal and maximally allowed values, only as explained in Eq. 19. Therefore, the error accumulates to

$$\delta\omega = \delta \sum_i \omega_i = \sum_i \Delta\omega_i\, \delta\omega_i' = \delta\omega' \sum_i \Delta\omega_i\,, \tag{26}$$

where in the last step we have used the fact that all the ω_i are constructed in the same way according to Eq. 20. This makes the error dependent on the individual min-max constraint, more precisely on the difference between maximum and minimum.

Keeping this in mind is important for the interpretation of the results of our experiments in Section 4.

3.2 Structure of objective function

Considering the terms including the constraints mentioned above, the objective function consists in our case of the following four terms:

- Returns to be maximized, ref. Equation 1

- Weights constrained as all budget needs to be invested, ref. Equation 3
- Linear constraints, ref. Equation 4
- Target volatility constraint, ref. Equation 2

This formulation allows to consider a single QUBO expression made up of 4 terms:

$$f_Q = \lambda_1 H_1 + \lambda_2 H_2 + \lambda_3 H_3 + \lambda_4 H_4, \tag{27}$$

where $\lambda_l > 0$ is the penalty coefficient incorporating the relative importance of the l^{th} term and the sign linked to the maximization or minimization; the H_l is the Hamiltonian derived from the QUBO matrix of the l^{th} term. We analyze each term in detail below.

3.2.1 Returns H_1

The optimization of returns consists of minimizing

$$H_1 = -r^T \omega, \tag{28}$$

where $r = (r_1, \ldots, r_N)$ is the vector of returns of assets $i = 1, \ldots, N$ and $\omega = (\omega_1, \ldots, \omega_N)$ is the vector of asset weights. The "return" term H_1 is the basic objective term of the optimization problem. In order to formulate the problem in the QUBO framework, one needs to discretize the weights as described in Section 3.1. Without losing generality the discretization expressed in Eq. 16 is used. With this discretization, the QUBO formulation of the basic objective term H_1 is written as

$$H_1 = -\sum_{i=1}^{N} \left(\omega_{i,\min} + (\omega_{i,\max} - \omega_{i,\min}) \cdot \sum_{k=1}^{K} p_K 2^{k-1} x_{i,k} \right) \cdot r_i \,. \tag{29}$$

3.2.2 Weight constraint H_2

This term is a hard constraint on the sum of investments of the initial budget, and it is expressed as follows:

$$1^T \cdot \omega = \sum_{i=1}^{N} \omega_i = 1. \tag{30}$$

With the discretization explained in Section 3.1, the QUBO formulation becomes

$$H_2 = \left[\sum_{i=1}^{N} \left(\omega_{i,\min} + (\omega_{i,\max} - \omega_{i,\min}) \cdot \sum_{k=1}^{K} p_K 2^{k-1} x_{i,k} \right) - 1 \right]^2 . \tag{31}$$

3.2.3 Linear constraints H_3

This term represents a set of linear constraints defined by the matrix A and the vector b in Eq. 4. These constraints are slightly different from H_2 because they include inequalities. One can always put them into a QUBO formulation by including auxiliary variables, so-called slack variables, which are also represented as a binary expansion using slack binary variables. Supposing that matrix A is of the type $M \times N$ where M is the total number of linear constraints and N the number of assets, the QUBO formulation becomes

$$\sum_{j=1}^{M} \lambda_{3j} \left(\sum_{i=1}^{N} a_{j,i}\, \omega_i + \alpha_j\, s_j - b_j \right)^2 , \tag{32}$$

where $a_{j,i}$ are the elements of matrix A, b_j are the constants of the various linear constraints j, s_j are slack terms, that are introduced to transform inequality constraints effectively into equality conditions, and α_j are the signs related to the slack terms, that depend on the relational operators of the constraints:

$$\alpha_j = \begin{cases} 1 & \text{if } j^{\text{th}} \text{ constraint is } \leq \\ 0 & \text{if } j^{\text{th}} \text{ constraint is } = \\ -1 & \text{if } j^{\text{th}} \text{ constraint is } \geq \end{cases}, \tag{33}$$

Using a binary formulation the slack term s_j is given as

$$s_j = \beta_j \sum_{k=1}^{S_j} p_{S_j} 2^{k-1} s_{j,k} \,, \tag{34}$$

where $s_{j,k} \in \{0, 1\}$, $j = 1, \ldots, M$, $k = 1, \ldots, S_j$, are the actual binary slack variables, and β_j is the maximum value of the continuous version of the slack term, given by

$$\beta_j = \begin{cases} \underset{x_i \in \{0,1\}}{\arg\max} \, [\, (b_j - \sum_i a_{j,i}\, x_i), 0 \,] & \text{if } j^{\text{th}} \text{ constraint is } \leq \\ \underset{x_i \in \{0,1\}}{\arg\max} \, [\, (\sum_i a_{j,i}\, x_i - b_j), 0 \,] & \text{if } j^{\text{th}} \text{ constraint is } \geq \end{cases} . \tag{35}$$

The number of slack variables S_j for constraint j depends on the effective slack term granularity

$$p_{S_j,\text{eff}} = \beta_j \cdot p_{S_j} \,, \quad p_{S_j} = 1/2^{S_j} \,. \tag{36}$$

For practical reasons, the number of slack variables S_j has been fixed for each linear constraint j to the number K of physical binary variables per asset. Thus, while the number of slack variables is always the same for the various constraints, the effective slack term granularity $p_{S_j,\text{eff}}$ varies.

Another approach which has been performed in the current activity is related to those linear constraints that act individually on each asset: these give lower and upper bounds on the values of such assets. In this way, the discretization described in Eq. 16 can be applied to the new range defined by the constraints. This allows to satisfy those constraints by construction and there is no need to include them in the QUBO formulation, leading to an ease of calibration and findings of feasible solutions. For example, if two constraints impose that the investment of an asset i must be within the range $[\omega_{i,\min}, \omega_{i,\max}]$, then the discretization will find a fraction of the value $(\omega_{i,\max} - \omega_{i,\min})$.

With the discretization explained in Section 3.1, Eq. 32 is transformed into

$$\sum_{j=1}^{M} \lambda_{3j} \left[\sum_{i=1}^{N} \left(a_{j,i}\, \omega_{i,\min} + a_{j,i}\, (\omega_{i,\max} - \omega_{i,\min}) \cdot \sum_{k=1}^{K} p_K 2^{k-1} x_{i,k} \right) + \right. \\ \left. + \alpha_j\, \beta_j \sum_{k=1}^{S_j} p_{S_j} 2^{k-1} s_{j,k} - b_j \right]^2 . \quad (37)$$

3.2.4 Target volatility constraint H_4

The QUBO formulation of this term strictly depends on the approach implemented to consider the target volatility constraint as in Section 2.

As a first example, we consider the constraint rewritten in Eq. 2 in which the target volatility is set to zero. In this way, the portfolio risk is handled via the minimization of the term

$$\omega^T\, \Sigma\, \omega = \sum_{i=1}^{N} \sum_{j=1}^{N} \sigma_{ij}\, \omega_i\, \omega_j\,, \quad (38)$$

which in the QUBO formulation with the discretization explained in Section 3.1 becomes

$$\sum_{i=1}^{N} \sum_{j=1}^{N} \tilde{\sigma}_{ij} \cdot \left(\omega_{i,\min} + (\omega_{i,\max} - \omega_{i,\min}) \cdot \sum_{k=1}^{K} p_K 2^{k-1} x_{i,k} \right) \cdot \\ \cdot \left(\omega_{j,\min} + (\omega_{j,\max} - \omega_{j,\min}) \cdot \sum_{k=1}^{K} p_K 2^{k-1} x_{j,k} \right), \quad (39)$$

where $\tilde{\sigma}_{ij}$ in an adjusted coefficient such that

$$\tilde{\sigma}_{ij} = \begin{cases} \sigma_{ij} & \text{if } \; i = j \\ 2\sigma_{ij} & \text{if } \; i < j \\ 0 & \text{otherwise} \end{cases} \quad (40)$$

When the maximal risk is not allowed to exceed a threshold value given by the target volatility σ^2_{target}, the following less-than-or-equal constraint has to be fulfilled

$$\omega^T\, \Sigma\, \omega \leq \sigma^2_{\text{target}}\,. \quad (41)$$

Putting this constraint into a non-constraint QUBO-like form, one gets the following:

$$H_4 = \left(\omega^T\, \Sigma\, \omega + s_{\text{vola}} - \sigma^2_{\text{target}} \right)^2 , \quad (42)$$

where, since a less-than-or-equal constraint (and not an exact equality condition) has to be handled, a slack variable term s_{vola} has to be introduced which, using binary slack variables $s_{\sigma,k} \in \{0, 1\}$, $k = 1, \ldots, S_\sigma$, is given by

$$s_{\text{vola}} = \sigma^2_{\text{target}} \sum_{k=1}^{S_\sigma} p_{S_\sigma}\, 2^{k-1}\, s_{\sigma,k}\,, \quad (43)$$

in which S_σ is the number of binary slack variables for the volatility constraint that depends on the effective volatility slack term granularity

$$p_{S_\sigma,\text{eff}} = \sigma^2_{\text{target}} \cdot p_{S_\sigma}\,, \quad p_{S_\sigma} = 1/2^{S_\sigma}\,. \quad (44)$$

After squaring Eq. 42, one obtains

$$H_4 = (\omega^T\, \Sigma\, \omega)^2 + 2\, (s_{\text{vola}} - \sigma^2_{\text{target}})\, \omega^T\, \Sigma\, \omega \\ -2\, s_{\text{vola}}\, \sigma^2_{\text{target}} + s^2_{\text{vola}} + \sigma^4_{\text{target}}\,. \quad (45)$$

Since the volatility risk term is itself a quadratic form, the first term contains quartic and cubic contributions in ω. Strictly speaking, the problem is no longer a QUBO (quadratic) problem, but it turned into a so-called PUBO (Polynomial Unconstrained Binary Optimization) problem (sometimes also called HUBO for Higher-Order Unconstrained Binary Optimization) (Glover et al. 2011a, b; Palmer et al. 2021). The existence of up to fourth-order polynomial terms represents a fundamental complication in the procedure since these terms cannot be mapped onto the Ising model of the quantum computer, whose interactions are by definition restricted to linear 1-body and quadratic 2-body terms. However, several workarounds for the PUBO problem are proposed in the literature:

1. Applying order-reduction techniques (Mugel et al. 2020): Using this method is rather expensive since one needs additional bits.
2. Linearization (Palmer et al. 2021): In this approach, the quadratic volatility term is replaced by a linearized expression.

$$H_4 = \left(k^T\, \Sigma\, \omega - \sigma^2_{\text{target}} \right)^2 , \quad (46)$$

where k is a vector of constants which are called linear weights. Due to the linearization, the whole term remains quadratic. However, finding an appropriate value of k is somehow arbitrary: one option is to find k in a self-consistent way, and another possibility is to fine-tune k starting from a convenient value like $k_i = 1/N \; \forall i$.

3. Replacement by am equality-to-zero condition (Grant et al. 2021): if the target volatility threshold value is sufficiently small, then it can be approximated by zero. When the right-hand-side is exactly zero, the $\leq$ operator can be replaced by the equality operator since the left-hand-side, the quadratic volatility term, is positive-definite. A constraint like $g(\omega) = 0$ with a positive-definite function $g(\omega) \geq 0$ can be handled in the optimization model very easily by just adding a term $\lambda \, g(\omega)$ to the objective. Thus, in this approach, one has

$$H_4 = \omega^T \, \Sigma \, \omega \,, \tag{47}$$

For the calculations employing (i) the classical QBsolv solver and (ii) D-Wave's Hybrid BQM solver, we used the latter workaround, namely the replacement of the PUBO term by an equality-to-zero constraint. By tuning the weight of the volatility constraint carefully, i.e., by choosing an appropriate Lagrange multiplier, we obtain a feasible formulation while being able to optimize the complete objective function including all the other constraints.

The calculations performed with (iii) D-Wave's new Hybrid CQM solver do not need such a replacement, because the CQM solver can handle both linear and quadratic conditions naturally as genuine constraints.

3.3 Computational method

To solve the PO problem, different approaches have been followed up such that the capabilities of quantum and quantum-inspired solutions could be thoroughly assessed and benchmarked. In order to check and quantify the quality of these solutions, the results have been compared with the global minimum of the optimization problem, which, given the limited size of the data at hand, could be easily found via classical strategies.

First and foremost, the QUBO model as described in Section 3 has been built and D-Wave's QBSolv library has been exploited to solve the optimization problem through classical optimization techniques. In order to do so, the binary quadratic model data structure has been used that stores each entry of the QUBO model, assigning biases and couplers as penalty coefficients to each variable and pair of variables, respectively.

Second, we have investigated the use of D-Wave's hybrid binary quadratic model (BQM), which decomposes the overall QUBO problem into subproblems suitable to be solved on a quantum processing unit (QPU). Those subproblems can be solved directly on the QPU, thus having the benefit of quantum effects such as quantum tunneling to best find high-quality solutions. The decomposition step is needed in order to have QUBO subproblems of sufficiently small size that match current QPU architecture; this procedure is handled automatically by D-Wave's hybrid software.

With the aforementioned QUBO solvers, one crucial step needed to find not only feasible but also optimal solutions is to fine-tune some significant QUBO parameters, namely the Lagrange multipliers λ_l from Eq. 27, that act as relative weights between the various optimization terms and constraints that build up the whole QUBO expression. This step is non-trivial and might lead to suboptimal solutions, especially when the number of optimization terms and constraints, and thus the overall complexity of the problem, increases.

As the next step, and particularly motivated to overcome the problem to fine-tune Lagrange parameters, we have investigated the usage of D-Wave's next-generation hybrid solution, the constrained quadratic model (CQM) solver. The CQM solver is the newest product of D-Waves's hybrid solver family. It enables to formulate constraints in their natural form as "they are," i.e., as real constraints

$$\text{term} \, \langle \text{op} \rangle \, b, \quad \langle \text{op} \rangle \; \in (=, \leq, \geq) \,. \tag{48}$$

where "term" can be any linear or quadratic form in the binary variables. Thus, even the maximal volatility constraint Eq. 41, which is quadratic and a less-than-or-equal condition (i.e., not an equality condition), can be entered into the CQM solver directly without any modification. Before, when using one of the previous solvers, e.g., the Hybrid BQM solver, one had to formulate constraints as penalty terms, which in case of an inequality condition even had to be supplied with an auxiliary variable term $\alpha \, s$. The constraint had to be put into a parabolic form multiplied by a Lagrange multiplier λ (cf. Sect. 3.2):

$$\lambda \, (\text{term} + \alpha \, s - b)^2 \,, \quad \alpha = \{0, 1, -1\} \text{ if } \langle \text{op} \rangle = \{=, \leq, \geq\}. \tag{49}$$

With the availability of the CQM solver from D-Wave, constraints are handled automatically. However, since the implementation details of D-Wave's CQM algorithm have not been publicly revealed by the software vendor, from a software end-user's perspective, it remains hidden under the surface how the constraints are in fact implemented or formulated.

4 Results

We have performed our investigation on multiple solutions ranging from classical strategies adopted by the QBSolv library to hybrid techniques for decomposing the optimization problem into suitable subproblems which are solved both on classical and quantum processing units (QPUs). In order to exploit the full potential of the available software for quantum optimization, and thus to reach the highest-performing solution strategy, two available strategies have been investigated, namely using a BQM and a CQM.

The former is used as a data structure to represent the QUBO modeling and hence underlies the same principles: one needs to fine-tune the model parameters in order to find feasible and optimized solutions, in terms of maximum return and minimum volatility. The latter allows to explicitly declare which terms of the optimization are genuine constraints and which constitute the objective function. The management of different terms is then delegated to the hybrid solver library, i.e., to the software side, and thus allows the software user to reduce the time spent calibrating QUBO parameters.

In the following paragraphs, we show multiple results using common notation and considerations:

- The scatter plots in Fig. 1 show the achieved return vs. volatility for different parameter sets, where one data point represents the best result of one experiment. The

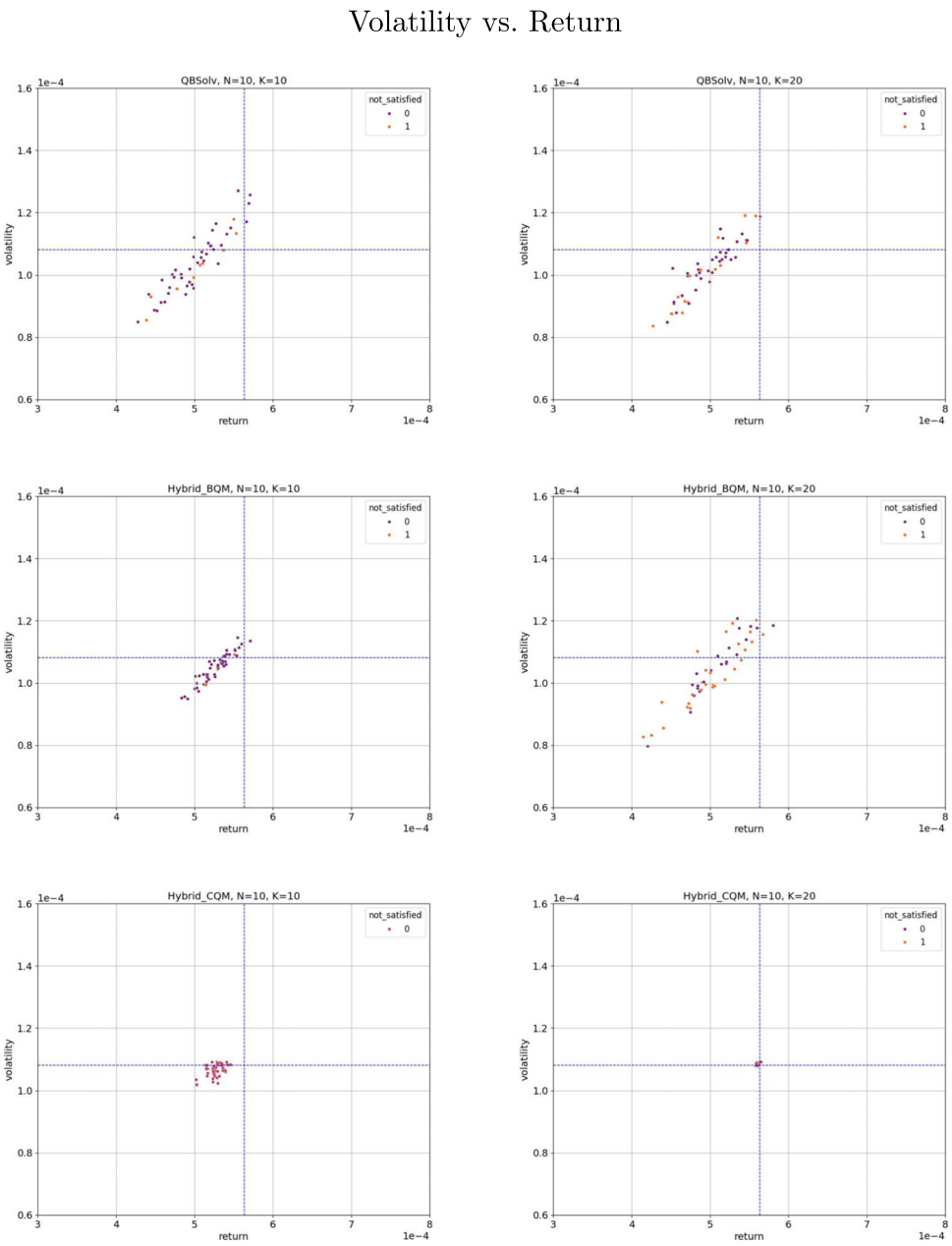


Fig. 1 Distribution of solutions of several runs for different solvers and granularity. The solvers employed are the purely classical QBSolv, and the Hybrid BQM and Hybrid CQM, that both have a quantum backend; N denotes the number of assets considered; K represents the number of binary variables, i.e., the number of qubits, used for each asset. This number also determines the granularity of the calculation. The dashed lines, horizontal for the volatility, and vertical for the expected return of the portfolio represent the solutions obtained by the classical CPLEX solver. The *not_satisfied* label on each solution indicates the number of constraints violated as described in paragraph Section 3.1. The spread of the solutions along an upwards slope makes sense from a business point of view, insofar as higher risk should be associated with higher return. It can be clearly seen that the results obtained with the Hybrid CQM solver are the most precise. A zoomed version of these plots with better visibility is given in Fig. 17

vertical dashed line marks the result achieved by the classical solver, which is expected to be close to the theoretical optimum of the return with the given volatility, marked by a horizontal dashed line. Points higher than the horizontal dashed line represent experiments which have yielded impermissible results (risk too high). Results to the right of the vertical dashed line would represent experiments which yield better-performing portfolios than the ones classically found, which is not expected. We are looking for the best experiment in the lower left quadrant, i.e., the one closest to the intersection of the dashed lines.

- The *not_satisfied* label in the plots refers to the number of constraints violated. The investment constraint is considered satisfied in those cases where the actual sum of investments deviates from the target (100%) no more than a small amount which is given by the effective granularity $p_{K,\text{eff}}$:

$$\left| \sum_{i=1}^{N} \omega_i - 1 \right| \leq p_{K,\text{eff}} \tag{50}$$

  The distribution of the sum of approximated weights is shown in Fig. 5 as measured in our experiments, which describes to what extent the normalization constraint is fulfilled.
- Our notation is such that:
  - N refers to the number of assets.
  - K refers to the number of (regular) qubits.
  - t refers to the maximum time provided to CQM to retrieve the solution.

4.1 Solvers comparison on business KPIs

In this paragraph, we focus on the comparison of the results based on the business KPIs, namely volatility and return of the optimized portfolios. Figure 1 reports volatility vs return plots (zoomed version with better visibility on details in Fig. 17 in the appendix). The classical optimization yields a volatility and return value reported as a horizontal dashed line and a vertical dashed line, respectively. The optimal solution lies at the intersection of the two lines. The dots represent the results (samples) from the QUBO formulation. Given the flexible nature of QUBO problems not setting constraints explicitly and given our approach to satisfy the volatility constraint, it is in principle possible to find results that do not satisfy such constraint. These solutions are represented in the plot with the dots lying above the horizontal dashed line. From a business perspective, these are solutions that must be discarded. We have however included them in the plots to report a detailed and complete overview of the outcome of the QUBO problems with a calibration of the QUBO weights as thorough as possible (note that the calibration has not been implemented when using the CQM solver). Among the samples found via the different solvers, we were able to find results fairly close to the global optimum found via a classical optimization procedure.

To produce the bottom plots of Fig. 1 regarding CQM performances, we have excluded the volatility constraint from the counting of the number of constraints not satisfied, which is shown via the dots' label within the plot. This is due to the fundamentally quartic nature of such constraint and the difficulty in treating this term within a QUBO formulation, which requires advanced procedures and cannot be reformulated as a quadratic term without the use of additional variables.

In Fig. 2, we see a comparison of business KPIs for the three solvers considered in this work, namely QBSolv, Hybrid BQM, and Hybrid CQM. We see that hybrid solvers perform better than QBSolv. While the average objective is higher for Hybrid CQM, so is the average risk. This has to be put into the perspective that behind the Hybrid BQM and QBSolv solution there is the need to fine-tune the QUBO weights, or Lagrange multipliers, which is handled automatically in the Hybrid CQM. Looking at the Sharpe ratio, we see that Hybrid BQM actually outperforms the other solvers. However the objective was to maximize the returns while satisfying the volatility constraint, rewritten as a risk minimization, and the Sharpe ratio has not been introduced as an explicit term of the objective functions. It also needs to be pointed out that the spread of the values is in the range of 10^{th} permille and thus very reduced, almost rendering the approaches on par.

4.2 Investigation of capability to scale number of assets

Scaling up the number of assets is crucial for industrial applications, and thus, we investigate the capabilities of the considered solvers for having 499 assets to choose from.

Figure 3 shows the comparison of results in terms of returns and volatility of QBSolv and CQM approaches, benchmarked against the classical solution, for 499 assets. We were not able to solve this problem with BQM. Not only is QBSolv likely to find infeasible solutions, but they are also of lower quality with respect to CQM results: the CQM's feature to automatically handle constraints proves to be a consistent approach to obtain feasible portfolios. At the same time, the objective (i.e., the expected return) is also close to the classical benchmark.

Figure 4 shows the distribution of expected return (objective), volatility, and the derived Sharpe ratios for multiple runs of the optimization with both solvers. For each, we report the distributions (histograms) for 499 assets and the median as well as the best result (in terms of objective, volatility, and

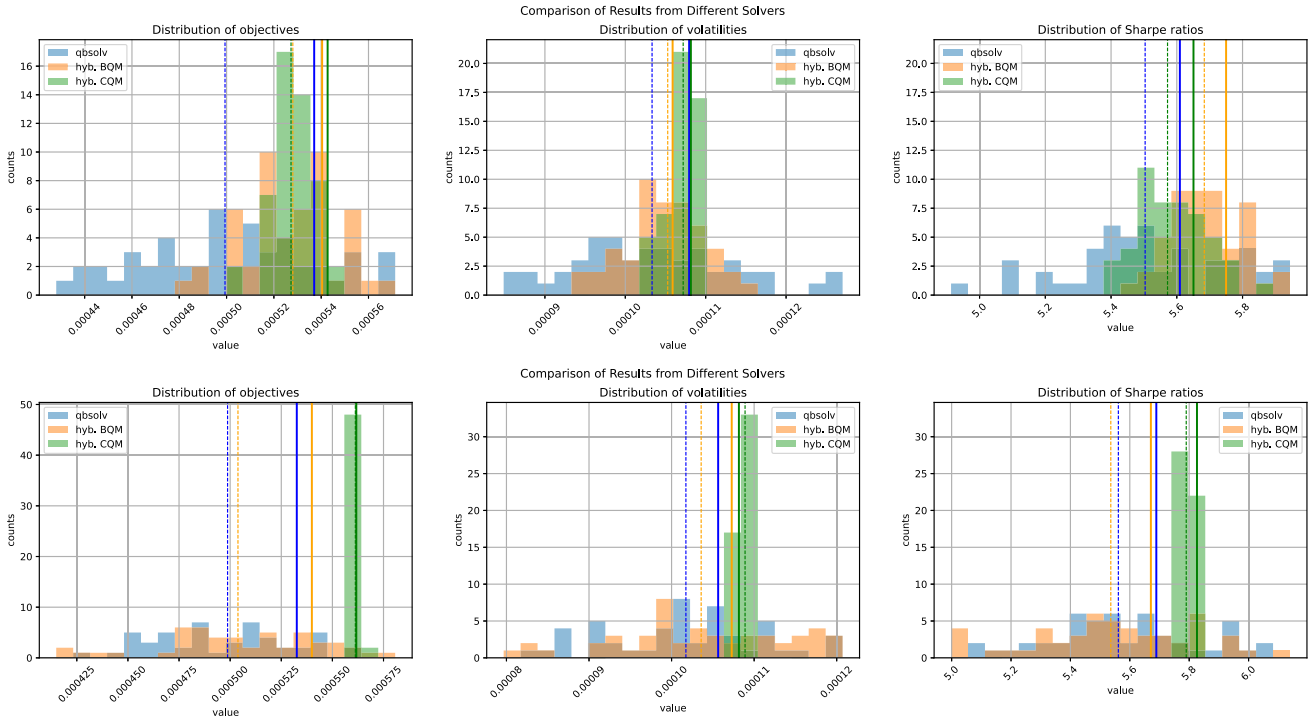


Fig. 2 We compare the distribution of business-relevant characteristics of portfolios obtained with different solvers. The solid lines of each color mark the classical result. The dotted lines mark the experiment yielding the highest objective while the volatility is below the set threshold, i.e. the best permissible portfolio obtained with these parameters. First row $K = 10$, second row $K = 20$

Sharpe ratio, respectively) of a feasible portfolio. The results we find are twofold:

1. The best results that we find are close to the classical solutions.
2. Hybrid CQM finds portfolio configurations with higher objectives which still satisfy all the constraints.

As we scale up the number of assets, so does the number of variables in the QUBO and thus the complexity of the problem. It therefore becomes more difficult for the solver to satisfy the constraints, which also requires more specific tuning of Lagrange multipliers to still perform well. On the other hand, we see that the CQM solver is able to find high-quality solutions and achieves higher returns than QBSolv for the

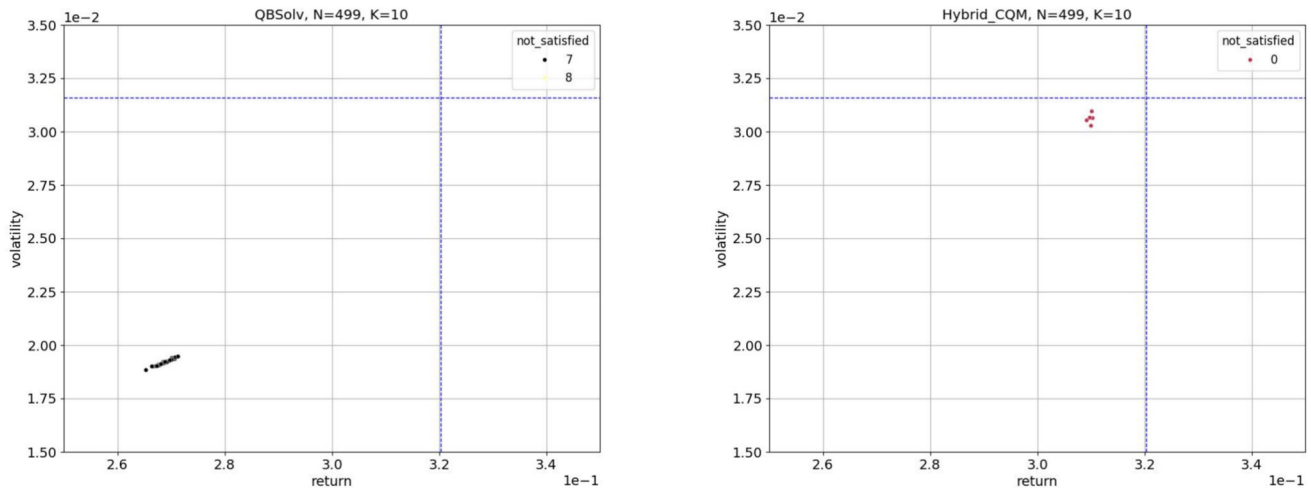


Fig. 3 Comparison of risk vs return for a more complicated problem with 499 assets. This problem could only be solved with QBSolv (left) and hybrid CQM (right)

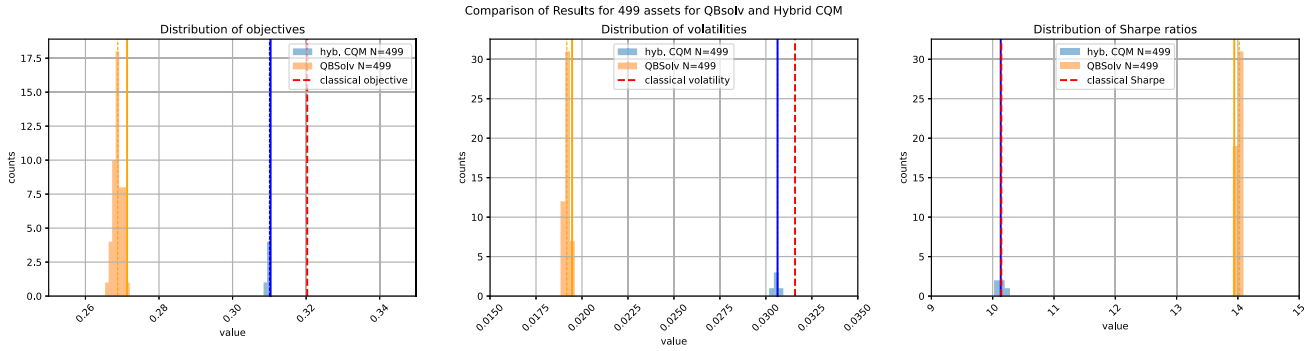


Fig. 4 Comparison of performance of QBSolv and Hybrid CQM with 499 assets. Performances are shown as the portfolios' expected returns, volatilities, and Sharpe Ratios, respectively. The dashed red lines mark the classical result. The dotted lines mark the experiment yielding the highest objective while the volatility is below the set threshold, i.e., the best permissible portfolio obtained with these parameters

configurations we have examined. Although the volatilities of these solutions are also higher, they are below the given threshold.

From a business perspective, the best solution would be the one that maximizes the expected return (while satisfying the volatility constraint). Alternatively, even though not directly optimized, the best portfolio would be the one that maximizes the Sharpe ratio (right panel). It appears as if QBSolv is delivering a better-performing portfolio than Hybrid CQM. However, all of these solutions violate at least one of the constraints (cf. Fig. 13) and therefore are in reality not usable. In fact, the solutions obtained with Hybrid CQM are closer to the classical solutions than those coming from QBSolv in general.

4.3 Solvers capabilities scaling with the number of qubits

Given a fixed number of assets, employing more variables allows to increase the granularity of the weights of the individual assets, which should we expect to help satisfying the constraint that the sum of investments should equal 1. This behavior is confirmed by Fig. 5, where we show the distributions of the sum of asset weights, i.e. the total amount of investment, both for $K = 10$ and $K = 20$ variables used to represent each asset. The peaks in the distribution for $K = 20$ suggest that, for multiple solutions, all the solvers are able to find portfolios in which the total investment is closer to the constraint target value 1. The statistics of the violation of the normalization constraint for the three solvers and for two granularities are reported in Table 1 along with our explicit calculation of the expected error in Appendix C.

Furthermore, while all solvers are able to find solutions having a relatively small deviation from the constraint target, QBSolv outputs solutions where such deviation is in the order of 10^{-4}, BQM in the order of 10^{-5} and CQM, as the best solver, in the order of 10^{-6}.

Comparing with the values measured from the distributions in Fig. 5 and reported in Table 1, we see that only a fraction of the error can be explained by the finite granularity. In Table 1, we also report the skewness of the solution distribution for the various methods for completeness. For the results to be conclusive, presumably larger statistics than available would be needed.

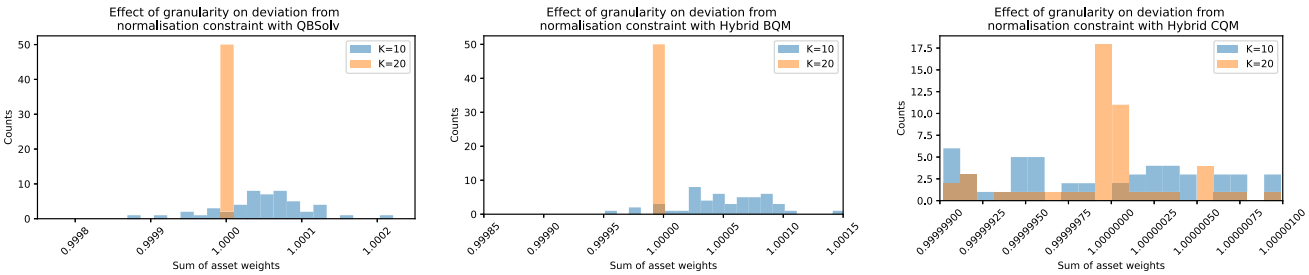


Fig. 5 Comparison of the distribution of portfolio weights using $K = 10$ and $K = 20$ variables for each asset, thus differing in the overall granularity of potential investments

Table 1 Comparison of the violation of the normalization constraint violation (deviation of mean of sum of weights from unity) and of the variance of the error for all the three solvers and for 10 and 20 binary variables, respectively

$1 - E(\sum w)$	QBSolv	Hybrid BQM	Hybrid CQM	Theory
$K = 10$	$-4.89 \cdot 10^{-5}$	$-4.97 \cdot 10^{-5}$	$7.28 \cdot 10^{-8}$	$4.77 \cdot 10^{-7}$
$K = 20$	$-7.20 \cdot 10^{-9}$	$-1.06 \cdot 10^{-7}$	$3.72 \cdot 10^{-8}$	$4.55 \cdot 10^{-13}$
MSE				Variance
$K = 10$	$3.67 \cdot 10^{-9}$	$1.42 \cdot 10^{-9}$	$3.46 \cdot 10^{-13}$	$7.97 \cdot 10^{-8}$
$K = 20$	$4.48 \cdot 10^{-14}$	$6.67 \cdot 10^{-15}$	$1.67 \cdot 10^{-13}$	$7.58 \cdot 10^{-14}$
Skew				
$K = 10$	$-3.29 \cdot 10^{-1}$	$-1.84 \cdot 10^{-1}$	$2.21 \cdot 10^{-2}$	$5.03 \cdot 10^{-3}$
$K = 20$	-5.36	2.01	$-2.72 \cdot 10^{-1}$	$4.96 \cdot 10^{-6}$

The results from an explicit calculation of the error expected due to finite granularity are reported as "Theory"

4.4 Observations

Building on the study of the effect of the number of variables considered for each asset, Fig. 6 reports the results in terms of the KPIs for $K = 10$ and $K = 20$. The first finding consists of the CQM solver providing more peaked distributions over multiple solutions when compared to other solvers, thus suggesting consistent—and high quality as can be seen from the measured values—results. Then, we show that QBSolv and BQM do not report substantial differences in the distributions when compared to one another, while slight disparity is shown when comparing the distributions for $K = 10$ and $K = 20$ for each solver.

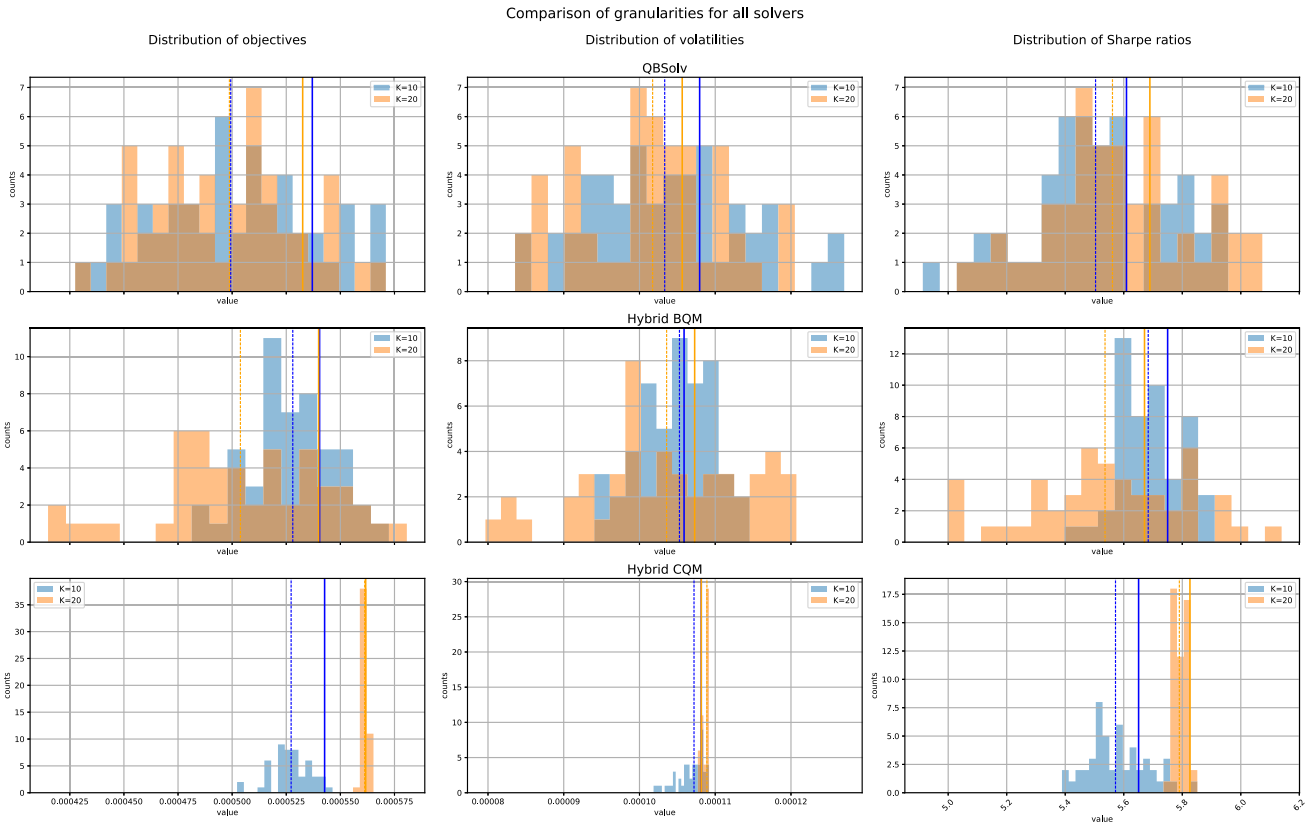


Fig. 6 Comparison of objective (left column), volatility (middle column), and Sharpe ratios (right column) for solutions obtained with QBSolv (top row), Hybrid BQM (middle row), and Hybrid CQM (bottom row). Results for granularities $K = 10$ (blue) and $K = 20$ (orange) are reported. In each plot, the median value of the sample of solutions obtained is marked with a dashed line. The best acceptable portfolio from the sample, i.e., the portfolio with the highest objective obeying the volatility bound, is marked with a solid line

5 Conclusions and outlook

In this work, we have analyzed the capabilities of current quantum and hybrid solvers in solving the portfolio optimization problem. The data used represents a production environment and consists of 10 assets that can be divided into 3 main classes: equity, fixed-income, and money market. This is a particularly interesting problem both in terms of common applicability in financial services as well as due to its nonlinear nature, which makes the QUBO formulation a particularly suitable model for the problem. We have thus detailed both the classical mathematical formulation of the portfolio optimization problem as well as the QUBO one.

We have explored the D-Wave's libraries and tackled the problem using the QBSolv, the Hybrid BQM, and the Hybrid CQM solvers, while benchmarking the solutions with one given by exact classical methods. We have found that the CQM solver and its automating handling of multiple optimization terms and constraints QUBO can lead to higher-quality solutions. Our satisfactory results show that the quantum computing approach is able to find solutions that are close to the exact optimum in terms of return and volatility.

These results pave the way for a broader applicability of the QUBO model using larger data sets, where a dramatic increase in the number of assets can lead classical solvers to yield only suboptimal solutions, while quantum computing is set to aim for high-performing scaling capabilities and may thus outperform classical solutions in much more computationally-complex scenarios. Concretely, the next steps will include increasing the number of assets and the complexity of the problem.

Appendix A: Solution quality with respect to various parameters

In this section, we examine how various model hyperparameters influence the solution quality.

First of all, the discrete approximation of the continuous weights can be expected to influence the solution quality. In Fig. 6, we report our findings on two choices of the granularity K. It appears that QBSolv does not profit a lot from increased granularity as the spread of the solution histograms is similar. However, the Sharpe ratio of the best usable portfolio is better for $K = 20$, given the increased volatility for coarser granularity.

The picture is similar for Hybrid BQM. The objectives are almost the same. Surprisingly, though, the Sharpe ratios are slightly better for lower granularity. Potentially, while using a larger granularity should in principle yield a better solution, it appears that the solver has trouble realizing this improvement as the solution space increases.

For Hybrid CQM, the effect is more pronounced. In general with increasing K the solution gets better. The quality seems to saturate at $K = 20$. It seems an accidental finding that $K = 5$ outperforms $K = 10$. It is clearly visible that for $K = 5$ the spread of the solutions is very large as can be seen in Fig. 7.

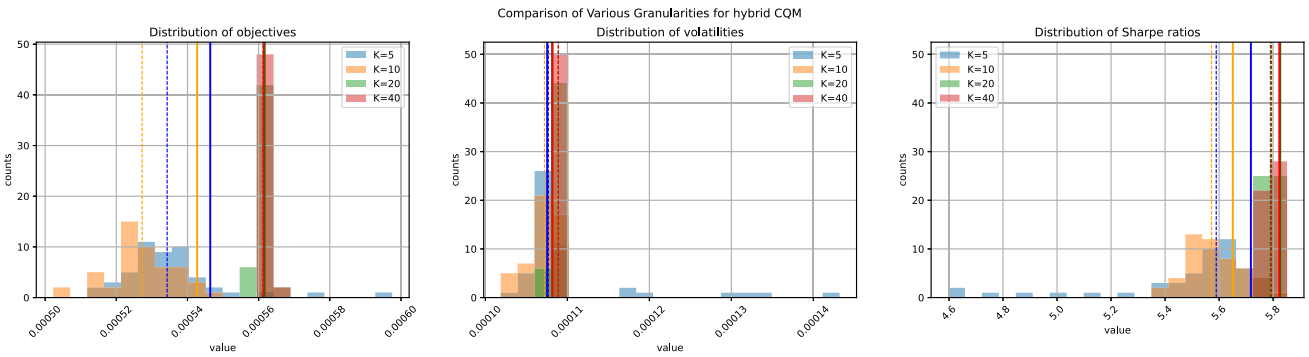


Fig. 7 The effect of varying the granularity of the weights approximation for CQM. In general, higher granularities improve sampling of the solutions

Appendix B: Iterations

It becomes clear from Fig. 8 that sampling the result for 50 iterations is not too far off a better result obtained with 5000 iterations at correspondingly higher costs. Figure 9 also shows this qualitatively. Therefore, it seems that a smaller sampling is already sufficient.

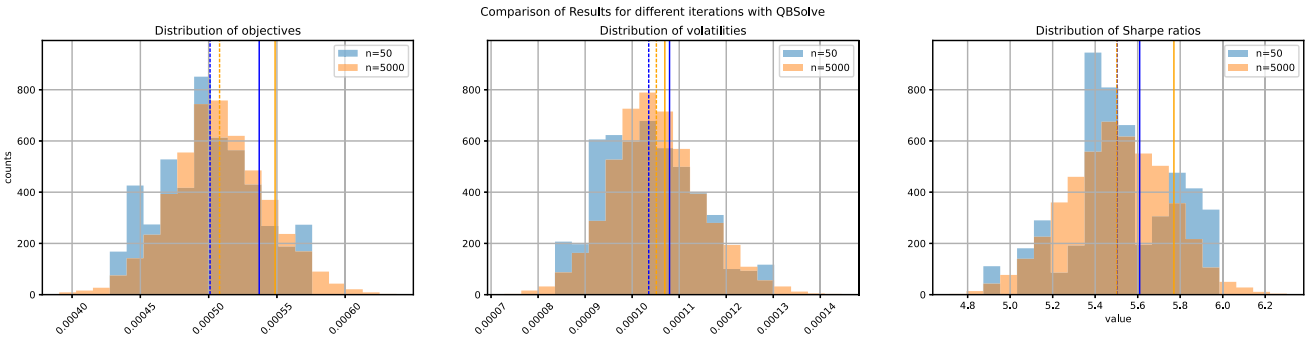


Fig. 8 Comparison of solution quality after increasing the number of iterations for QBSolv. A slightly better solution can be found by sampling more often. Histograms are scaled for comparability

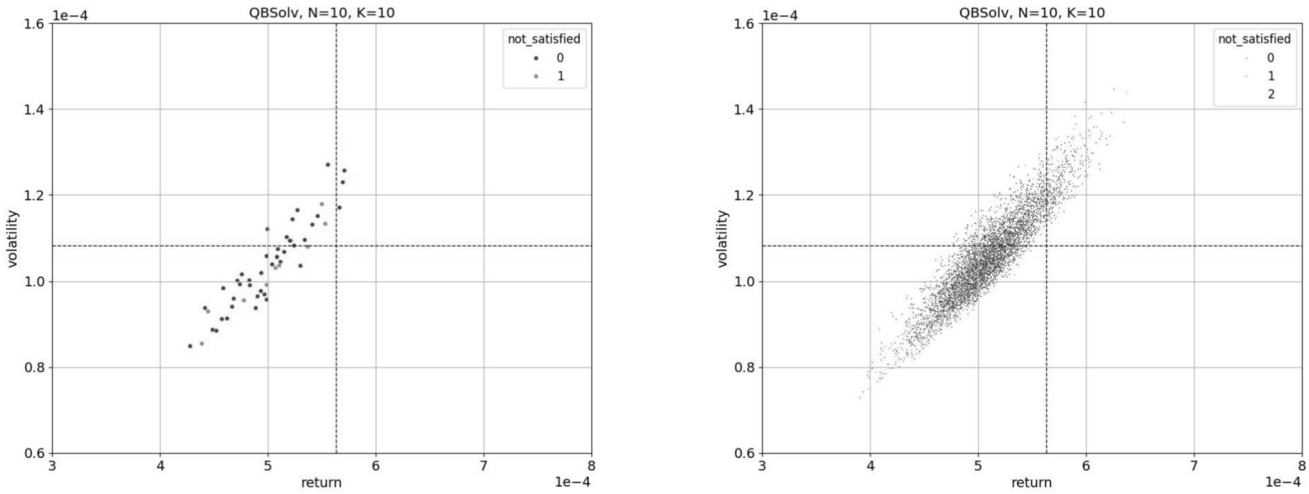


Fig. 9 Comparing the solutions for QBSolv in risk vs. volatility for 50 (left) and 5000 (right) iterations, respectively. It can be seen qualitatively that a lower sampling provides already a good representation of the solution space

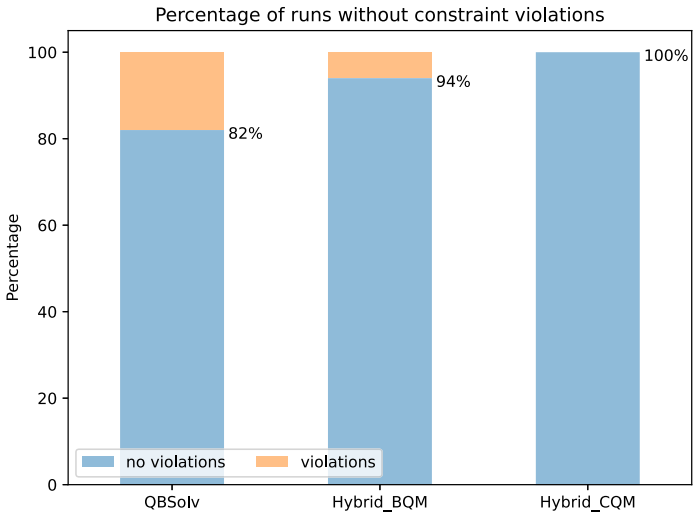


Fig. 10 Comparison of success probability for each solver. Any constraint violation will be counted. While in QBSolv and Hybrid BQM, 82% and 96% of the runs were satisfying all the constraints, respectively, this was true for all runs with Hybrid CQM

Appendix C: Constraint violation

In Section 2, we have presented the constraints on the problem solution. Those constraints are hard constraints for the business context and a solution can only be used if they are obeyed. Due to the nature of the solution strategy, however, violation of some of the constraints is to be expected. This is because the constraints are included in QUBO by imposing an energy penalty, which does not guarantee it is obeyed. Therefore, post-selection of the results is necessary. For the application in a production context, it is relevant to understand the success probability in the sense of which fraction of the results do not violate any constraints.

We are evaluating the constraint violations found in our experiments in Fig. 10. With success probabilities between 82 and 100%, the procedure is usable with a large enough sample size. The hybrid methods have a slightly higher success probability than the simulation, which manifests the utility of using the QPU in the calculation. Hybrid CQM (100%) has a slight advantage over Hybrid BQM (94%). Here, we have evaluated only runs with 10 assets and 10 qubits.

Including also higher number of assets and binary variables shows a much more diversified picture. In Fig. 11, we make the rather unexpected observation that satisfying all constraints becomes more difficult with more binary variables. For Hybrid CQM, we have examined even more values for K in Fig. 12.

An interesting experiment is to scale the number of assets, because we expect the quantum computer to be more performant than the classical solver when we increase this number. In Fig. 13, we see that QBSolv is not able to find any permissible solutions with 499 assets while Hybrid CQM still always finds a permissible solution.

For the application of the procedure in a business context, not all violations are equally problematic. The volatility and normalization constraints can be slightly violated, for instance, while regulatory constraints must not. We therefore

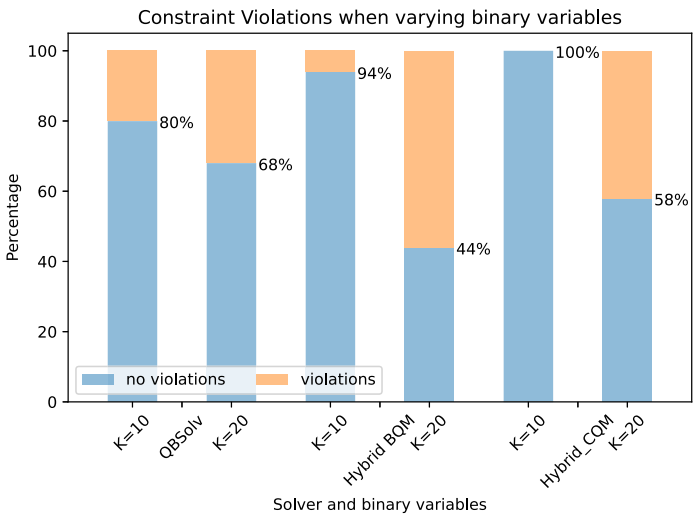


Fig. 11 Comparison of success probability for different numbers of binary variables for each of the solvers. It is striking to see that adding more variables does not necessarily lead to better performance. Presumably, this is due to the increase in search space, which makes it more difficult for the solver to find a solution satisfying all constraints

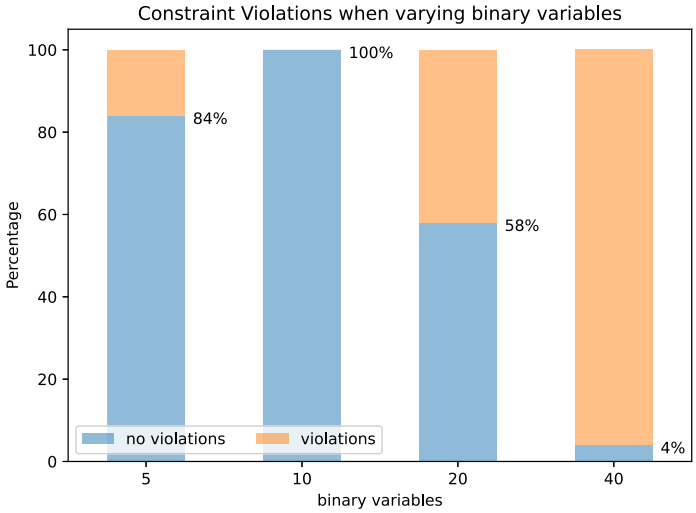


Fig. 12 The effect of the granularity on constraint violation for Hybrid CQM

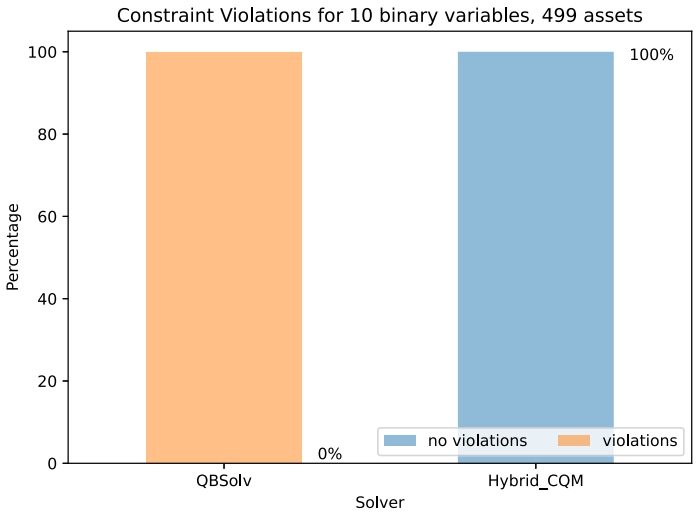


Fig. 13 Examination of the success probability for a large number of assets ($N = 499$) shows that Hybrid CQM can still find solutions which satisfy the constraints as opposed to QBSolv

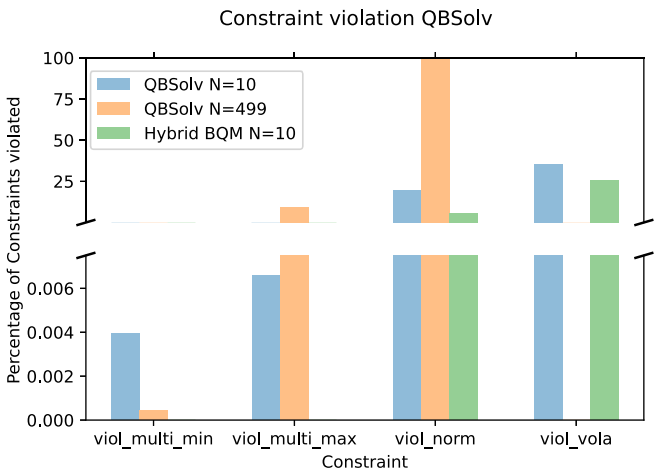


Fig. 14 Percentage of experiments with constraint violations per constraint type and solver. Multi-min and multi-max constraints are violated at a much lower frequency than normalization and volatility constraint

also examine which constraints are violated in each context in Fig. 14.

Given the construction in Eq. 16, the single-min and single-max constraints are automatically and always satisfied and are therefore not displayed. This is not true for the other constraints. We see violations on two levels. While the multi-min and multi-max constraints are violated on the $10^{-3}\%$ level (except for QBSolv with many assets), normalization and volatility constraints are violated above 20%.

The normalization constraint is affected by the granularity as we have already detailed in Section 3.1. We report the violation of the normalization constraint in our experiments in Fig. 5.

It is obvious that the granularity affects the violation of the normalization constraint. For the calculation of the error stemming from the finite granularity of representing the binary expansion of the weights, we are applying a telescope procedure explained in Fig. 15.

In principle, every rational number between 0 and 1 is equally probable for a specific weight, so the probability distribution is uniform. The error produced when representing the continuous number by a discrete binary variable depends in a linear fashion on its distance from the number. The representation in Fig. 15 makes it directly obvious, that the expected error cancels in every part of the telescope.

We are first calculating the expected value of the error ϵ as depicted in Fig. 15. Each contribution to the expectation value $\mathrm{E}[\epsilon]$ from the first $2^K - 1$ integrals cancel. We see that

$$\begin{aligned} \mathrm{E}_0 &= \int_0^{\frac{p_K}{2}} x \,\mathrm{d}x + \int_{\frac{p_K}{2}}^{p_K} (x - p_K)\,\mathrm{d}x \\ &= 0 \\ &= \mathrm{E}_i \quad \text{for } i \leq 2^K - 1\,. \end{aligned} \tag{C1}$$

The non-zero contribution to the expected value is

$$\mathrm{E}_{2^K} = \mathrm{E}[\epsilon] = \int_{1-p_K}^{1} (x - (1 - p_K))\,\mathrm{d}x = \frac{p_K^2}{2} \sim \mathcal{O}(p_K^2)\,. \tag{C2}$$

Note that this is the expected error for an individual unconstrained weight with possible values in [0, 1]. We obtain the expected value of the sum of weights according to Eq. 26. In the case at hand in this study, it turns out that the expected error of the normalization constraint is the same as the prefactor just turns out to be unity.

Concerning the standard deviation, we are determining the second moment of our error distribution $\mathrm{Var}[\epsilon] = \mathrm{E}[\epsilon^2] - \mathrm{E}[\epsilon]^2$. We are following the prescription outlined in Fig. 15. The first integral of $\mathrm{E}[\epsilon^2]$

$$\begin{aligned} \mathrm{E}[\epsilon^2]_0 &= \int_0^{\frac{p_K}{2}} x^2\,\mathrm{d}x + \int_{\frac{p_K}{2}}^{p_K} (x - p_K)^2\,\mathrm{d}x = \frac{p_k^3}{12} \\ &= \mathrm{E}[\epsilon^2]_i \quad \text{for } i \leq 2^k - 1\,. \end{aligned} \tag{C3}$$

Given the structure of the telescope, this is the contribution to the variance for all the integrals up to the final one

$$\mathrm{E}[\epsilon^2]_{2^K} = \int_{1-p_K}^{1} (x - (1 - p_K))^2\,\mathrm{d}x = \frac{p_K^3}{3}\,. \tag{C4}$$

Summing $2^K - 1$ parts of Eqs. C3 and C4, we obtain

$$\begin{aligned} \mathrm{E}[\epsilon^2] &= \left(\frac{1}{p_K} - 1\right)\frac{p_K^3}{12} + \frac{p_K^3}{3} \\ &= \frac{p_K^2}{12} + \frac{p_K^3}{4} \sim \mathcal{O}(p_K^2)\,. \end{aligned} \tag{C5}$$

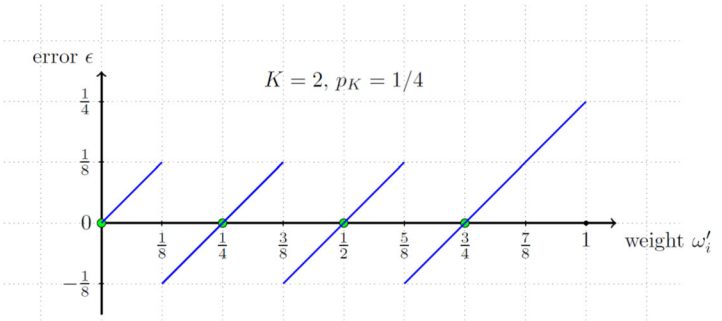
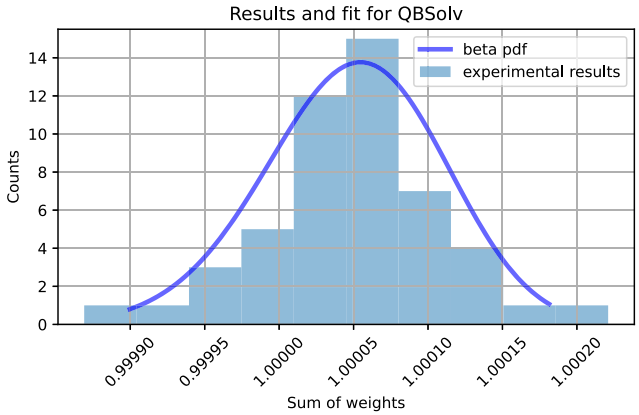


Fig. 15 This sketch explains the calculation of the error in the approximation of a rational number between 0 and 1 using binary variables. For the purpose of the illustration, we are using $K = 2$ variables without loss of generality. Potential representations (0, 0), (1, 0), (0, 1), (1, 1) which represent the numbers 0, $\frac{1}{4}$, $\frac{1}{2}$, $\frac{3}{4}$ (green dots). Each possible continuous rational number is represented by a discrete binary value that has the smallest difference. The difference/error behaves as depicted and grows larger close to unity. This is because in the construction of Section 3.1 unity is not reached due to a trade-off for higher resolution



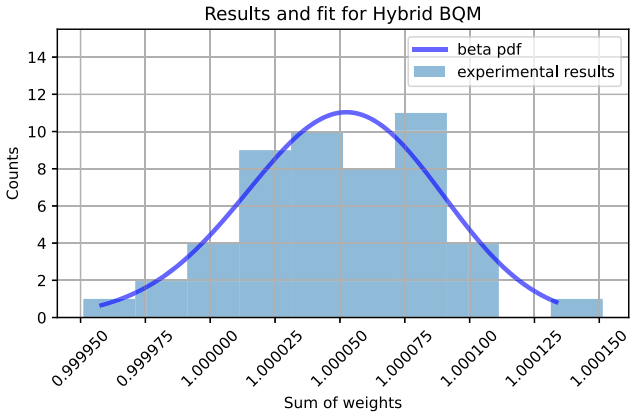


Fig. 16 Comparison of experimental results of the sum of weights for QBSolv and HybridBQM for 10 binary variables with a corresponding beta distribution fit. In principle, a skewed beta distribution seems to be suitable to generate the experimental data. To get a more conclusive understanding, larger statistics would be necessary, presumably

Putting it all together, we obtain for the variance for the error

$$\mathrm{Var}[\epsilon] = \frac{p_K^2}{12} + \frac{p_K^3}{4} - \frac{p_K^4}{4} \sim \mathcal{O}(p_K^2)\,. \tag{C6}$$

For determining the skewness

$$\mathrm{Skew}[\epsilon] = \frac{\mathrm{E}[\epsilon^3] - 3\mathrm{E}[\epsilon]\mathrm{Var}[\epsilon] - \mathrm{E}[\epsilon]^3}{(\mathrm{Var}[\epsilon])^{3/2}}\,, \tag{C7}$$

we calculate the third moment $\mathrm{E}[\epsilon^3]$. Due to the anti-symmetry of the error function, the first integrals vanish when partitioning in the same way as when calculating the expectation value

$$\mathrm{E}[\epsilon^3]_i = \int_0^{\frac{p_K}{2}} x^3\,\mathrm{d}x + \int_{\frac{p_K}{2}}^{p_K} (x - p_K)^3\,\mathrm{d}x = 0 \quad \text{for } i \leq 2^K - 1\,. \tag{C8}$$

The non-vanishing contribution comes from

$$\mathrm{E}[\epsilon^3]_{2^K} = \int_{1-p_K}^{1} (x - (1 - p_K))^3\,\mathrm{d}x = \frac{p_K^4}{4}\,. \tag{C9}$$

Inserting the results of Eqs. C2, C6, and C9 into Eq. C7, one gets

$$\begin{aligned}\mathrm{Skew}[\epsilon] &= \frac{\frac{p_K^4}{4} - 3\frac{p_K^2}{2}(\frac{p_K^2}{12} + \frac{p_K^3}{4} - \frac{p_K^4}{4}) - (\frac{p_K^2}{2})^3}{(\frac{p_K^2}{12} + \frac{p_K^3}{4} - \frac{p_K^4}{4})^{\frac{3}{2}}} = \\ &= 3\sqrt{3}\,(p_K - 3p_K^2 + 2p_K^3)\cdot(1 + 3p_K - 3p_K^2)^{-\frac{3}{2}} \\ &= 3\sqrt{3}\,p_K + \mathcal{O}(p_K^2) \sim \mathcal{O}(p_K).\end{aligned} \tag{C10}$$

As reflected in the data of Table 1, the higher moments of theory and experiments do not match. In Fig. 16, we show how a skewed beta distribution coincides with the experimental data. For the results to be conclusive, presumably larger statistics would be needed.

All in all, the considerations in this section show good usability of the approach for a large enough sampling, which is reflected already in Fig. 6, where the best usable portfolio, which is marked with a solid line is chosen such that no constraints are violated.

Appendix D: Zoomed plots

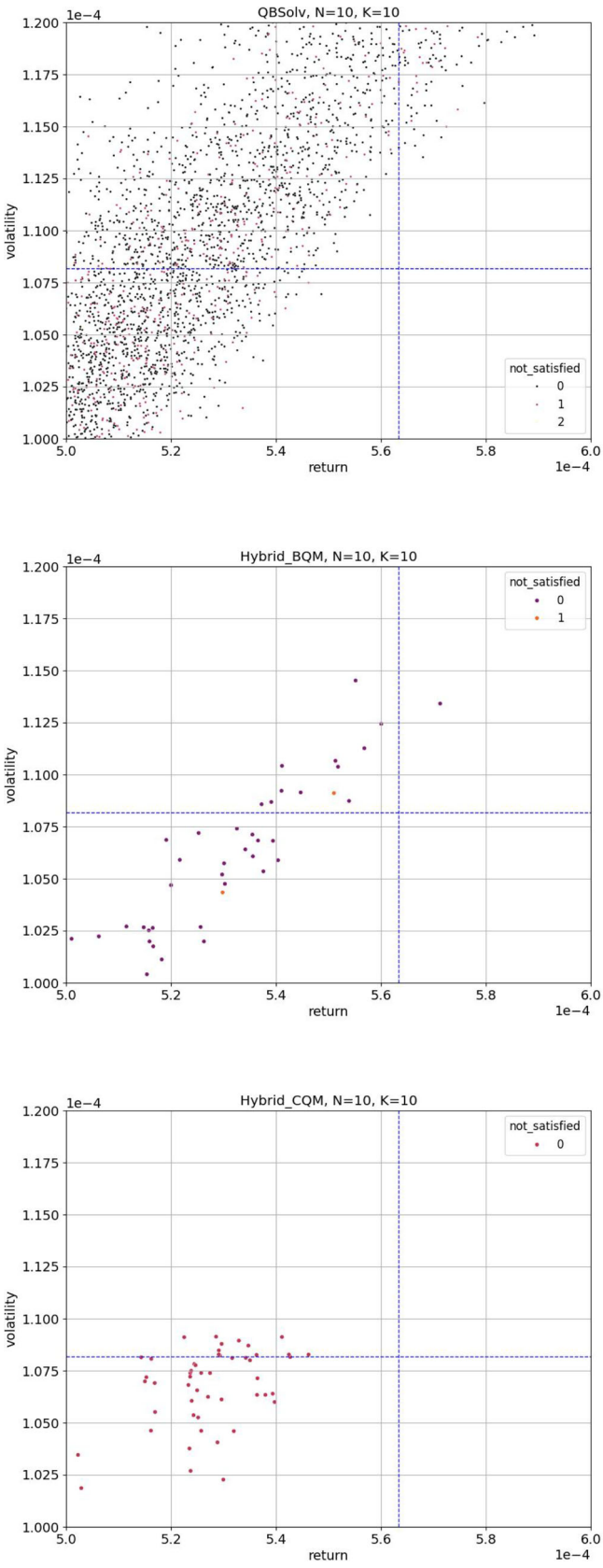


Fig. 17 Return is plotted against volatility like in Fig. 1 but zoomed in around the center of the solutions for better visibility. It can be seen very clearly that the variance of the solutions is greatly reduced by the use of a QPU. In particular, for Hybrid CQM, the solutions become more clustered around the volatility threshold line and the shape is more circular than elongated, which gives a hint at the different way the quantum algorithm is used

Acknowledgements The data and classical results were kindly provided by Simon Haller and Björn Chyba from RBI Research. We are also grateful for guidance on the business context and application. We also thank Vjekoslav Bonic for fruitful discussions.

Author contribution W.S., J.M.O., R.A., L.A. and J.S. devised the concept of the research project, worked out the theory, devised the solution strategy and worked on the interpretation of the results and the conclusion. W.S. ran the simulations on the quantum backend. B.R. provided the business context and contributed to the discussion in the initial stage of the work. W.S., J.M.O. and L.A. wrote the main manuscript text and prepared all the figures. All authors reviewed the manuscript.

Funding J.M.O. was partially funded by the grant BenchQC (DIK-2210-0011// DIK0425/02) of the Bavarian Ministry for Commerce (StMWi).

Data availability No datasets were generated or analysed during the current study.

Declarations

Conflict of interest All authors are employed by commercial companies, who are in principle investigating the use of quantum computing for applications with the aim of its commercialisation. The results of this work inform future commercial activities of the companies in the application of quantum computation. Machine Learning Reply has received a government grant by the Bavarian Ministry of Commerce for benchmarking quantum algorithms. The results of this work have been used in the application of this grant and will inspire the research carried out. Furthermore, the authors have no direct financial benefit from the research reported in this manuscript.

References

Asproni L, Caputo D, Silva B, Fazzi G, Magagnini M (2020) Accuracy and minor embedding in subqubo decomposition with fully connected large problems: a case study about the number partitioning problem. Quant Mach Intell 2(1):4. https://doi.org/10.1007/s42484-020-00014-w

Bienstock D (1999) A computational study of a family of mixed-integer quadratic programming problems. Math Program Ser B 74. https://doi.org/10.1007/BF02592208

Black F, Litterman R (1992b) Global portfolio optimization. Financ Anal J 48(5):28–43

Black F, Litterman RB (1991a) Asset allocation. J Fixed Income 1(2):7–18. https://doi.org/10.3905/jfi.1991.408013https://jfi.pm-research.com/content/1/2/7.full.pdf

Brown D (2011) Learning and control techniques in portfolio optimization. thesis, Brigham Young University

Cesarone F, Scozzari A, Tardella F (2009) Efficient algorithms for mean-variance portfolio optimization with hard real-world constraints. G Ist Ital Attuari 72

Cesarone F, Scozzari A, Tardella F (2011) Portfolio selection problems in practice: a comparison between linear and quadratic optimization models

Farhi E, Goldstone J, Gutmann S, Lapan J, Lundgren A, Preda D (2001) A quantum adiabatic evolution algorithm applied to random instances of an NP-complete problem. Science 292(5516):472–476. https://doi.org/10.1126/science.1057726 arXiv:quant-ph/0104129 [quant-ph]

Farhi E, Goldstone J, Gutmann S, Sipser M (2000) Quantum computation by adiabatic evolution. Quantum Physics

Glover FW, Kochenberger, GA (2018) A tutorial on formulating QUBO models. arXiv:1811.11538

Glover F, Hao J-K, Kochenberger G (2011) Polynomial unconstrained binary optimisation - part 1. Int J Metaheuristics 1:232–256. https://doi.org/10.1504/IJMHEUR.2011.041196

Glover F, Hao J-K, Kochenberger G (2011) Polynomial unconstrained binary optimisation â part 2. Int J Metaheuristics 1:317. https://doi.org/10.1504/IJMHEUR.2011.044356

Grant E, Humble TS, Stump B (2021) Benchmarking quantum annealing controls with portfolio optimization. Phys Rev Appl 15:014012. https://doi.org/10.1103/PhysRevApplied.15.014012

Heim B, Brown EW, Wecker D, Troyer M (2017) Designing adiabatic quantum optimization: a case study for the traveling salesman problem 1702–06248. arXiv:1702.06248 [quant-ph]

Heim B, Rønnow TF, Isakov SV, Troyer M (2015) Quantum versus classical annealing of Ising spin glasses. Science 348(6231):215–217. https://doi.org/10.1126/science.aaa4170https://www.science.org/doi/pdf/10.1126/science.aaa4170

IBM (2022a) CPLEX. https://www.ibm.com/analytics/cplex-optimizer. Accessed: 14-Nov-2022

IBM (2022b) Rcplex. https://cran.r-project.org/web/packages/Rcplex/Rcplex.pdf. Accessed: 14-Nov-2022

Jin Y, Qu R, Atkin J (2016) Constrained portfolio optimisation: the state-of-the-art Markowitz models, pp 388–395. https://doi.org/10.5220/0005758303880395

Kadowaki T, Nishimori H (1998) Quantum annealing in the transverse Ising model. Phys. Rev. E 58:5355–5363. https://doi.org/10.1103/PhysRevE.58.5355

Katzgraber HG, Hamze F, Zhu Z, Ochoa AJ, Munoz-Bauza H (2015) Seeking quantum speedup through spin glasses: the good, the bad, and the ugly. Phys Rev X00(5):031026. https://doi.org/10.1103/PhysRevX.5.031026

Kerenidis I, Prakash A, Szilágyi D (2019) Quantum algorithms for portfolio optimization

Kochenberger G, Hao J-K, Glover F, Lewis M, Lü Z, Wang H, Wang Y (2014) The unconstrained binary quadratic programming problem: a survey. J Comb Optim 28(1):58–81. https://doi.org/10.1007/s10878-014-9734-0

Lucas A (2014) Ising formulations of many NP problems. Front Phys 2. https://doi.org/10.3389/fphy.2014.00005

Maringer D (2008) Heuristic optimization for portfolio management. Comput Intell Mag IEEE 3(4):31–34. https://doi.org/10.1007/b136219

Markowitz H (1952) Portfolio selection. J Finance 7(1):77–91

Morita S, Nishimori H (2008) Mathematical foundation of quantum annealing. J Math Phys 49. https://doi.org/10.1063/1.2995837

Mugel S, Kuchkovsky C, Sánchez E, Fernández-Lorenzo S, Luis-Hita J, Lizaso E, Orús R (2022) Dynamic portfolio optimization with real datasets using quantum processors and quantum-inspired tensor networks. Phys Rev Res 4:013006. https://doi.org/10.1103/PhysRevResearch.4.013006

Mugel S, Lizaso E, Orus R (2020) Use cases of quantum optimization for finance

Nesterov Y, Nemirovski A (1994) Interior-point polynomial algorithms in convex programming. In: Siam studies in applied mathematics

Palmer S, Sahin S, Hernandez R, Mugel S, Orus R (2021) Quantum portfolio optimization with investment bands and target volatility

Pedersen M (2021) Simple portfolio optimization that works! SSRN Electronic Journal. https://doi.org/10.2139/ssrn.3942552

Rosenberg G, Haghnegahdar P, Goddard P, Carr P, Wu K, de Prado ML (2016) Solving the optimal trading trajectory problem using a quantum annealer. IEEE J Sel Top Signal Process 10(6):1053–1060. https://doi.org/10.1109/jstsp.2016.2574703