UPPSALA UNIVERSITET

A TIME-VARYING FACTOR APPROACH TO COVARIANCE ESTIMATION

Submitted by
Erik Lillrank

A thesis submitted to the Department of Statistics in partial fulfillment of the requirements for the two-year degree of Master of Arts in Statistics in the Faculty of Social Sciences

Supervisor
Yukai Yang

Spring, 2025

ABSTRACT

Financial volatility changes continuously, yet most portfolio optimization strategies rely on static covariance estimates. This thesis introduces a time-dependent covariance matrix that updates for each time period via kernel-weighted local principal components which is applied in a *time-varying minimum-variance portfolio* (TV-MVP). The full workflow is implemented in the open-source R package `TVMVP`. The package also includes a test for constant factor loadings, evaluated through Monte Carlo simulations across six data-generating processes. Results confirm its statistical reliability: the test accepts constant loadings in over 80% of runs and consistently rejects time-varying structures. Empirically, TV-MVP is applied to 50-250 Swedish stocks and benchmarked against sample, Ledoit-Wolf, EWMA, POET, and graphical-lasso covariance estimates. In calm markets (2017-2019), TV-MVP achieves competitive risk levels and returns. In the volatile 2022-2024 window, it delivers the lowest or second-lowest standard deviation across all asset pools while preserving competitive Sharpe ratios and drawdowns. TV-MVP therefore offers a practical, low-volatility alternative with accessible implementation through its accompanying software package.

Keywords: R-Package, Modern Portfolio Theory, Time-Varying Portfolio Optimization, Minimum Variance Portfolio

Contents

1 Introduction

Constructing an optimal portfolio is a complex and inherently subjective task. In practice, the term *optimal portfolio* is often used to denote the portfolio that maximizes returns for a given level of risk. In this context, risk is typically decomposed into two components: systematic risk, which affects the entire market (e.g., changes in interest rates or inflation), and idiosyncratic risk, which is unique to individual assets, companies, or industries. While systematic risk cannot be diversified away, idiosyncratic risk can be mitigated by constructing a diversified portfolio that minimizes inter-asset correlations (Campbell et al., 2001).

There exists a wide array of methods to achieve an optimal trade-off between risk and return. Modern Portfolio Theory (MPT), or the mean-variance framework introduced by Markowitz (1952), is one of the most influential approaches to this problem. In the mean-variance setting, the portfolio optimization problem is formulated as:

$$\begin{aligned} \max_{\mathbf{w}} \quad & \mathbf{w}'\boldsymbol{\mu} - \frac{\lambda}{2}\mathbf{w}'\boldsymbol{\Sigma}_r\mathbf{w}, \\ \text{s.t.} \quad & \mathbf{w}'\mathbf{1}_p = 1, \end{aligned} \tag{1}$$

where $\mathbf{w}$ is a $p \times 1$ vector of asset weights, $\boldsymbol{\Sigma}_r$ is the covariance matrix of the asset returns, $\mathbf{1}_p$ is a $p \times 1$ vector of 1's, λ is a parameter reflecting the investor's risk aversion (Palomar, 2025, Sec. 7.1.2). Although the original framework assumes portfolio weights form a convex combination of assets, the solution may be extended to allow affine combinations when short selling is permitted.

A key limitation of the classical mean-variance portfolio is its reliance on precise estimates of $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}_r$. In practice, estimates of expected returns are notoriously noisy, often leading to suboptimal performance (Chopra and Ziemba, 1993, Michaud, 1989). Furthermore, the assumption of a static covariance matrix, $\boldsymbol{\Sigma}_r$, is overly restrictive given that the relationships between assets evolve over time due to shifts in economic conditions, structural breaks, and regime changes (e.g. R. F. Engle et al., 2001 Bollerslev et al., 1988 & Pelletier, 2006). An approach that dynamically captures these shifts would allow for more accurate volatility estimation and, consequently, better portfolio construction.

This thesis aims to develop a method for estimating a time-dependent covariance matrix, $\boldsymbol{\Sigma}_{r,t}$, to construct optimal portfolios under evolving market conditions. In the literature, several methods have been proposed to incorporate the time dimension in volatility and covariance estimation. Examples include multivariate GARCH models that capture the time varying volatility and covariance structures between multiple assets simultainiously (Bollerslev et al., 1988, NG, 1991), the Dynamic Conditional Correlation (DCC) models introduced by R. Engle (2002) which extends GARCH to capture dynamic correlations, and Exponentially Weighted Moving Average (EWMA) models that downweight older observations (Longerstaey and Spencer, 1996).

In this thesis, a time-varying factor model as proposed by Su and Wang (2017) is implemented to estimate $\Sigma_{r,t}$. This approach is closely related to the methods described in Q. Fan et al. (2024) and Wang et al. (2021), both of which utilize time-varying factor models to estimate covariance matrices for constructing Minimum Variance Portfolios (MVP). Although their results are promising both in simulation studies and when applied to the empirical data, the practical implementation of these models is challenging, which limits their widespread use by practitioners. To address this, an R package has been developed that simplifies the implementation of Time-Varying Minimum Variance Portfolios (TV-MVP). We aim to contribute to the field of dynamic covariance estimation by further developing the method of time-dependent covariance estimation introduced by Q. Fan et al. (2024), and lower the barriers of entry by writing and R package.

Given that the primary focus of this thesis is in the estimation of the covariance matrix, performance evaluation is based on the Minimum Variance Portfolio (MVP), a variation of the mean-variance model that disregards expected asset returns and focuses solely on minimizing volatility. The MVP problem is formulated as:

$$\begin{aligned} \min_{\mathbf{w}} \quad & \mathbf{w}'\mathbf{\Sigma}_r\mathbf{w}, \\ \text{s.t.} \quad & \mathbf{w}'\mathbf{1}_p = 1, \end{aligned} \tag{2}$$

with the analytical solution:

$$w^* = \frac{\mathbf{\Sigma}_r^{-1}\mathbf{1}_p}{\mathbf{1}_p'\mathbf{\Sigma}_r^{-1}\mathbf{1}_p}, \tag{3}$$

given that there are no shorting restrictions (Palomar, 2025, Sec. 6.5.1).

The overarching question addressed in this thesis is: Is the performance of a Time Varying Minimum Variance Portfolio (TV-MVP) competitive with other popular portfolio selection methods? Moreover, can TV-MVP be implemented in an R package in a manner that is accessible and practical for the average investor?

The methodology, implementation, and how to use the package in practice are described in the following section. The simulation design is presented in the section *Monte Carlo Simulation*, followed by the setup for the *Empirical Evaluation*, and lastly, the *Results and Discussion*, and the *Concluding Remarks*.

2 Methodology and Implementation

In this section, the necessary equations and computations is described in depth. The methods described in this section is closely related to those presented in the articles by Q. Fan et al. (2024), Wang et al. (2021), and Su and Wang (2017). Some inspiration has also been taken from the GitHub repository containing the MatLab code base used by Q. Fan et al. (2024) in their article (Wu and Fan, 2024).

This section discusses the *Time Varying Factor model*, a method for *Determining the Number of Factors*, *Testing for Time-Invariance in Factor Loadings*, *Time-Dependent Covariance estimation*, how we construct a portfolio optimization strategy for *Out-of-Sample Prediction*, and lastly, a short installation guide for the R package.

2.1 Time Varying Factor Model

The time-varying factor model introduced by Su and Wang (2017) is used to estimate the time-varying factor loadings $\boldsymbol{\lambda}_{it}$ and the factors $\mathbf{F}_t$. We assume that the *p*-dimensional time series of the asset returns with *T* observations $\{r_{it}, i = 1, 2, ..., p;\; t = 1, 2, ..., T\}$ follows the time-varying factor model with *m* latent factors $\mathbf{F}_t = (F_{1t}, ..., F_{mt})'$:

$$r_{it} = \boldsymbol{\lambda}'_{it}\mathbf{F}_t + e_{it}, \tag{4}$$

where e_{it} is the idosyncratic error. To estimate the time-varying factor loadings and common factors, Su and Wang (2017) propose a local weighted least squares method:

$$\min_{\{\boldsymbol{\lambda}_{ix}\}_{i=1}^{p}, \{\mathbf{F}_t\}_{t=1}^{T}} (pT)^{-1} \sum_{i=1}^{p} \sum_{t=1}^{T} (r_{it} - \boldsymbol{\lambda}'_{ix}\mathbf{F}_t)^2 K_h\left(\frac{t-x}{T}\right), \tag{5}$$

where we assume $\boldsymbol{\lambda}_i : [0,1] \rightarrow \mathbb{R}$ is a smooth function such that

$$\boldsymbol{\lambda}_{it} = \boldsymbol{\lambda}_i\left(\frac{t}{T}\right) \approx \boldsymbol{\lambda}_i\left(\frac{x}{T}\right) = \boldsymbol{\lambda}_{ix} \text{ when } \frac{t}{T} \approx \frac{x}{T}. \tag{6}$$

$K_h = h^{-1}K(\cdot/h),\ K : \mathbb{R} \rightarrow \mathbb{R}^+$ is a kernel function and $h = h(T,p)$ is a bandwidth parameter. The minimization problem in equation 5 can be rewritten as:

$$\min_{\mathbf{F}^{(x)},\boldsymbol{\Lambda}^{(x)}} \operatorname{tr}\left[\left(\mathbf{r}^{(x)} - \mathbf{F}^{(x)}\boldsymbol{\Lambda}_x\right)\left(\mathbf{r}^{(x)} - \mathbf{F}^{(x)}\boldsymbol{\Lambda}_x\right)'\right], \tag{7}$$

where $\mathbf{r}^{(x)} = (\mathbf{r}_1^{(x)}, ..., \mathbf{r}_p^{(x)})$, $\mathbf{r}_i^{(x)} = (k_{h,1x}^{1/2} r_{i1}, ..., k_{h,Tx}^{1/2} r_{iT})'$, $\mathbf{F}^{(x)} = (k_{h,1x}^{1/2}\mathbf{F}_1^0, ..., k_{h,Tx}^{1/2}\mathbf{F}_T^0)'$, and $\boldsymbol{\Lambda}_x$ is the factor loadings at time x. Here $k_{h,tx}$ refers to $k_{h,tx} = h^{-1}K((t-x)/(Th))$. Under the identification restrictions $\mathbf{F}^{(x)\prime}\mathbf{F}^{(x)}/T = \mathbb{I}_m$ and $\boldsymbol{\Lambda}_x'\boldsymbol{\Lambda}_x$ is a diagonal matrix, we can concentrate out $\boldsymbol{\Lambda}_x = \mathbf{r}^{(x)\prime}\mathbf{F}^{(x)}(\mathbf{F}^{(x)\prime}\mathbf{F}^{(x)})^{-1} = \mathbf{r}^{(x)\prime}\mathbf{F}^{(x)}/T$, and rewrite equation 7 as:

$$\operatorname{tr}\left[\mathbf{r}^{(x)\prime}\mathbf{r}^{(x)}\right] - T^{-1}\operatorname{tr}\left[\mathbf{F}^{(x)\prime}\mathbf{r}^{(x)}\mathbf{r}^{(x)\prime}\mathbf{F}^{(x)}\right]. \tag{8}$$

This is the conventional PCA problem: maximizing $\operatorname{tr}\left[\mathbf{F}^{(x)\prime}\mathbf{r}^{(x)}\mathbf{r}^{(x)\prime}\mathbf{F}^{(x)}\right]$ with the restriction $\mathbf{F}^{(x)\prime}\mathbf{F}^{(x)}/T = \mathbb{I}_m$. As stated by Su and Wang (2017), $\hat{\mathbf{F}}^{(x)}$ is $\sqrt{T}$ times the eigenvectors of the *m* largest eigenvalues of $\mathbf{r}^{(x)}\mathbf{r}^{(x)\prime}$, and $\hat{\boldsymbol{\Lambda}}_x = (\mathbf{F}^{(x)}\mathbf{F}^{(x)\prime})^{-1}\mathbf{F}^{(x)\prime}\mathbf{r}^{(x)} = \mathbf{F}^{(x)\prime}\mathbf{r}^{(x)}/T$.

$\hat{\mathbf{F}}_t^{(x)}$ is a consistent estimator of the weighted factor $\mathbf{F}^{(x)} = k_{h,tx}\mathbf{F}_t$, following Su and Wang (2017) we use a two stage estimation to find a consistent estimator of $\mathbf{F}_t$. In the first step above, we find consistent estimators for $\boldsymbol{\Lambda}_t$, $\hat{\boldsymbol{\Lambda}}_x$, which we use to compute the consistent estimator for $\mathbf{F}_t$:

$$\hat{\mathbf{F}}_t = \left(\hat{\boldsymbol{\Lambda}}_t'\hat{\boldsymbol{\Lambda}}_t\right)^{-1}\hat{\boldsymbol{\Lambda}}_t'\boldsymbol{r}_t. \tag{9}$$

Before moving on, we need to define the boundary kernel $k*_{h,tx}$:

$$k_{h,tx}^* = h^{-1}K_x^*\left(\frac{t-x}{Th}\right) = \begin{cases} h^{-1}K\left(\frac{t-x}{Th}\right)/\int_{-(x/Th)}^{\infty} K(u)\,du, & \text{if } x \in [0, \lfloor Th \rfloor] \\ h^{-1}K\left(\frac{t-x}{Th}\right), & \text{if } x \in [\lfloor Th \rfloor, T - \lfloor Th \rfloor] \\ h^{-1}K\left(\frac{t-x}{Th}\right)\int_{-\infty}^{(1-x/T)/h} K(u)\,du, & \text{if } x \in (T - \lfloor Th \rfloor, T]. \end{cases} \tag{10}$$

The boundary correction is applied as proposed by Su and Wang (2017), and is necessary in order not to lose efficiency near $t = 1$ or $t = T$ when weighting the data. The kernel function used in the analysis is the Epanechnikov kernel $K(u) = 0.75(1-u^2)\mathbf{1}\{|u| \leqslant 1\}$ where $\mathbf{1}\{\cdot\}$ is the indicator function, the bandwidth used is Silverman's rule of thumb: $h =$

$(2.35/\sqrt{12})T^{-1/5}p^{-1/10}$. The functions for Epanechnikov kernel and Silverman's rule of thumb are implemented in the package, however, the package also allows the user to specify their own kernel and bandwidth functions to be used in the local PCA. The choice of bandwidth is due to ease of use, however, other alternatives, such as a cross-validation approach as described by Su and Wang (2017) and Q. Fan et al. (2024) could also be used.

These are the necessary steps for estimating the time-varying factor loadings and common factors. The practical implementation of the procedure is described in algorithm 1.

2.2 Determining the Number of Factors

A BIC-type information criterion is used to determine the number of factors, *m*. What this aims to estimate is m_0, the true number of factors, which is assumed to be bounded by an above finite integer m_{max}. $\hat{m}_0$ is estimated in the following manner:

$$\begin{aligned} \hat{m}_0 &= \arg\min_m IC(m) \\ \text{where } IC(m) &= \log\ V(m, \{\breve{\mathbf{\Lambda}}_x(m)\}) + \frac{p+Th}{pTh}\log\left(\frac{pTh}{p+Th}\right)m, \\ \text{and } V(m, \{\breve{\mathbf{\Lambda}}_t(m)\}) &= \min_{\breve{\mathbf{F}}=(\breve{\mathbf{F}}_1,\ldots,\breve{\mathbf{F}}_T)'} \frac{1}{pT}\sum_{i=1}^{p}\sum_{x=1}^{T}\left[r_{it} - \breve{\mathbf{F}}_t'\breve{\mathbf{\lambda}}_{it}(m)\right]^2. \end{aligned} \tag{11}$$

Here, $\breve{\mathbf{\Lambda}}_t^m = (pT)^{-1}\mathbf{r}^{(x)\prime}\mathbf{r}^{(x)}\hat{\mathbf{\Lambda}}_t^m$, and it follows from equation 9 that $\breve{\mathbf{F}}_t^m = (\breve{\mathbf{\Lambda}}_t^{m\prime}\breve{\mathbf{\Lambda}}_t^m)\breve{\mathbf{\Lambda}}_t^{m\prime}\mathbf{r}_t$. The asymptotic rank of $\breve{\mathbf{\Lambda}}_t^m$ is given by $\min(m, m_0)$, compared to $\hat{\mathbf{\Lambda}}_t^m$, which is always a full rank matrix. This lends itself to be useful here as $\breve{\mathbf{\Lambda}}_t^m$ is informative on m_0 when $m > m_0$. More information on this can be found in Lemma A.8 in Su and Wang (2017). Given some assumptions stated by Su and Wang (2017), it is shown that $P(\hat{m}_0 = m_0) \rightarrow 1\ as\ (p, T) \rightarrow \infty$.

The pseudo-code for this is given in algorithm 2. Using the package, this can easily be implemented: `determine_factors(returns, max_m, bandwidth)` which outputs `optimal_m` and `IC_values`. Through our own experience using the package, we have found that setting `max_m=10` is often sufficient.

Algorithm 1 Local PCA

```
Require: A returns matrix $\mathbf{r} \in \mathbb{R}^{T \times p}$, kernel parameter $x$, bandwidth $h$, number of factors $m$, kernel function $K(\cdot)$.
Ensure: A factors matrix $\mathbf{F} \in \mathbb{R}^{T \times m}$, a list of loadings $\{\mathbf{\Lambda}_t\}_{t=1}^T$, and a list of kernel weights $\{\mathbf{w}_t\}_{t=1}^T$.
 1: Initialize $\mathbf{F} \leftarrow$ empty $T \times m$ matrix.
 2: Initialize loadings list $\{\mathbf{\Lambda}_t\}_{t=1}^T \leftarrow$ empty.
 3: Initialize weights list $\{\mathbf{w}_t\}_{t=1}^T \leftarrow$ empty.
 4: Set previous factors $\mathbf{F}^{prev} \leftarrow$ NULL.
 5: for $t = 1$ to $T$ do
 6:     // Compute kernel weights for each time point
 7:     For $x = 1, \dots, T$, compute $k^*_{h,tx}(x)$
 8:     Compute weighted returns: $\mathbf{r}^{(x)} = (k_{h,x}^{1/2}\mathbf{r}_1, \dots, k_{h,x}^{1/2}\mathbf{r}_T)'$.
 9:     // Eigen decomposition to obtain local factors
10:     Compute the eigen decomposition: $\mathbf{r}^{(x)}\mathbf{r}^{(x)'} = \mathbf{VDV}'$.
11:     Order the eigenvalues in descending order and select the first $m$ eigenvectors, and compute the factors: $\hat{F}^{(x)} = [\mathbf{v}_1, \dots, \mathbf{v}_m] \cdot \sqrt{T}$
12:     if $\mathbf{F}^{prev} \neq$NULL then    ▷ Sign consistency ensures interpretability over time and maintains consistent rotations.
13:         for $j = 1$ to $m$ do
14:             if cor($\mathbf{F}_j^{prev}, \hat{\mathbf{F}}_j^{(x)}) < 0$ then
15:                 Flip the sign of the $j$th column: $\hat{\mathbf{F}}_j^{(x)} \leftarrow -\hat{\mathbf{F}}_j^{(x)}$.
16:             end if
17:         end for
18:     end if
19:     // Compute loadings and factor for time $t$
20:     $\hat{\mathbf{\Lambda}}_t = T^{-1}\hat{\mathbf{F}}^{(x)'}\mathbf{r}^{(x)}$
21:     $\hat{\mathbf{F}}_t = \left(\hat{\mathbf{\Lambda}}_t'\hat{\mathbf{\Lambda}}_t\right)^{-1}\hat{\mathbf{\Lambda}}_t'\mathbf{r}_t$
22:     Set $\mathbf{F}[t, :] \leftarrow \hat{\mathbf{F}}_t'$.
23:     Store $\hat{\mathbf{\Lambda}}_t$ in the loadings list.
24:     Store the weights vector $\mathbf{w}_t = (k_h(1), \dots, k_h(T))'$.
25:     Update $\mathbf{F}^{prev} \leftarrow \hat{\mathbf{F}}^{(x)}$.
26: end for
27: return $\{\mathbf{F}, \{\hat{\mathbf{F}}^{(x)}\}_{t=1}^T, \{\mathbf{\Lambda}_t\}_{t=1}^T, m, \{\mathbf{w}_t\}_{t=1}^T\}$.
```

2.3 Testing for Time-Invariance in Factor Loadings

Before employing the out-of-sample prediction for investment purposes, it is useful to first investigate the time-varyingness of the covariance structure in the data. The hypothesis test

Algorithm 2 Determine the Optimal Number of Factors

```
Require: Data matrix $r \in \mathbb{R}^{T \times p}$, maximum number of factors $m_{max}$, bandwidth $h$
Ensure: Optimal number of factors $\hat{m}_0$ and the information criterion values $\{IC(m)\}_{m=1}^{m_{max}}$
 1: Initialize vectors: $V \leftarrow \mathbf{0} \in \mathbb{R}^{m_{max}}$, $penalty \leftarrow \mathbf{0} \in \mathbb{R}^{max}$, and $IC\_values \leftarrow \mathbf{0} \in \mathbb{R}^{max}$.
 2: for $m = 1$ to $m_{max}$ do
 3:     Initialize residuals matrix $\mathbf{e} \in \mathbb{R}^{T \times p}$ (e.g., with NA or zeros).
 4:     Set $\mathbf{F}^{prev} \leftarrow$ NULL.
 5:     for $t = 1$ to $T$ do
 6:         Perform local PCA with $m$ factors at time $t$. Lines 6-33 in algorithm 1.
 7:         Normalize factor loadings: $\tilde{\boldsymbol{\lambda}}_{it}^{m} = \sqrt{p}\frac{\hat{\boldsymbol{\lambda}}_{it}^{m}}{||\hat{\boldsymbol{\lambda}}_{it}^{m}||}$ s.t. $\frac{1}{p}\hat{\boldsymbol{\Lambda}}_t^{m'}\hat{\boldsymbol{\Lambda}}_t^{m} = \mathbb{I}_m$
 8:         Compute estimate of factor loadings by $\breve{\boldsymbol{\Lambda}}_t^{m} = (pT)^{-1}\mathbf{r}^{(x)'}\mathbf{r}^{(x)}\tilde{\boldsymbol{\Lambda}}_t^{m}$
 9:         Compute factor estimates by $\breve{\mathbf{F}}_t^{m} = (\breve{\boldsymbol{\Lambda}}_t^{m'}\breve{\boldsymbol{\Lambda}}_t^{m})\breve{\boldsymbol{\Lambda}}_t^{m'}\mathbf{r}_t$
10:         Compute the residuals: $\mathbf{e}_t \leftarrow \mathbf{r}_t - \breve{\mathbf{F}}_t^{m'}\breve{\boldsymbol{\Lambda}}_t^{m}$
11:         Update $\mathbf{F}^{prev} \leftarrow \hat{\mathbf{F}}^{(x)}$.
12:     end for
13:     Compute the average sum of squared residuals: $V_m \leftarrow (pT)^{-1}e'e$
14:     Compute the penalty term: penalty$_m \leftarrow m \cdot \left(\frac{N+T\cdot h}{N\cdot T\cdot h}\right) \cdot \log\left(\frac{N\cdot T\cdot h}{N+T\cdot h}\right)$.
15:     Form the information criterion: IC$_m \leftarrow \ln\left(V_m\right) +$ penalty$_m$.
16: end for
17: Determine the optimal number of factors: $\hat{m}_0 \leftarrow \arg\min_m IC(m)$.
18: Return $(\hat{m}_0, IC)$.
```

proposed by Su and Wang (2017) is therefore included in the package. The hypotheses are:

$$\begin{aligned} &\mathbb{H}_0 : \boldsymbol{\lambda}_{it} = \boldsymbol{\lambda}_{i0} \text{ for } i = 1, 2, ..., p \text{ and } t = 1, 2, ..., T \\ &\mathbb{H}_1 : \boldsymbol{\lambda}_{it} \neq \boldsymbol{\lambda}_{i0} \text{ for some } i, t, \end{aligned} \tag{12}$$

i.e, we test whether the factor loadings are constant over time. Here, $\hat{\boldsymbol{\lambda}}_{it}$ are estimated using algorithm 1, and $\hat{\boldsymbol{\lambda}}_{i0}$ is estimated using regular PCA. This test statistic, J_{pT}, is shown by Su and Wang (2017) to be asymptotically standard normal under the null, if certain assumptions hold (see Theorems 4.1 and 4.2 in Su and Wang, 2017). In the finite sample setting, there is a risk that $\hat{J}_{pT}$ does not converge in distribution, yielding a statistic that does not follow the standard normal distribution under the null. To ensure the performance of the test in finite sample settings, a bootstrap version is included in the package.

The test statistic is given by

$$\hat{J}_{pT} = \hat{\mathbb{V}}_{pT}^{-1/2}\left(Tp^{1/2}h^{1/2}\hat{M} - \hat{\mathbb{B}}_{pT}\right), \tag{13}$$

which is a scaled and centered version of $\hat{M}$:

$$\hat{M} = \frac{1}{pT} \sum_{i=1}^{p} \sum_{t=1}^{T} \left(\hat{\boldsymbol{\lambda}}_{it}' \hat{\mathbf{F}}_t - \tilde{\boldsymbol{\lambda}}_{i0}' \tilde{\mathbf{F}}_t \right), \tag{14}$$

And the asymptotic variance and bias are given by

$$\begin{aligned} \hat{\mathbb{V}}_{pT} &= \frac{2}{phT^2} \sum_{1 \leqslant s \neq r \leqslant T} \bar{k}_{sr}^2 \left(\hat{\mathbf{F}}_s' \hat{\boldsymbol{\Sigma}}_F \hat{\mathbf{F}}_r \right)^2 \left(\hat{\mathbf{e}}_r' \hat{\mathbf{e}}_s \right)^2, \ and \\ \hat{\mathbb{B}}_{pT} &= \frac{h^{1/2}}{T^2 p^{1/2}} \sum_{i=1}^{p} \sum_{t=1}^{T} \sum_{s=1}^{T} \left(k_{h,st} \hat{\mathbf{F}}_s' \hat{\mathbf{F}}_t - \tilde{\mathbf{F}}_s' \tilde{\mathbf{F}}_t \right)^2 \hat{e}_{is}^2 \end{aligned} \tag{15}$$

respectively.

This, and the construction of the wild bootstrap, is summarized in algorithm 3. When implementing the function in practice you run `hyptest1(returns, m, B, kernel_func)` which outputs J_{pT}, $p-$value, and bootstrap statistics $J_{pT}^{bootstrap}$.

2.4 Time-Dependent Covariance Matrix Estimation

Stock market data is inherently high-dimensional, with the number of variables *p* large relative to the sample size *T*. With this comes the problem of estimating the covariance matrix; the sample covariance matrix often performs poorly when *p* is large in finite samples (Ledoit and Wolf, 2004). A common approach to this problem is to assume a sparse covariance matrix, i.e., assume that the covariances are zero, or close to zero, for many of the assets. As such, many regularization techniques have been introduced to impose such sparsity in the covariance matrix, such as graphical LASSO, shrinkage, and thresholding methods (e.g. Bickel and Levina, 2008; Friedman et al., 2007; Ledoit and Wolf, 2003). Wang et al. (2021) argue that the sparsity assumption is often too restrictive in finance and economics settings, as the variables are often highly correlated. As such, regularizing the covariance matrix too heavily could yield inaccurate results. In the case of this thesis and package, it could yield inaccurate weights, which would produce portfolios with uncertain performance. Financial data is also inherently dynamic, with covariance structures changing over time due to extreme events such as pandemics and war, as well as changes in production and gradual market shifts. To address these problems, both Wang et al. (2021) and Q. Fan et al. (2024) propose a covariance estimator constructed using time-varying factors and factor loadings, with some slight differences. In our package we use a similar method where we kernel smooth the data using the boundary kernel specified in Eq. 10 around $x = T$ when performing the out-of-sample prediction, i.e. we use $\hat{\mathbf{F}}$ and $\hat{\mathbf{\Lambda}}_t$ in the construction of the covariance matrix $\hat{\mathbf{\Sigma}}_{r,t}$. Our method distinguishes

Algorithm 3 Hypothesis Test for Time-Varying Factor Loadings

```
Require: Data matrix $\mathbf{r} \in \mathbb{R}^{T \times p}$, number of factors $m$, bootstrap iterations $B$ (default: 200), kernel function $K$
    (default: epanechnikov_kernel)
Ensure: Test statistic $\hat{J}_{pT}$, bootstrap p-value, and bootstrap statistics $\{J_{pT}^{*(b)}\}_{b=1}^{B}$
 1: Standardize Data:
 2:     $\mathbf{r} \leftarrow \text{scale}(\mathbf{r})$
 3:     $T \leftarrow$ number of rows of $\mathbf{r}$, $p \leftarrow$ number of columns of $\mathbf{r}$
 4:     $h \leftarrow \text{silverman}(\mathbf{r})$
 5: Local PCA:
 6:     $(\hat{\mathbf{F}}, \hat{\mathbf{\Lambda}}) \leftarrow \text{local PCA}(\mathbf{r}, h, m)$
 7: Global Factor Analysis:
 8:     Compute truncated SVD: $\mathbf{r} \approx \mathbf{U}_m \, \mathbf{D}_m \, \mathbf{V}_m^{\top}$
 9:     $\tilde{\mathbf{F}} \leftarrow \sqrt{T}\, \mathbf{U}_m$
10:     $\tilde{\mathbf{\Lambda}} \leftarrow \left(T^{-1}\, \tilde{\mathbf{F}}'\mathbf{r}\right)'$
11: Compute residuals and Error Scale:
12:     $\hat{\mathbf{e}} \leftarrow \sum_{t=1}^{T}(\mathbf{r}_t - \hat{\mathbf{F}}_t'\hat{\Lambda}_t)$
13:     $\tilde{\mathbf{\Sigma}} \leftarrow \tilde{\sigma}_{ij}^{0}(1-\epsilon)^{|i-j|}\ for\ all\ i, j = 1, .., p$      ▷ Let $\tilde{\sigma}_{ij}^{0}$ be the (i,j)th element of $\tilde{\mathbf{\Sigma}}_0 = T^{-1}\sum_{t=1}^{T}\tilde{\mathbf{e}}_t\tilde{\mathbf{e}}_t'$
14: Compute Test Statistic:
15:     $\hat{M} \leftarrow \frac{1}{pT}\sum_{i=1}^{p}\sum_{t=1}^{T}\left(\hat{\mathbf{\lambda}}_{it}'\hat{\mathbf{F}}_t - \tilde{\mathbf{\lambda}}_{i0}'\tilde{\mathbf{F}}_t\right)$
16:     $\hat{\mathbb{B}}_{pT} \leftarrow \frac{h^{1/2}}{T^2 p^{1/2}}\sum_{i=1}^{p}\sum_{t=1}^{T}\sum_{s=1}^{T}\left(k_{h,st}\hat{\mathbf{F}}_s'\hat{\mathbf{F}}_t - \tilde{\mathbf{F}}_s'\tilde{\mathbf{F}}_t\right)^2 \hat{e}_{is}^2$
17:     $\hat{\mathbb{V}}_{pT} \leftarrow \frac{2}{phT^2}\sum_{1 \leqslant s \neq r \leqslant T}\bar{k}_{sr}^2\left(\hat{\mathbf{F}}_s'\hat{\mathbf{\Sigma}}_F\hat{\mathbf{F}}_r\right)^2(\hat{\mathbf{e}}_r'\hat{\mathbf{e}}_s)^2$
18:     $\hat{J}_{pT} \leftarrow \dfrac{T\sqrt{p}\sqrt{h}\,\hat{M} - \hat{\mathbb{B}}_{pT}}{\sqrt{\hat{\mathbb{V}}_{pT}}}$
19: Bootstrap Procedure:
20:     Initialize $J_{pT}^{\text{bootstrap}} \leftarrow$NULL
21: for $b = 1$ to $B$ do
22:     Generate $\boldsymbol{\zeta} \in \mathbb{R}^{T \times p}$ with i.i.d. $N(0, 1)$ entries
23:     Compute bootstrap errors: $\mathbf{e}^* \leftarrow \tilde{\mathbf{\Sigma}}^{1/2}\boldsymbol{\zeta}$
24:     Generate bootstrap sample: $\mathbf{r}^* \leftarrow \tilde{\mathbf{F}}\tilde{\mathbf{\Lambda}}' + \mathbf{e}^*$
25:     Repeat steps on lines 5-18 for $\mathbf{r}^*$
26:     Append $J_{pT}^*$ to $J_{pT}^{\text{bootstrap}}$
27: end for
28: Compute Bootstrap p-value:
29:     $p_{value} \leftarrow \frac{1}{B}\sum_{b=1}^{B}\mathbf{1}\{J_{pT}^{*(b)} \geqslant J_{pT}\}$
30: return $\{J_{pT},\ p\_value,\ J_{pT}^{\text{bootstrap}}\}$
```

itself in that we use a different method for calculating Σ_F compared the Wang et al. (2021), and a different method for regularizing Σ_e compared to Q. Fan et al. (2024). The original plan was to implement the same regularization of Σ_e as Q. Fan et al. (2024), however due to the computational complexity, we opted for a simpler approach.

We start by constructing a naive estimate of the residual covariance. Let $\hat{\mathbf{e}}_t = \mathbf{r}_t - \hat{\mathbf{F}}_t'\hat{\mathbf{\Lambda}}_t$, the sample covariance matrix is then:

$$\hat{\Sigma}_e = \frac{1}{T}\hat{\mathbf{e}}'\hat{\mathbf{e}}. \tag{16}$$

To solve the problem of ill-behaved sample covariance matrices, we apply a general shrinkage to the residual covariance matrix (see Z. Chen and Leng, 2016; Wang et al., 2021).

$$\tilde{\Sigma}_e^{(x)} = [\tilde{\sigma}_{e,ij}^{(x)}]_{p\times p}, \quad \tilde{\sigma}_{e,ij}^{(x)} = \begin{cases} \hat{\sigma}_{e,ij}^{(x)}, & \text{if } i = j \\ S_{\rho(x)}(\hat{\sigma}_{e,ij}^{(x)}) & \text{if } i \neq j \end{cases} \tag{17}$$

Where $s_\rho(z)$ is a shrinkage function which satisfies the following three conditions for all $z \in R$: (i) $|s_\rho(z)| \leqslant |z|$; (ii) $s_{rho} = 0$ for $z \leqslant \rho$; (iii) $|s_\rho(z) - z| \leqslant \rho$ (Z. Chen and Leng, 2016). In this thesis the focus is on soft thresholding: $s_\rho = \text{sign}(z)(|z| - \rho)_+$, however hard thresholding, adaptive lasso, and smoothly clipped absolute deviation (SCAD) could easily be implemented in the package and would be a good addition to the current functionality.

After estimating the residual covariance, we can compute the covariance matrix of the returns:

$$\tilde{\Sigma}_{r,t} = \mathbf{\Lambda}_t\widehat{\Sigma}_F\mathbf{\Lambda}_t' + \tilde{\Sigma}_e, \tag{18}$$

which is dependent on time by the time-dependent factor loadings. Here, $\widehat{\Sigma}_F$ is simply:

$$\frac{1}{T}\mathbf{F}'\mathbf{F}. \tag{19}$$

2.4.1 Choice of Tuning Parameter ρ

Following Wang et al. (2021) and J. Chen et al. (2019), we implement a three-step procedure that allows the tuning variable to vary with *x*. The steps for choosing the tuning parameter ρ are as follows:

(i) Let $\lfloor\cdot\rfloor$ denote the floor function, for a given *x*, divide the sample into $m = 1, \ldots, \lfloor T/2M_0\rfloor$ groups, split the data in each group into two sub-samples, $T_1 = \lfloor\frac{T}{2}(1 - 1/\log(T/2))\rfloor$ and

$T_2 = \lfloor T/2 \rfloor - T_1$, leaving M_0 observations out in between the two sub-samples. (ii) Compute the shrunk residual covariance matrix of the first sub-sample $\tilde{\mathbf{\Sigma}}_{e,1,m}$, and the sample covariance matrix of the second sub-sample $\hat{\mathbf{\Sigma}}_{e,2,m}$. (iii) Choose tuning parameter ρ which minimizes the sum of the squared Frobenius norm:

$$\sum_{m=1}^{\lfloor T/2M_0 \rfloor} ||\tilde{\mathbf{\Sigma}}_{e,1,m} - \hat{\mathbf{\Sigma}}_{e,2,m}||_F^2 \tag{20}$$

for $\rho \in [\rho_1, \rho_2]$, where $\rho_1 = \epsilon + \inf\{\rho_* > 0 | \lambda_{min}(\tilde{\mathbf{\Sigma}}_{e,1,m} > 0, \forall \rho > \rho_*\}$. Here, ϵ is a small positive constant, and ρ_2, the upper bound of ρ, should be a sufficiently large positive constant. The Frobenius is used as the criterion as it balances bias (over-shrinking) and variance (under-shrinking). The reason for excluding M_0 observations between the two sub-samples is to ensure that the correlation between the two is negligible. Both J. Chen et al. (2019) and Wang et al. (2021) use $M_0 = 10$, and we follow this lead.

The full process of computing the covariance matrix and the optimal tuning parameter can be seen in algorithms 4 and 5, respectively. These functions are internal, and a wrapper has been written that incorporates the local PCA procedure and the covariance estimator:

```r
cov_mat <- time_varying_cov(returns,
                            m,
                            bandwidth = silverman(returns),
                            kernel_func = epanechnikov_kernel,
                            M0 = 10,
                            rho_grid = seq(0.005, 2,
                                           length.out = 30),
                            floor_value = 1e-12,
                            epsilon2 = 1e-6,
                            full_output = FALSE)
```

This function computes the covariance matrix for the last time period in `returns`. The default output is the covariance matrix, but if more detailed output is needed, the user can run the function with `full_output=TRUE`.

Algorithm 4 Estimate Covariance

```
Require: Local PCA results $(\mathbf{\Lambda}_x, \mathbf{F})$, returns $\mathbf{r}$, $M_0$, $\rho_{\text{grid}}$, floor value $\epsilon$
Ensure: Estimated covariance $\tilde{\mathbf{\Sigma}}_{r,t}$ and residual covariance $\tilde{\mathbf{\Sigma}}_e$
 1: Extract factor loadings $\mathbf{\Lambda}_t$ and factors $F$
 2: Compute residuals $\mathbf{e}_t = \mathbf{r} - \mathbf{F}'\mathbf{\Lambda}_t$
 3: Compute best shrinkage parameter $\rho^*$ using ADAPTIVERHO on $\mathbf{e}$        ▷ Algorithm 5
 4: Compute raw residual covariance: $\hat{\mathbf{\Sigma}}_e = \frac{1}{T}\hat{\mathbf{e}}'\hat{\mathbf{e}}$
 5: Compute shrinkage threshold: $\tau = \rho^* \cdot \text{mean}(|\hat{\mathbf{\Sigma}}_e|_{\text{off-diagonal}})$
 6: Apply soft-thresholding to off-diagonal elements of $\hat{\mathbf{\Sigma}}_e$:
        $$\tilde{\mathbf{\Sigma}}_e = [\tilde{\sigma}_{e,ij}]_{p\times p},\ \tilde{\sigma}_{e,ij} = \text{sign}(\hat{\sigma}_{e,ij} \max(|\hat{\sigma}_{e,ij}| - \tau, 0), \quad i \neq j$$
 7: Compute total covariance estimate: $\tilde{\mathbf{\Sigma}}_{r,t} = \mathbf{\Lambda}_t \hat{\mathbf{\Sigma}}_F \mathbf{\Lambda}_t' + \tilde{\mathbf{\Sigma}}_e$
 8: Perform eigenvalue flooring to ensure positive semidefiniteness:
        Compute eigen-decomposition: $\tilde{\mathbf{\Sigma}}_{r,t} = \mathbf{Q\Lambda Q}'$
        Floor eigenvalues: $\lambda_i = \max(\lambda_i, \epsilon)$
        Reconstruct PSD covariance: $\tilde{\mathbf{\Sigma}}_{r,t} = \mathbf{Q\Lambda Q}'$
 9: Return $\tilde{\mathbf{\Sigma}}_{r,t}$
```

Algorithm 5 Adaptive Shrinkage Tuning Parameter Selection

```
Require: Residual matrix $\mathbf{e}$, tuning parameter $M_0$, candidate shrinkage values $\boldsymbol{\rho}_{\text{grid}}$, a small
    positive constant $\epsilon$
Ensure: Optimal shrinkage parameter $\rho^*$
 1: Partition data into overlapping sub-groups of size $\lfloor T/(2M_0) \rfloor$
 2: for each $\rho \in \boldsymbol{\rho}_{\text{grid}}$ do
 3:     Divide into sub-samples $T_1 = \lfloor \frac{T}{2}(1 - 1/\log(T/2)) \rfloor$ and $T_2 = \lfloor T/2 \rfloor - T_1$
 4:     Apply soft-thresholding to the first sub-sample's covariance $T_1$
 5:     Compute $\tilde{\mathbf{\Sigma}}_{e,1,m}$ and $\hat{\mathbf{\Sigma}}_{e,2,m}$
 6:     Compute Frobenius norm difference: $\sum_{m=1}^{\lfloor T/2M_0 \rfloor} ||\tilde{\mathbf{\Sigma}}_{e,1,m} - \hat{\mathbf{\Sigma}}_{e,2,m}||_F^2$
 7: end for
 8: Compute $\rho_1 = \epsilon + \inf\{\rho_* > 0 | \lambda_{min}(\tilde{\mathbf{\Sigma}}_{e,1,m} > 0, \forall \rho > \rho_*\}$
 9: Select $\rho^* \in [\rho_1, \rho_2]$ that minimizes Frobenius norm difference
10: Return $\rho^*$
```

2.5 Out-of-Sample Prediction

The out-of-sample prediction uses many of the previously described methods. In this section, the functions of the package that are used for prediction and how they have been constructed are presented. Note that log returns are used throughout the analysis and are the expected input for the functions included discussed in this section. The use of log returns are due to personal preference, and the results could easily be converted to simple returns.

2.5.1 Portfolio Optimization and Prediction

The metrics of interest are risk, expected excess returns (ER), and the Sharpe ratio (SR):

$$\text{Risk}_t = \mathbf{w}_t' \mathbf{\Sigma}_{r,t} \mathbf{w}_t = \frac{1}{\mathbf{1}_p' \mathbf{\Sigma}_{r,t}^{-1} \mathbf{1}_p}, \quad EER_t = \mathbf{w}_t'(\boldsymbol{\mu}_t - r_f), \quad SR = \frac{\mathbf{1}_p' \mathbf{\Sigma}_{r,t}^{-1}(\boldsymbol{\mu}_t - r_f)}{\sqrt{\mathbf{1}_p' \mathbf{\Sigma}_{r,t}^{-1} \mathbf{1}_p}}, \tag{21}$$

where r_f refers to the risk-free rate. SR is a metric commonly used when evaluating equity portfolio performance, and is a metric of the returns gained compared to the risk taken.

Out-of-sample forecasting aims to predict excess returns $r_{t+j},\ j = 1, \dots, J$, where J is the forecasting horizon. As we only have information up until $t = T$, we estimate r_{t+j} as:

$$\mathbf{r}_{t+j} = \widehat{EER}_t, \quad t = T, \tag{22}$$

and the cumulative excess returns:

$$\widehat{CER}_J = J \times \widehat{EER}_t, \quad t = T. \tag{23}$$

Similarly, for the cumulative risk:

$$\widehat{CR}_J = \sqrt{J} \times \widehat{Risk}_t, \quad t = T. \tag{24}$$

The covariance matrix is computed as in algorithm 4, which is used to compute the weights as in eq. 3. The package also offers the user to set a minimum returns constraint, i.e., eq. 3 is subject to both $w'\mathbf{1}_p = 1$ and $\mathbf{w}'(\boldsymbol{\mu} - r_f) \geqslant r_{min}/J$, as well as the option to compute the maximum SR portfolio. The full process can be seen in algorithm 6.

While the main focus of this thesis lies in the estimation of time-varying covariance matrices and their application in portfolio optimization, portfolio construction also requires estimates

Algorithm 6 Predict Portfolio

```
Require: A returns matrix $\mathbf{r} \in \mathbb{R}^{T\times p}$, forecast horizon, maximum number of factors, kernel function $K(\cdot)$,
    minimum returns constrataint (optional), maximum Sharpe ratio portfolio (TRUE or FALSE), risk free rate
    $r_f$.
Ensure: Portfolio weights $\mathbf{w}^*$, forecast estimates of cumulative excess log returns $CER$, cumulative risk $CR$,
    and Sharpe ratio $SR$.
 1: Select bandwidth using Silverman's rule
 2: Determine optimal number of factors via DETERMINE_FACTORS
 3: Perform Local PCA                                                    ▷ see Algorithm 1
 4: Compute covariance matrix:                                           ▷ Algorithm 4
 5: Compute expected excess returns: $\hat{\boldsymbol{\mu}} \leftarrow$ expected value of returns minus risk free rate $\hat{\mathbf{r}}_{t+j} - r_f$
 6: Global Minimum Variance Portfolio (GMV):
 7: $\mathbf{w}_T^{*GMV} = \frac{\tilde{\boldsymbol{\Sigma}}_{r,T}^{-1}\mathbf{1}_p}{\mathbf{1}_p'\tilde{\boldsymbol{\Sigma}}_{r,T}^{-1}\mathbf{1}_p}$              ▷ Normalize
 8: Compute GMV performance:
 9: $\widehat{CER}_J^{GMV} = J \times \mathbf{w}_T^{*GMV}\hat{\boldsymbol{\mu}}$
10: $\widehat{CR}_J^{GMV} = \sqrt{\mathbf{w}_T^{*GMV}\tilde{\boldsymbol{\Sigma}}_{r,T}\mathbf{w}_T^{*GMV}} \times \sqrt{J}$
11: $\widehat{SR}^{GMV} = \frac{\widehat{CER}_J^{GMV}}{\sqrt{J}\times\widehat{CR}_J^{GMV}}$
12: if $max\_SR$ = TRUE then
13:     Maximum SR Portfolio:
14:     $\mathbf{w}_T^{*sr} \propto \tilde{\boldsymbol{\Sigma}}_{r,T}^{-1}\hat{\boldsymbol{\mu}}$              ▷ Normalize
15:     Compute Maximum SR performance                                   ▷ See lines 9-11
16: end if
17: if $min\_return \neq$ null then
18:     Minimum Variance Portfolio with Return Constraint
19:     $\boldsymbol{A} \leftarrow [\mathbf{1}, \hat{\boldsymbol{\mu}}]_{p\times 2}$                ▷ Constraint matrix
20:     $\mathbf{b} \leftarrow [1, min\_return/J]_{2\times 1}$                 ▷ Constraint values
21:     $\mathbf{w}_T^{*constr.} = \tilde{\boldsymbol{\Sigma}}_{r,T}^{-1}\mathbf{A}(\mathbf{A}'\tilde{\boldsymbol{\Sigma}}_{r,T}^{-1}\mathbf{A})^{-1}\mathbf{b}$     ▷ Normalize
22:     Compute Constrained Portfolio performance                        ▷ See lines 9-11
23: end if
24: return { Minimum Variance Portfolio: $(\mathbf{w}_T^{*GMV}, CER^{GMV}, CR^{GMV}, SR^{GMV})$,
25:         Maximum SR Portfolio: $(\mathbf{w}_T^{*sr}, CER^{sr}, CR^{sr}, SR^{sr})$ [if computed],
26:         Return-Constrained Portfolio: $(\mathbf{w}_T^{*constr.}, CER^{constr.}, CR^{constr.}, SR^{constr.})$ [if computed] }
```

of expected returns. To this end, we employ a simple univariate ARIMA-based forecasting approach. In the package, the expected returns are computed by ARIMA prediction. The ARIMA specification is defined using a simple grid search to find the model with the lowest AIC out of ARIMA(0,0,0), (1,0,0), (0,0,1), and (1,0,1). This is used to forecast the expected portfolio returns and in the optimization of the maximum SR portfolio and the MVP with minimum returns constraint.

ARIMA serves as a pragmatic choice as a general-purpose forecasting method. The use of more sophisticated methods of estimating the expected returns and rigorous evaluation of the maximum SR portfolio and the MVP with minimum returns constraint is left for future research.

An example of how to run the function from the package can be seen below:

```r
optimal_portfolio <- predict_portfolio(returns = returns,
                                        horizon = 5,
                                        max_factors = 5,
                                        min_return = 0.015,
                                        max_SR = TRUE)
```

2.5.2 Expanding Window Evaluation

The expanding window function `expanding_tvmvp` is simply an extension of `predict_portfolio` used to evaluate the performance. For this function, the user states the initial window to use for estimation of the covariance. The function then computes the weights, which are applied for the duration of the rebalancing period, rebalance, and start over. Using this expanding window allows us to investigate the performance of the method over time.

The evaluation metrics implemented in the function are cumulative log excess returns, standard deviation, and Sharpe ratio. The cumulative log excess returns are calculated as:

$$CER = \sum_{t=1}^{K \times J} (er_t^*). \tag{25}$$

Where K denotes the number of windows, J denotes the length of the rebalancing period, and er^* denotes the weighted excess returns. The risk is simply the standard deviation:

$$\text{SD} = \frac{1}{K \times J - 1} \sum_{t=1}^{K \times J} (er_t^* - \bar{er}^*), \tag{26}$$

and the Sharpe ratio is SR $= \bar{er}^*/\text{SD}$. These three metrics gives a good representation of how well the method works for the given data.

Algorithm 7 Expanding Window Time-Varying Minimum Variance Portfolio (MVP)

```
Require: A returns matrix $\mathbf{r} \in \mathbb{R}^{T \times p}$, length of initial window used for estimation, maximum number of factors, return type (daily, weekly, monthly), kernel function $K(\cdot)$, risk free rate $r_f$ (scalar or vector of length $T - initial\ window$).
Ensure: Cumulative excess log returns (CER), mean excess log returns (MER), standard deviation, Sharpe ratio, and annualized versions of the metrics.
 1: $rebalance\_dates \leftarrow \{initial\_window + 1, initial\_window + 1 + rebal\_period, \ldots, T\}$
 2: $RT \leftarrow$ |rebalance_dates|
 3: Determine number of factors based on initial window          ▷ Algorithm 2
 4: for $j = 1$ to $RT$ do
 5:     $reb\_t \leftarrow$ rebalance_dates[$j$]
 6:     $est\_data \leftarrow$ returns[1:($reb\_t - 1$)]
 7:     Predict length of rebalancing period ahead, $m$ determined outside loop          ▷ Algorithm 6
 8:     Save $\mathbf{w}^*_{T*}$ into weights list
 9:     $hold\_end \leftarrow \min(reb\_t + rebal\_period - 1, T)$
10:     $\mathbf{r}^*_j \leftarrow$ {for $t = reb\_t$ to $hold\_end$, compute $\hat{\mathbf{w}}'\mathbf{r}_t$}
11: end for
12: er = $\mathbf{r}^*_j - r_f$
13: $CER \leftarrow \sum_{k=1}^{K} \sum_{j=1}^{J} \mathbf{w}^*_k er_j$, for $K = RT$, $J = rebal\_period$
14: Compute performance metrics: mean, standard deviation, Sharpe ratio
15: Annualize standard deviation and Sharpe ratio
16: return list of computed metrics and portfolio weights
```

The complete sequence can be seen in algorithm 7. To use the function, simply run: `expanding_tvmvp(returns, initial_window, rebal_period, max_factors, return_type = "daily")`. For brevity, it is not shown in the algorithm, but the function also outputs the same metrics for an equal weights portfolio as a benchmark to evaluate the performance against.

2.6 Package Installation

The methods described in section 2 are implemented in the R package `TVMVP` [1] which can be installed using:

[1]The package version discussed in this thesis is a beta that I developed independently under the academic guidance of my supervisor, Yukai Yang. A fully-refined release is being prepared jointly by Yang and me for submission to CRAN.

```r
devtools::install_github("erilill/TV-MVP", ref = "beta", build_
   vignettes = TRUE),
```

given that the package `devtools` or `remotes` is installed.

And then be attached by running:

```r
library(TVMVP)
```

Before using the package, it is recommended to read the vignette `TVMVP-package`, which gives a brief presentation of how the package is intended to be used. This can be accessed by running `vignette("TVMVP_overview", package = "TVMVP")`. The source code can be found in our GitHub repository (Lillrank and Yang, 2025). The package was written, and the analysis conducted, using R 4.4.2 (R Core Team, 2021).

Important to note is that the functions that offer out-of-the-box portfolio optimization, `predict_portfolio`, and `expanding_tvmvp`, require log excess returns, r_t; the remaining functions can be used interchangeably with simple excess returns.

3 Monte Carlo Simulation

To test that the hypothesis test works as intended, a simulation study is conducted. The performance of the hypothesis test has been well documented by Su and Wang (2017) and Q. Fan et al. (2024); because of this, only a smaller-scale simulation study is conducted to make sure that the hypothesis test is correctly implemented in the package.

The simulation is run with 500 replications, T=200, p=100, m=2, and 200 bootstrap draws when conducting the test. As the size of the test statistic is less reliable in the finite sample, the main result presented is the rejection rate at 1, 5, and 10% significance, based on the bootstrap p-value.

Six data-generating processes (DGP) have been chosen to evaluate the performance of the package in varied settings of time-invariant and time-varying covariance. The DGP's used are mimicking the simulation design used by Su and Wang (2017).

3.1 Data Generating Processes

A subset of the DGPs designed by Su and Wang (2017) is used, namely the first six DGPs included in their simulation study. The choice to use this simulation design was made to ensure comparability with the results found by Su and Wang (2017).

DGP 1 (IID):

$\boldsymbol{\lambda}_{it} \sim N(0, \mathbb{I}_2) \quad \text{and} \quad e_{it} \sim N(0,1).$

DGP 2 (Heteroskedastic):

$\boldsymbol{\lambda}_{it} \sim N(0, \mathbb{I}_2) \quad \text{and} \quad e_{it} = \sigma_i \nu_{it}, \quad \text{where} \quad \sigma_i \sim N(0,1) \quad \text{and} \quad \nu_{it} \sim U(0.5, 1.5).$

DGP 3 (Cross-Sectional Dependence)

$\boldsymbol{\lambda}_{it} \sim N(0, \mathbb{I}_2) \quad \text{and} \quad e_t \sim N(0, \Sigma_e), \quad \text{where} \quad \Sigma = (c)_{ij}, c_{ij} = 0.5^{|i-j|}.$

DGP 4 (Structural Break):

$$\boldsymbol{\lambda}_{it} = \begin{cases} \lambda_{i0,k}, & \text{for } 1, \ldots, T/2 \\ \lambda_{i0,k} + b, & \text{for } T/2+1, \ldots, T \end{cases},$$

$\lambda_{i0,k} \sim N(1,1)$ for k=1,2, $e_{it} = \sigma_i \nu_{it}, \quad$ where $\quad \sigma_i \sim N(0,1) \quad$ and $\quad \nu_{it} \sim U(0.5, 1.5).$

DGP 5 (Multiple Structural Breaks):

$$\lambda_{it,1} = \begin{cases} \lambda_{i0,1} + 1 & \text{for } 0.6T < t \leqslant 0.8T \\ \lambda_{i0,1} - 0.5b & \text{for } 0.2T < t \leqslant 0.4T \end{cases}$$

Where $\lambda_{i0,1} \sim N(1,1)$, $\lambda_{it,2} = \lambda_{i0,2} \sim N(0,1)$ and $e_{it} \sim N(0,1)$.

DGP 6 (Smooth Structural Changes):

$\lambda_{it,1} = \lambda_{i0,1} \sim N(0,1)$, $\lambda_{it,2} = b \times G(10t/T, 2, 5i/p + 2)$, where $G(z : \kappa, \gamma) = \big\{1 + \exp\left[-\kappa \prod_{l=1}^{p}(z - \gamma_l)\right]\big\}^{-1}$, and $e_i \sim N(0,1)$.

The scenarios that are tested are both time-invariant (DGP 1-3) and time-varying (DGP 4-6). The degree of time varyingness that is tested is $b = 2$. Su and Wang (2017) also includes $b = 1$, however this is omitted due to time constraints.

With DGP 1-3, the aim is to test whether the hypothesis test correctly identifies time-invariant time series with different structures. DGP 4-6 models time series with different time varying structures: A singular structural break, multiple structural breaks, and a smooth structural break.

The factors are constructed as:

$$\begin{aligned} f_{1t} &= 0.6 f_{1,t-1} + u_1 \\ f_{2t} &= 0.3 f_{2,t-1} + u_2 \end{aligned} \tag{27}$$

where u_1 and u_2 are IID $N(0, 1-0.6^2)$ and $N(0, 1-0.3^2)$ respectively, which is the same as in Su and Wang (2017).

The simulated data is then constructed as: $\mathbf{r}_t^{sim} = \mathbf{\Lambda}_t^{sim}\mathbf{F}_t^{sim} + \mathbf{e}_t^{sim}$.

4 Empirical Evaluation

The out-of-sample performance of the method is evaluated using data consisting of stocks traded on the Swedish stock market during two periods, between 2015-12-31 and 2019-12-31, and 2020-12-31 and 2024-12-31. The reason is that we want to compare the performance when applied during a financially stable period, to when it is applied during a less financially stable period. After omitting assets that were not traded for the entire period, the first data set consists of 261 stocks, and the second set consists of 347 stocks. After cleaning the data of bank holidays, the first set includes 1004 time points, and the second set includes 1008, i.e., approximately 251-252 trading days per year. The Stockholm Interbank Offered Rate (STIBOR) is used as a proxy for the risk-free rate in the analysis. This is a benchmark of the rates of unsecured bonds between banks issued by the Swedish Financial Benchmark facility (SFBF) (Swedish Financial Benchmark Facility, n.d.). Using STIBOR as a proxy for the risk-free rate is a common approach when studying the Swedish market (e.g. Dahlquist et al., 2000; Engström, 2004). The STIBOR data consists of annualized interest rates reported as tomorrow/next. The data was retrieved from Refinitiv Eikon (2025).

Log returns $\mathbf{r}_t = \log(\mathbf{P}_t) - \log(\mathbf{P}_{t-1})$, and logged risk free rate $\mathbf{r}_f = \log(1 + \mathbf{r}_f^{ann}/252)$ is used throughout the analysis. The risk-free rate is rescaled to reflect the daily rate, there are approximately 252 trading days every year.

To mimic a real-world investment environment, the performance is evaluated using weekly and monthly rebalancing. The different scenarios that is used to evaluate the performance are 50, 150, and 250 randomly selected stocks traded on the Swedish stock market. The performance is evaluated using a expanding window that uses the first year (2016/2021) as the initial window (252 time points), and rolls over the next three years (752/756 time points). The number of factors is updated yearly using `determine_factors()` with `max_m=10`. We assume no transaction cost and that there are no restrictions on shorting. The metrics that are used to evaluate the model are cumulative log excess returns (CER), mean log excess returns (MER), standard deviation (SD), Sharpe ratio (SR), as well as maximum drawdown (MDD), which is computed as:

$$\begin{aligned} MDD &= \max_t \{\text{Drawdown}_t\} \\ \text{where Drawdown}_t &= 1 - \frac{V_t}{\max_{\tau \in [0,t]} V_\tau}. \end{aligned} \tag{28}$$

Here, V_t denotes the cumulative simple returns at time t. MDD is a metric of how stable the portfolio returns are over time, indicating whether there have been large negative spikes in returns during the period. Since it measures the difference between the peak and the largest through in percentage, we have to convert the log excess returns to simple excess returns to compute the metric.

As the weights are optimized to minimize the covariance between the assets in the portfolio, the most important metric is the standard deviation (risk). However, CER and MDD are also be important when evaluating the performance, as low-risk portfolios would be expected to have stable excess returns over time.

The results of TV-MVP are compared to other popular methods of estimating the covariance of the returns, namely: Sample covariance, Ledoite-Wolf shrinkage (Schafer et al., 2021; Schäfer and Strimmer, 2005), exponentially weighted moving average (EWMA) (Longerstaey and Spencer, 1996; Reckziegel, 2025), Principal Orthogonal ComplEment Thresholding (POET) (J. Fan et al., 2013, 2016), and Graphical Lasso (Glasso) (Friedman et al., 2007, 2019). As a baseline, an equally weighted portfolio is also included. Shrinkage and Glasso are methods which both aim to regularize an ill-conditioned sample covariance matrix, where Ledoit-Wolf Shrinkage aims to shrink the sample covariance towards a well-conditioned covariance matrix, and Glasso aims to estimate a sparse inverse covariance matrix (Friedman et al., 2007; Schäfer and Strimmer, 2005). EWMA and POET share characteristics of TV-MVP, which makes the comparison of these methods interesting. EWMA apply larger weights to recent observations when estimating the covariance matrix, similar to the kernel-weighting approach used in TV-

MVP, while POET, similairly to TV-MVP, uses a factor approach to estimate the covariance (J. Fan et al., 2013; Longerstaey and Spencer, 1996).

5 Results and Discussion

In this section, the results from the simulation study and the empirical evaluation are presented and discussed. The complete results can be found in the *results* folder of the GitHub repository *CodeForThesis* (Lillrank, 2025). In the same repository, you also find the necessary scripts for conducting the simulation study and the empirical analysis.

5.1 Simulation Results

Table 1 presents the proportions of bootstrap p-values exceeding various thresholds for each data generating process (DGP) in our test of constant factor loadings, where $H_0 : \boldsymbol{\lambda}_{it} = \boldsymbol{\lambda}_{i0}$ for $i = 1, 2, \ldots, p$ and $t = 1, 2, \ldots, T$ vs. $H_1 : \boldsymbol{\lambda}_{it} \neq \boldsymbol{\lambda}_{i0}$ for some i, t.

DGPs 1-3 feature time-invariant loadings. As shown in the table, the hypothesis test correctly classifies these cases as time-invariant more than 80% of the time. In contrast, DGPs 4-6 are generated with time-varying factor loadings. As can be seen from table 1, the hypothesis test correctly identifies the time-varying DGP's 100% of the time, for all p-value thresholds. This suggests that the test reliably distinguishes between time-invariant and time-varying loadings.

Table 1: Rejection rates of hypothesis test in Monte Carlo simulation: Proportions of bootstrap p-values above thresholds for each DGP in hypothesis test of constant loadings (see section 2.3 and Su and Wang (2017)). $H_0 : \boldsymbol{\lambda}_{it} = \boldsymbol{\lambda}_{i0}$ for $i = 1, 2, \ldots, p$ and $t = 1, 2, \ldots, T$ vs. $H_1 : \boldsymbol{\lambda}_{it} \neq \boldsymbol{\lambda}_{i0}$ for some i, t.

Threshold	DGP1	DGP2	DGP3	DGP4	DGP5	DGP6
$p < 0.1$	0.148	0.112	0.198	1.000	1.000	1.000
$p < 0.05$	0.114	0.080	0.154	1.000	1.000	1.000
$p < 0.01$	0.066	0.056	0.096	1.000	1.000	1.000

These results are in line with those found by Su and Wang (2017). Looking at the results of their simulation study in Tables 3 and 4 (Su and Wang, 2017), we see that their results show slightly better accuracy in identifying the time-invariant case, but with very similar results for identifying the time-varying case. The discrepancies found when comparing the results for DGP's 1-3 could be due to some slight differences in the implementation of the data-generating processes. Except for these small discrepancies, the results found here match the ones found by Su and Wang (2017), which confirms that the hypothesis test has been correctly implemented in the package.

5.2 Empirical Results

The results from the empirical analysis can be seen in tables 2 and 3, for 2017-2019 and 2022-2024, respectively. Within the tables, subtables show the results for weekly and monthly rebalancing separately. The asset pools that have been used to construct the portfolios consist of 50, 150, and 250 stocks, which were randomly chosen out of the 261 (table 2) and 347 (table 3) stocks in the data set.

In table 2 we see the results from the expanding window during 2017-2019. For both weekly and monthly rebalancing, we see low, positive returns for the equal weights portfolio and larger positive returns for the optimized portfolios, except for Glasso, which performs poorly concerning cumulative log excess returns (CER). Looking at the SD, we see that the shrinkage method of estimating the covariance matrix yields the lowest SD for all asset pools, with TV-MVP tied for lowest SD with p=150, and second lowest SD for p=250. We see quite similar results between the portfolios in regards to risk, with only slightly lower SD compared to the equal weights portfolio for all optimized portfolios.

If we instead look at the other metrics of interest, we see that POET has the largest CER and SR, and the lowest MDD, for all asset pools. This indicates that this portfolio has the steadiest upward development during the period, which would indicate that this is the portfolio that has had the best performance. In this aspect, TV-MVP performs slightly worse compared with Sample, Shrink, EWMA, and POET covariance estimation. These have larger CER and SR, while having lower MDD for p=50 when using both weekly and monthly rebalancing. For p=150, we see that TV-MVP has the second lowest MDD in table 2a, while for the monthly rebalancing in table 2b, it has slightly higher MDD than the other portfolios (except Glasso and 1/N). We also see that TV-MVP has lower CER and SR for p=150 when compared to Sample, Shrink, EWMA, and POET, for both weekly and monthly rebalancing. For p=250, we see that the performance of TV-MVP is more competitive compared to the other portfolios, with the third highest CER and SR, and the second lowest MDD, for both weekly and monthly rebalancing.

Looking at the results for 2022-2024 in table 3, for all three asset pools, we see that the equal weights portfolio produces negative cumulative excess log returns, mean excess log returns, and Sharpe ratio. This shows that, on average, the assets within the pools have had a negative development during the period. As such, it should be no surprise that the optimized portfolios also show negative returns during the period. The MVP is optimized based on minimizing the inter-asset correlation rather than maximizing returns; hence, in a declining market, we would not expect these portfolios to perform well concerning excess returns. However, what we do see is that the negative CER is smaller for the optimized portfolios, indicating that some of the loss in returns has been avoided due to diversification.

Looking at SD, our metric measuring risk, what we see in the tables 3a and 3b is that for p=50, the SD is quite similar between the portfolios, with the portfolios using the sample and EWMA covariance estimation producing the lowest risk at 0.0094. The similarity in results could be an indication that the variation in the asset pool is small; this would explain why all portfolios perform similarly to the equal weights portfolio. For p=150 and p=250, we see that TV-MVP produces the lowest SD with POET being a close second.

For all portfolios, we see large maximum drawdowns, indicating financial instability within the asset pools. What we see is that POET and TV-MVP have the lowest MDD out of all portfolios, however, POET has better performance in this aspect for $p = 50$ and 150, for both weekly and monthly rebalancing, and for $p = 250$, the resulting MDD are very similar between the two portfolios. If we also look at CER, we see in table 3a that POET has positive returns during the period when using p=50, and the smallest losses when $p = 150$, while Sample covariance has the lowest negative CER with $p = 250$, with TV-MVP being a close second. For monthly rebalancing (table 3b), POET has the lowest loss of returns for $p = 50$ and 150, and EWMA for $p = 250$. TV-MVP does not outperform the other methods in terms of portfolio returns and drawdowns; however, it is consistently among the top three portfolios in this regard, and is the top performer in terms of SD.

These results are similar to those found by Q. Fan et al. (2024), and Wang et al. (2021), which have used very similar methods for estimation of a time-varying covariance matrix to optimize the minimum variance portfolio. Q. Fan et al. (2024) compares TV-MVP to different established methods and found TV-MVP to be competitive but not better than methods such as POET and DCC. Wang et al. (2021) compared the performance of their time-varying covariance against POET in their empirical analysis, trying different numbers of factors, and found that the methods perform similarly. These results are in line with what can be seen here, where TV-MVP generally has some of the lowest risk, especially in financially turbulent times, but it is not clearly better than the other methods with which it is being compared.

6 Concluding Remarks

In this thesis, we further developed and evaluated a recently introduced time-varying factor model for estimating the covariance matrix in a dynamic setting, with particular emphasis on portfolio optimization. Our methodology adapts a kernel-weighted principal component approach to capture evolving factor loadings and uses a shrinkage-based residual covariance estimation to address potential high-dimensional challenges. We package this in a user-friendly R package to facilitate practical adoption by researchers and practitioners alike.

Through simulation experiments, we confirmed that the included hypothesis test for time-invariance in factor loadings effectively distinguishes between constant and time-varying covariance structures. Empirical applications to Swedish Stock data from stable (2017-2019) and volatile (2022-2024) market conditions showed that the proposed time-varying minimum variance portfolio (TV-MVP) can achieve comparatively low volatility. While it does not outperform established methods (e.g., POET and shrinkage estimators) in terms of portfolio returns and Sharpe ratio, it remains broadly competitive.

For future research, we leave several opportunities for expansion and improvements of the method. First, evaluation of the maximum Sharpe ratio portfolio as well as other portfolio optimization techniques, as well as more sophisticated modelling of the expected returns, which could enhance the performance of these portfolios. Second, implementing different regularization options for the residual covariance. Finally, expanding the package with more methods of portfolio optimization and refining the code base for more efficient computation. By providing a user-friendly implementation of the method and proving its competitiveness against established methods, we anticipate that time-varying factor approaches will become increasingly central to portfolio management.

References

Bickel, P. J., & Levina, E. (2008). Regularized estimation of large covariance matrices. *The Annals of Statistics*, *36*(1), 199–227.

Bollerslev, T., Engle, R. F., & Wooldridge, J. M. (1988). A capital asset pricing model with time-varying covariances. *Journal of Political Economy*, *96*(1), 116–131.

Campbell, J. Y., Lettau, M., Malkiel, B. G., & Xu, Y. (2001). Have individual stocks become more volatile? an empirical exploration of idiosyncratic risk. *The Journal of Finance*, *56*(1), 1–43.

Chen, J., Li, D., & Linton, O. (2019). A new semiparametric estimation approach for large dynamic covariance matrices with multiple conditioning variables. *Journal of Econometrics*, *212*(1), 155–176.

Chen, Z., & Leng, C. (2016). Dynamic covariance models. *Journal of the American Statistical Association*, *111*(515), 1196–1207.

Chopra, V. K., & Ziemba, W. T. (1993). The effect of errors in means, variances, and covariances on optimal portfolio choice. *The Journal of Portfolio Management*, *19*(2), 6–11.

Dahlquist, M., Engström, S., & Söderlind, P. (2000). Performance and characteristics of swedish mutual funds. *Journal of Financial and Quantitative Analysis*, *35*(3), 409–423.

Engle, R. (2002). Dynamic conditional correlation. *Journal of Business & Economic Statistics*, *20*(3), 339–350.

Engle, R. F., National Bureau of Economic Research, & Sheppard, K. (2001). *Theoretical and empirical properties of dynamic conditional correlation multivariate garch*. National Bureau of Economic Research.

Engström, S. (2004). *Investment strategies, fund performance and portfolio characteristics* (SSE/EFI Working Paper Series in Economics and Finance No. 554). Stockholm School of Economics (SSE).

Fan, J., Liao, Y., & Mincheva, M. (2016). *Poet: Principal orthogonal complement thresholding (poet) method* [R package version 2.0]. https://CRAN.R-project.org/package=POET

Fan, J., Liao, Y., & Mincheva, M. (2013). Large covariance estimation by thresholding principal orthogonal complements. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, *75*(4), 603–680.

Fan, Q., Wu, R., Yang, Y., & Zhong, W. (2024). Time-varying minimum variance portfolio. *Journal of Econometrics*, *239*(2), 105339.

Friedman, J., Hastie, T., & Tibshirani, R. (2019). *Glasso: Graphical lasso: Estimation of gaussian graphical models* [R package version 1.11]. https://CRAN.R-project.org/package=glasso

Friedman, J., Hastie, T., & Tibshirani, R. (2007). Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, *9*(3), 432–441.

Ledoit, O., & Wolf, M. (2003). Improved estimation of the covariance matrix of stock returns with an application to portfolio selection. *Journal of Empirical Finance*, *10*(5), 603–621.

Ledoit, O., & Wolf, M. (2004). A well-conditioned estimator for large-dimensional covariance matrices. *Journal of Multivariate Analysis*, *88*(2), 365–411.

Lillrank, E. (2025). Codeforthesis [GitHub repository. Accessed 9 Jun 2025]. https://github.com/erilill/CodeForThesis

Lillrank, E., & Yang, Y. (2025). TVMVP: Time–varying minimum variance portfolio (beta) [GitHub repository. Accessed 9 Jun 2025]. https://github.com/erilill/TV-MVP/tree/beta

Longerstaey, J., & Spencer, M. (1996). *Riskmetrics technical document* (tech. rep. No. 51) (p. 54). Morgan Guaranty Trust Company of New York. New York.

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

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

NG, L. (1991). Tests of the capm with time-varying covariances: A multivariate garch approach. *The Journal of Finance*, *46*(4), 1507–1521.

Palomar, D. P. (2025). *Portfolio optimization* [Bookdown web edition, accessed 9 May 2025]. https://bookdown.org/palomar/portfoliooptimizationbook/

Pelletier, D. (2006). Regime switching for dynamic correlations. *Journal of Econometrics*, *131*(1), 445–473.

R Core Team. (2021). *R: A language and environment for statistical computing*. R Foundation for Statistical Computing. Vienna, Austria. https://www.R-project.org/

Reckziegel, B. (2025). *Portfoliomoments: Functions to be used in conjuction with portfolioanalytics* [R package version 0.1.0]. https://github.com/Reckziegel/PortfolioMoments

Refinitiv Eikon. (2025). Refinitiv eikon financial data platform [Accessed: 2025-02-26]. https://eikon.refinitiv.com/

Schafer, J., Opgen-Rhein, R., Zuber, V., Ahdesmaki, M., Silva, A. P. D., & Strimmer., K. (2021). *Corpcor: Efficient estimation of covariance and (partial) correlation* [R package version 1.6.10]. https://CRAN.R-project.org/package=corpcor

Schäfer, J., & Strimmer, K. (2005). A shrinkage approach to large-scale covariance matrix estimation and implications for functional genomics. *Statistical Applications in Genetics and Molecular Biology*, *4*(1).

Su, L., & Wang, X. (2017). On time-varying factor models: Estimation and testing. *Journal of Econometrics*, *198*(1), 84–101.

Swedish Financial Benchmark Facility. (n.d.). *Stibor* [Accessed: 2023-03-10]. https://swfbf.se/stibor/

Wang, H., Peng, B., Li, D., & Leng, C. (2021). Nonparametric estimation of large covariance matrices with conditional sparsity. *Journal of Econometrics*, *223*(1), 53–72.

Wu, R., & Fan, Q. (2024). TV-MVP: Code for time-varying minimum-variance portfolio [GitHub repository. Accessed 20 May 2025]. https://github.com/RuikeWu/TV-MVP

Table 2: Out-of-sample performance of minimum variance portfolios estimated with six covariance methods ($1/N$, Sample, Ledoit-Wolf Shrinkage, EWMA, POET, Glasso, and the proposed TV-MVP). Portfolios are built on three asset-pool sizes - $p = 50,\ 150,\ 250$ stocks randomly drawn from the 261 Swedish equities in the data set, and rebalanced either **weekly** (Panel A) or **monthly** (Panel B). The initial estimation window is calendar year 2016; the evaluation spans 2017-2019. Reported metrics are cumulative excess log returns (CER), mean excess log returns (MER), standard deviation (SD), Sharpe ratio (SR), and maximum drawdown (MDD).

(a) Weekly Rebalancing

	p = 50					**p = 150**					**p = 250**				
Method	**CER**	**MER**	**SD**	**SR**	**MDD**	**CER**	**MER**	**SD**	**SR**	**MDD**	**CER**	**MER**	**SD**	**SR**	**MDD**
1/N	0.0030	3.969e-06	0.0070	0.0006	0.2401	0.0491	6.523e-05	0.0066	0.0099	0.2422	0.1063	1.413e-04	0.0065	0.0218	0.2071
Sample	0.4210	5.599e-04	0.0064	0.0875	0.1076	0.2975	3.956e-04	0.0059	0.0668	0.1218	0.1642	2.184e-04	0.0070	0.0311	0.1568
Shrink	0.3687	4.903e-04	0.0063	0.0780	0.1081	0.2385	3.171e-04	0.0056	0.0567	0.1346	0.2438	3.242e-04	0.0051	0.0635	0.1485
EWMA	0.4156	5.527e-04	0.0064	0.0864	0.1071	0.2991	3.978e-04	0.0060	0.0668	0.1275	0.1718	2.285e-04	0.0070	0.0325	0.1569
POET	0.6072	8.075e-04	0.0071	0.1143	0.1003	0.4294	5.710-04	0.0058	0.0978	0.0990	0.4506	5.992e-04	0.0056	0.1067	0.0798
Glasso	0.0309	4.114e-05	0.0068	0.0060	0.2292	0.0634	8.429e-05	0.0066	0.0128	0.2345	0.1210	1.609e-04	0.0065	0.0249	0.2045
TV-MVP	0.3129	4.160e-04	0.0066	0.0632	0.1181	0.2348	3.123e-04	0.0056	0.0558	0.1143	0.2785	3.704e-04	0.0053	0.0703	0.1142

(b) Monthly Rebalancing

	p = 50					**p = 150**					**p = 250**				
Method	**CER**	**MER**	**SD**	**SR**	**MDD**	**CER**	**MER**	**SD**	**SR**	**MDD**	**CER**	**MER**	**SD**	**SR**	**MDD**
1/N	0.0030	3.969e-06	0.0070	0.0006	0.2401	0.0491	6.523e-05	0.0066	0.0099	0.2422	0.1063	1.413e-04	0.0065	0.0218	0.2071
Sample	0.4250	5.652e-04	0.0064	0.0881	0.1090	0.2968	3.946e-04	0.0060	0.0658	0.1256	0.1565	2.082e-04	0.0082	0.0255	0.1488
Shrink	0.3687	4.903e-04	0.0063	0.0778	0.1095	0.2476	3.292e-04	0.0057	0.0582	0.1336	0.2681	3.565e-04	0.0051	0.0694	0.1419
EWMA	0.4325	5.752e-04	0.0064	0.0900	0.1070	0.3199	4.254e-04	0.0060	0.0707	0.1207	0.1726	2.295e-04	0.0082	0.0281	0.1436
POET	0.5644	7.505e-04	0.0070	0.0982	0.1124	0.4359	5.796e-04	0.0059	0.0990	0.0989	0.4462	5.934e-04	0.0056	0.1055	0.0798
Glasso	0.0300	3.987e-05	0.0069	0.0058	0.2295	0.0625	8.316e-05	0.0066	0.0127	0.2350	0.1205	1.603e-04	0.0065	0.0248	0.2046
TV-MVP	0.2337	3.108e-04	0.0066	0.0469	0.1352	0.2252	2.994e-04	0.0057	0.0522	0.1421	0.2566	3.413e-04	0.0054	0.0637	0.1399

Table 3: Out-of-sample performance of minimum variance portfolios estimated with six covariance methods ($1/N$, Sample, Ledoit-Wolf Shrinkage, EWMA, POET, Glasso, and the proposed TV-MVP). Portfolios are built on three asset-pool sizes - $p = 50,\ 150,\ 250$ stocks randomly drawn from the 261 Swedish equities in the data set, and rebalanced either **weekly** (Panel A) or **monthly** (Panel B). The initial estimation window is calendar year 2021; the evaluation spans 2022-2024. Reported metrics are cumulative excess log returns (CER), mean excess log returns (MER), standard deviation (SD), Sharpe ratio (SR), and maximum drawdown (MDD).

(a) Weekly Rebalancing

	p = 50					**p = 150**					**p = 250**				
Method	**CER**	**MER**	**SD**	**SR**	**MDD**	**CER**	**MER**	**SD**	**SR**	**MDD**	**CER**	**MER**	**SD**	**SR**	**MDD**
1/N	-0.4863	-6.432e-4	0.0120	-0.0536	0.5170	-0.6028	-7.974e-4	0.0114	-0.0700	0.5338	-0.6576	-8.700e-4	0.0109	-0.0795	0.5250
Sample	-0.1858	-2.457e-4	0.0094	-0.0263	0.3037	-0.2382	-3.151e-4	0.0074	-0.0423	0.3727	-0.2798	-3.701e-4	0.0118	-0.0314	0.4859
Shrink	-0.2616	-3.461e-4	0.0096	-0.0360	0.3882	-0.4404	-5.825e-4	0.0076	-0.0764	0.4683	-1.0124	-1.339e-3	0.0134	-0.0998	0.6592
EWMA	-0.1795	-2.375e-4	0.0094	-0.0251	0.2955	-0.2459	-3.253e-4	0.0074	-0.0437	0.3769	-0.3007	-3.977e-4	0.0121	-0.0329	0.4932
POET	0.0064	8.449e-6	0.0100	0.0009	0.2314	-0.0498	-6.589e-5	0.0070	-0.0096	0.1950	-0.3250	-4.312e-4	0.0071	-0.0603	0.3651
Glasso	-0.4597	-6.081e-4	0.0118	-0.0514	0.5026	-0.5766	-7.626e-4	0.0113	-0.0675	0.5253	-0.6304	-8.339e-4	0.0109	-0.0767	0.5188
TV-MVP	-0.1500	-1.985e-4	0.0097	-0.0205	0.2848	-0.1999	-2.645e-4	0.0067	-0.0397	0.2859	-0.3248	-4.297e-4	0.0070	-0.0618	0.3642

(b) Monthly Rebalancing

	p = 50					**p = 150**					**p = 250**				
Method	**CER**	**MER**	**SD**	**SR**	**MDD**	**CER**	**MER**	**SD**	**SR**	**MDD**	**CER**	**MER**	**SD**	**SR**	**MDD**
1/N	-0.4863	-6.432e-4	0.0120	-0.0536	0.5170	-0.6028	-7.974e-4	0.0114	-0.0700	0.5338	-0.6576	-8.699e-4	0.0109	-0.0795	0.5250
Sample	-0.1994	-2.638e-4	0.0094	-0.0280	0.3143	-0.2779	-3.675e-4	0.0076	-0.0482	0.3887	-0.1928	-2.551e-4	0.0112	-0.0228	0.459
Shrink	-0.2716	-3.593e-4	0.0097	-0.0371	0.3964	-0.4535	-5.999e-4	0.0077	-0.0779	0.4733	-1.0047	-1.329e-3	0.0134	-0.0990	0.6571
EWMA	-0.1763	-2.332e-4	0.0094	-0.0249	0.3012	-0.2756	-3.646e-4	0.0076	-0.0477	0.3895	-0.1926	-2.548e-4	0.0112	-0.0228	0.4550
POET	-0.0564	-7.455e-5	0.0100	-0.0075	0.2291	-0.0817	-1.081e-4	0.0070	-0.0155	0.2156	-0.3531	-4.671e-4	0.0073	-0.0641	0.3809
Glasso	-0.4588	-6.068e-4	0.0118	-0.0513	0.5022	-0.5778	-7.642e-4	0.0113	-0.0676	0.5254	-0.6319	-8.359e-4	0.0109	-0.0769	0.5190
TV-MVP	-0.0595	-7.870e-5	0.0102	-0.0077	0.2301	-0.1974	-2.611e-4	0.0067	-0.0391	0.3038	-0.3385	-4.500e-4	0.0066	-0.0675	0.3772