

# Mean-CVaR Portfolio Optimisation in Whole Shares

From the Rockafellar–Uryasev linear programme in portfolio weights to the mixed-integer programme with rebalancing and transaction costs implemented in `quantcore`

AISE502 – AI-Augmented Portfolio Intelligence Platform  
Deterministic quant core, module `quantcore.optimization`

September 2026

## Abstract

The portfolio optimisation service of the platform selects a portfolio of stocks such that the expected profit over a horizon is maximal while the Conditional Value-at-Risk (expected shortfall) of the loss over that horizon stays below an absolute risk budget in CHF. Positions are whole numbers of shares, sector and single-stock limits apply, short positions are not allowed, and an existing portfolio can be rebalanced at proportional and fixed transaction costs. This note states the classical scenario-based mean-CVaR linear programme of Rockafellar and Uryasev in portfolio weights, and derives from it, step by step, the mixed-integer linear programme (MILP) that `quantcore/optimization/problem.py` builds and that is solved on NVIDIA cuOpt (with HiGHS as fallback). Every constraint of the implementation is traced back to its origin, a worked example shows why rounding a continuous solution is not an option, a section on scenario sets quantifies the sampling error of the approximation and derives the variance-reduction, importance-sampling and row-generation techniques in the code together with the exact (mean-variance-equivalent) solution of the Gaussian case, and the last section lists the complete model with the variable and row names used in the code.

## Contents

|          |                                                                 |          |
|----------|-----------------------------------------------------------------|----------|
| <b>1</b> | <b>Setting and notation</b>                                     | <b>2</b> |
| <b>2</b> | <b>Conditional Value-at-Risk and its linear representation</b>  | <b>2</b> |
| <b>3</b> | <b>The classical model in portfolio weights</b>                 | <b>4</b> |
| <b>4</b> | <b>From weights to numbers of shares</b>                        | <b>4</b> |
| 4.1      | Money instead of fractions . . . . .                            | 5        |
| 4.2      | Budget, cash and the constraints . . . . .                      | 5        |
| 4.3      | The mean-CVaR MILP in shares . . . . .                          | 6        |
| 4.4      | Why the integrality cannot be handled by rounding . . . . .     | 6        |
| <b>5</b> | <b>Rebalancing an existing portfolio with transaction costs</b> | <b>7</b> |
| 5.1      | Trades, linking and the cash balance . . . . .                  | 7        |
| 5.2      | Transaction costs enter the loss . . . . .                      | 7        |
| 5.3      | Limits on the rebalancing itself . . . . .                      | 8        |
| 5.4      | The new portfolio as a special case . . . . .                   | 8        |

|          |                                                                                    |           |
|----------|------------------------------------------------------------------------------------|-----------|
| 5.5      | Optional cardinality and minimum position size . . . . .                           | 8         |
| <b>6</b> | <b>Scenario sets: sampling error, variance reduction and large sets</b>            | <b>8</b>  |
| 6.1      | How noisy is a plain Monte-Carlo set? . . . . .                                    | 9         |
| 6.2      | Quasi-Monte Carlo and moment matching . . . . .                                    | 9         |
| 6.3      | Importance sampling for the tail . . . . .                                         | 10        |
| 6.4      | One scenario set for a family of runs . . . . .                                    | 12        |
| 6.5      | How many scenarios, and how many assets? . . . . .                                 | 12        |
| 6.6      | Tail-row generation for large scenario sets . . . . .                              | 13        |
| 6.7      | The Gaussian special case: mean–variance equivalence and an exact solution . . . . | 14        |
| 6.8      | Rank of the covariance estimate . . . . .                                          | 15        |
| 6.9      | Reproducibility . . . . .                                                          | 15        |
| <b>7</b> | <b>The complete model as implemented</b>                                           | <b>16</b> |

## 1 Setting and notation

There are  $N$  candidate stocks  $i = 1, \dots, N$  with current prices  $P_i > 0$  in the base currency (CHF). The investor has cash  $c^0 \geq 0$  and, in the rebalancing case, existing holdings  $n_i^0 \in \mathbb{Z}_{\geq 0}$  shares; the total wealth is

$$W = c^0 + \sum_{i=1}^N P_i n_i^0. \quad (1)$$

The uncertainty is described by  $S$  equiprobable scenarios of simple returns over the horizon,

$$R_{s,i} = e^{r_{s,i}} - 1, \quad s = 1, \dots, S, \quad (2)$$

where the log-returns  $r_{s,i}$  (in CHF) are simulated by one of the stock-price models of the platform (geometric Brownian motion or empirical innovations) with the expected returns of the Black–Litterman model as drift; see `docs/METHODOLOGY.md`, Sections 6–7. Each scenario has probability  $1/S$ . The confidence level of the risk measure is  $\beta \in (0, 1)$  (0.95 or 0.99).

Throughout, *losses are positive numbers*: a loss of  $L$  means the portfolio is worth  $L$  less than at the start of the horizon. Vectors are columns,  $[z]^+ = \max(z, 0)$ , and  $\mathbf{1}$  is the vector of ones.

## 2 Conditional Value-at-Risk and its linear representation

Let  $L(x, Y)$  be the loss of a decision  $x \in X \subseteq \mathbb{R}^N$  under the random outcome  $Y$ , with distribution function  $\Psi(x, \zeta) = \mathbb{P}\{L(x, Y) \leq \zeta\}$ .

**Definition 2.1** (VaR and CVaR). The *Value-at-Risk* at level  $\beta$  is the  $\beta$ -quantile of the loss,  $\text{VaR}_\beta(x) = \min\{\zeta \in \mathbb{R} : \Psi(x, \zeta) \geq \beta\}$ . The *Conditional Value-at-Risk* (expected shortfall) is the mean of the  $\beta$ -tail of the loss distribution,

$$\text{CVaR}_\beta(x) = \frac{1}{1-\beta} \int_\beta^1 \text{VaR}_\gamma(x) d\gamma, \quad (3)$$

which for a continuous loss distribution equals  $\mathbb{E}[L(x, Y) \mid L(x, Y) \geq \text{VaR}_\beta(x)]$ . For distributions with atoms (in particular for a finite set of scenarios) the definition (3) splits the atom at the VaR appropriately [2, 4].

CVaR is a coherent risk measure (monotone, sub-additive, positively homogeneous, translation invariant), which VaR is not. Its value for optimisation purposes rests on the following representation.

**Theorem 2.2** (Rockafellar–Uryasev [1, 2]). *Define the auxiliary function*

$$F_\beta(x, \zeta) = \zeta + \frac{1}{1-\beta} \mathbb{E} \left[ [L(x, Y) - \zeta]^+ \right]. \quad (4)$$

*Then  $F_\beta(x, \cdot)$  is finite and convex, and*

$$\text{CVaR}_\beta(x) = \min_{\zeta \in \mathbb{R}} F_\beta(x, \zeta), \quad \text{VaR}_\beta(x) = \text{left endpoint of } \arg \min_{\zeta} F_\beta(x, \zeta). \quad (5)$$

*If  $L(x, Y)$  is convex in  $x$ , then  $F_\beta$  is jointly convex in  $(x, \zeta)$  and*

$$\min_{x \in X} \text{CVaR}_\beta(x) = \min_{(x, \zeta) \in X \times \mathbb{R}} F_\beta(x, \zeta), \quad (6)$$

*with  $(x^*, \zeta^*)$  minimising the right-hand side if and only if  $x^*$  minimises the left-hand side and  $\zeta^* \in \arg \min_{\zeta} F_\beta(x^*, \zeta)$ .*

The theorem removes the need to know the VaR before computing the CVaR: both come out of one minimisation over the scalar  $\zeta$ , and  $\zeta$  can simply be added to the decision variables. A second consequence is used for risk *constraints* [3]: for any budget  $\omega$ ,

$$\text{CVaR}_\beta(x) \leq \omega \iff \exists \zeta : F_\beta(x, \zeta) \leq \omega, \quad (7)$$

because the minimum over  $\zeta$  is at most  $\omega$  exactly when some  $\zeta$  attains at most  $\omega$ .

**Scenario approximation and linearisation.** With  $S$  equiprobable scenarios the expectation in (4) becomes an average,

$$\tilde{F}_\beta(x, \zeta) = \zeta + \frac{1}{(1-\beta)S} \sum_{s=1}^S [L_s(x) - \zeta]^+, \quad L_s(x) = L(x, Y_s). \quad (8)$$

The positive parts are linearised with auxiliary variables  $u_s \geq 0$ : since  $u_s \geq L_s(x) - \zeta$  and  $u_s \geq 0$  imply  $u_s \geq [L_s(x) - \zeta]^+$ , and the objective/constraint pushes  $u_s$  down,

$$\min_{\zeta} \tilde{F}_\beta(x, \zeta) = \min_{\zeta, u} \left\{ \zeta + \frac{1}{(1-\beta)S} \sum_s u_s : u_s \geq L_s(x) - \zeta, u_s \geq 0, s = 1, \dots, S \right\}. \quad (9)$$

Whenever  $L_s(x)$  is linear in  $x$ , (9) is a linear programme in  $(x, \zeta, u)$ .

*Remark 2.3* (The discrete tail and the meaning of  $\zeta$ ). For  $S$  scenarios and  $k = (1-\beta)S$ , the minimum of (8) equals the average of the  $k$  largest losses, where a fractional  $k$  gives the  $(\lfloor k \rfloor + 1)$ -th largest loss the fractional weight  $k - \lfloor k \rfloor$ . This is exactly what `quantcore.measurement.risk.scenario_cvar` computes, and it is the value the LP returns. When  $k$  is an integer,  $\tilde{F}_\beta(x, \cdot)$  is flat between the  $(k+1)$ -th and the  $k$ -th largest loss, so the minimiser  $\zeta$  is *not unique*: any point of that interval is optimal, and the VaR is its left endpoint. Consequently the  $\zeta$  (called  $\alpha$  in the code) reported by a solver need not equal the VaR, in particular when the CVaR constraint is not tight; the implementation therefore recomputes VaR and CVaR of the decoded portfolio from the scenarios rather than reading them off the LP variables.

### 3 The classical model in portfolio weights

The textbook formulation [1, 3] describes a portfolio by weights  $w \in \mathbb{R}^N$ , the fractions of wealth invested in each stock. Its loss over the horizon, relative to the initial wealth, is minus the portfolio return,

$$L_s(w) = -\sum_{i=1}^N R_{s,i} w_i = -R_s^\top w, \quad (10)$$

which is linear in  $w$ , and the expected return is  $\bar{R}^\top w$  with  $\bar{R}_i = \frac{1}{S} \sum_s R_{s,i}$ . With a risk budget  $\omega$  expressed as a fraction of wealth (e.g. “the 95 % CVaR over one week may not exceed 4 % of the portfolio value”), the *mean-CVaR linear programme* reads

$$\max_{w, \zeta, u} \quad \bar{R}^\top w \quad (11a)$$

$$\text{s.t.} \quad u_s \geq -R_s^\top w - \zeta, \quad u_s \geq 0, \quad s = 1, \dots, S, \quad (11b)$$

$$\zeta + \frac{1}{(1-\beta)S} \sum_{s=1}^S u_s \leq \omega, \quad (11c)$$

$$\mathbf{1}^\top w \leq 1, \quad w \geq 0, \quad (11d)$$

$$w_i \leq \bar{w}, \quad i = 1, \dots, N, \quad (11e)$$

$$\sum_{i \in G_k} w_i \leq \bar{w}_k, \quad k = 1, \dots, K. \quad (11f)$$

Here (11b)–(11c) are (9) and (7) combined, (11d) is the budget with a non-negative cash residual  $1 - \mathbf{1}^\top w$  (earning nothing over the horizon) and the no-short-sale condition, (11e) caps each stock at the fraction  $\bar{w}$  (10 % in the platform) and (11f) caps every GICS sector  $G_k$  at  $\bar{w}_k$ .

**Three equivalent objectives.** Krokhtmal, Palmquist and Uryasev [3] show that the three problems

$$(a) \max \bar{R}^\top w \text{ s.t. } \text{CVaR}_\beta \leq \omega, \quad (b) \min \text{CVaR}_\beta \text{ s.t. } \bar{R}^\top w \geq \rho, \quad (c) \max \bar{R}^\top w - \lambda \text{CVaR}_\beta \quad (12)$$

trace out the same efficient frontier as  $\omega$ ,  $\rho$  and  $\lambda \geq 0$  vary. All three are linear after the substitution (9); the implementation offers them as the objectives `max_return`, `min_cvar` and `utility`. The platform’s primary mode is (a), because the risk-management specification is an *absolute* loss level for a period.

**Properties.** Problem (11) has  $N + 1 + S$  variables and  $S + 1 + N + K + 1$  constraints (the weight caps are variable bounds). Its constraint matrix is dense in the  $S$  scenario rows:  $S(N + 2)$  non-zeros. It is a linear programme, so every local optimum is global, the solution is a vertex, and it can be solved for thousands of scenarios and hundreds of assets in seconds.

### 4 From weights to numbers of shares

Weights are a convenient abstraction, but a real portfolio consists of whole shares: one cannot buy 2.66 shares of a stock, and for a stock priced at CHF 2000 the difference between 2 and 3 shares is 10 % of a CHF 10 000 portfolio. The platform therefore optimises directly over the integer numbers of shares.

#### 4.1 Money instead of fractions

Let  $n_i \in \mathbb{Z}_{\geq 0}$  be the number of shares of stock  $i$  held over the horizon. The value of the position is  $P_i n_i$ , and the weight of the classical model is recovered as

$$w_i = \frac{P_i n_i}{W}. \quad (13)$$

Substituting (13) into the relative loss (10) and multiplying by  $W$  turns it into a loss *in money*:

$$WL_s(w) = - \sum_i R_{s,i} P_i n_i = - \sum_i g_{s,i} n_i, \quad g_{s,i} := P_i R_{s,i}, \quad (14)$$

where  $g_{s,i}$  is the CHF gain of one share of stock  $i$  in scenario  $s$ . Because CVaR is positively homogeneous,  $\text{CVaR}_\beta(WL) = W \text{CVaR}_\beta(L)$ : the CVaR of the money loss is the money amount the risk manager specifies (“the 95 % CVaR over a week may not exceed CHF 40 000”), and the relative budget  $\omega$  of (11c) becomes the absolute budget  $\Lambda = \omega W$ . The expected return  $\bar{R}^\top w$  becomes the expected profit

$$\mathbb{E}[\text{P\&L}] = \sum_i \bar{g}_i n_i, \quad \bar{g}_i = \frac{1}{S} \sum_s g_{s,i} = P_i \bar{R}_i. \quad (15)$$

Both the objective and the risk constraint use the *same* scenario set, so the trade-off they describe is internally consistent (the drift of the scenarios is the expected-return model).

#### 4.2 Budget, cash and the constraints

Multiplying each constraint of (11) by  $W$  and substituting (13) gives:

- budget:  $\sum_i P_i n_i \leq W$  — the purchases may not exceed the available cash, what is not invested stays in cash;
- single stock:  $P_i n_i \leq \bar{w} W$ ; sector:  $\sum_{i \in G_k} P_i n_i \leq \bar{w}_k W$ ;
- no short sales:  $n_i \geq 0$ .

The single-stock cap deserves a closer look. As a linear constraint it is a row of the model, but for an integer variable it is equivalent to the *variable bound*

$$n_i \leq \bar{n}_i := \left\lfloor \frac{\bar{w} W}{P_i} \right\rfloor, \quad (16)$$

which is strictly tighter than the linear row whenever  $\bar{w} W / P_i$  is not an integer: the rounding down is free information that a MILP solver exploits (tighter LP relaxation, smaller branch-and-bound tree). The implementation uses (16) with  $\bar{w}$  replaced by  $\min(\bar{w}, f_{\max})$  if a maximum invested fraction  $f_{\max} < 1$  is given, and never leaves the cap to a row.

### 4.3 The mean–CVaR MILP in shares

Putting the pieces together, the new-portfolio problem (all wealth in cash,  $n^0 = 0$ ,  $W = c^0$ ) reads

$$\max_{n, \zeta, u} \sum_i \bar{g}_i n_i \quad (17a)$$

$$\text{s.t.} \quad \sum_i g_{s,i} n_i + \zeta + u_s \geq 0, \quad u_s \geq 0, \quad s = 1, \dots, S, \quad (17b)$$

$$\zeta + \frac{1}{(1-\beta)S} \sum_s u_s \leq \Lambda, \quad (17c)$$

$$\sum_i P_i n_i \leq c^0, \quad (17d)$$

$$\sum_{i \in G_k} P_i n_i \leq \bar{w}_k W, \quad k = 1, \dots, K, \quad (17e)$$

$$n_i \in \{0, 1, \dots, \bar{n}_i\}, \quad i = 1, \dots, N, \quad (17f)$$

$$\zeta \in \mathbb{R}.$$

Row (17b) is (11b) written as “ $\geq 0$ ” with the money loss (14) inserted ( $u_s \geq L_s - \zeta \Leftrightarrow \sum_i g_{s,i} n_i + \zeta + u_s \geq 0$ ). Apart from the integrality (17f) the problem is the LP (11) scaled by  $W$ ; with the integrality it is a mixed-integer linear programme, and the LP obtained by dropping (17f) is its *LP relaxation*.

### 4.4 Why the integrality cannot be handled by rounding

The optimum of the LP relaxation is a vertex, and its coordinates  $n_i$  are generically fractional. Rounding them is not a substitute for solving the MILP:

- rounding *up* (or to the nearest integer) can violate the budget, a cap, or the CVaR constraint — the risk budget is exceeded;
- rounding *down* is always feasible (all constraints are “ $\leq$ ” in  $n$ , and the scenario rows are satisfied by re-optimising  $\zeta, u$ ) but can be far from optimal, because the “lost” fractions can often be re-invested in other, cheaper shares;
- the true integer optimum can have a different *structure* than the relaxed one, not just rounded coordinates.

The gap is largest when  $W/P_i$  is small, i.e. for expensive stocks or small portfolios; for a CHF 1 000 000 portfolio and stocks below CHF 1 000 it is negligible, but the 10 % cap keeps each position to at most  $\bar{w}W/P_i$  shares, and a few stocks in the universe trade above CHF 10 000 (Lindt & Sprüngli participation certificates above CHF 100 000).

**Example 4.1** (Two stocks, eight scenarios). Prices  $P = (572, 2208)$ , cash  $W = c^0 = 10\,000$ ,  $\beta = 0.75$  (so  $k = (1-\beta)S = 2$ : the CVaR is the mean of the two worst scenarios), risk budget  $\Lambda = 250$ , caps  $\bar{w} = 0.7$ , no costs. The simple returns of the eight scenarios are

$$R = \begin{pmatrix} 0.005 & -0.082 \\ 0.014 & -0.052 \\ -0.069 & -0.020 \\ -0.002 & 0.055 \\ -0.009 & 0.083 \\ 0.048 & 0.064 \\ 0.044 & 0.028 \\ -0.011 & 0.042 \end{pmatrix}, \quad \bar{R} = (0.0025, 0.01475).$$

Stock 2 has six times the expected return of stock 1 but a heavier left tail.

| portfolio                                       | $n_1$  | $n_2$  | invested | $\mathbb{E}[\text{P\&L}]$ | $\text{CVaR}_{0.75}$ | feasible                     |
|-------------------------------------------------|--------|--------|----------|---------------------------|----------------------|------------------------------|
| LP relaxation ( $n \in \mathbb{R}_{\geq 0}^2$ ) | 2.6605 | 1.7876 | 5 468.90 | 62.02                     | 250.00               | yes, CVaR binding            |
| nearest rounding                                | 3      | 2      | 6 132    | 69.43                     | 280.13               | <b>no</b> : CVaR $> \Lambda$ |
| rounding down                                   | 2      | 1      | 3 352    | 35.43                     | 149.22               | yes, $-43\%$ vs. LP          |
| MILP optimum                                    | 7      | 1      | 6 212    | 42.58                     | 240.74               | yes                          |

Rounding to the nearest integers breaks the risk budget (the CVaR limit, not the cash: 6 132 of 10 000 are invested). The relaxed solution buys 1.79 shares of the attractive stock 2; the second whole share of it is too large a block of risk for the budget, so the integer optimum keeps one share of stock 2 and fills the remaining risk budget with *seven* shares of stock 1 — a portfolio that no rounding of (2.66, 1.79) produces. The enumeration of all feasible integer portfolios confirms (7, 1) as the optimum; the integrality gap  $62.02 - 42.58$  is 31 % of the relaxed value. (The numbers were computed with `MeanCVaRProblem` and `HiGHS` and are pinned by `tests/test_problem.py::test_documentation_example_rounding_fails`.)

## 5 Rebalancing an existing portfolio with transaction costs

When holdings  $n^0 \neq 0$  exist, the decision is no longer “which shares to buy” but “which shares to buy and which to sell”, and every trade costs money. Weights cannot express this naturally (a weight change of  $+2\%$  says nothing about the number of shares traded, and the cost depends on the traded value), whereas in share numbers it is immediate.

### 5.1 Trades, linking and the cash balance

Introduce the integer trade variables  $b_i \geq 0$  (shares bought) and  $s_i \geq 0$  (shares sold), linked to the final position by

$$n_i = n_i^0 + b_i - s_i, \quad s_i \leq n_i^0 \text{ (implied by } n_i \geq 0), \quad (18)$$

and the transaction cost with proportional rates  $c^b, c^s$  (10 basis points by default) and an optional fixed cost  $F$  per traded stock,

$$TC(b, s, t) = \sum_i (c^b P_i b_i + c^s P_i s_i) + F \sum_i t_i, \quad b_i + s_i \leq M_i t_i, \quad t_i \in \{0, 1\}, \quad (19)$$

where  $t_i$  indicates whether stock  $i$  is traded at all and  $M_i = \max(\bar{n}_i, n_i^0) + 1$  is a valid big-M (no trade can exceed it). The fixed cost makes small trades unattractive, which is what it should do; it is the only place where binary variables enter the cost model.

The budget (17d) becomes a *cash balance*: sales bring cash in, purchases and costs take it out, and the cash at the end of the rebalancing may not be negative,

$$c^{\text{end}} = c^0 + \sum_i P_i s_i - \sum_i P_i b_i - TC \geq 0 \iff \sum_i P_i b_i - \sum_i P_i s_i + TC \leq c^0. \quad (20)$$

*Remark 5.1* (Buying and selling the same stock). With positive proportional costs an optimal solution never has  $b_i > 0$  and  $s_i > 0$  simultaneously: reducing both by one share leaves  $n_i$  unchanged and saves  $(c^b + c^s)P_i$  in cost and cash. With zero costs the decomposition of a net trade into  $b_i$  and  $s_i$  is not unique but harmless. The decoded solution reports the net trade  $b_i - s_i$ .

### 5.2 Transaction costs enter the loss

The costs are paid at the start of the horizon and are therefore a deterministic part of the loss relative to the initial wealth  $W$  of (1):

$$L_s(n, b, s, t) = TC(b, s, t) - \sum_i g_{s,i} n_i. \quad (21)$$

By translation invariance,  $\text{CVaR}_\beta(TC - G) = TC + \text{CVaR}_\beta(-G)$ : the costs shift the whole loss distribution and consume part of the risk budget, exactly as they reduce the expected profit,

$$\mathbb{E}[\text{P\&L}] = \sum_i \bar{g}_i n_i - TC(b, s, t). \quad (22)$$

The scenario rows (17b) therefore become

$$\sum_i g_{s,i} n_i - TC(b, s, t) + \zeta + u_s \geq 0, \quad s = 1, \dots, S. \quad (23)$$

### 5.3 Limits on the rebalancing itself

Two constraints of the specification concern the trading rather than the resulting portfolio: a cap on the total transaction cost and a cap on the turnover (the traded value as a fraction of wealth),

$$TC(b, s, t) \leq TC_{\max}, \quad \sum_i P_i (b_i + s_i) \leq \tau W. \quad (24)$$

Both are optional. The single-stock and sector caps apply to the *final* positions  $n = n^0 + b - s$ : a legacy position above the cap has to be sold down to it, and if a turnover cap prevents that, the problem is (correctly) infeasible.

### 5.4 The new portfolio as a special case

With  $n^0 = 0$  nothing can be sold, so  $s = 0$  and  $b = n$ ; the linking rows (18) become trivial and are dropped, the costs become  $TC = c^b \sum_i P_i n_i + F \sum_i t_i$ , and the cash balance (20) reduces to  $\sum_i P_i (1 + c^b) n_i + F \sum_i t_i \leq c^0$ . This is (17) with proportional costs, i.e. mode **new** of the implementation.

### 5.5 Optional cardinality and minimum position size

A portfolio of 30 stocks is easier to monitor than one of 120 with many tiny positions; the integer model can express both a maximum number of positions and a minimum value per open position with one binary  $y_i$  per stock:

$$n_i \leq \max(\bar{n}_i, 1) y_i, \quad P_i n_i \geq v_{\min} y_i, \quad \sum_i y_i \leq K_{\max}, \quad y_i \in \{0, 1\}. \quad (25)$$

These constraints are optional (`max_positions`, `min_position_value`); they make the problem combinatorially harder and are usually not needed — the CVaR constraint and the integrality already produce concentrated portfolios.

## 6 Scenario sets: sampling error, variance reduction and large sets

The MILP sees the return distribution only through the scenario matrix  $g_{s,i} = P_i R_{s,i}$  and the probabilities of its rows. It is a *sample-average approximation* (SAA) of the problem one would like to solve — the one with the true distribution of the horizon returns [8] — and its solution inherits the sampling error of the scenario set. This section quantifies that error on the real universe, derives the three remedies implemented in `quantcore.models.sampling` and `quantcore.optimization`, and closes with the Gaussian special case, in which the exact solution is available and the mean-CVaR problem coincides with the mean-variance problem.

## 6.1 How noisy is a plain Monte-Carlo set?

Let the scenarios be  $S$  independent draws from the model with mean  $\mu$  (per asset, over the horizon) and covariance  $\Sigma$ . The sample mean of asset  $i$  has standard error  $\sigma_i/\sqrt{S}$ . With  $S = 2000$  and weekly volatilities of 3–11 % this is 0.07–0.25 % per week, i.e. 3–13 % per year — the same order as the expected returns themselves (5–12 % p.a. from the Black–Litterman model). The optimiser cannot tell signal from noise: it overweights the assets whose sample mean happens to be high, the classical “error maximisation” of [9]. The tail is only the second problem: the CVaR at 95 % is estimated from the  $(1 - \beta)S = 100$  worst scenarios, with a relative standard error of about  $0.37/\sqrt{(1 - \beta)S} \approx 2\%$  for a Gaussian tail.

Table 1 measures the consequences on the 30 largest stocks of the universe (CHF 1 million, weekly horizon,  $\text{CVaR}_{0.95} \leq 5\%$  of wealth, eight seeds per method). The reference is the exact optimum of Section 6.7, so “true” figures are exact. The comparison is made in portfolio weights (the continuous relaxation) so that integrality neither hides nor causes noise; the integer results are the same to within a percent.

| $S = 2048, \beta = 0.95$                               | plain MC         | antithetic       | MC + mm          | Sobol’           | Sobol’ + mm       |
|--------------------------------------------------------|------------------|------------------|------------------|------------------|-------------------|
| in-sample $\mathbb{E}[\text{P\&L}]$ , mean $\pm$ sd    | $2791 \pm 552$   | $2228 \pm 19$    | $2221 \pm 10$    | $2221 \pm 8$     | $2220 \pm 5$      |
| true $\mathbb{E}[\text{P\&L}]$ of the chosen portfolio | $1991 \pm 119$   | $2228 \pm 19$    | $2221 \pm 10$    | $2220 \pm 5$     | $2220 \pm 5$      |
| loss versus the optimum (2219)                         | −228             | −9               | −2               | −0.6             | −0.4              |
| true CVaR over the limit, mean $\pm$ sd                | $-0.5 \pm 5.6\%$ | $+1.8 \pm 2.8\%$ | $+0.6 \pm 1.7\%$ | $+0.2 \pm 0.7\%$ | $+0.15 \pm 0.7\%$ |
| turnover between two seeds                             | 101 %            | 12.6 %           | 11.7 %           | 7.2 %            | 5.8 %             |
| turnover to the optimum                                | 75 %             | 8.9 %            | 8.3 %            | 5.0 %            | 4.1 %             |

Table 1: Sampling noise of the mean–CVaR optimisation (30 stocks, eight seeds, CHF per week; “turnover” is  $\sum_i |w_i - w'_i|$ ). “mm” = moment matching.

With plain Monte Carlo two seeds give essentially disjoint portfolios (turnover 101 %), the in-sample expected profit is 26 % too high and the true expected profit 10 % below the optimum. Fixing the first two moments of the set removes the illusion entirely; Sobol’ points halve the remaining tail noise. The residual 4–6 % turnover is the flatness of the optimum (many portfolios within 0.02 % of the optimal objective), not sampling. Increasing  $S$  does not rescue plain Monte Carlo: at  $S = 8192$  its turnover between seeds is still 76 % (the mean error decays like  $1/\sqrt{S}$ ), while Sobol’ + mm is at 4.0 %.

## 6.2 Quasi-Monte Carlo and moment matching

**Sobol’ points.** Instead of pseudo-random  $z_s \sim N(0, I)$  the implementation uses scrambled Sobol’ points [10]:  $S$  (rounded up to a power of two) points  $u_s \in (0, 1)^N$  that fill the unit cube far more evenly than random points, mapped through the inverse normal distribution function,  $z_s = \Phi^{-1}(u_s)$ . The scrambling is seeded, so the set is reproducible. The first coordinates of a Sobol’ sequence are the best distributed, so the draws are coloured with the eigen-root  $A = V\Lambda^{1/2}$  of  $\Sigma$  with eigenvalues in descending order: the first coordinate drives the market factor, which dominates every long-only portfolio’s loss [11, ch. 5].

**Moment matching.** Whatever the base sample, its first two moments can be made *exact* [12]. With probabilities  $p_s$  (equal unless importance sampling is used, Section 6.3), let  $m = \sum_s p_s z_s$  and  $C = \sum_s p_s (z_s - m)(z_s - m)^\top$ ; then

$$\tilde{z}_s = C^{-1/2}(z_s - m), \quad \sum_s p_s \tilde{z}_s = 0, \quad \sum_s p_s \tilde{z}_s \tilde{z}_s^\top = I, \quad (26)$$

and the scenarios  $r_s = hm_0 + \sqrt{h} A \tilde{z}_s$  have mean  $hm_0$  and covariance  $h\Sigma$  exactly, for any  $S > N$ . Only the *shape* of the tail remains sampled. For the empirical innovation model the same

transform is applied to the resampled sums of innovations (target covariance  $hI$ ), so that both models reproduce the latest covariance exactly.

**The objective uses the model mean.** Once the mean is exact there is no reason to estimate it from the scenarios at all. The implementation puts the model's expected simple return  $\mu_i$  (the Black–Litterman posterior, converted to the horizon) directly into the objective,  $\bar{g}_i = P_i \mu_i$ , and uses the scenarios only in the CVaR rows. This also keeps the objective exact under importance sampling, where a tail-focused sample estimates the mean *worse* than a plain one. The convention behind  $\mu_i$  matters: Black–Litterman returns are arithmetic expectations, so the log-drift of the scenarios is  $\ln(1 + h\mu_i)/h - \sigma_i^2/2$  (Itô correction) and  $\mathbb{E}[e^{r_i} - 1] = h\mu_i$ ; using  $\mu_i$  as the log-drift would add  $\sigma_i^2/2$  to the expected simple return — 5 % p.a. for a typical stock and 34 % for the most volatile one in the universe — and tilt the optimiser towards volatile names.

### 6.3 Importance sampling for the tail

Only the scenarios in the tail of the loss matter for the CVaR: at  $\beta = 0.99$ , 1 % of them. Importance sampling draws more scenarios where they matter and corrects with likelihood ratios [13]. Three things are specific to the optimisation setting.

*The tail depends on the decision.* The loss of portfolio  $v = P \circ n$  is  $L = -v^\top r = -v^\top h m_0 - (\sqrt{h} A^\top v)^\top z$ , so its tail is the half-space  $\{z : d^\top z \geq c\}$  with the unit direction  $d = -A^\top v / \|A^\top v\|$ . Because all long-only, capped, sector-limited candidates load mainly on the market factor, one direction serves all of them. The default takes  $v$  from the *cap-weighted portfolio of the candidate stocks* (`importance_sampling="market"`), which depends on the universe and the covariance but on none of the parameters a user varies; the alternative `"adaptive"` takes it from a first-pass solution of the problem itself, which is marginally sharper at  $\beta = 0.99$  but makes the scenario set depend on the risk limit and the constraints (Section 6.4).

*Bounded likelihood ratios.* A share  $\varepsilon$  (default one half) of the Sobol' points is shifted,  $z_s \mapsto z_s + \theta$  with  $\theta = z_\beta d$ , which centres the loss of  $v^{(1)}$  at its VaR; the rest stay nominal. The proposal is the defensive mixture  $g = (1 - \varepsilon)\varphi + \varepsilon\varphi(\cdot - \theta)$  [14], whose likelihood ratios

$$\ell_s = \frac{\varphi(z_s)}{g(z_s)} = \frac{1}{(1 - \varepsilon) + \varepsilon e^{\theta^\top z_s - \|\theta\|^2/2}} \leq \frac{1}{1 - \varepsilon} \quad (27)$$

are bounded, so a candidate whose tail points elsewhere is never wildly mis-estimated (the optimiser would exploit any such region). The scenario probabilities are the self-normalised ratios  $p_s = \ell_s / \sum_{s'} \ell_{s'}$ , and the moment matching (26) is applied with these weights.

*Weighted scenarios in the MILP.* Theorem 2.2 holds for any discrete distribution: with probabilities  $p_s$  the CVaR row (17c) becomes

$$\zeta + \frac{1}{1 - \beta} \sum_s p_s u_s \leq \Lambda, \quad (28)$$

and nothing else changes; the decoded VaR and CVaR are computed from the weighted sample in the same way (the tail is the set of largest losses whose probability adds up to  $1 - \beta$ , the last one taken fractionally).

For the empirical innovation model the tilt acts on the days: the resampling probabilities are  $q_t \propto \exp(\vartheta d^\top \varepsilon_t)$  with  $\vartheta$  chosen so that the tilted mean of  $d^\top \varepsilon$  equals  $z_\beta / \sqrt{h}$  per day (a one-dimensional root), the mixture draws all  $h$  days of a scenario either from  $q$  or uniformly, and the likelihood ratio of a scenario is  $1 / ((1 - \varepsilon) + \varepsilon \prod_k T q_{t_k})$ .

At equal  $S$  the tail noise falls by a factor 2–2.5 at 95 % and 3–4 at 99 %; equivalently, 2048 tilted scenarios are as good as 8192 plain ones. The gain is largest exactly where it is needed,

| $\beta, S$                       | 0.95, 2048 |        | 0.95, 8 192 |        | 0.99, 2048 |        | 0.99, 8 192 |        |
|----------------------------------|------------|--------|-------------|--------|------------|--------|-------------|--------|
|                                  | S+mm       | + IS   | S+mm        | + IS   | S+mm       | + IS   | S+mm        | + IS   |
| scenarios beyond the VaR         | 101        | 560    | 410         | 2 249  | 19         | 512    | 80          | 2 075  |
| true CVaR over the limit         | 0.15 %     | 0.21 % | 0.08 %      | 0.01 % | 2.45 %     | 0.50 % | 0.75 %      | 0.03 % |
| sd of the true CVaR (% of limit) | 0.74       | 0.30   | 0.36        | 0.15   | 1.67       | 0.40   | 0.79        | 0.20   |
| loss versus the optimum (CHF)    | 0.4        | 1.1    | 0.3         | 0.0    | 10.7       | 2.7    | 3.6         | 0.1    |
| turnover between two seeds       | 5.9 %      | 3.6 %  | 4.0 %       | 1.6 %  | 29 %       | 11.5 % | 15.4 %      | 5.7 %  |

Table 2: Importance sampling (“+ IS”: mixture  $\varepsilon = 0.5$ , two-stage tilt) versus moment-matched Sobol’ points (“S+mm”) at equal  $S$ ; 30 stocks, eight seeds, exact Gaussian reference.

at  $\beta = 0.99$ , where a plain set of 2048 scenarios has 19 tail points. The tilted LP is slower per scenario (a quarter of its rows are active instead of  $1 - \beta$ ), but the market tilt is on by default (`importance_sampling="auto" = "market"`) because it costs nothing in accuracy at 95 % and halves the noise at 99 %. Table 3 compares the two directions. Under the Gaussian model importance sampling is unnecessary — the exact solution of Section 6.7 needs no scenarios at all.

| $S = 2048$                       | $\beta = 0.95$ |        |            | $\beta = 0.99$ |        |            |
|----------------------------------|----------------|--------|------------|----------------|--------|------------|
|                                  | no tilt        | market | adaptive   | no tilt        | market | adaptive   |
| true CVaR over the limit         | 0.14 %         | 0.37 % | 0.20 %     | 2.43 %         | 1.37 % | 0.50 %     |
| sd of the true CVaR (% of limit) | 0.74           | 0.72   | 0.30       | 1.69           | 0.61   | 0.42       |
| turnover between two seeds       | 5.7 %          | 3.2 %  | 3.6 %      | 28.5 %         | 17.4 % | 11.6 %     |
| turnover to the optimum          | 4.1 %          | 2.6 %  | 2.4 %      | 22.4 %         | 13.0 % | 8.7 %      |
| depends on the risk parameters   | no             | no     | <b>yes</b> | no             | no     | <b>yes</b> |

Table 3: Tilt direction: the cap-weighted candidate portfolio (“market”, two blocks at  $z_{0.95}$  and  $z_{0.99}$ ) versus a first-pass solution (“adaptive”, one block at  $z_\beta$ ); 30 stocks, eight seeds.

*How far does one direction carry?* The tilt enriches the tail of the *reference* portfolio; another portfolio benefits in proportion to how much its loss is correlated with the reference’s. Table 4 measures this for portfolios from market-like to deliberately orthogonal. Long-only portfolios under single-stock and sector caps are market-like in practice — the optimiser’s solutions come out at an alignment of about 0.90 — and the reference is the cap-weighted portfolio *of the candidate stocks*, so a defensive universe gets a defensive reference. Where the alignment is nevertheless low the tilt does not help and costs a little precision (the nominal half of the mixture bounds the damage); the optimisation result reports the realised alignment (`tilt_alignment`) and warns below 0.3, where `importance_sampling="adaptive"` or `"off"` is the better choice.

| portfolio                      | alignment | scenarios beyond the VaR |        | noise of the CVaR estimate |        |           |
|--------------------------------|-----------|--------------------------|--------|----------------------------|--------|-----------|
|                                |           | plain                    | tilted | plain                      | tilted | reduction |
| cap-weighted (the reference)   | 1.00      | 41                       | 789    | 0.99 %                     | 0.19 % | 5.3×      |
| equal weight                   | 0.89      | 41                       | 644    | 0.79 %                     | 0.30 % | 2.6×      |
| mean-CVaR optimum              | 0.90      | 41                       | 630    | 1.39 %                     | 0.61 % | 2.3×      |
| five highest-beta stocks       | 0.68      | 41                       | 368    | 0.22 %                     | 0.34 % | 0.6×      |
| minimum variance               | 0.69      | 41                       | 350    | 1.34 %                     | 1.25 % | 1.1×      |
| five lowest-beta stocks        | −0.09     | 41                       | 34     | 1.66 %                     | 1.96 % | 0.8×      |
| three least market-like stocks | −0.15     | 41                       | 31     | 1.93 %                     | 1.95 % | 1.0×      |

Table 4: Reach of one tilt direction: “alignment” is the correlation between the portfolio’s loss and the reference’s;  $S = 4096$ ,  $\beta = 0.99$ , 12 seeds, 30 stocks. Noise is the standard deviation of the CVaR estimate across seeds, relative to its mean.

## 6.4 One scenario set for a family of runs

A portfolio proposal is rarely produced once: a risk limit is varied, a constraint tightened, a view added, two confidence levels compared. If each of those runs drew its own scenarios, the differences between the results would mix the effect of the parameter with the sampling noise of a new set — precisely the noise Sections 6.2 and 6.3 remove. The implementation therefore makes the scenario set a function of the *model* only. Its *scenario key* (reported with every result) contains

| in the key (changing it draws new scenarios)                                                                                                                                                                              | not in the key (the set is reused)                                                                                                                                                                                              |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| candidate stocks, data as-of date, frequency and horizon, covariance model, scenario model, number of scenarios, sampling scheme, moment matching, seed, tilt mode and mixture, innovation history and re-standardisation | CVaR limit (absolute or as a fraction), confidence level $\beta$ , objective, risk aversion, target return, all constraints, transaction costs, cash, existing holdings, expected-return model and views, solver and time limit |

Two properties make this possible. First, the tilt of Section 6.3 uses the cap-weighted candidate portfolio and blocks at both usual confidence levels, so it does not look at the optimisation parameters. Second, the drift enters *after* the random part: scenarios are generated with zero expected return and the expected-return model is added as a per-asset constant,  $r_s \mapsto r_s + hm$ , by `ScenarioSet.with_drift`. Changing a view therefore shifts every scenario by a constant and leaves the shocks — and hence the whole risk picture — untouched; the result reports a fingerprint of the shocks (`shock_hash`) next to the one of the scenarios (`hash`). Sets are cached per key inside the service, so a family of runs also pays the expensive innovation filter only once (measured: 14s for the first run, 0s afterwards).

Only `importance_sampling="adaptive"` breaks the property, since its tilt comes from a first-pass solve; the key then lists the parameters of that solve, so runs that differ in them are not silently compared on different scenarios.

## 6.5 How many scenarios, and how many assets?

**The count has to grow with the candidate set.** The approximation searches an  $N$ -dimensional space, so a fixed  $S$  buys less precision as  $N$  grows. Measured against the exact optimum of Section 6.7 (GBM model, four seeds):

| $N$ | $S$  | $S/N$ | turnover between two seeds | turnover to the optimum | sd of $\mathbb{E}[P\&L]$ |
|-----|------|-------|----------------------------|-------------------------|--------------------------|
| 30  | 512  | 17    | 9.6 %                      | 7.2 %                   | 17.6 CHF                 |
| 30  | 2048 | 68    | 3.5 %                      | 4.2 %                   | 2.8                      |
| 30  | 8192 | 273   | 2.6 %                      | 3.8 %                   | 1.1                      |
| 100 | 512  | 5     | 26.5 %                     | 21.5 %                  | 15.5                     |
| 100 | 2048 | 20    | 12.5 %                     | 10.0 %                  | 2.8                      |
| 100 | 8192 | 82    | 6.2 %                      | 6.2 %                   | 2.0                      |
| 300 | 512  | 1.7   | 22.7 %                     | 23.4 %                  | 18.5                     |
| 300 | 2048 | 6.8   | 9.0 %                      | 6.8 %                   | 4.2                      |

Two readings of the same table matter. The *reported figures* are stable everywhere: from 2000 scenarios on, the expected profit of the proposal varies by less than 0.1 % between seeds. The *composition* is what moves, and at fixed  $S$  it moves roughly in proportion to  $N$  (3.5 % at  $N = 30$  against 12.5 % at  $N = 100$ , both at  $S = 2048$ ) — with hundreds of candidates and a 10 % cap there are many portfolios within a fraction of a percent of the optimum, and the sample decides between them. About 64 scenarios per candidate keeps the disagreement near 5 %; as a default that is too expensive ( $N = 300$  would mean 32 768 scenarios and minutes per solve), so the implementation draws 16 per candidate within [4096, 16 384] and says in the result when the ratio is low. Where the composition itself has to be reproducible — a backtest, a teaching example — either raise  $S$  or use `scenario_model="gaussian_exact"`, which has no sampling error at all.

**Cost at the top end, and which solver.** A model with  $N = 300$  and  $S = 32\,768$  has 9.9M non-zeros: building it takes 3.3s, encoding it as msgpack 0.6s, and the 108MB body transfers to the cuOpt server at about 10 MB/s. Transport is therefore not the constraint; proving optimality is. Table 5 compares the two back ends with both running on the Spark, end to end through the service (scenario generation, build, solve, decode).

| case                                   | cuOpt (GPU)          | HiGHS (CPU, 4 threads) |
|----------------------------------------|----------------------|------------------------|
| $N = 30, S = 8\,192$                   | 15.6 s               | 14.5 s                 |
| $N = 30, S = 32\,768$                  | 171.4 s              | 171.0 s                |
| $N = 100, S = 4\,096$                  | 10.0 s               | 8.9 s                  |
| $N = 100, S = 32\,768$                 | 606.9 s (time limit) | 350.7 s (optimal)      |
| $N = 300, S = 8\,192$ , row generation | 139 s                | 64 s                   |

Table 5: The same models on the same machine, weekly horizon,  $\beta = 0.95$ , 600s solver time limit.

HiGHS is at least as fast in every case and twice as fast on the two largest. This is not a statement about GPU optimisation in general. cuOpt’s published speedups concern *linear* programmes solved with its PDLP first-order method; for mixed-integer problems its documentation states that the GPU runs primal heuristics while *branch and bound runs on the CPU*, that the MIP solver is in beta, and that proving feasible solutions optimal “remains under active development”. The models here have an easy LP relaxation and a small integrality gap, so nearly all the work is the optimality proof — the part that is not accelerated — whereas HiGHS warm-starts its dual simplex across branch-and-bound nodes and, with row generation, solves a sequence of small problems in-process while cuOpt pays a fixed cost per request. HiGHS is a demanding reference: in the Mittelmann benchmark of 7 July 2026 it solves 179 of 240 instances at a scaled shifted geometric mean of 5.44, against 1.72 for the commercial COPT and 9.93 for SCIP. Three caveats: cuOpt’s time limit covers only its internal solve, it ignores the MIP-gap settings sent over REST and may therefore prove optimality to a tighter tolerance, and this is one problem family on one machine rather than a benchmark.

## 6.6 Tail-row generation for large scenario sets

The MILP has one row and one variable  $u_s$  per scenario; with  $N + 2$  non-zeros per row the matrix grows linearly in  $S$ , but the simplex work does not: HiGHS’ dual simplex on the LP relaxation scales roughly like  $S^{1.9}$  and the branch-and-bound like  $S^{1.65}$ , cuOpt like  $S^{1.2}$ – $S^{1.6}$  (Table 6). Going from 2 000 to 100 000 scenarios costs about 45× on cuOpt and more than 200× on HiGHS.

| $S$       | non-zeros | HiGHS LP | HiGHS MILP | cuOpt MILP | payload | row generation (HiGHS) |
|-----------|-----------|----------|------------|------------|---------|------------------------|
| $N = 30$  |           |          |            |            |         |                        |
| 2 048     | 68 k      | 0.03 s   | 1.5 s      | 5.1 s      | 1.7 MB  | –                      |
| 8 192     | 270 k     | 0.4 s    | 14.5 s     | 10.8 s     | 6.6 MB  | 5.4 s (2 iterations)   |
| 32 768    | 1.1 M     | 6.0 s    | 166 s      | 66.6 s     | 27 MB   | 25 s (3 iterations)    |
| 100 000   | 3.3 M     | 56.5 s   | > 300 s    | 220 s      | 82 MB   | 158 s (2 iterations)   |
| $N = 100$ |           |          |            |            |         |                        |
| 2 048     | 211 k     | 0.1 s    | 9.5 s      | 7.0 s      | 5 MB    |                        |
| 8 192     | 844 k     | 1.2 s    | 50 s       | 11.5 s     | 21 MB   |                        |
| 32 768    | 3.4 M     | 16.5 s   | 232 s      | 101 s      | 82 MB   |                        |

Table 6: Solve times versus the number of scenarios (MIP gap  $10^{-4}$ ,  $\beta = 0.95$ ). The HiGHS column was measured on a development laptop whose SciPy bundles HiGHS 1.2.0 and the cuOpt column on the DGX Spark, so the two columns show the growth with  $S$  for each solver but are not a like-for-like comparison; Table 5 compares them on the same machine.

At the optimum only the scenarios in the tail have  $u_s > 0$ ; every other scenario row is slack. *Row*

generation [15] exploits this:

1. Start with a working set  $\mathcal{W}$ : the  $2(1 - \beta)S$  worst scenarios of a reference portfolio (the current holdings, or an equal-weight allocation).
2. Solve the MILP with the scenario rows of  $\mathcal{W}$  only, keeping the full-set probabilities  $p_s$  in (28) (omitted scenarios contribute  $u_s = 0$ ) and the full-set objective.
3. Compute the losses  $L_s$  of all  $S$  scenarios for the solution  $(n^*, \zeta^*)$ ; add every  $s \notin \mathcal{W}$  with  $L_s > \zeta^*$  to  $\mathcal{W}$  and repeat; stop when there is none.

The restricted problem drops constraints, so it is a relaxation and its objective bounds the full one from above; when no omitted scenario is violated the restricted solution is feasible for the full problem with  $u_s = 0$  on the omitted rows, hence optimal (within the MIP gap). The working set stays at about 11 % of  $S$  at  $\beta = 0.95$  (2–3 % at  $\beta = 0.99$ ), two or three iterations suffice, and the speed-up over the full MILP is 6–7 $\times$  from 32 768 scenarios on; only the working-set rows are sent to the solver. The default (`row_generation="auto"`) uses it from 8 192 scenarios on.

## 6.7 The Gaussian special case: mean–variance equivalence and an exact solution

**Proposition 6.1.** *Let the horizon returns be elliptically distributed with mean  $m$  and dispersion  $C$ , e.g.  $R \sim N(m, C)$ . Then for every portfolio  $\text{CVaR}_\beta(L(w)) = -m^\top w + k_\beta \sigma(w)$  with  $\sigma(w) = \sqrt{w^\top C w}$  and a constant  $k_\beta$  ( $k_\beta = \varphi(z_\beta)/(1 - \beta)$  in the normal case, 2.063 at 95 % and 2.665 at 99 %). Consequently the solution of “maximise  $m^\top w$  subject to  $\text{CVaR}_\beta \leq \Lambda$ ” lies on the mean–variance efficient frontier, and it is also the solution of “maximise  $m^\top w$  subject to  $\sigma(w) \leq \bar{\sigma}$ ” with the implicit bound  $\bar{\sigma} = (\Lambda + m^\top w^*)/k_\beta$ .*

*Proof.* For elliptical returns the loss of every portfolio is a location–scale transform of one and the same symmetric distribution, so every positively homogeneous, translation-invariant risk measure is an affine function of  $\sigma(w)$  [16]. Let  $w^*$  be mean–CVaR optimal with mean  $m^*$ . Any feasible  $w$  with  $m^\top w \geq m^*$  and  $\sigma(w) \leq \bar{\sigma}$  has  $\text{CVaR}_\beta \leq -m^* + k_\beta \bar{\sigma} = \Lambda$ , hence is mean–CVaR feasible and cannot beat  $m^*$ : the two problems have the same optimal value, and the same solutions up to ties. The same argument applies over the integer feasible set.  $\square$

The variance bound is only known *ex post* (it depends on  $m^*$ ), so the mean–CVaR problem with a Gaussian model is solved directly. In whole shares it is

$$\max_n m^\top v - TC \quad \text{s.t.} \quad g(n) := TC - m^\top v + k_\beta \sqrt{v^\top C v} \leq \Lambda, \quad v = P \circ n, \quad (29)$$

plus the linear constraints of Section 4: a convex mixed-integer second-order-cone programme (a mixed-integer QP in the utility form), which neither cuOpt nor HiGHS solves directly. The function  $g$  is convex, so it can be outer-approximated by tangent cuts on the same MILP [17, 18]. At an exposure  $v_k$  with  $\sigma_k = \sqrt{v_k^\top C v_k} > 0$  the gradient of  $\sigma$  is  $C v_k / \sigma_k$ , and because  $(C v_k)^\top v_k / \sigma_k = \sigma_k$  the tangent simplifies to

$$g(n) \geq TC + \left( k_\beta \frac{C v_k}{\sigma_k} - m \right)^\top v. \quad (30)$$

With an epigraph variable  $t$  (`cvar` in the code) the MILP carries the rows  $TC + (k_\beta C v_k / \sigma_k - m)^\top (P \circ n) - t \leq 0$  for the cuts collected so far and  $t \leq \Lambda$  (or  $t$  in the objective for `min_cvar` and `utility`). The loop starts with a cut at a reference portfolio, solves the MILP, compares the true  $g(n^*)$  with  $t^*$ , and adds a cut at  $v^*$  until they agree (relative tolerance  $10^{-6}$ ). Each cut removes the current infeasible point and the integer set is finite, so the loop terminates at the exact optimum within the MIP gap. On the universe it takes 20–70 cuts (each a MILP with  $N$  integers and a few dozen rows, milliseconds on HiGHS) and reproduces Gurobi’s direct MISOCP solution to the cent. This is `scenario_model="gaussian_exact"`: no scenarios, no sampling error, fully deterministic.

*Remark 6.2.* The GBM scenario model is lognormal in simple returns ( $R = e^r - 1$  with  $r$  normal), the exact mode is normal in simple returns. At weekly volatilities of 4–5 % the lognormal left tail is 2–3 % thinner, so the scenario optimum takes a few percent more exposure; the sharp test of the sampling machinery therefore uses Gaussian scenarios linearly, where moment-matched Sobol’ points reproduce the exact optimum within 1 % and plain Monte Carlo does not (`tests/test_rowgen_gaussian.py`). For the empirical innovation model no closed form exists — that is the point of using it — and the equivalence serves only as a calibration of the sampling mechanism.

## 6.8 Rank of the covariance estimate

An estimator with  $T_{\text{eff}}$  effective observations cannot support more than  $T_{\text{eff}}$  assets: a 200-day window with  $N = 300$  gives a sample covariance of rank 200, and a portfolio in its null space has *zero* estimated variance — an optimiser maximising return under a risk cap will find it. The shrinkage of the measurement note repairs exactly this, and its data-driven intensity rises with  $N/T$  on its own:

| estimator, $N$                   | $T_{\text{eff}}$ | intensity | $\lambda_{\min}/\overline{\sigma^2}$ | condition          | zero eigenvalues |
|----------------------------------|------------------|-----------|--------------------------------------|--------------------|------------------|
| rolling 200, $N = 300$           | 200              | 0         | $-3.4 \cdot 10^{-15}$                | $4 \cdot 10^{298}$ | 101              |
| rolling 200, $N = 300$           | 200              | 0.080     | 0.023                                | $3.0 \cdot 10^3$   | 0                |
| rolling 200, $N = 745$           | 200              | 0         | $-2.0 \cdot 10^{-14}$                | $6 \cdot 10^{298}$ | 546              |
| rolling 200, $N = 745$           | 200              | 0.128     | 0.025                                | $4.3 \cdot 10^3$   | 0                |
| EWM $\lambda = 0.99$ , $N = 745$ | 300              | 0         | $2.2 \cdot 10^{-5}$                  | $5.5 \cdot 10^6$   | 44 (numerically) |
| EWM $\lambda = 0.99$ , $N = 745$ | 300              | 0.114     | 0.023                                | $4.8 \cdot 10^3$   | 0                |

With `shrinkage="auto"` — the default everywhere — the smallest eigenvalue never falls below 2 % of the average variance, whatever the number of candidates. Without it the degeneracy is real and silent: a null-space portfolio comes out at variance  $-1.6 \cdot 10^{-20}$  and the Sobol’ draws span only 222 of 300 dimensions. `CovarianceModel.conditioning` reports the figures and the optimisation warns when the estimate is singular or ill-conditioned for the size of the candidate set.

Two things soften the problem in this particular model, and neither should be relied on. An EWM estimator over the full history is formally full rank (every observation keeps a positive weight), so its degeneracy is numerical rather than exact; and no-short-selling with a 10 % cap makes null directions hard to reach, since exploiting one generally requires a short leg. Measured across  $N = 30 \dots 745$ , the optimised portfolio’s out-of-sample-to-in-sample volatility ratio exceeds the cap-weighted portfolio’s by a factor of only 0.89–1.18, with or without shrinkage — the optimiser is not systematically mining the estimate. The exactly singular case does bite elsewhere, though: with a 200-day window and  $N \geq 300$  the outer approximation of Section 6.7 stops converging and returns a time-limited solution.

## 6.9 Reproducibility

Two runs of the same request use the same scenario set, and so do two runs that differ only in the parameters listed on the right of Section 6.4: the base sample depends only on the seed (scrambled Sobol’ points and the PCG64 generator are deterministic across platforms), the transformations are deterministic, and the result reports a fingerprint of the final matrix and its probabilities (`scenario_hash`). The fingerprint is exact on one platform; between BLAS implementations (laptop versus the DGX Spark) the GBM sets agree to  $10^{-11}$  and keep the same hash, while the empirical model’s innovation filter — thousands of eigendecompositions — can differ at the  $10^{-10}$  level, which changes the hash but none of the reported figures. Whether the *solution* is the same depends on the solver. HiGHS’ branch-and-bound is deterministic: identical inputs give

identical shares (`reproducibility="exact"`) as long as the HiGHS build is the same — across SciPy versions the search may end on a different optimal vertex among ties, at the same objective. `cuOpt` combines GPU heuristics with a time-limited search and, in version 26.08, ignores the MIP-gap settings sent over its REST interface; it proves optimality within its default relative gap of  $10^{-4}$ , so the objective, expected profit and CVaR of two runs agree to that gap (about 0.2 CHF per week on a CHF 1 million portfolio) while near-tied share counts can differ by a few shares (`reproducibility="within_gap"`). A result that hit the time limit (`FeasibleFound`) is not reproducible at all. The Gaussian exact mode runs on HiGHS by default and is exact in this sense.

## 7 The complete model as implemented

`MeanCVaRProblem.build()` assembles the following MILP from a `ProblemSpec`. The variable vector is laid out as  $x = (n, [b, s], \alpha, u, [y], [t])$ ; blocks in brackets exist only in rebalancing mode ( $b, s$ ), with cardinality/minimum-position rules ( $y$ ) or with a fixed cost per trade ( $t$ ). The CVaR level  $\zeta$  is called  $\alpha$  in the code.

| block             | name in the code          | type, bounds              | meaning                                        |
|-------------------|---------------------------|---------------------------|------------------------------------------------|
| $n_i$             | <code>n[symbol]</code>    | integer, $[0, \bar{n}_i]$ | final number of shares                         |
| $b_i$             | <code>buy[symbol]</code>  | integer, $[0, \bar{n}_i]$ | shares bought (rebalancing)                    |
| $s_i$             | <code>sell[symbol]</code> | integer, $[0, n_i^0]$     | shares sold (rebalancing)                      |
| $\alpha$          | <code>alpha</code>        | continuous, free          | the $\zeta$ of Theorem 2.2                     |
| $u_s$             | <code>u[s]</code>         | continuous, $[0, \infty)$ | excess loss of scenario $s$ over $\alpha$      |
| $t_{\text{cvar}}$ | <code>cvar</code>         | continuous, free          | Gaussian CVaR epigraph (Section 6.7)           |
| $y_i$             | <code>y[symbol]</code>    | binary                    | position open (cardinality / minimum position) |
| $t_i$             | <code>t[symbol]</code>    | binary                    | stock traded (fixed cost)                      |

With  $\bar{n}_i = \lfloor \min(\bar{w}, f_{\max}) W / P_i \rfloor$ ,  $g_{s,i} = P_i R_{s,i}$ ,  $\bar{g}_i = P_i \mu_i$  (the model's expected simple return over the horizon; the probability-weighted scenario mean if no model mean is supplied),  $\kappa_s = \frac{p_s}{1-\beta}$  with  $p_s = 1/S$  unless the scenarios carry importance-sampling probabilities, and  $TC = \sum_i (c^b P_i b_i + c^s P_i s_i) + F \sum_i t_i$  (in mode `new`:  $c^b \sum_i P_i n_i + F \sum_i t_i$ ), the rows are:

| row name                 | constraint                                                                                 | count                         | origin            |
|--------------------------|--------------------------------------------------------------------------------------------|-------------------------------|-------------------|
| <code>cvar_scen_s</code> | $\sum_i g_{s,i} n_i - TC + \alpha + u_s \geq 0$                                            | $S$                           | (9), (23)         |
| <code>cvar_limit</code>  | $\alpha + \sum_s \kappa_s u_s \leq \Lambda$<br>(Gaussian: $t_{\text{cvar}} \leq \Lambda$ ) | 1, if a limit is given        | (7), (28)         |
| <code>cut_j</code>       | $TC + (k_\beta C v_j / \sigma_j - m)^\top (P \circ n) - t_{\text{cvar}} \leq 0$            | one per cut,<br>Gaussian mode | (30)              |
| <code>budget</code>      | $\sum_i P_i b_i - \sum_i P_i s_i + TC \leq c^0$<br>(new: $\sum_i P_i n_i + TC \leq c^0$ )  | 1                             | (20)              |
| <code>link_i</code>      | $n_i - b_i + s_i = n_i^0$                                                                  | $N$ , rebalancing             | (18)              |
| <code>invested</code>    | $f_{\min} W \leq \sum_i P_i n_i \leq f_{\max} W$                                           | $\leq 1$ , if bounds given    | invested fraction |
| <code>sector_k</code>    | $\sum_{i \in G_k} P_i n_i \leq \bar{w}_k W$                                                | $K$ , binding sectors         | (17e)             |
| <code>max_cost</code>    | $TC \leq TC_{\max}$                                                                        | $\leq 1$                      | (24)              |
| <code>turnover</code>    | $\sum_i P_i (b_i + s_i) \leq \tau W$                                                       | $\leq 1$ , rebalancing        | (24)              |
| <code>open_i</code>      | $n_i - \max(\bar{n}_i, 1) y_i \leq 0$                                                      | $N$ , optional                | (25)              |
| <code>minpos_i</code>    | $P_i n_i - v_{\min} y_i \geq 0$                                                            | $N$ , optional                | (25)              |
| <code>cardinality</code> | $\sum_i y_i \leq K_{\max}$                                                                 | $\leq 1$ , optional           | (25)              |
| <code>trade_i</code>     | $b_i + s_i - M_i t_i \leq 0$<br>(new: $n_i - M_i t_i \leq 0$ )                             | $N$ , fixed cost only         | (19)              |
| <code>target_pnl</code>  | $\sum_i \bar{g}_i n_i - TC \geq \rho$                                                      | 1, <code>min_cvar</code> only | (12)(b)           |

The objective is one of

$$\text{max\_return} : \quad \max \sum_i \bar{g}_i n_i - TC \quad \text{subject to } \text{cvar\_limit}, \quad (31)$$

$$\text{min\_cvar} : \quad \min \alpha + \sum_s \kappa_s u_s \quad \text{subject to } \text{target\_pnl}, \quad (32)$$

$$\text{utility} : \quad \max \sum_i \bar{g}_i n_i - TC - \lambda \left( \alpha + \sum_s \kappa_s u_s \right), \quad (33)$$

the three forms of (12) (in Gaussian mode  $\alpha + \sum_s \kappa_s u_s$  is replaced by  $t_{\text{cvar}}$ ). All coefficients are in CHF; the scenario rows carry  $N(+2N) + 2$  non-zeros each, so the matrix has about  $S(N + 2)$  non-zeros (420 k for  $N = 100$ ,  $S = 4096$ ; 3 M for the full universe of 745 stocks with  $S = 4096$ ). Duplicate index pairs that arise when a variable appears in several terms of a row (e.g.  $n_i$  in the gain and in the cost of mode **new**) are summed when the sparse matrix is assembled.

**The two solvers.** The same `MILPModel` is accepted by both back ends. *cuOpt* is NVIDIA’s GPU optimisation library, reached over HTTP as a self-hosted server (version 26.08 here) running as a container on the DGX Spark; it is the only component that uses the GPU. *HiGHS* is the open-source LP/MILP solver of ERGO-Code at the University of Edinburgh (MIT licence, <https://highs.dev>) [7]; it is not installed or started separately but ships compiled inside SciPy, so `scipy.optimize.milp` runs it *in the calling process on the CPU* — in production inside the same container as the service (SciPy 1.18.1, HiGHS 1.12.0, four threads). It serves as the fallback when *cuOpt* is unreachable, as the solver of the small cut MILPs of the Gaussian mode, and as the oracle of the test suite.

**Solving and decoding.** The model is sent to the *cuOpt* server as an LP/MILP request (CSR matrix, bounds, variable types, a mandatory time limit; msgpack-encoded) and polled until a solution arrives; if the server is unreachable the same `MILPModel` is handed to HiGHS. From 8192 scenarios on the service solves by row generation (Section 6.6), i.e. a sequence of such MILPs on the working set; in Gaussian mode by the outer-approximation loop (Section 6.7). A status **Optimal** means proven optimality within the solver’s relative gap; **FeasibleFound** means the time limit stopped the search at an incumbent whose `mip_gap` is reported. The result states the scenario fingerprint and the level of reproducibility (Section 6.9). From the solution vector, `MeanCVaRProblem.decode()` rounds the integer blocks (they are integral up to solver tolerance), rebuilds trades, costs, cash, position values and weights, and *recomputes* the expected profit (from the model means), VaR and CVaR of the decoded portfolio from the weighted scenario set (Remark 2.3) or, in Gaussian mode, from the closed form; these recomputed figures are what the service reports, together with the provenance of every model input (covariance model, scenario model and seed, expected returns and views, constraints, costs, solver statistics).

**What the model does not do.** It assumes the scenario set represents the horizon distribution and that all trades execute at today’s prices; it holds the portfolio unchanged over the horizon (no intra-horizon rebalancing); cash earns nothing over the horizon (the CHF risk-free rate is zero); it is single-period. These are deliberate simplifications of a teaching platform, not limitations of the formulation — multi-period extensions replace the scenario rows by a scenario tree.

## References

- [1] R. T. Rockafellar and S. Uryasev, *Optimization of conditional value-at-risk*, Journal of Risk 2(3), 21–41, 2000.

- [2] R. T. Rockafellar and S. Uryasev, *Conditional value-at-risk for general loss distributions*, Journal of Banking & Finance 26(7), 1443–1471, 2002.
- [3] P. Krokmal, J. Palmquist and S. Uryasev, *Portfolio optimization with conditional value-at-risk objective and constraints*, Journal of Risk 4(2), 43–68, 2002.
- [4] C. Acerbi and D. Tasche, *On the coherence of expected shortfall*, Journal of Banking & Finance 26(7), 1487–1503, 2002.
- [5] M. S. Lobo, M. Fazel and S. Boyd, *Portfolio optimization with linear and fixed transaction costs*, Annals of Operations Research 152, 341–365, 2007.
- [6] H. Kellerer, R. Mansini and M. G. Speranza, *Selecting portfolios with fixed costs and minimum transaction lots*, Annals of Operations Research 99, 287–304, 2000.
- [7] Q. Huangfu and J. A. J. Hall, *Parallelizing the dual revised simplex method*, Mathematical Programming Computation 10, 119–142, 2018.
- [8] A. Shapiro, D. Dentcheva and A. Ruszczyński, *Lectures on Stochastic Programming: Modeling and Theory*, SIAM, Philadelphia, 2009.
- [9] R. O. Michaud, *The Markowitz optimization enigma: is ‘optimized’ optimal?*, Financial Analysts Journal 45(1), 31–42, 1989.
- [10] A. B. Owen, *Scrambling Sobol’ and Niederreiter–Xing points*, Journal of Complexity 14(4), 466–489, 1998.
- [11] P. Glasserman, *Monte Carlo Methods in Financial Engineering*, Springer, New York, 2004.
- [12] K. Høyland and S. W. Wallace, *Generating scenario trees for multistage decision problems*, Management Science 47(2), 295–307, 2001.
- [13] P. Glasserman, P. Heidelberger and P. Shahabuddin, *Variance reduction techniques for estimating value-at-risk*, Management Science 46(10), 1349–1364, 2000.
- [14] T. Hesterberg, *Weighted average importance sampling and defensive mixture distributions*, Technometrics 37(2), 185–194, 1995.
- [15] A. Küenzi-Bay and J. Mayer, *Computational aspects of minimizing conditional value-at-risk*, Computational Management Science 3(1), 3–27, 2006.
- [16] P. Embrechts, A. J. McNeil and D. Straumann, *Correlation and dependence in risk management: properties and pitfalls*, in: M. Dempster (ed.), Risk Management: Value at Risk and Beyond, Cambridge University Press, 176–223, 2002.
- [17] J. E. Kelley, *The cutting-plane method for solving convex programs*, Journal of the SIAM 8(4), 703–712, 1960.
- [18] M. A. Duran and I. E. Grossmann, *An outer-approximation algorithm for a class of mixed-integer nonlinear programs*, Mathematical Programming 36, 307–339, 1986.
- [19] R. Mansini, W. Ogryczak and M. G. Speranza, *Twenty years of linear programming based portfolio optimization*, European Journal of Operational Research 234(2), 518–535, 2014.