ABSTRACT

INTRODUCING DCC-SVR, A HYBRID FORECASTING METHOD FOR LARGE COVARIANCE MATRICES

Michael Nebor, Ph.D.
Department of Economics
Northern Illinois University, 2025

This paper introduces DCC-SVR, a hybrid Dynamic Conditional Correlation (DCC) and Support Vector Regression (SVR) method of forecasting the covariance matrix. This paper shows that DCC-SVR is able to outperform the traditional methods of DCC and rolling historical on multiple data sets. Performance is shown for both standard GARCH and GJR-GARCH methods. This paper also analyzes performance when dimensions are increased to 49 dimensions and when an application using equal weighted portfolio allocation is used.

NORTHERN ILLINOIS UNIVERSITY
DE KALB, ILLINOIS

AUGUST 2025

INTRODUCING DCC-SVR, A HYBRID FORECASTING METHOD FOR LARGE COVARIANCE MATRICES

BY

MICHAEL NEBOR
© 2025 Michael Nebor

A DISSERTATION SUBMITTED TO THE GRADUATE SCHOOL

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE

DOCTOR OF PHILOSOPHY

DEPARTMENT OF ECONOMICS

Dissertation Director:

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 2

INTRODUCING DCC-SVR, A HYBRID FORECASTING METHOD FOR LARGE COVARIANCE MATRICES

2.1 Introduction

Accurately forecasting the covariance matrix is an important task as it can lead to accurate portfolio allocations that will help achieve investor goals. Machine learning (ML) methods have shown to be excellent tools to use in various forecasting situations. In Dudek, Fiszeder, Kobus, and Orzeszko (2024), various machine learning methods were applied to forecast the variances of cryptocurrencies and the performance of each method was compared. The authors found there to be no single superior forecasting method as the best method changed depending on the currency, the forecast length, and the error method used. They found that support vector regression (SVR) typically performed well and remained among the best performing methods.

A promising ML research paper is Fiszeder and Orzeszko (2021) where the authors introduced a covariance matrix forecasting method using SVR. The method works where past covariance matrices are decomposed using their Cholesky decomposition, then SVR is used to predict each point of the Cholesky matrix for a future period. This matrix is then converted into the forecasted covariance matrix. This method performed well and showed able to outperform Dynamic Conditional Correlation (DCC), but these forecasts were based on using different amounts of input data making them not directly comparable.

DCC models are statistical time series models used to forecast correlation and covariance matrices. They are known to work well and are able to outperform other statistical methods. An issue with DCC models is that they become complex when expanded into large dimensions. This causes them to require numerous calculations and take a longer time to obtain.

A positive attribute of DCC models is that they are able to convert units within the model. DCC models require the input of asset returns and are able to convert the units so that the resulting outputs are in the form of variances and covariances.

Proposed ML methods for forecasting the covariance matrix have been based on requiring historical variance and covariance values as the inputs needed to train the ML model. Since they are trained on historical variances, the asset returns need to be converted externally prior to being fed into the model. This external conversion requires additional data to be provided. This additional data can be either historical return data of a higher frequency to compute historical covariance values, or it can be the high and low values within each period to compute historical range covariance values (Fiszeder & Orzeszko, 2021). Either way, additional data is required for this external form of unit conversion.

The goal of this paper is to introduce the hybrid DCC-SVR method of forecasting the covariance matrix and to compare its results and run time to other methods. This is a hybrid method which combines positive aspects of both statistical and machine learning methods. From use of DCC, it is able to convert the units internally and is based soley on historical return data. From its use of SVR comes increased accuracy in predictions over traditional DCC.

The forecasting methods used in this paper are rolling historical, Dynamic Conditional Correlation (DCC-) GARCH, and DCC-SVR. Additionally, there are two variations of the DCC and DCC-SVR methods included in analysis. The first version is based on standard GARCH models and the second version is based on asymmetric GJR-GARCH models.

The results of this paper establish superiority of the DCC-SVR method over the traditional method of DCC. Unlike the ML methods used in previous papers, DCC-SVR is able to outperform traditional DCC while using the exact same base data set information. DCC-SVR is able to perform well with both forms of standard GARCH and GJR-GARCH models. When expanded to large dimensions, performance of the DCC-SVR method continues to hold.

Since the DCC-SVR method is built off of DCC, it also adopts the DCC weakness of being computationally burdensome when extended to large dimensions. To address this issue and show the reader an easy and practical solution, the large matrix re-representation method of Fan is introduced (Fan, Fan, & Lv, 2008). The results show that the Fan expansion method has comparable forecasting accuracy while requiring much less time to compute.

2.2 Method

A rolling window is used to create the forecasts. Monthly data within the rolling window is used to create the one month ahead covariance matrix forecast, $\hat{\Sigma}$. Realized daily data within the month is used to compute the proxy for the true values of the monthly covariance matrix, Σ. The variance and covariance entries, $\sigma_{ij,t}$, within the proxy matrix, Σ_t, are found using Eq. 2.1.

$$\sigma_{ij,t} = \sum_{\ell=1}^{d_t}(y_{i,\ell,t} - \overline{y}_{i,t})(y_{j,\ell,t} - \overline{y}_{j,t}) \qquad \forall\, i, j \in 1, 2, ..., P \tag{2.1}$$

In Eq. 2.1, d_t is the total number of days within month t, $y_{i,\ell,t}$ represents the return of asset i on day ℓ during month t, and $\overline{y}_{i,t}$ is the daily mean return of asset i during month t. P represents the number of assets in the portfolio.

To measure the error between the forecasted variance and covariance matrix values, $\hat{\sigma}_{ij,t}$, and their true values, $\sigma_{ij,t}$, a distance measure is used. The absolute forecast error (AFE) is the absolute error of each variance and covariance value within the matrix summed together. This is shown in Eq. 2.2. When applied to each month t forecast this becomes a time series vector of AFE loss values. The mean absolute forecast error (MAFE) is the average of the vector of loss values. This is shown in Eq. 2.3. While these equations show the loss for all values in the matrix, they can be extended so variance and covariance values are analyzed separately. The variances are examined by summing the error of the diagonal matrix values when $i = j$. The covariances are examined by summing the error of the lower triangular matrix values when $i < j$. In Eq. 2.3, N represents the number of out of sample forecasts made.

$$\mathrm{AFE}_t = \sum_{i=1}^{P}\sum_{j=1}^{P} |\sigma_{ij,t} - \hat{\sigma}_{ij,t}| \tag{2.2}$$

$$\mathrm{MAFE} = \frac{1}{N}\sum_{t=1}^{N} \mathrm{AFE}_t \tag{2.3}$$

The model confidence set (MCS) test is used to statistically compare the various methods against each other (Hansen, Lunde, & Nason, 2011). The idea is to create a superior set of models, where all models start in the superior set, then an iterative process is used to test for statistical equality of all methods and if not equal, then the worst performing model is removed. At the end of this process, the methods that remain in the superior set are statistically superior to the methods that were removed. These tests were performed using the rugarch package.

2.2.1 Forecasting Methods

2.2.1.1 Rolling Historical

The rolling historical method is shown in Eq. 2.4. This method is used as a benchmark for comparing the results. It works by using the monthly rolling window return data to create a sample covariance matrix. I obtained these using cov() in the R programming language. This covariance matrix is then used as the forecasted covariance matrix for the next period.

$$\hat{\Sigma}_{t+1}^{RollingHistorical} = \frac{1}{R-1}\sum_{r=t-R}^{t}(X_r - \overline{X}_t)(X_r - \overline{X}_t)' \tag{2.4}$$

Where:

$$X_r = \begin{pmatrix} X_{1,r} \\ X_{2,r} \\ \vdots \\ X_{P,r} \end{pmatrix} \qquad \overline{X}_t = \begin{pmatrix} \overline{X}_{1,t} \\ \overline{X}_{2,t} \\ \vdots \\ \overline{X}_{P,t} \end{pmatrix}$$

R is the total number of months of the rolling window and r is a specific point within the current rolling window of time t. X_r representing a vector of monthly returns within the current rolling window period for all assets. $\overline{X}_t$ represents a vector of the rolling window means for all assets.

2.2.1.2 DCC-GARCH(1,1)

Another benchmark used for comparing results is Dynamic Conditional Correlation (DCC-) GARCH (Engle & Sheppard, 2001). DCC-GARCH(1,1) is a well known method for covariance matrix forecasting and is often used as a benchmark to test new methods. This method is based on using historical rolling window monthly returns to forecast 1 month ahead.

To understand DCC it is important to first understand GARCH. GARCH is able to predict next periods variance using weighted past residual observations and past realized variances (Bollerslev, 1986). Eq. 2.5 shows the equation for the GARCH(1,1) model used in this paper. In the equation the error term can be rewritten as $a_{i,t} = \sqrt{\sigma_{i,t}}\epsilon_{i,t}$ with $\epsilon_{i,t} \sim$ Normal(0,1). The condition $\alpha_{i,1} + \beta_{i,1} < 1$ ensures it is a stationary and mean reverting process. The scalar values of α_0, α_1, and β_1 are found using maximum likelihood estimation.

$$\hat{\sigma}_{i,t+1}^{GARCH} = \alpha_{i,0} + \alpha_{i,1}a_{i,t}^2 + \beta_{i,1}\sigma_{i,t} \tag{2.5}$$

The results of this paper also include a popular variation of GARCH known as GJR-GARCH. GJR was developed to incorporate asymmetric effects into GARCH models (Glosten, Jagannathan, & Runkle, 1993; Ghalanos, 2024). The formula for GJR-GARCH is shown in Eq. 2.6.

$$\hat{\sigma}_{i,t+1}^{GJR-GARCH} = \alpha_{i,0} + \alpha_{i,1}a_{i,t}^2 + \gamma_{i,t}I_{i,t}a_{i,t}^2 + \beta_{i,1}\sigma_{i,t} \tag{2.6}$$

Where $I_{i,t}$ is an indicator function which equals 1 when $a_{i,t} < 0$ and 0 otherwise. $\gamma_{i,t}$ is the leverage term which is used to incorporate the asymmetrically effects of shocks.

DCC then builds on GARCH to create a covariance matrix forecast using the first line of Eq. 2.7.

$$\begin{aligned}
\hat{\Sigma}^{DCC}_{t+1} &= D_{t+1}\Omega_{t+1}D_{t+1} \\
D_{t+1} &= diag(\sqrt{\hat{\sigma}^{GARCH}_{1,t+1}}, ..., \sqrt{\hat{\sigma}^{GARCH}_{P,t+1}}) \\
\Omega_{t+1} &= Q^{*-1}_{t+1}Q_{t+1}Q^{*-1}_{t+1} \\
Q_{t+1} &= (1-a-b)\overline{Q} + ae_te'_t + bQ_t
\end{aligned} \tag{2.7}$$

where Ω_{t+1} is the dynamic conditional correlation matrix. The notation a and b are fitted weights found by maximum likelihood estimation, $\hat{\sigma}_{i,t+1}$ are the univariate standard GARCH or GJR-GARCH variance forecasts, $\epsilon_t = (\epsilon_{1,t}, ..., \epsilon_{P,t})'$ are the univariate GARCH error terms, $e_t = D_t^{-1}\epsilon_t$ are the standardized GARCH error terms, $\overline{Q} = E[e_te'_t]$ is the unconditional covariance matrix, and $Q^* = (diag(Q_{t+1}))^{\frac{1}{2}}$.

2.2.1.3 Hybrid DCC-SVR Method

This paper focuses on the ML method of SVR because it performed well at forecasting variance when compared to other ML methods (Dudek, Fiszeder, Kobus, & Orzeszko, 2024). SVR is the regression version of support vector machine (SVM). SVM was developed by Cortes and Vapnik (1995) as a classification method. Data points are projected into a higher dimension (the feature space) and a hyperplane is then used to separate the different classes of data and act as a decision boundary. The data points are projected to the higher dimension with use of a specified kernel function. While linear kernels can be used, nonlinear kernels such as Gaussian or polynomial kernels can also be used.

The only parameters that need to be specified for a linear kernel are ϵ and C. ϵ controls how many misclassified training observations become penalized. It essentially specifies an insensitive zone for when an observation is misclassified yet still within the grace zone, there will not be any penalty. C specifies the penalty that will be imposed for misclassified observations that are outside the insensitive zone. When the Gaussian kernel is used, it requires an additional parameter of γ. This parameter controls for the influence of a single training example on determining the decision boundary. This relates to decision boundary complexity due to it controlling how closely the hyperplane should be fit to the data. Compared to neural networks, SVR is known to be more robust to outliers and missing data, requires less training data samples to perform optimally, and does not require a complex architecture to be specified.

The following paragraph describes the SVR process. Similar descriptions of the SVR process can be found in related papers (Fiszeder & Orzeszko, 2021; Sun & Yu, 2020; Chen, Härdle, & Jeong, 2010).

The SVR function is given in Eq. 2.8.

$$f(x) = w'\varphi(x) + b \tag{2.8}$$

Where:

$$w = \begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_\ell \end{pmatrix} \qquad \varphi(x) = \begin{pmatrix} \varphi(x_1) \\ \varphi(x_2) \\ \vdots \\ \varphi(x_\ell) \end{pmatrix}$$

The training data set is $(x_t, y_t)_{t=1,\cdots,n}$, where the explanatory variables $x_t \in R^\ell$ and the response variable $y_t \in R^1$. The total number of training samples is n with t being an individual observation of the training data. $\varphi(x)$ is a vector of functions which projects the inputs to the feature space, w is a vector of weights that connect the feature space to the output space, and b is a bias term.

The primal soft margin optimization problem approach to solving SVR is shown in Eq. 2.9. In this equation $||w||^2 = w'w$. ξ_t and ξ_t^* are nonnegative slack variables used in combination with the loss function to represent upper and lower constraints. Note, b is contained within $f(x_t)$.

$$\min_{w,b,\xi_t,\xi_t^*} \frac{1}{2}||w||^2 + C\sum_{t=1}^{n}(\xi_t + \xi_t^*) \tag{2.9}$$

Subject to constraints:

$$f(x_t) - y_t \leq \epsilon + \xi_t$$

$$y_t - f(x_t) \leq \epsilon + \xi_t^*$$

$$0 \leq \xi_t, \xi_t^*$$

Taking a dual approach to solving the optimization problem leads to the solution of Eq. 2.10.

$$f(x, \alpha, \alpha^*) = \sum_{t=1}^{T_{sv}}(\alpha_t - \alpha_t^*)K(x_t, x) + b \tag{2.10}$$

Where α_t and α_t^* are Lagrange multipliers, T_{sv} is the number of support vectors, and K is the specified kernel function. The Gaussian kernel is used in this paper and its equation is shown in Eq. 2.11.

$$K(x_t, x) = \exp(-\gamma||x_t - x||^2), \qquad \gamma > 0, \tag{2.11}$$

The GARCH-SVR forecasting method was proposed in Sun and Yu (2020) as a hybrid univariate method that combines the forecasts of traditional GARCH with SVR. To show the effectiveness of their method they use the datasets of the daily closing price of the S&P 500 index and the daily exchange rate of the British pound against the US dollar. The authors showed their method outperforms both traditional GARCH and a previously proposed hybrid method known as SVR-GARCH (Peng, Albuquerque, Camboim de Sá, Padula, & Montenegro, 2018).

The hybrid DCC-SVR method proposed in this paper builds on GARCH-SVR by applying the GARCH-SVR process to each element of the DCC matrix. This paper uses the same SVR input specifications as the original paper. This is because their specified values worked well when used for all data sets. The specifications use a Gaussian kernel with the parameters of $C = .03$, $\epsilon = .0009$, and $\gamma = .1$. The SVR regressions are performed using the e1071 library, which is an R library with the specific purpose of allowing for application of SVR regression methods in the R programming language.

The following Eqs. 2.12 - 2.14 and the listed 7 steps demonstrate how to implement the DCC-SVR forecasting method. When reading the notation, t relates to the point in time of the forecast out of the entire dataset. The additional variable r relates to the specific point within the current time t rolling window data. The symbol v_t represents a list of $v_{ij,t,r}$ values. Eq. 2.12 shows that $v_{ij,t,r}$ is the variance and covariance values found using the 5 return periods of r through $r-4$. This represents a proxy of monthly variance and covariance values at the point in time of r based on using monthly data. Similar proxies have been used in previous papers (Pérez-cruz, Afonso-rodríguez, & Giner, 2003; Ou & Wang, 2010; Li, Liang, Li, Wang, & Wu, 2009; Santamaría-Bonfil, Frausto-Solís, & Vázquez-Rodarte, 2015; Sun & Yu, 2020).

$$v_{ij,t,r} = \frac{1}{4}\sum_{\ell=0}^{4}(y_{i,t,r-\ell} - \bar{y}_{i,t,r})(y_{j,t,r-\ell} - \bar{y}_{j,t,r}) \tag{2.12}$$

Where $y_{i,t,r}$ represents the monthly return of asset i during month t and $\bar{y}_{i,t,r}$ represents the mean return of asset i during the 5 periods of r through $r-4$.

In Eq. 2.13, the symbol $h_{t,r}$ represents a list of the diagonal and lower triangular entries of the DCC matrix. The entries of the resulting list are $h_{ij,t,r}$.

$$h_{t,r} = vech(DCC_{t,r}) \tag{2.13}$$

$z_{t,r}$ is a list containing the entries of $z_{ij,t,r}$. Eq. 2.14 shows $z_{ij,t,r}$ as the difference between $v_{ij,t,r}$ and $h_{ij,t,r}$. The difference between these two values can be thought of as a measurement of how far off the DCC forecasted values are from its true proxy values.

$$z_{ij,t,r} = v_{ij,t,r} - h_{ij,t,r} \tag{2.14}$$

Steps for hybrid DCC-SVR:

1. Use monthly rolling window return data of periods $(t-R), (t-R+1), \cdots, t$ to obtain the in sample fitted values $\hat{\Sigma}^{DCC}_{t,r}$ for $r = (t-R), (t-R+1), \cdots, t$ and the out of sample forecast $\hat{\Sigma}^{DCC}_{t,r=t+1}$

2. Obtain $v_{t,r}$, $h_{t,r}$, $z_{t,r}$ for $r = (t-R), (t-R+1), \cdots, t$.

3. Forecast $\hat{z}_{ij,t,r=t+1}$ values. Where $\hat{z}_{ij,t,r=t+1} = f_{ij,t}(z_{ij,t,r=t}, v_{ij,t,r=t}, y_{i,t,r=t} \times y_{j,t,r=t}, h_{ij,t,r=t})$ with f being an SVR function that was fitted using historical values of $r = (t-R), (t-R+1), \cdots, (t-1)$.

4. Combine DCC and SVR forecasted values.

$$\hat{v}_{ij,t,r=t+1} = \hat{z}_{ij,t,r=t+1} + \hat{h}_{ij,t,r=t+1}$$

Where $\hat{v}_{ij,t,r=t+1}$ represents the forecast of the true $\hat{\sigma}_{ij,t,r=t+1}$ values, $\hat{z}_{ij,t,r=t+1}$ is the SVR forecasted values, and $\hat{h}_{ij,t,r=t+1}$ represents the entries of the DCC forecasted matrix.

5. Recompose all forecasted $\hat{v}_{ij,t,r=t+1}$ values from list form back into symmetrical matrix form to obtain $\hat{\Sigma}^{DCC-SVR}_{t,r=t+1}$.

6. (OPTIONAL) Use $\hat{\Sigma}^{DCC-SVR}_{t,r=t+1}$ and the Fan method to obtain forecast of a larger portfolio.

7. This becomes the forecast for next period, $\hat{\Sigma}^{DCC-SVR}_{t+1} = \hat{\Sigma}^{DCC-SVR}_{t,r=t+1}$

Since the entries of the DCC-SVR method are created by adding a forecast and a residual term together, it is possible for the resulting matrix to not be positive definite and even potentially have negative values for its diagonal variance terms. If desired, an optional step between steps 5 and 6, can be added. This step would be to check the forecasted matrix for positive definiteness and if necessary convert the matrix. There are several methods available to convert a matrix to become positive definite. The choice of which method is best to use is up to the individual user. All results included in this paper portray results without the use of the optional step. When this step was implemented and the results were compared, there was almost no change to the results.

2.2.2 Expansion of Forecasted Matrix

Any positive semi-definite matrix that is also positive definite will have an inverse. When a covariance matrix has dimensions that are greater than the number of return samples, the matrix will no longer be positive definite and will no longer be invertible. To address this issue, Fan, Fan, and Lv (2008) shows that a factor model can be used to re-represent a large non-invertible matrix so that it is guaranteed to be invertible.

While originally meant to allow for matrix inversion, this paper will apply the Fan method in a forecasting context. Essentially, a user will first forecast a smaller dimensional factor matrix, then use OLS regression to convert the matrix into that of a higher dimensional matrix forecast. The author is unaware of any papers that have previously applied the Fan method in this manner. The Fan method can be applied to other matrix forecasting methods outside of the ones used in this paper. The Fan method is an ideal companion to forecasting methods that are highly accurate, yet computationally complex and hard to apply to large dimensions.

The Fan method is used in this paper as the optional step 6 listed in the hybrid DCC-SVR forecasting method. This optional step will also be applied to the DCC forecasting method. First, the factor covariance matrix $\hat{\Sigma}_{\mathrm{F},t}$ will be forecasted ($\hat{\Sigma}_{\mathrm{F},t+1}$) using a rolling window. The performance of the obtained $\hat{\Sigma}_{\mathrm{F},t+1}$ values will be analyzed (Tables 2.1 - 2.3). Next, the Fan method will be applied to the previously analyzed factor matrix forecasts ($\hat{\Sigma}_{\mathrm{F},t+1}$) to calculate the covariance matrix forecasts for the larger portfolios of equities ($\hat{\Sigma}_{t+1}$). The performance of these larger portfolio forecasts, $\hat{\Sigma}_{t+1}$, will then be analyzed in various ways.

The relationship between these two matrices can be seen clearly in the final line of Eq. 2.15. Analyzing both the factor and the larger matrices will show that when more accurate methods of forecasting the intermediary factor matrix are used, then the resulting forecasts for the larger portfolio $\hat{\Sigma}_{t+1}$ will become more accurate as well.

The process used for the Fan expansion method is illustrated in Eq. 2.15. These equations show that returns can be decomposed into what is explained by factors and an error term. This can then be expressed using matrix notation. The covariance of each portion can then be applied to show how this translates into the decomposition of a covariance matrix. This is then applied to the context of forecasting by replacing the time series values with forecasted values.

$$
\begin{aligned}
&y_{i,t,r} = \hat{b}_{1,i,t} f_{1,t,r} + \hat{b}_{2,i,t} f_{2,t,r} + ... + \hat{b}_{K,i,t} f_{K,t,r} + \epsilon_{i,t,r} \\
&\text{where} \quad i = 1, 2, ..., P, \quad r = (t-R), (t-R+1), ..., t \\
&Y_t = \hat{B}_t \mathrm{F}_t + \epsilon_t \\
&\mathrm{cov}(Y_t) = \mathrm{cov}(\hat{B}_t \mathrm{F}_t) + \mathrm{cov}(\epsilon_t) = \hat{B}_t \mathrm{cov}(\mathrm{F}_t) \hat{B}_t' + \mathrm{cov}(\epsilon_t) \\
&\Sigma_t = \hat{B}_t \Sigma_{\mathrm{F},t} \hat{B}_t' + \Sigma_{\epsilon,t} \\
&\hat{\Sigma}_{t+1} = \hat{B}_t \hat{\Sigma}_{\mathrm{F},t+1} \hat{B}_t' + \hat{\Sigma}_{\epsilon,t+1}
\end{aligned}
\tag{2.15}
$$

In the equations, b represents factor loading's obtained by OLS with use of the rolling window data. y is the dependent variable of the regression, representing a stock included in a portfolio. The subscript K indicates the number of factors used. The subscript i represents an individual asset, and uppercase letters indicate they are in matrix form.

If this method were to be applied using $K = 4$ factors and the $P = 10$ assets, this would result in $\hat{\Sigma}_{t+1}$ being a $[10, 10]$ matrix, $\hat{B}$ being a $[10, 4]$ matrix, $\hat{\Sigma}_{\mathrm{F},t+1}$ being a $[4, 4]$ matrix, and $\hat{\Sigma}_{\epsilon,t+1}$ being a $[10, 10]$ matrix.

The method used for $\hat{\Sigma}_{\epsilon,t+1}$ was held constant for all expanded forecasting methods. $\hat{\Sigma}_{\epsilon,t+1}$ can be found using the following steps. To help illustrate the idea, dimensions are given based on using the K factors, P assets, and a R rolling window size.

1. $\epsilon_{i,t,r} = y_{i,t,r} - (\hat{b}_{1,i,t} f_{1,t,r} + \hat{b}_{2,i,t} f_{2,t,r} + ... + \hat{b}_{K,i,t} f_{K,t,r})$, for each asset $i \in 1, 2, \cdots, P$ and for each period of the rolling window $r \in (t-R), (t-R+1), \cdots, t$ of time t which results in matrix ϵ_t of $[R,\ P]$

2. Multiply matrices, $\mathrm{cov}(\epsilon_t) = \epsilon_t' \times \epsilon_t$, results in a $[P,\ P]$ matrix

3. Divide, $\hat{\Sigma}_{\epsilon,t+1} = \frac{\mathrm{cov}(\epsilon_t)}{R}$, results in a $[P,\ P]$ matrix

The results of this paper were obtained using OLS regressions that do not include an intercept coefficient. The inclusion of an intercept term did not affect the results.

2.2.3 Application Method

The equal weighted portfolio allocation method was chosen to be used because of its simple nature. Eq. 2.16 shows the portfolio weights of this method are determined soley by the number of assets in the portfolio. This allows for the weights to remain easily interpretable, have minimal introduction of additional information that can confuse results, and will allow for the weighting values to remain constant across forecasting methods (as long as they are of the same portfolio size).

The portfolio weights are applied to the forecasted and proxy covariance matrices to find their portfolio standard deviation values. Eqs. 2.17 and 2.18 show how the matrix values are transformed into single values. The error distance measures of Eqs. 2.19 and 2.21 are then applied. When applied to all out of sample forecasts, this produces a series of error values. The mean squared tracking error (MSTE) and the mean absolute tracking error (MATE) of Eqs. 2.20 and 2.22 are averages of the series. The application of portfolio weighting allows for the use of new forms of measurement that help to support the robustness of results.

$$w_i = \frac{1}{P} \qquad \forall\, i \in \{1, 2, \cdots, P\} \tag{2.16}$$

$$\sigma_{port,t} = w_t' \Sigma_t w_t \tag{2.17}$$

$$\hat{\sigma}_{port,t} = w_t'\hat{\Sigma}_t w_t \tag{2.18}$$

$$\text{STE}_t = (\sqrt{\sigma_{port,t}} - \sqrt{\hat{\sigma}_{port,t}})^2 \tag{2.19}$$

$$\text{MSTE} = \frac{1}{N}\sum_{t=1}^{N}\text{STE}_t \tag{2.20}$$

$$\text{ATE}_t = |\sigma_{port,t} - \hat{\sigma}_{port,t}| \tag{2.21}$$

$$\text{MATE} = \frac{1}{N}\sum_{t=1}^{N}\text{ATE}_t \tag{2.22}$$

2.3 Data

The data used in this paper is from July 1963 through December 2022. A 240 month rolling window is used to create a 1 month ahead forecast. The first forecasted month is July 1983 and there are a total of 474 forecasts. Multiple data sets are analyzed to show robustness of the results. All data sets include dividends and all returns are in percentage form. Both the daily and monthly versions of the data sets are sourced.

The factor data is sourced from CRSP and consists of the 4 factors of Market Minus Risk Free Rate (Mkt-RF), Small Minus Big (SMB), High Minus Low (HML), and Momentum (Mom). These factors are known to work well at explaining the variance in equity returns (Fama & French, 1993; Carhart, 1997).

The Beta, Standard Deviation (SD), and Market Capitalization (Mkt) data sets were sourced from CRSP. Each of these data sets contain 10 dimensions of portfolio data. They are used as the larger portfolios because they are a larger dimension than the factor matrix, but still small enough that it is manageable to work with.

The Beta, SD, and Mkt-Cap portfolios were formed from real returns. They were created by sorting firms by their Beta, SD, or market capitalization into portfolios where the lowest 10% of firms would be contained in portfolio 1 and the highest 10% of firms would be sorted into portfolio 10. These 10 portfolios would contain the 10 dimensions for each covariance matrix.

The methods and results of this paper can be applied to portfolios that are much larger than 10 dimensions. To show the results hold when dimensions increase, the industry data sets of size 5, 12, 30, and 49 were used. They were formed by sorting real firms into portfolios based on their industry. This data is sourced from the Kenneth French data library.

The analysis was performed in R and the packages used were glmnet, e1071, rugarch, rmgarch.

2.4 Results

2.4.1 Factor Matrix Forecasts

Table 2.1 shows the performance of each method at forecasting the factor covariance matrix. In the table, the methods are grouped based on their comparability. The first group contains the standard GARCH methods and the second group contains the GJR-GARCH methods. The MCS test is performed on each group.

The table shows the hybrid method to be the overall best performing method of each group. This is because they have the lowest error levels for both groups. The MCS test results is also able to statistically support the superiority of the hybrid methods by including them in the superior set. Hybrid is then followed in performance by DCC and lastly rolling historical.

To gain a greater understanding of where the superior performance comes from, the matrix is segmented so the variance (Table 2.2) and the covariance (Table 2.3) values can be analyzed. The results of Tables 2.1, 2.2, and 2.3 are related to each other by Eq. 2.23. The results of these tables show the superiority of the hybrid methods is due to a superiority at forecasting both variance and covariance values.

$$\text{Full matrix error value} = \text{Variance error value } + 2 \times \text{Covariance error value} \tag{2.23}$$

Table 2.1: MAFE Full Matrix: 4 Factor Results

Method	
Rolling-Historical	136.66
DCC	121.57
Hybrid DCC-SVR	113.35*
Rolling-Historical	136.66
DCC GJR	113.47
Hybrid DCC-GJR-SVR	106.52*

This table analyzes the 4 dimensional factor matrix for all values of the matrix. Error is computed using Eq. 2.3. The * represents method is in the superior set of models obtained by MCS test with 95% confidence. The MCS test is performed on each group of comparable methods. Group 1 consists of the top 3 methods and group 2 consists of the bottom 3 methods.

Table 2.2: MAFE Variances: 4 Factor Results

Method	
Rolling-Historical	54.88
DCC	49.33
Hybrid DCC-SVR	44.48*
Rolling-Historical	54.88
DCC GJR	43.76
Hybrid DCC-GJR-SVR	39.86*

This table analyzes the variance values of the 4 dimensional factor matrix. Error is computed using Eq. 2.3 when $i = j$. The * represents method is in the superior set of models obtained by MCS test with 95% confidence. The MCS test is performed on each group of comparable methods. Group 1 consists of the top 3 methods and group 2 consists of the bottom 3 methods.

Table 2.3: MAFE Covariances: 4 Factor Results

Method	
Rolling-Historical	40.89
DCC	36.12
Hybrid DCC-SVR	34.43*
Rolling-Historical	40.89
DCC GJR	34.85
Hybrid DCC-GJR-SVR	33.33*

This table analyzes the covariance values of the 4 dimensional factor matrix. Error is computed using Eq. 2.3 when $i < j$. The * represents method is in the superior set of models obtained by MCS test with 95% confidence. The MCS test is performed on each group of comparable methods. Group 1 consists of the top 3 methods and group 2 consists of the bottom 3 methods.

2.4.2 Large Matrix Forecasts

Table 2.4 analyzes the performance of the forecasting methods when applied to various new data sets. The results of Table 2.4 sections 2.4A Direct and 2.4B Direct support the results of Table 2.1. These sections show the performance of each method and their relative rankings to be in line with the previous factor data set results. Appendix Tables A.1 and A.2 show the results when variances and covariances are specifically analyzed.

It is important to look at how the performance will change when the number of dimensions change. To keep numbers comparable under changing dimensions all error values on Tables 2.4, A.1, and A.2 are scaled by the number of values summed together. Entire matrix error values are divided by P^2, variance error values are divided by P, and covariance error values are divided by $P * (P - 1)/2$.

Use of the Ind 5 - Ind 49 data sets show that when the dimensions increase, the rankings of performance continue to hold with the DCC-SVR methods performing the best. This can be seen in Table 2.4 sections 2.4A Direct and 2.4B Direct. In these sections, when comparing the percentage increase in error between the 5 and 49 portfolios it can be seen that the hybrid DCC-SVR method actually shows comparable improvement when expanded to larger dimensions. While the error for DCC increased by 26.56%, the hybrid DCC-SVR method only showed a 21.15% increase in error. For the GJR methods, DCC error increased by 27.72% and Hybrid DCC-GJR increased by 20.51%. These results hold for variances and covariances as well.

Table 2.4: MAFE Full Matrix: Large Matrix Results, Scaled

		Method	**Beta 10**	**SD 10**	**Mkt 10**	**Ind 5**	**Ind 12**	**Ind 30**	**Ind 49**
2.4A.	Direct	Rolling-Historical	27.66	26.55	27.22	20.10	20.43	24.24	25.32
		DCC	22.46	22.06	23.60	17.81	18.45	21.91	22.54
		Hybrid DCC-SVR	16.60	15.87	15.43	16.41	17.01	19.54	19.88
	Fan Expansion	DCC	25.28	24.12	24.40	18.19	18.55	21.86	22.96
		Hybrid DCC-SVR	18.78	17.99	18.03	16.55	16.85	19.19	19.81
2.4B.	Direct	Rolling-Historical	27.66	26.55	27.22	20.10	20.43	24.24	25.32
		DCC GJR	21.41	20.84	23.56	16.92	17.45	20.85	21.61
		Hybrid DCC-GJR-SVR	16.01	15.49	16.27	16.09	16.53	18.99	19.39
	Fan Expansion	DCC GJR	22.87	21.87	22.59	17.04	17.35	20.31	21.33
		Hybrid DCC-GJR-SVR	17.53	16.79	17.17	15.94	16.19	18.31	18.93

This table analyzes the larger matrix forecasts for all values of the matrix. Error is computed using Eq. 2.3.

The methods within Table 2.4 have been grouped into Direct methods and Fan Expansion methods. This is to highlight the comparison of performance between these two types of forecasting methods. The Direct methods are when the forecasting method is applied directly to the larger portfolio data. The Fan Expansion methods are when the forecasting method is first applied to the factor data to obtain a factor covariance matrix forecast, this forecast is then expanded using the Fan method to become a 10 dimensional matrix forecast. For details of the Fan method refer to Eq. 2.15.

When looking at the first three data sets of Beta, SD, and Mkt in Table 2.4 section 2.4A, it can be seen that the direct methods had lower error levels than their comparative Fan Expansion methods. An example of this is in the Beta data set, where DCC Direct had an error level of 22.46 and DCC using Fan Expansion had an error level of 25.28.

While performance based soley on accuracy show the direct method to be better, a draw back to using the direct method with DCC based methods is that it becomes computationally complex when expanded to large dimensions. This can be seen in the run times of the Ind 5, Ind 12, Ind 30, and Ind 49 datasets on Table 2.5. The results are in minutes and represent the total amount of time to complete all testing forecasts. As the portfolio size increases from 5 assets to 49 assets the run time of the DCC direct method drastically increases. The Fan Expansion methods however did not have this issue and the run times remained constant as dimensions increased. This is shown in the table when DCC direct for Ind 49 had a run time of 903.73 minutes while DCC with Fan expansion on the same data set is only 16.43 minutes.

When accounting for run times, the Fan Expansion method becomes a much better method to use than the direct method. This can be seen visually for the DCC-SVR method on Figures 2.1 and 2.2. Figure 2.1 shows the comparison of the methods error levels as portfolio dimensions increase. The direct method shows to perform slightly better but they have similar results. Figure 2.2 shows the run times for these same methods when dimension size increases and it can now be seen that the Fan method runs much faster. Note, for the specific case of the DCC-SVR method and use of the Ind datasets (shown in Figure 2.1), the Fan Expansion method actually shows to outperform direct.

Since the run times are presented, it should be noted the runs are performed on a device with an i7 processor, 12 GB of available RAM, and a 64-bit operating system.

In the results something noticed was that the Hybrid DCC-SVR methods produced better forecasts when unscaled data was used as an input for the SVR function. It is unusual for a machine learning method to perform worse when scaled data is used, however since that was the case, the unscaled versions were used.

Table 2.5: Total Run Times

	Method	**Ind 5**	**Ind 12**	**Ind 30**	**Ind 49**
Direct	DCC	13.21	56.26	316.71	903.73
	DCC GJR	25.51	95.19	492.81	1324.04
	Hybrid DCC-SVR	18.44	63.34	353.82	998.69
	Hybrid DCC-GJR-SVR	32.51	101.53	520.04	1412.61
Fan Expansion	DCC	16.09	16.16	16.27	16.43
	DCC GJR	28.85	28.92	29.05	29.2
	Hybrid DCC-SVR	15.49	15.53	15.71	15.82
	Hybrid DCC-GJR-SVR	25.58	25.63	25.8	25.92

Run time in minutes for all testing forecasts to be made.

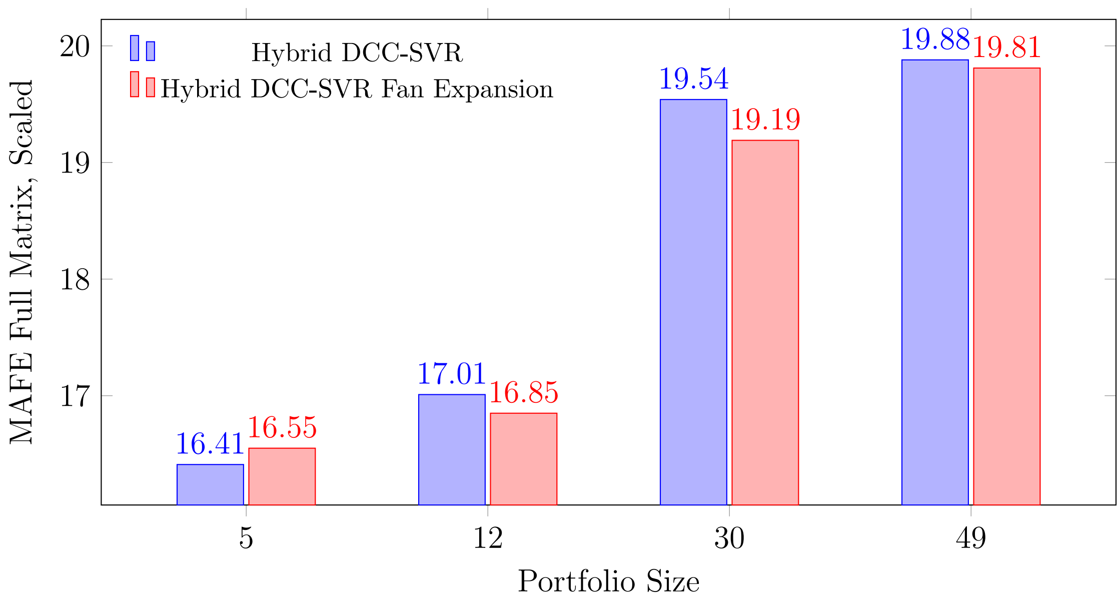

Figure 2.1: Error Values

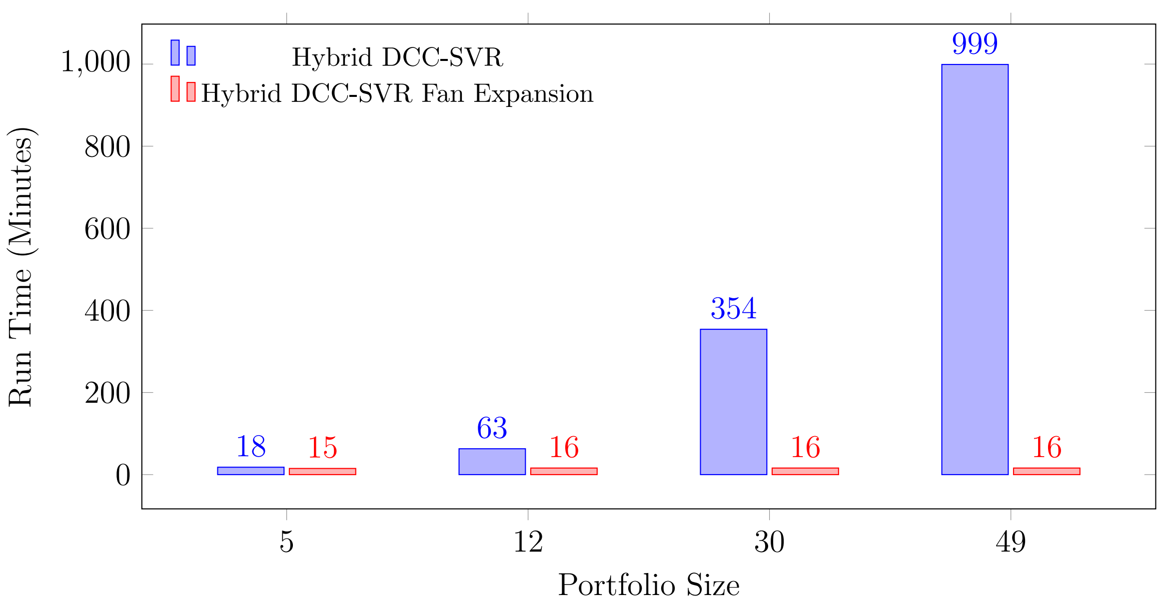

Figure 2.2: Run Time

2.4.3 Application Forecasts

The application results using the MSTE (Eq. 2.20) are shown in Table 2.6. The application results using MATE (Eq. 2.22) are shown in appendix Table A.3. These results align with the previous results of this paper. These results show that improving the forecasting method used will affect the resulting portfolio standard deviation values.

2.5 Conclusion

The results of this paper show that the newly introduced method of DCC-SVR is able to outperform traditional methods of forecasting such as DCC and Rolling Historical. The results show that the hybrid method is able to improve upon both standard GARCH and GJR-GARCH versions of DCC. This was first shown using the factor dataset, then it was shown using alternative datasets, then it was shown using alternative loss functions. While DCC-SVR is shown to outperform DCC in large dimensions, this paper also demonstrates a key issue with DCC models which is its run time for large dimensions. This paper shows how the Fan expansion method can be a practical solution to this issue because it is able to have similar results while requiring much less run time. When paired together, this will help to increase the practical application of the hybrid DCC-SVR method and allow for fast and accurate large covariance matrix forecasts to be made.

Table 2.6: Application MSTE: Equal Weighting

		Method	**Beta 10**	**SD 10**	**Mkt 10**	**Ind 5**	**Ind 12**	**Ind 30**	**Ind 49**
2.6A.	Direct	Rolling-Historical	12.33	11.93	11.10	7.32	7.55	8.69	8.95
		DCC	8.59	8.54	8.84	5.96	6.14	6.91	7.07
		Hybrid DCC-SVR	6.07	5.86	5.17	5.88	6.04	6.32	6.31
	Fan Expansion	DCC	10.11	9.76	9.07	5.96	6.06	6.80	7.04
		Hybrid DCC-SVR	6.90	6.71	5.78	5.80	5.85	6.08	6.00
2.6B.	Direct	Rolling-Historical	12.33	11.93	11.10	7.32	7.55	8.69	8.95
		DCC GJR	8.10	7.86	8.71	5.50	5.74	6.32	6.54
		Hybrid DCC-GJR-SVR	5.72	5.49	5.31	5.60	5.77	5.87	5.91
	Fan Expansion	DCC GJR	8.93	8.66	8.10	5.52	5.59	6.12	6.27
		Hybrid DCC-GJR-SVR	6.23	6.07	5.32	5.57	5.58	5.68	5.57

This table shows the MSTE (Eq. 2.20) when equal portfolio weighting is applied to obtain portfolio standard deviation values.

REFERENCES

Bollerslev, T. (1986). Generalized autoregressive conditional heteroskedasticity. *Journal of Econometrics*, *31*(3), 307–327.

Carhart, M. M. (1997). On Persistence in Mutual Fund Performance. *The Journal of Finance*, *52*(1), 57–82.

Chen, S., Härdle, W. K., & Jeong, K. (2010). Forecasting volatility with support vector machine-based GARCH model. *Journal of Forecasting*, *29*(4), 406–433.

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

Dudek, G., Fiszeder, P., Kobus, P., & Orzeszko, W. (2024). Forecasting cryptocurrencies volatility using statistical and machine learning methods: A comparative study. *Applied Soft Computing*, *151*, 111132.

Engle, R., & Sheppard, K. (2001, October). *Theoretical and Empirical properties of Dynamic Conditional Correlation Multivariate GARCH* (tech. rep. No. w8554). National Bureau of Economic Research. Cambridge, MA.

Fama, E. F., & French, K. R. (1993). Common risk factors in the returns on stocks and bonds. *Journal of Financial Economics*, *33*(1), 3–56.

Fan, J., Fan, Y., & Lv, J. (2008). High dimensional covariance matrix estimation using a factor model. *Journal of Econometrics*, *147*(1), 186–197.

Fiszeder, P., & Orzeszko, W. (2021). Covariance matrix forecasting using support vector regression. *Applied Intelligence*, *51*(10), 7029–7042.

Ghalanos, A. (2024, September). Introduction to the rugarch package. (Version 1.4-3).

Glosten, L. R., Jagannathan, R., & Runkle, D. E. (1993). On the Relation between the Expected Value and the Volatility of the Nominal Excess Return on Stocks. *The Journal of Finance*, *48*(5), 1779–1801.

Hansen, P. R., Lunde, A., & Nason, J. M. (2011). The Model Confidence Set. *Econometrica*, *79*(2), 453–497.

Li, N., Liang, X., Li, X., Wang, C., & Wu, D. D. (2009). Network Environment and Financial Risk Using Machine Learning and Sentiment Analysis. *Human and Ecological Risk Assessment: An International Journal*, *15*(2), 227–252.

Ou, P., & Wang, H. (2010). Financial Volatility Forecasting by Least Square Support Vector Machine Based on GARCH, EGARCH and GJR Models: Evidence from ASEAN Stock Markets. *International Journal of Economics and Finance*, *2*(1), p51.

Peng, Y., Albuquerque, P. H. M., Camboim de Sá, J. M., Padula, A. J. A., & Montenegro, M. R. (2018). The best of two worlds: Forecasting high frequency volatility for cryptocurrencies and traditional currencies with Support Vector Regression. *Expert Systems with Applications*, *97*, 177–192.

Pérez-cruz, F., Afonso-rodríguez, J. A., & Giner, J. (2003). Estimating GARCH models using support vector machines*. *Quantitative Finance*, *3*(3), 163–172.

Santamaría-Bonfil, G., Frausto-Solís, J., & Vázquez-Rodarte, I. (2015). Volatility Forecasting Using Support Vector Regression and a Hybrid Genetic Algorithm. *Computational Economics*, *45*(1), 111–133.

Sun, H., & Yu, B. (2020). Forecasting Financial Returns Volatility: A GARCH-SVR Model. *Computational Economics*, *55*(2), 451–471.

APPENDIX

ADDITIONAL RESULTS

Table A.1: MAFE Variances: Large Matrix Results, Scaled

		Method	**Beta 10**	**SD 10**	**Mkt 10**	**Ind 5**	**Ind 12**	**Ind 30**	**Ind 49**
A.1A.	Direct	Rolling-Historical	31.00	32.16	31.05	24.34	27.00	34.97	42.91
		DCC	25.67	27.94	27.22	20.82	23.81	30.78	33.16
		Hybrid DCC-SVR	19.84	19.78	18.62	19.25	22.10	28.08	30.13
	Fan Expansion	DCC	28.09	29.76	27.90	21.52	23.96	31.33	40.00
		Hybrid DCC-SVR	21.45	22.71	21.42	19.56	21.86	28.44	36.49
A.1B.	Direct	Rolling-Historical	31.00	32.16	31.05	24.34	27.00	34.97	42.91
		DCC GJR	24.32	27.31	27.32	19.86	22.31	29.76	32.07
		Hybrid DCC-GJR-SVR	18.90	20.18	19.58	18.88	21.29	27.72	29.56
	Fan Expansion	DCC GJR	25.45	27.29	26.06	20.27	22.67	29.72	38.33
		Hybrid DCC-GJR-SVR	20.02	21.30	20.54	18.84	21.05	27.42	35.49

This table analyzes the variance values of the 10 dimensional matrix forecasts. Error is computed using Eq. 2.3 when $i = j$.

Table A.2: MAFE Covariances: Large Matrix Results, Scaled

		Method	Beta 10	SD 10	Mkt 10	Ind 5	Ind 12	Ind 30	Ind 49
A.2A.	Direct	Rolling-Historical	27.28	25.92	26.79	19.04	19.83	23.87	24.95
		DCC	22.10	21.41	23.20	17.06	17.96	21.61	22.32
		Hybrid DCC-SVR	16.24	15.44	15.07	15.70	16.55	19.25	19.66
	Fan Expansion	DCC	24.97	23.49	24.01	17.35	18.06	21.53	22.61
		Hybrid DCC-SVR	18.49	17.46	17.65	15.80	16.40	18.87	19.46
A.2B.	Direct	Rolling-Historical	27.28	25.92	26.79	19.04	19.83	23.87	24.95
		DCC GJR	21.08	20.12	23.14	16.19	17.00	20.54	21.40
		Hybrid DCC-GJR-SVR	15.69	14.97	15.90	15.39	16.09	18.69	19.18
	Fan Expansion	DCC GJR	22.58	21.27	22.20	16.23	16.87	19.98	20.98
		Hybrid DCC-GJR-SVR	17.25	16.29	16.79	15.21	15.75	17.99	18.59

This table analyzes the covariance values of the 10 dimensional matrix forecasts. Error is computed using Eq. 2.3 when $i < j$.

Table A.3: Application MATE: Equal Weighting

		Method	Beta 10	SD 10	Mkt 10	Ind 5	Ind 12	Ind 30	Ind 49
A.3A.	Direct	Rolling-Historical	2.73	2.68	2.81	1.80	1.79	2.02	2.12
		DCC	2.20	2.21	2.41	1.55	1.55	1.76	1.82
		Hybrid DCC-SVR	1.55	1.50	1.57	1.38	1.36	1.45	1.47
	Fan Expansion	DCC	2.48	2.43	2.47	1.57	1.55	1.75	1.85
		Hybrid DCC-SVR	1.80	1.77	1.80	1.39	1.36	1.43	1.48
A.3B.	Direct	Rolling-Historical	2.73	2.68	2.81	1.80	1.79	2.02	2.12
		DCC GJR	2.12	2.10	2.42	1.44	1.46	1.65	1.73
		Hybrid DCC-GJR-SVR	1.52	1.49	1.66	1.33	1.32	1.39	1.42
	Fan Expansion	DCC GJR	2.27	2.22	2.31	1.47	1.45	1.61	1.70
		Hybrid DCC-GJR-SVR	1.69	1.66	1.71	1.36	1.33	1.37	1.41

This table shows the MATE (Eq. 2.22) when equal portfolio weighting is applied to obtain portfolio standard deviation values.