

Package: WeightCraft (via r-universe)

May 26, 2026

Title Portfolio Choice: Estimation, Construction, and Evaluation

Version 1.0.0

Description Providing quantitative tools for input estimation,
portfolio construction, and performance evaluation.

License GPL-3

Encoding UTF-8

RoxygenNote 7.3.3

Depends R (>= 4.0.0)

Imports quadprog, stats

NeedsCompilation no

Author Pavlos Pantatosakis [aut, cre], Georgios Skoulakis [aut]

Maintainer Pavlos Pantatosakis <pantatosakisp@yahoo.com>

Repository <https://vlospa99.r-universe.dev>

Date/Publication 2026-03-26 11:16:20 UTC

RemoteUrl <https://github.com/cran/WeightCraft>

RemoteRef HEAD

RemoteSha f44e88bf75322b9f364ced4a3626b27a8285a487

Contents

EWMA	2
PerformanceMeasures	3
solve_cwls	6
solve_mean_variance	7
WeightCraft	10

Index	12
--------------	-----------

EWMA

*Exponentially Weighted Moving Average***Description**

This function calculates an exponentially weighted moving average vector, and an exponentially weighted moving covariance matrix for a given numeric matrix or data.frame. It is designed for applications where recent observations receive higher weights than past ones.

Usage

```
EWMA(x, lambda, center = TRUE)
```

Arguments

x	A numeric matrix or data.frame. Rows represent time periods, columns represent variables.
lambda	A single numeric value in the interval (0, 1] representing the decay factor. When lambda = 1, all observations receive equal weight.
center	Logical; if TRUE (default), the weighted mean is subtracted from the observations before computing the covariance matrix. If FALSE, the covariance matrix is computed around zero.

Details

The weight for each time t is calculated as:

$$w_t = \frac{1 - \lambda}{1 - \lambda^T} \lambda^{T-t}$$

When lambda = 1, all weights equal $1/T$ and the function returns the equal-weighted mean and covariance. The weighted mean vector $\hat{\mu}^w$ is:

$$\hat{\mu}^w = \sum_{t=1}^T w_t x_t$$

When center = FALSE, $\hat{\mu}^w$ is set to zero and the covariance matrix is computed around zero instead.

The weighted covariance matrix $\hat{\Sigma}$ is:

$$\hat{\Sigma} = \sum_{t=1}^T w_t (x_t - \hat{\mu}^w)(x_t - \hat{\mu}^w)'$$

Value

A list with three elements:

- `weights`: A numeric vector of length T containing the exponential weights, ordered from oldest (smallest weight) to most recent (largest weight). $\sum_{t=1}^T w_t = 1$
- `EWMA_mu`: A numeric vector of length p containing the weighted mean of each column of x . If `center = FALSE`, this is a zero vector.
- `EWMA_sigma`: A $p \times p$ numeric matrix containing the weighted covariance matrix.

Author(s)

Pavlos Pantatosakis <pantatosakis@yahoo.com>, Georgios Skoulakis <georgios.skoulakis@unipi.gr>

References

J.P.Morgan/Reuters (Longerstaey, J., & Spencer, M., 1996). Riskmetrics—Technical Document.
 Pafka, S., Potters, M., & Kondor, I. (2004). Exponential weighting and random-matrix-theory-based filtering of financial covariance matrices for portfolio optimization. arXiv preprint cond-mat/0402573.

Examples

```
y <- matrix(rnorm(600, 0, 1), ncol = 3)
colnames(y) <- c("X1", "X2", "X3")
example1 <- EWMA(y, 0.97)
print(example1$EWMA_mu) # Weighted Mean
print(example1$EWMA_sigma) # Weighted Covariance Matrix

# Weights sum to 1 and decrease exponentially over time.
sum(example1$weights) ; ts.plot(example1$weights)

#Equal Weight
example2 <- EWMA(y, 1)
print(example2$EWMA_mu) # Mean
print(example2$EWMA_sigma) # Covariance Matrix

sum(example2$weights) ; ts.plot(example2$weights)
```

Description

This function computes various performance measures for one or more return series. It is designed for evaluating and comparing different investment strategies.

Usage

PerformanceMeasures(R, rf, risk_aversion = 3, FREQ = "Monthly")

Arguments

R	A numeric matrix or data.frame with the returns of each strategy. Rows represent time periods, and columns represent different strategies or assets.
rf	A numeric scalar or a vector representing the risk-free rate. If a vector, its length must match the number of rows in R.
risk_aversion	A single positive numeric value representing the coefficient of risk aversion (γ). Default is 3.
FREQ	The frequency of the data. Can be a character string: 'Annual', 'Monthly', 'Weekly', or 'Daily', or a numeric value (1 for Annual, 12 for Monthly, 52 for Weekly, and 252 for Daily). Default is 'Monthly'.

Details

The function computes annualized metrics based on the specified frequency f . Let r_t be the return and $r_{f,t}$ be the risk-free rate at time t . The performance measures are defined as follows:

- **Certainty Equivalent Return (CER):**

$$CER = f \times (\hat{\mu} - 0.5\gamma\hat{\sigma}^2)$$

where $\hat{\mu} = \bar{r} = \frac{1}{T} \sum_{t=1}^T r_t$ and $\hat{\sigma}^2 = \frac{1}{T} \sum_{t=1}^T (r_t - \bar{r})^2$.

- **Compound Annual Growth Rate (CAGR):**

$$CAGR = \left(\prod_{t=1}^T (1 + r_t) \right)^{f/T} - 1$$

- **Risk Premium (Annualized Excess Returns):**

$$Premium = f \times \frac{1}{T} \sum_{t=1}^T (r_t - r_{f,t})$$

- **Annualized Volatility:**

$$Volatility = \sqrt{f} \times \sqrt{\frac{1}{T} \sum_{t=1}^T (x_t - \bar{x})^2}$$

where $x_t = r_t - r_{f,t}$.

- **Sharpe Ratio:**

$$Sharpe = \frac{Premium}{Volatility}$$

- **Sortino Ratio:**

$$Sortino = \frac{Premium}{\sigma_{DS}}$$

where $\sigma_{DS} = \sqrt{f \times \frac{1}{T} \sum_{t=1}^T \min(x_t, 0)^2}$.

- **Maximum Drawdown (MaxDD):**

$$MaxDD = \left| \min \left(\frac{W_t}{RMax_t} - 1 \right) \right|$$

where $W_t = \prod_{s=1}^t (1+r_s)$ is the cumulative wealth series, and $RMax_t = \max\{W_1, \dots, W_t\}$.

- **Terminal Wealth:**

$$Wealth = \prod_{t=1}^T (1 + r_t)$$

Value

A data.frame where each row corresponds to a strategy. The columns CER, CAGR, Premium, Volatility, and MaxDD are returned as percentages (multiplied by 100).

Author(s)

Pavlos Pantatosakis <pantatosakis@yahoo.com>, Georgios Skoulakis <georgios.skoulakis@unipi.gr>

Examples

```
set.seed(123)

# Time-varying risk-free rate, monthly data, default risk aversion = 3
example1 <- matrix(rnorm(120, 0.01, 0.04), ncol = 2)
colnames(example1) <- c("Strategy_A", "Strategy_B")
rf_vec <- rnorm(60, 0.005, 0.001)
perf1 <- PerformanceMeasures(example1, rf = rf_vec, FREQ = "Monthly")
print(perf1)

# Constant risk-free rate, annual data, risk aversion = 4
example2 <- matrix(rnorm(100, 0.10, 0.15), ncol = 2)
colnames(example2) <- c("Strategy_C", "Strategy_D")
perf2 <- PerformanceMeasures(example2, rf = 0.02, risk_aversion = 4, FREQ = "Annual")
print(perf2)
```

 solve_cwls

 Constrained Weighted Least Squares (CWLS)

Description

This function performs a Weighted Least Squares regression with optional sign constraints on the slope coefficients. It is useful for Return-Based Style Analysis (RBSA) where factor exposures are required to be non-negative or non-positive.

Usage

```
solve_cwls(y, X, w, slope_sign_constraint, intercept = TRUE)
```

Arguments

<code>y</code>	A numeric vector of the dependent variable.
<code>X</code>	A numeric matrix or data.frame of independent variables.
<code>w</code>	A numeric vector of non-negative weights for the observations.
<code>slope_sign_constraint</code>	A numeric vector of length $\text{ncol}(X)$. Use 1 for $\beta \geq 0$, -1 for $\beta \leq 0$, and 0 for no constraint. The intercept (if included) is always unconstrained.
<code>intercept</code>	Logical; if TRUE (default), an intercept term is added to X.

Details

The function minimizes the weighted sum of squared residuals:

$$\min_{\beta} \sum_{t=1}^T w_t (y_t - x_t^T \beta)^2$$

subject to the sign constraints specified in `slope_sign_constraint`.

If `intercept = TRUE`, an intercept column is prepended to the design matrix. The intercept is always unconstrained. If `intercept = FALSE`, the design matrix is used as supplied and constraints are applied to all columns as indexed.

The optimization is solved via Quadratic Programming:

$$\min_{\beta} \left(\frac{1}{2} \beta^T (X^T W X) \beta - (X^T W y)^T \beta \right)$$

where W is a diagonal matrix of weights.

Value

A list containing:

- `coefficients`: Vector of estimated constrained coefficients.
- `fitted`: The fitted values (\hat{y}).
- `residuals`: The residuals ($y - \hat{y}$).
- `weighted_r_squared`: The weighted R-squared of the fit.
- `obj_value`: The value of the quadratic objective function at the solution.

Author(s)

Pavlos Pantatosakis <pantatosakis@yahoo.com>, Georgios Skoulakis <georgios.skoulakis@unipi.gr>

Examples

```
# Simulate fund returns and 3 factor benchmarks
set.seed(123)
x_factors <- matrix(rnorm(300), ncol = 3)
colnames(x_factors) <- c("Market", "Value", "Size")
y_fund <- 0.5 * x_factors[,1] + 0.2 * x_factors[,2] + rnorm(100, 0, 0.1)

# Define weights (e.g., more weight on recent data)
obs_weights <- seq(0.1, 1, length.out = 100)

# Constraint: all slopes must be non-negative
constraints <- c(1, 1, 1)

fit <- solve_cwls(y_fund, x_factors, obs_weights, constraints)
print(fit$coefficients)

# Unconstrained slope signs reduces to base R lm().
fit_uncon <- solve_cwls(y_fund, x_factors, obs_weights, rep(0, 3))
all.equal(as.numeric(fit_uncon$coefficients),
          as.numeric(coef(lm(y_fund ~ x_factors, weights = obs_weights))))

# Unconstrained slope signs with equal weights reduces to Ordinary Least Squares (OLS).
fit_ols <- solve_cwls(y_fund, x_factors, rep(1,100), rep(0,3))
fit_ols$coefficients
coef(lm(y_fund ~ x_factors))
```

Description

Solves a mean-variance optimization problem to find the optimal risky asset weights that maximize a quadratic utility function, subject to individual asset bounds and risk-free asset allocation constraints. The risk-free asset absorbs any wealth not invested in risky assets, with its weight determined implicitly as the residual of the risky weights.

Usage

```
solve_mean_variance(
    mu,
    Sigma,
    risky_lb,
    risky_ub,
    rf_lb,
    rf_ub,
    risk_aversion = 3
)
```

Arguments

mu	A numeric vector of length N representing the expected excess returns of the risky assets (i.e., simple returns minus the risk-free rate)
Sigma	A symmetric positive definite $N \times N$ numeric covariance matrix of the risky asset returns.
risky_lb	A numeric vector of length N specifying the lower bounds for each risky asset weight. Set to 0 to disallow short selling.
risky_ub	A numeric vector of length N specifying the upper bounds for each risky asset weight.
rf_lb	A single numeric value specifying the minimum weight of the risk-free asset. Must satisfy $rf_lb \leq rf_ub$.
rf_ub	A single numeric value specifying the maximum weight of the risk-free asset.
risk_aversion	A single positive numeric value representing the investor's risk aversion coefficient (γ). Default is 3.

Details**Utility Function**

The investor maximizes a mean-variance utility defined over the full portfolio, including the risk-free asset. Let w be the $N \times 1$ vector of risky weights. The risk-free weight is the implicit residual:

$$w_{rf} = 1 - \mathbf{1}^\top w$$

The full portfolio utility is:

$$U(w) = r_f \cdot (1 - \mathbf{1}^\top w) + w^\top \hat{\mu} - \frac{\gamma}{2} w^\top \hat{\Sigma} w$$

Expanding the risk-free term:

$$U(w) = r_f + w^\top (\hat{\mu} - r_f \cdot \mathbf{1}) - \frac{\gamma}{2} w^\top \hat{\Sigma} w$$

Since $\hat{\mu}$ is supplied as **excess returns** and r_f is a constant that does not affect the optimizer solution, the effective objective passed to the solver is:

$$\max_w \left\{ w^\top \hat{\mu} - \frac{\gamma}{2} w^\top \hat{\Sigma} w \right\}$$

QP Mapping

Internally, the problem is mapped to the quadprog : : solve.QP minimization framework:

$$\min_w \left(\frac{1}{2} w^\top D w - d^\top w \right)$$

where $D = \gamma \hat{\Sigma}$ and $d = \hat{\mu}$.

Constraints

- **Asset-specific bounds:**

$$lb_i \leq w_i \leq ub_i, \quad \forall i \in \{1, \dots, N\}$$

- **Risk-free lower bound:**

$$1 - \mathbf{1}^\top w \geq rf_{lb} \iff -\mathbf{1}^\top w \geq rf_{lb} - 1$$

- **Risk-free upper bound:**

$$1 - \mathbf{1}^\top w \leq rf_{ub} \iff \mathbf{1}^\top w \geq 1 - rf_{ub}$$

Value

A list containing:

- **weights:** Full weight vector for all assets, including the risk-free asset.
- **risky_weights:** Vector of optimal risky asset weights w^* .
- **rf_weight:** Scalar weight allocated to the risk-free asset: $1 - \mathbf{1}^\top w^*$.
- **expected_return:** Expected **excess** return of the risky portfolio $w^\top \hat{\mu}$.
- **variance:** Portfolio variance $w^\top \hat{\Sigma} w$.
- **volatility:** Portfolio volatility $\sqrt{w^\top \hat{\Sigma} w}$.

Author(s)

Pavlos Pantatosakis <pantatosakis@yahoo.com>, Georgios Skoulakis <georgios.skoulakis@unipi.gr>

Examples

```
# 1. Define Inputs (3 Assets: Stocks, Bonds, Real Estate)
# Note: mu must be excess returns (raw return - rf), computed upstream
example_mu <- c(Stocks = 0.08, Bonds = 0.04, RealEstate = 0.06)

example_Sigma <- matrix(c(0.040, 0.005, 0.015,
                          0.005, 0.010, 0.002,
                          0.015, 0.002, 0.025),
                        nrow = 3, byrow = TRUE)
colnames(example_Sigma) <- rownames(example_Sigma) <- names(example_mu)

# 2. Set Constraints
# No short-selling (lb = 0), max 60% in any single risky asset (ub = 0.6)
example_lb <- rep(0, 3)
example_ub <- rep(0.6, 3)

# Risk-free asset must be between 10% and 30% of total wealth
example_rf_low <- 0.10
example_rf_high <- 0.30

# 3. Solve for gamma = 3
example_result <- solve_mean_variance(example_mu,
                                      example_Sigma,
                                      example_lb,
                                      example_ub,
                                      example_rf_low,
                                      example_rf_high,
                                      risk_aversion = 3)

# 4. View Results
print(example_result$weights)
cat("Portfolio Volatility:", round(example_result$volatility, 4), "\n")
```

WeightCraft

WeightCraft: Portfolio Choice Problems - Estimation, Construction, and Evaluation

Description

The **WeightCraft** package provides functions for quantitative portfolio management, covering return and risk estimation, mean-variance optimization, and performance evaluation.

Details

Package: WeightCraft
 Type: Package
 Version: 1.0.0

Date: 2026-03-22
License: GPL-3

Maintainers

Pavlos Pantatosakis <pantatosakisp@yahoo.com>

Author(s)

Pavlos Pantatosakis <pantatosakisp@yahoo.com>, Georgios Skoulakis <georgios.skoulakis@unipi.gr>

Index

EWMA, [2](#)

PerformanceMeasures, [3](#)

solve_cwls, [6](#)

solve_mean_variance, [7](#)

WeightCraft, [10](#)

WeightCraft-package (WeightCraft), [10](#)