# Comparision Between Algorithms On Stock Price Modeling and Votality Dependence Modeling

*Yingying Chen*
*Chen Yuan*

**Abstract**

The stock of a corporation is constituted of the equity stock of its owners. A single share of the stock represents fractional ownership of the corporation in proportion to the total number of shares. Given the history price, we would like to forecast the stock prices. In time series, $SARIMA$ is popular choice. However, if we apply $SARIMA$ into real data, the prediction is not always as good as expected. In this project, we would like to use a different method, decompose the stock price into three pieces, conquer each one and unit all three. In addition, for usual GARCH models, the marginal distributions are estimated. However, usually in real case, stocks from same industry may have their prices depend on each other. Therefore, in this project, we tried to model dependence of stock prices using copula approach. Our analysis shows that k-nearest neighbour is our best model in fitting one single stock, and GARCH dependence model works best when there are more stocks in the same industry.

## 1 Introduction

Votality is one of the key variables in modelling time series data. It is necessary in stock exchange, asset evaluation and portfolio management. One of the simple ways to modeling votality is using a GARCH model. However, in this case, only marginal distribition is estimated. A copula model is usually a good choice of modelling dependence.[18] Therefore, a Copula-Garach model to model residual dependence can be used.[9]. For past works, copula is fitted using maximum likelihood or pseudo-likelihood (MPL) method[20], but in our analysis, we are using hierarchical modeling.

Our data comes from yahoo finance, where the 91 selected stocks are both listed in The New York Stock Exchange (NYSE) and Toronto Stock Exchange (TSX). However, since we are interested in modeling based on data in the past 5 years, stocks that are on the market before 2012 were used, which eventually leads us to 63 stocks.
The file "stock price.dta" contains 5923 rows and 63 columns, where the number of "NA" varies in different columns. The data can be categorized by industry:

- **Column 1-2**: Stocks from Electricity industry.
- **Column 3-19**: Stocks from Energy industry.
- **Column 20-28**: Stocks from Finance industry.
- **Column 29-51**: Stocks from Mining industry.
- **Column 52-53**: Stocks from Paper Production industry.
- **Column 54-56**: Stocks from Technology industry.
- **Column 57-60**: Stocks from Telecommunications industry.
- **Column 61-63**: Stocks from Transportation industry.

To facilitate comparison between different models, we will pick one stock price data and ecompose the time series into three pieces, seasonality, trend and random residuals; build a model to each of them and forecast; unit all three pieces(including predictions). To pick on specific stock, we used *stress* as our measurement, where

$$stress = \frac{(\mathbf{Y}_t - \hat{\mathbf{Y}}_t)^T(\mathbf{Y}_t - \hat{\mathbf{Y}}_t)}{(\mathbf{Y}_t - \bar{\mathbf{Y}}_t)^T(\mathbf{Y}_t - \bar{\mathbf{Y}}_t)}$$

*stress* is the smaller the better (dividing the denominator is to better scale stress). Notice, since it is not the ordinary least squares regression, *stress* is not bounded by 0 to 1

In the paper, we will pick the stock `CAE`.

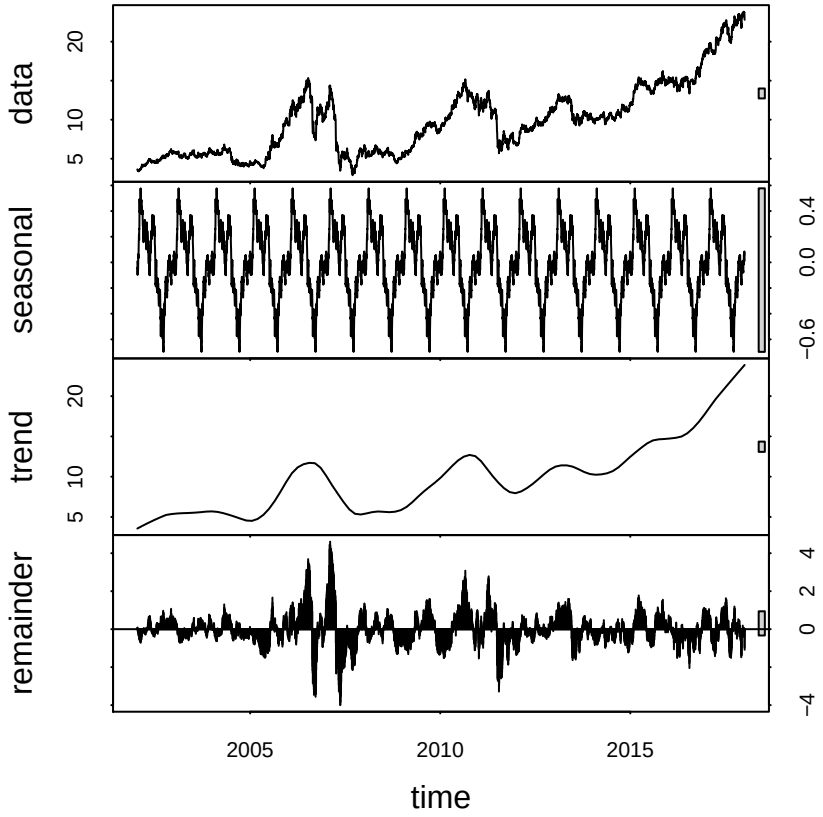
# 2 Methodology

## 2.1 Comparison of models based on on single stock data

A time series can be decomposed into three pieces, seasonality, trend and random residual[8].

$$Y_t = S_t + T_t + R_t$$

If we decompose *CAE* time series, the visualization of these three pieces are shown as follows:




### 2.1.1 For seasonal $S_t$

Since seasonality is repeated, it is not necessary to build a model (in other words, the fit residuals are zero). When we do prediction, we just follow the seasonal trend and predict.

### 2.1.2 For trend $T_t$

The trend is clearly not linear. There are several methods to predict this part

- linear regression

We can build model

$$\mathbf{T} = \mathbf{X}\boldsymbol{\beta} + \mathbf{e}$$

where $\mathbf{X}$ represent time. $\hat{\boldsymbol{\beta}}$ can be found by minimizing $\mathbf{e}^T\mathbf{e}$

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{T}$$

In our model

$$\begin{aligned} \widehat{T}_t &= -1607.6913508 + 0.8047257t \\ (t) &\quad (-117.7)\ (118.4) \end{aligned}$$

- Natural Spline[12]

In the mathematical field of numerical analysis, Spline interpolation is a form of interpolation where the interpolate is a special type of piece wise polynomial called a spline. Spline interpolation is often preferred over polynomial interpolation because the interpolation error can be made small even when using low degree polynomials for the spline

We cut the data into several pieces, or call neighborhood. In each neighborhood, we fit a polynomial $p$ model. For example, in the $j$th neighborhood a $p$ degree polynomial would look like

$$\mu^{(j)}(x) = \beta_0^{(j)} + \beta_1^{(j)}x + ... + \beta_P^{(j)}x^p, \forall x \in Nbhd_j$$

Then $\mu(x)$ can be found as

$$\mu(x) = \sum_{j=1}^{J} I_{Nbhd_j}(x)\mu^{(j)}(x)$$

Next, we would like to restrict the parameters by forcing the curves to meet each boundary, first derivatives, second derivatives, ... , $p-1$ derivatives. the degree freedom would be

$$J(p+1) - (p-1+1)K = K + p + 1$$

and this one called the $p$ degree spline. In natural spline, it forces the polynomials at either end to be fluctuate less by severely reducing their degree. A popular choice would be cubic smoothing spline, with the end to be a straight line.

- KNN fitting[3]

The easy way to proceed is to fit a local average at every value $x = x_i$ in the data set. Values for other values of $x$ might be found by simple linear interpolation between these fitted values (i.e. simply "connect the dots"). One way to define a local neighborhood would be to find the $k$ nearest neighbors of that $x$ value.

There are also some other methods to fit the trend, like `basic spline`, `smoothing spline`, `loess`, ... They are either cutting the field into pieces and connecting with restrictions or resetting the objective function and minimizing through optimization methods. We will not show all the details of the rest methods, but we would apply them in our package `tsDecom`.

### 2.1.3 For residuals

A run test is to test the series stationary or not. The null hypothesis test is the series is stationary. Through the result we can find the p-value is smaller than 0.05, which means there is evidence against the null hypothesis test, the series is not stationary. In general, we take the difference of the series The p value is larger than 0.05. The series is stationary. Then, we would like to fit $\triangle R_t$ with $ARMA$ model.

$$\triangle R_t = R_t - R_{t-1}$$

Auto regressive moving average (`ARMA`) model is fitted to time series data either to better understand the data or to predict future points in the series (forecasting). This model can only work on the stationary data.

$$\triangle R_t - \phi_1 \triangle R_{t-1} - ... - \phi_p \triangle R_{t-p} = \epsilon_t + \theta_1 \epsilon_{t-1} + \theta_q \epsilon_{t-q}$$

$AIC$ is a criterion for model selection; the model with the lowest $AIC$ is preferred and it can be expressed as:

$$AIC = 2k - 2log(\hat{L})$$

where $k$ is the number of parameters and $\hat{L}$ is maximum value of the likelihood function for the model.

Actually, most of the time, to our prediction, the effect of $ARMA$ order is quite small. The main reason is that the residuals are too small comparing with trend and seasonality. Sometimes, the change of residuals can be ignored. What's more, high order model can make our computation too complex and sometimes even overfitted. Hence, in this paper, we would like to use $AR(1)$ model to fit the residual part. We have three ways to do that:

#### 2.1.3.1 Basic $AR(1)$ model

$$\triangle R_t = \phi \triangle R_{t-1} + \epsilon_t$$

where $\epsilon_t \sim N(0, \sigma^2)$. It gives us $AR(1)$ model.

$$\begin{aligned} \triangle R_t &= 0.0006 \triangle R_{t-1} + \epsilon_t \\ (se) &\qquad (0.0131) \end{aligned}$$

$$R_t = 0.0006(R_{t-1} - R_{t-2}) + R_{t-1} + \epsilon_t, \quad \epsilon_t \sim N(0, 0.0304)$$

#### 2.1.3.2 $AR(1)$ with $t$ distribution

$$\triangle R_t = \phi \triangle R_{t-1} + \sigma \epsilon_t$$

However, this time, we would like to set $\epsilon_t \sim t(v)$ distribution, $v$ is the degree of freedom. Thus

$$f(\epsilon_t) = \frac{\Gamma(\frac{v+1}{2})}{\sqrt{v\pi}\Gamma(\frac{v}{2})} \left(1 + \frac{\epsilon_t^2}{v}\right)^{-\frac{v+1}{2}}$$

Consider the pdf of $\triangle R_1$, the first observations in the sample. This is a random variable with 0 mean and variance

$$Var(\triangle R_1) = \frac{1}{1-\phi^2} Var(\epsilon_t) = \frac{1}{1-\phi^2} \frac{v}{v-2} \sigma^2 \qquad \text{when v } \geq 2$$

Since $\epsilon_t$ is t distribution. $\triangle R_1$ follows scaled $t$ distribution with variance $\frac{1}{1-\phi^2}\frac{v}{v-2}\sigma^2$

$$f(\triangle R_1) = \frac{\Gamma(\frac{v+1}{2})}{\sqrt{v\pi}\Gamma(\frac{v}{2})}\left(1+\frac{(1-\phi^2)\triangle R_1^2}{v\sigma^2}\right)^{-\frac{v+1}{2}}\frac{\sqrt{1-\phi^2}}{\sigma}$$

and

$$f(\triangle R_k|\triangle R_{k-1},\ldots,\triangle R_1) = f(\triangle R_k|\triangle R_{k-1})$$

based on the Jacobian matrix and determinant and coefficient transformation

$$f(\triangle R_k|\triangle R_{k-1}) = \frac{\Gamma(\frac{v+1}{2})}{\sqrt{v\pi}\sigma\Gamma(\frac{v}{2})}\left(1+\frac{(\triangle R_k-\phi\triangle R_{k-1})^2}{v\sigma^2}\right)^{-\frac{v+1}{2}}$$

The likelihood can be found as:

$$L = \left(\frac{\Gamma(\frac{v+1}{2})}{\sqrt{v\pi}\Gamma(\frac{v}{2})}\right)^n\frac{1}{\sigma^n}\sqrt{1-\phi^2}\prod_{i=2}^{n}\left(1+\frac{(\triangle R_i-\phi\triangle R_{i-1})^2}{v\sigma^2}\right)^{-\frac{v+1}{2}}\left(1+\frac{(1-\phi^2)\triangle R_1^2}{v\sigma^2}\right)^{-\frac{v+1}{2}}$$

log-likehood can be denoted as:

$$\mathcal{L} = log(L)$$

$\epsilon_t$ follows t distribution but unknown degree freedom $v$. By setting

$$\frac{\partial\mathcal{L}}{\partial\phi}=0;\quad\frac{\partial\mathcal{L}}{\partial v}=0;\quad\frac{\partial\mathcal{L}}{\partial\sigma}=0;\quad\sigma>0$$

we can use quasi-Newton method, Nelder Mead or some other optimization methods to find $\widehat{\phi}$, $\widehat{v}$ and $\widehat{\sigma}$. From `optim_proj()`[13], we can find the likelihood function reaches the maximum.

$$\widehat{\phi}\approx -0.0397\ \ \widehat{v}\approx 3.296\ \text{ and }\ \widehat{\sigma}\approx 0.115$$

$$R_t = -0.0397(R_{t-1}-R_{t-2})+R_{t-1}+0.115\epsilon_t,\quad \epsilon_t\sim t(3.296)$$

#### 2.1.3.3 Regime switching model $AR(1)$

[11]

$$\triangle R_t = \delta_t\phi^E\triangle R_{t-1}+(1-\delta_t)\phi^R\triangle R_{t-1}+\epsilon_t,\qquad \epsilon_t\sim N(0,\sigma^2)$$

where $\delta_t$ is

$$\delta_t = \begin{cases} 1 & \triangle R_{t-1}\geq 0 \\ 0 & \triangle R_{t-1}\leq 0 \end{cases}$$

the first observations in the sample $\triangle R_1$ is a random variable with 0 mean and variance

$$Var(\triangle R_1) = \frac{\sigma^2}{1-\left(\delta_t\phi^E+(1-\delta_t)\phi^R\right)^2} = \kappa^2$$

Hence, $f(\triangle R_1)$ can be found as

$$f(\triangle R_1) = \frac{1}{\sqrt{2\pi}\kappa}e^{-\frac{\triangle R_1^2}{2\kappa^2}}$$

and $f(\triangle R_k|\triangle R_{k-1},\ldots,\triangle R_1) = f(\triangle R_k|\triangle R_{k-1})$

$$f(\triangle R_k|\triangle R_{k-1}) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{\left(\triangle R_k - (\delta_t \phi^E \triangle R_{k-1} + (1-\delta_t)\phi^R \triangle R_{k-1})\right)^2}{2\sigma^2}} \qquad \text{where k} \geq 2$$

Loglikehood function can be found as

$$\mathcal{L} = log(f(\triangle R_1)) + log(f(\triangle R_2|\triangle R_1)) + \ldots + log(f(\triangle R_n|\triangle R_{n-1}))$$

Thus

$$\frac{\partial \mathcal{L}}{\partial \phi^E} = 0; \quad \frac{\partial \mathcal{L}}{\partial \phi^R} = 0; \quad \frac{\partial \mathcal{L}}{\partial \sigma} = 0; \quad \sigma > 0$$

$\widehat{\phi}$, $\widehat{v}$ and $\widehat{\sigma}$ can be found through `optim`[4, 6, 10, 15, 16, 17]

$$\widehat{\phi}^E \approx -0.020 \;\; \widehat{\phi}^R = 0.022 \;\; \text{and} \;\; \widehat{\sigma} = 0.174$$

From `optim_ proj()`, we can find the likelihood function reaches the maximum.

$$R_t = -0.02\delta_t(R_{t-1} - R_{t-2}) + 0.022(1-\delta_t)(R_{t-1} - R_{t-2}) + R_{t-1} + \epsilon_t, \qquad \epsilon_t \sim N(0, 0.174^2)$$

## 2.2 Dependence modelling

### 2.2.1 Modelling Votality

Heteroscedasticity generally means that the variance of time-series data is not constant over time. To model the votality of stock price, we apply a simple GARCH(1,1) model with central t-distribution as residual disribution. The model architecture is as follows. Let $s_{ti}$ denote the price of stock i on day t. In our analysis, we used the log return which can be calculated as $y_{ti} = log(s_{ti}) - log(s_{t-1i})$. Then the marginal model for returns on stock i is:

$$\begin{aligned}
y_{ti} &= \mu_i + \epsilon_{ti} \\
\epsilon_{ti} &= \sigma_{ti} x_{ti} \\
\sigma_{ti}^2 &= \omega_i + \alpha_i \epsilon_{t-1i}^2 + \beta_i \sigma_{t-1i}^2 \\
x_{ti} &\overset{iid}{\sim} t(\nu_i)
\end{aligned}$$

where the log-likelihood is[1][5]

$$l(\theta) = Tlog[\frac{\Gamma(\frac{\nu+1}{2})}{\sqrt{\pi(\nu-2)}\Gamma(\frac{\nu}{2})}] - \frac{1}{2}\Sigma_{t=1}^T log\sigma_t^2 - \frac{\nu+1}{2}\Sigma_{t=1}^T log[1 + \frac{\epsilon_t^2}{\sigma_t^2(\nu-2)}]$$

The basic GARCH(1,1) model was fitted using `rugarch` package. Our test using optim_proj shows that the degrees of freedom is not always optimized, but to facilite our analysis, we will stick with this estimate.

### 2.2.2 Modeling Dependence

Copula is one way of modelling the dependence between random variables. It is the joint distribution between marginally uniformly distributed variables. Considering $X_1, X_2, ..., X_p$ are random variables with distribution function $F_1(x_1), ...F_p(x_p)$, then by Probability Integral Transform, $F_i(x_i) \overset{iid}{\sim} U(0,1)$. Then there exist a copula such $C$ such that $F(x_1, x_2, ..., x_p) = C(F_1(x_1), F_2(x_2), ..., F_p(x_p))$.

#### 2.2.2.1 Gaussian Copula

Consider $u_i = F_i(x_i)$. Then the cdf of Gaussian Copula is defined by:

$$C(u_1, u_2, ..., u_p; \Omega) = \Phi_\Omega(\Phi^{-1}(u_1), \Phi^{-1}(u_2), ..., \Phi^{-1}(u_p))$$

and the density function of Gaussian Copula is.

$$c(u_1, u_2, .., u_p; \Omega) = \frac{\psi_\Omega(\Phi^{-1}(u_1), \Phi^{-1}(u_2), ..., \Phi^{-1}(u_p))}{\psi(\Phi^{-1}(u_1))\psi(\Phi^{-1}(u_2))...\psi(\Phi^{-1}(u_p))}$$

#### 2.2.2.2 t-Copula

Similarly, the cdf of t-Copula is defined by:

$$C(u_1, u_2, ..., u_p; \Omega, n) = T_{\Omega,n}(T_n^{-1}(u_1), T_n^{-1}(u_2), ..., T_n^{-1}(u_p))$$

And the density function is:

$$c(u_1, u_2, .., u_p; \Omega, n) = \frac{t_{\Omega,n}(T_n^{-1}(u_1), T_n^{-1}(u_2), ..., T_n^{-1}(u_p))}{t_n(T_n^{-1}(u_1))t_n(T_n^{-1}(u_2))t_n(T_n^{-1}(u_p))}$$

### 2.2.3 Copula-Garch Model

In ordinary Garch model, the marginal distributions are modeled. But in real life, stock prices usually depend on other stock prices within the same industry. To modeling the dependence of stock price, we model the dependence of through model residuals.

Consider $x_1, x_2, .., x_n$ are the residuals fitted using the garch model indicated above, then $x_{ti} \overset{iid}{\sim} t(x|\nu_i)$. By Probability Integral Transform, $u_{ti} = T(x|\nu_i) \overset{iid}{\sim} U(0, 1)$. Dependency can be then modeled using Copula.

Let $\theta_i = (\alpha_i, \beta_i, \omega_i, \mu_i, \nu_i)$ denote the marginal GARCH parameters and $\xi$ denote the copula parameters, then the full set of parameters of the Copula-Garch model is $\Theta = (\theta_1, \theta_2, ..., \theta_D, \xi)$.

The log likelihood of Copula Garch model is given as

$$l(\Theta|Y) = \Sigma_{t=1}^{T} \left( log f_D(z_t^{\Theta}|\Omega) + \Sigma_{i=1}^{D}[logt(x_{ti}^{(\theta_i)}|\nu_i) - log\sigma_{ti}^{(\theta_i)} - logf(z_{ti}^{(\theta_i)})] \right)$$

where $f_D$ stands for the multvariate density distribution and $f$ denotes the univaiate density distribution.

### 2.2.4 Sector-level hierarchical modeling

Since the number of parameters in $\Theta$ is usually to large that `optim` will overwhelm, we employed a sector-level hierarchical modeling algorithm using Expectation-Maximation algorithm to obtain an estimate of covariance matrix.

Suppose we have $D$ stocks and $K$ sectors. Consider $S_{tk}$ denotes the mean of sector k on day t and $S_t = (S_{t1}, ..., S_{tk})$. Consider $z_t^{(k)} = (z_{t1}^{(k)}, ..., z_{tN_K}^{(k)})$ denote the transformed residuals on day $t$ of $N_k$ assets and $\sum_{k=1}^{K} N_k = D$. The hierarchical model for the transformed residuals is

$$S_t \overset{iid}{\sim} N(0, \Sigma)$$
$$z_t^{(k)}|S_t \overset{iid}{\sim} N(S_{tk}, \Psi_k)$$

Let $\Omega = (\Sigma, \Psi_1, ..., \Psi_k)$ denote the parameters of this model, then we can write an EM algorithm to find the MLE of $\Omega$.

In the E setup of EM algorithm, consider the missing data are $y_{obs} = Z$ and $y_{miss} = S = (S_1, ..., S_T)$. The log likelihood is

$$\begin{aligned} l(\Omega|S,Z) &= -\frac{1}{2}\Sigma_{t=1}^{T}[S_t'\Sigma^{-1}S_t + log|\Sigma|] - \frac{1}{2}\Sigma_{k=1}^{K}\Sigma_{t=1}^{T}[(z_t^{(k)} - S_{tK})'\Psi_k^{-1}(z_t^{(k)} - S_{tK} + log|\Psi_k|)] \\ &= -\frac{1}{2}[Tlog|\Sigma| + \Sigma_{t=1}^{T}tr(\Sigma^{-1}S_tS_t')] \\ &- \frac{1}{2}\Sigma_{k=1}^{K}[Tlog|\Psi_k| + \Sigma_{t=1}^{T}[tr(\Psi_k^{-1}z_t^{(k)}z_t'^{(k)}) - S_{tk}tr(\Psi_k^{-1}(z_t^{(k)}J_{N_k}' + J_{N_k}z_t^{(k)'})) + S_{tk}^2tr(\Psi_k^{-1}J_{N_k}J_{Nk}')]] \end{aligned}$$

The expectation of this loglikelihood is

$$Q_m(\Omega) = -\frac{1}{2}[Tlog|\Sigma| + tr(\Sigma^{-1}\hat{A}^{(m)})] - \frac{1}{2}[Tlog|\Psi_k| + tr(\Psi_k^{-1}\hat{B_k}^{(m)})]$$

where $\hat{A}^{(m)} = \Sigma_{t=1}^{T}\hat{\Sigma}^{(m)} + \hat{\mu}_t^{(m)}\hat{\mu}_t^{(m)'}$ and $\hat{B}^{(m)} = \Sigma_{t=1}^{T}z_t^{(k)}z_t^{(k)'} - \Sigma_{t=1}^{T}\hat{\mu_t}^{(m)}(z_t^{(k)}J_{Nk}' + J_{N_k}z_t^{(k)'}) + \Sigma_{t=1}^{T}\hat{A_{kk}}^{(m)}J_{Nk}J_{Nk}'$.

For the M step, we update $\hat{\Sigma}^{(m+1)} = T^{-1}\hat{A}^{(m)}$ and $\hat{\Psi}_k^{(m+1)} = T^{-1}\hat{B}_k^{(m)}$.

#### 2.2.5 Evaluation of Forecasting Models

Considering that $S_t^{(W)} = (s_{t-W+1}, ..., s_t)$ denote the stock prices in the last W days, then we can estimate the parameters for GarchGC using algorithms described above. Thus, we are able to forecast the votalities and stock prices in the future. For doing $B$ times of forecasting, we are able to obtain a 95% prediction interval of the forecast distribution.

Let $p_{FC}(P_{t+k}|S_t^{(W)})$ denote the forescast distribution of $P_{t+k}$, where $P_{t+k}$ is the value of a given portfolio on day $t+k$. Considering $P_{t+k}^{(obs)}$ be the actual observed value on day $t+k$, then er are able to estimate the cdf of $P_{FC}$ by drawing $B$ iid samples from prediction.

If the GarchGC model is the true model for modeling dependence for our data, then we have a null hypothesis $H_0$ that $p_t \sim U(0,1)$. And based of Probability Integral Transform, we have $z_t = \Phi^{-1}(p_t) \sim N(0,1)$, which can be visually tested using QQ-plots and numerically tested by Shapiro-Wilk test.

# 3 Results

## 3.1 Model-wised comparison

In above section, we decompose our data into three pieces and analysis each of them. Then, we would like to unit all of three to check the *stress*.

| $R^2$ | lm | natal spine | KNN |
|---|---|---|---|
| AR(1) | 2.025 | 1.69 | 1.65 |
| AR(1) with $t$ | 2.63 | 2.87 | 2.83 |
| Regime switching model | 26.02 | 25.82 | 25.75 |

From this table, we can find that basic AR(1) with KNN gives the best prediction.
Next we would like to test the residuals

1. Box-Pierce test: Box-Pierce is an independent test. The null hypothesis is the time series independent.
2. Jarque-Bera test: The Jarque-Bera test is a test for normality. The null hypothesis is the time series follows normal distribution:
$$JB = N(\frac{\widehat{\kappa}_3^2}{6} + \frac{\widehat{\kappa}_4^2}{24}) \sim \chi_2^2$$

where $\widehat{\kappa}_3 = \frac{1}{N}\sum_{i=1}^{N}\widehat{\epsilon}_i^3$ and $\widehat{\kappa}_4 = \frac{1}{N}\sum_{i=1}^{N}\widehat{\epsilon}_i^4$, and $\widehat{\epsilon}_i$ is the centralized data, $\widehat{\epsilon}_i = \frac{x_i-\mu}{\sigma}$. If $JB$ is larger than 6, we reject the null hypothesis at 0.05 confidence level.

3. Overfitting: The null hypothesis test is that the parameters of added terms are all equal to zero.

$$\Lambda = N \times ln\left(\frac{\widehat{\sigma}_p^2}{\widehat{\sigma}_{p+r}^2}\right) \sim \chi_r^2$$

Residuals $\mathbf{Y} - \hat{\mathbf{Y}}$ fail all the test.

### 3.2 Dependence Modeling

Based on the qq-plots we have and the Shapiro-Wilk tests result(as in Appendix), we can generally conclude that the GarchGC model works better when there are more companies in the industry. In our case, modeling dependence works best on the mining sector. However, one thing to notice is that when analyzing the distribution of the forecast distribution model, at some somulation of $t + k$ days, none or all of our simulated values are smaller than or equal to the true stock price, and thus will give us `Inf` or `-Inf` when applying probability integral transform. To facilitate the hypothesis test, we remove these values in our normality analysis, which means that our analysis could be potentially biased.

## 4 Discussion

The prediction is not as good as expected, but if we go back to check the fitted residuals, we can find the residuals are too large and not stationary at all. We have two ways to fix this situation, one is to set residuals as zero, like hard margin in support vector machine or to give the bound of the $\epsilon_t$ generation, then the random $\epsilon_t$ would not dominate the residuals.

| $R^2$ | lm | natal spine | KNN |
|---|---|---|---|
| AR(1) | 0.456 | 0.197 | 0.171 |
| AR(1) with $t$ | 0.471 | 0.162 | 0.149 |
| Regime switching model | 0.461 | 0.208 | 0.184 |
| Hard Margin | 0.358 | 0.089 | 0.04 |

The KNN gives the best prediction and all residuals test has passed. The fitted line is very close to the original data. However, this method has several short-comes:

1. The data is decomposed into three pieces. However, these three parts are not necessarily independent to each other. Then, it gives us a lot of difficulty to estimate the variance. In finance, the volatility sometimes interests us more than the values.

2. The smoothing can always fit the training set well. But to the test set, it may cause overfitting problem.

Our package `tsDecom` is based on the whole process.

For dependence modelling, our analysis did relatively succeesful on stocks in mining industry while remian unsuccessful in telecommunication industry. One thing to notice is that in our GARCH model, we assume that the residual is a centered t-distribution, but in real life, a skewed-t distribution is usually the case. One thing we could try in the future is using a archimedean copulas to fit the residuals, which may get us a better result rather than elliptical copulas.

# References


[1] Maximum likelihood estimation for conditional variance models. https://www.mathworks.com/help/econ/maximum-likelihood-estimation-for-conditional-variance-models.html.

[2] Quantitative risk management. http://www.qrmtutorial.org/. Accessed: 2018-04-12.

[3] N. S. Altman. "an introduction to kernel and nearest-neighbor nonparametric regression". *The American Statistician*, 46(3):175–185, 1992.

[4] C. J. P. Belisle. Convergence theorems for a class of simulated annealing algorithms. *Rd. J. Applied Probability*, 29:885–895, 1992.

[5] Tim Bollerslev. A conditionally heteroskedastic time series model for speculative prices and rates of return. *The Review of Economics and Statistics*, 69(3):542–547, 8 1987.

[6] R. H. Byrd. Convergence theorems for a class of simulated annealing algorithms. *SIAM J. Scientific Computing*, 16:1190–1208, 1995.

[7] Khashanah K. Chen KH. Analysis of systemic risk: A dynamic vine copula-based arma-egarch model. *Transactions on Engineering Technologies*, pages 15–30, 2017.

[8] R. B. Cleveland. Stl: A seasonal-trend decomposition procedure based on loess. *Journal of Official Statistics*, 6:3–73, 1990.

[9] Michael Rockinger Eric Jondeau. The copula-garch model of conditional dependencies: An international stock market application. *Journal of International Money and Finance*, pages 827–853, 2006.

[10] R. Fletcher and C. M. Reeves. Function minimization by conjugate gradients. *Computer Journal*, 7:148–154, 1964.

[11] B.E. Hansen. Inference in tar models. *tudies in Nonlinear Dynamics and Econometrics*, 1997.

[12] ed. Hazewinkel, Michiel. "spline interpolation", encyclopedia of mathematics. *Springer Science+Business Media B.V. / Kluwer Academic Publishers*, 2001.

[13] Martin Lysy. optimcheck. https://github.com/mlysy/optimCheck.

[14] Guido Masarotto and Cristiano Varin. Gaussian copula regression in r. *JournalofStatisticalSoftware*, 77(8), 4 2017.

[15] J. C. Nash. Compact numerical methods for computers. linear algebra and function minimisation. 1990.

[16] J. A. Nelder and R. Mead. A simplex algorithm for function minimization. *Computer Journal*, 7:308–313, 1965.

[17] J. Nocedal and S. J. Wright. Numerical optimization. *Springer*, 1990.

[18] YANG Xiang-qun OUYANG Zi-sheng, LIAO Hui. Modeling dependence based on mixture copulas and its application in risk management. *Appl. Math. J. Chinese Univ.*, 24(4):393–401, 2009.

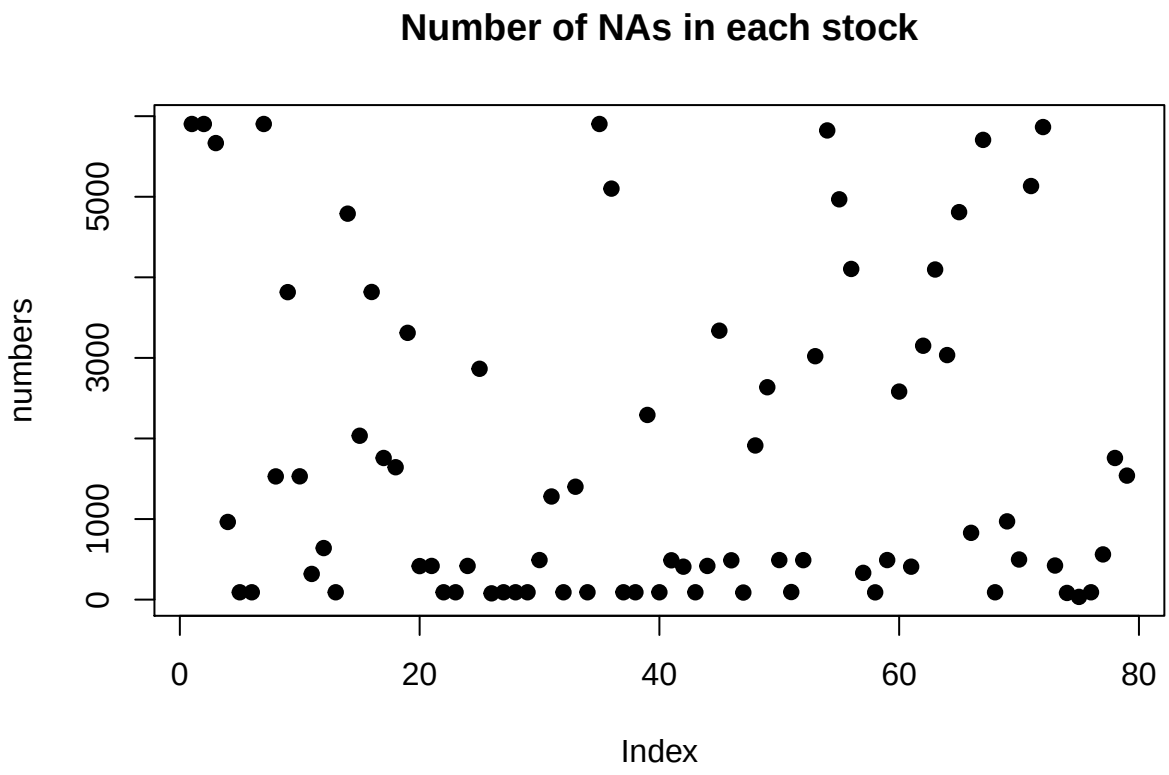
[19] Sedigheh Shams and Fatemeh K. Haghighi. A copula-garch model of conditional dependencies: Estimating tehran market stock exchange value-at-risk. *Journal of Statistical and Econometric Methods*, 2(2):39–50, 2013.

[20] Erin M. Conlon Zheng Wei, Daeyoung Kim. Parallel computing for copula parameter estimation with big data: A simulation study. 9 2016.

# 5 Appendix

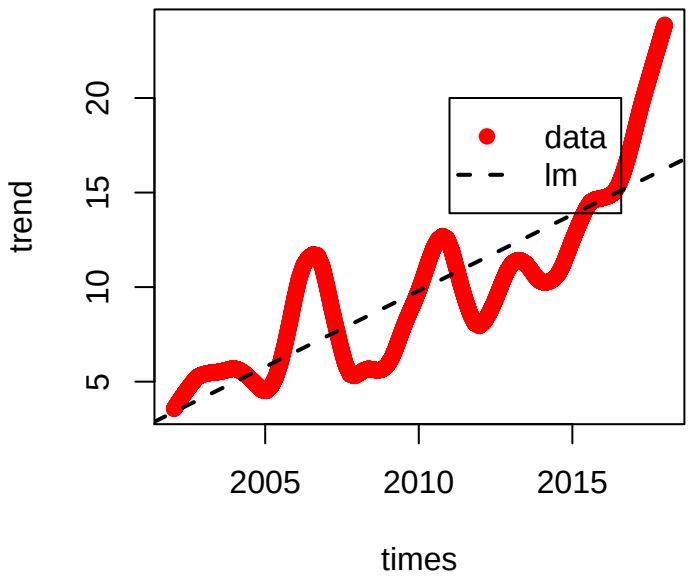
## 5.1 Plot related to analysis

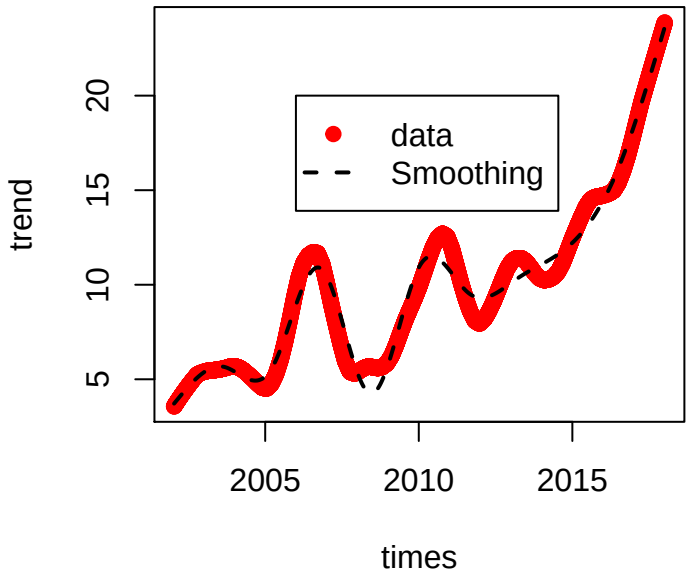
NA check

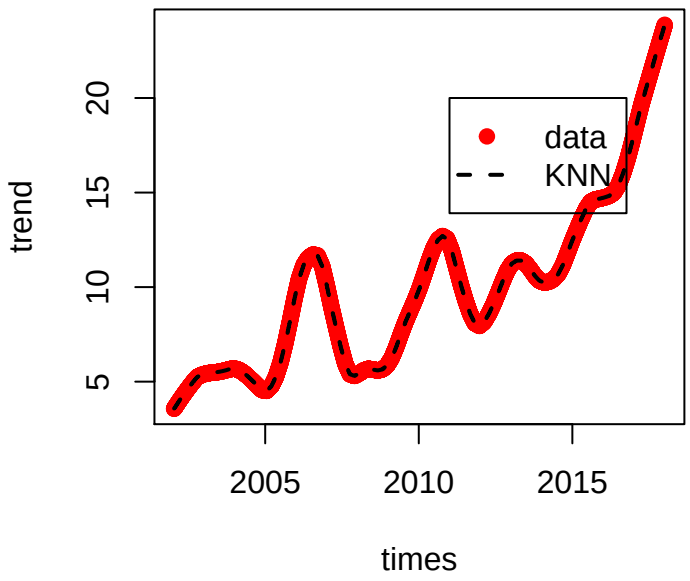



```
## [1] "CAE"
```

Trend Fitting using different algorithms


trend
5
10
15
20
2005
2010
2015
times
data
lm


trend
5
10
15
20
2005
2010
2015
times
data
Smoothing

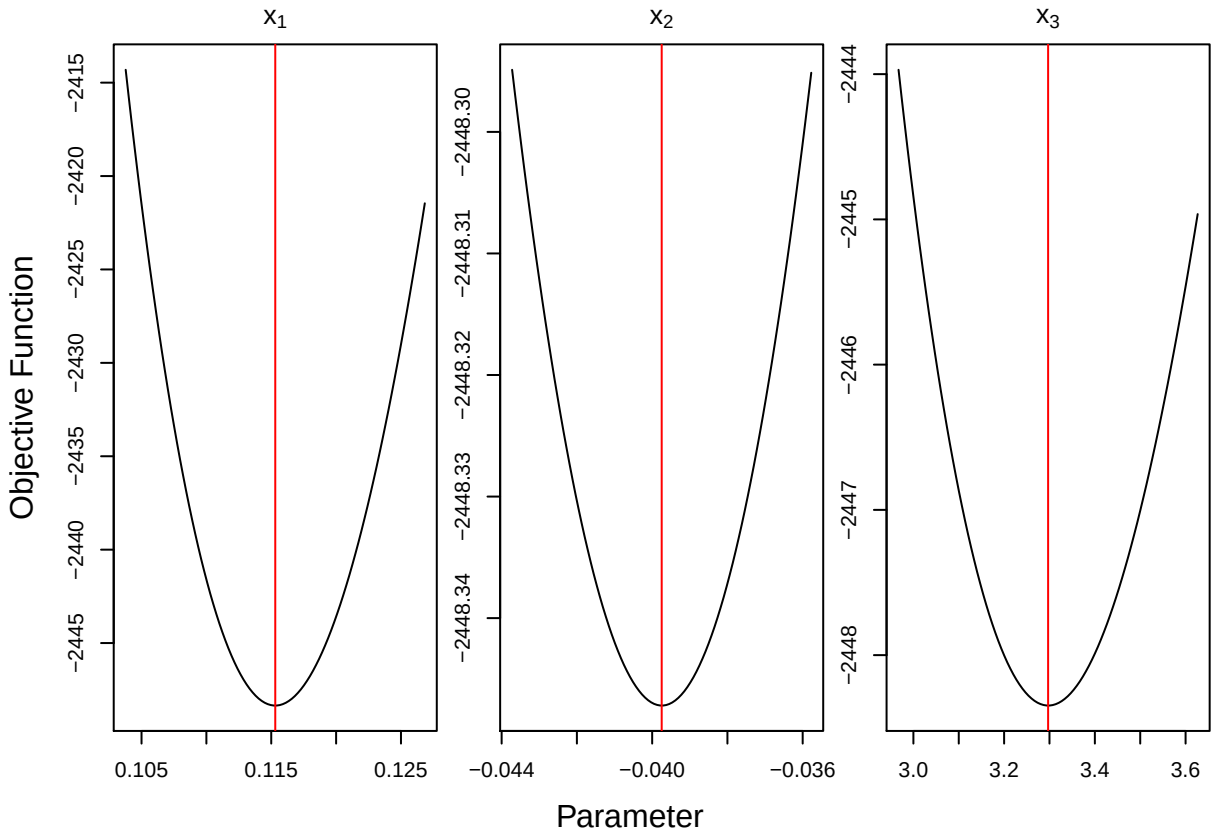



```
## Warning: package 'tseries' was built under R version 3.4.4
```

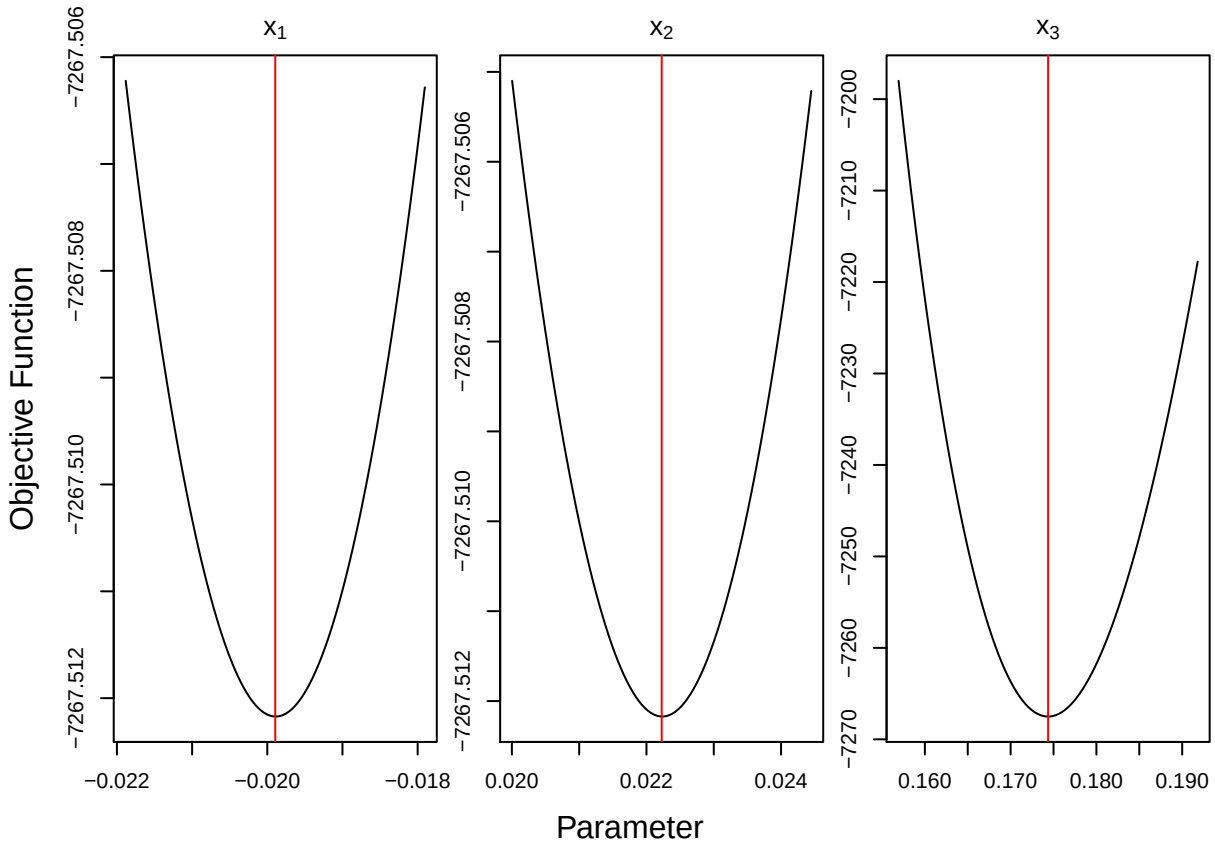
```
##
##  Runs Test
##
## data:  factor(sign(res))
## Standard Normal = -67.155, p-value < 2.2e-16
## alternative hypothesis: two.sided
```

```
##
##  Runs Test
##
## data:  factor(sign(diffRes))
## Standard Normal = 0.84736, p-value = 0.3968
## alternative hypothesis: two.sided
```

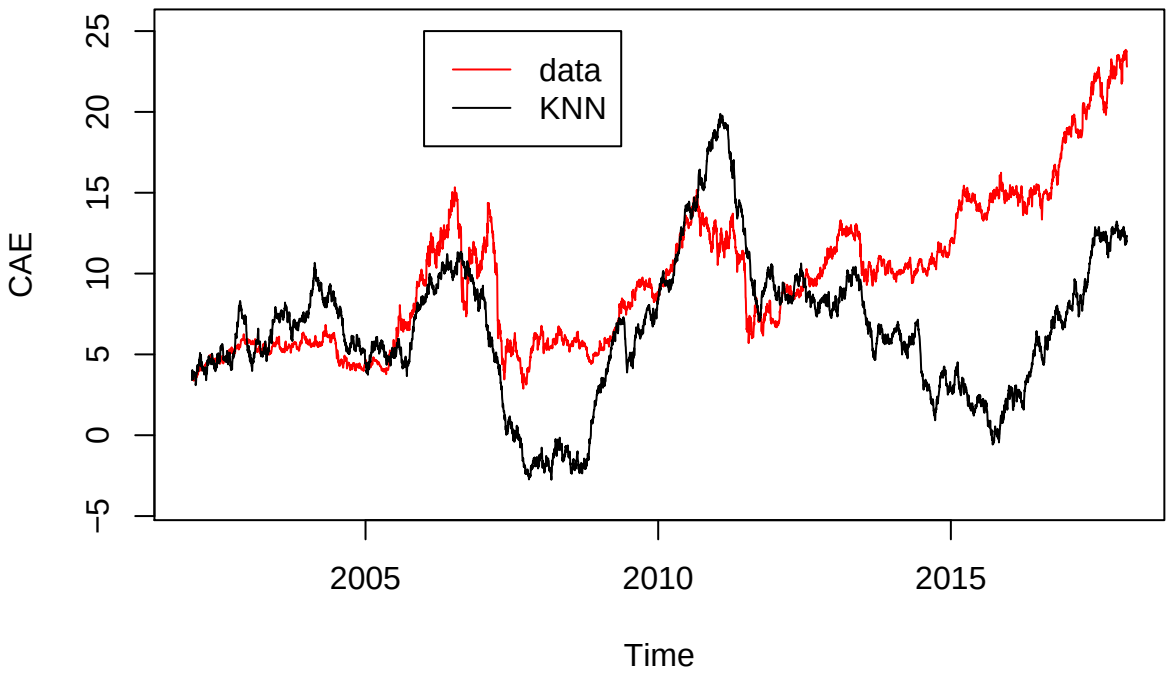
Optim Check for AR(1) Model with t distribution

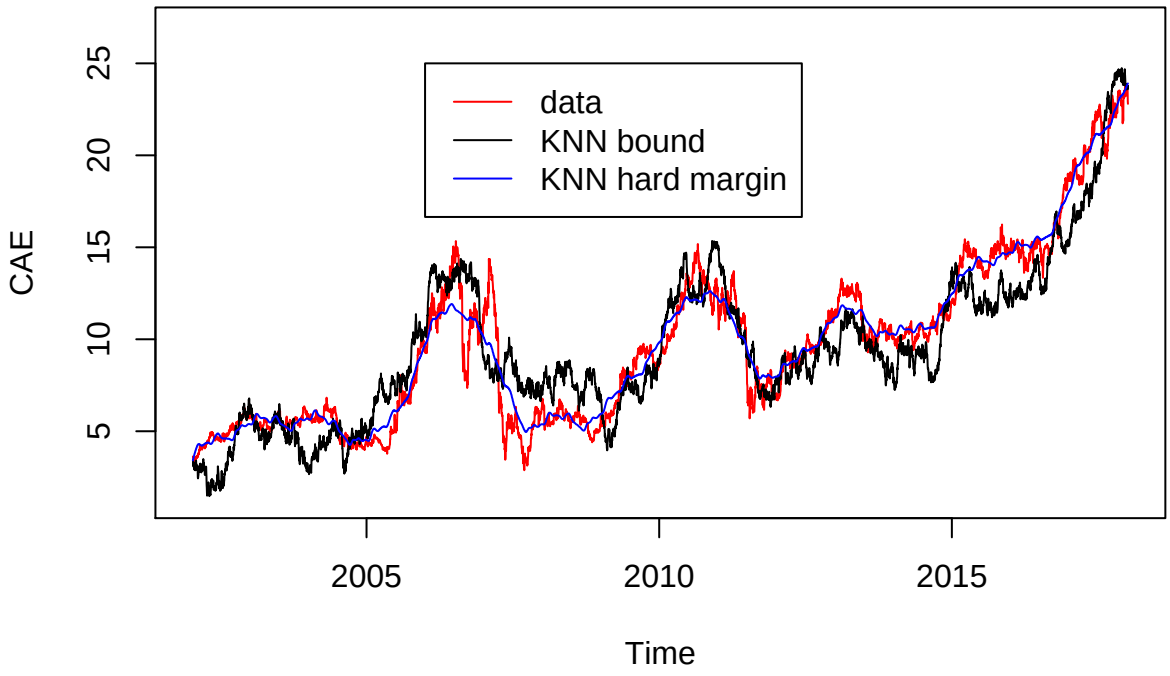



Optim check for regime switching model AR(1)




Fitted AR(1) with KNN


data
KNN
CAE
−5
0
5
10
15
20
25
2005
2010
2015
Time




Optim check for the rugarch package

```
##
## 'optim_proj' check on 4-variable minimization problem.
##
## Top 4 relative errors in potential solution:
##
##             xsol D=xopt-xsol R=D/|xsol|
## omega  2.041e-06  -1.021e-06       -0.5
## alpha1 1.061e-01  -5.303e-02       -0.5
## beta1  8.744e-01  -4.372e-01       -0.5
## shape  5.684e+00  -2.842e+00       -0.5
##
##
## 'optim_proj' check on 4-variable minimization problem.
##
## Top 4 relative errors in potential solution:
##
##             xsol D=xopt-xsol R=D/|xsol|
## omega  4.129e-06  -2.064e-06    -0.5000
## alpha1 8.183e-02  -4.091e-02    -0.5000
## beta1  9.149e-01  -4.574e-01    -0.5000
## shape  3.893e+00  -1.868e+00    -0.4798
##
##
## 'optim_proj' check on 4-variable minimization problem.
##
```

```
## Top 4 relative errors in potential solution:
##
##             xsol D=xopt-xsol R=D/|xsol|
## omega  1.045e-05  -5.225e-06       -0.5
## alpha1 1.462e-01  -7.308e-02       -0.5
## beta1  8.084e-01  -4.042e-01       -0.5
## shape  4.285e+00  -2.142e+00       -0.5
##
##
## 'optim_proj' check on 4-variable minimization problem.
##
## Top 4 relative errors in potential solution:
##
##             xsol D=xopt-xsol R=D/|xsol|
## omega  5.349e-06  -2.675e-06    -0.5000
## alpha1 9.750e-02  -4.875e-02    -0.5000
## beta1  9.015e-01  -4.507e-01    -0.5000
## shape  3.565e+00  -1.530e+00    -0.4293
##
##
## 'optim_proj' check on 4-variable minimization problem.
##
## Top 4 relative errors in potential solution:
##
##             xsol D=xopt-xsol R=D/|xsol|
## omega  3.227e-06  -1.613e-06       -0.5
## alpha1 8.734e-02  -4.367e-02       -0.5
## beta1  9.014e-01  -4.507e-01       -0.5
## shape  4.360e+00  -2.180e+00       -0.5
##
##
## 'optim_proj' check on 4-variable minimization problem.
##
## Top 4 relative errors in potential solution:
##
##             xsol D=xopt-xsol R=D/|xsol|
## omega  3.189e-06  -1.594e-06       -0.5
## alpha1 8.572e-02  -4.286e-02       -0.5
## beta1  9.133e-01  -4.566e-01       -0.5
## shape  6.247e+00  -3.123e+00       -0.5
##
##
## 'optim_proj' check on 4-variable minimization problem.
##
## Top 4 relative errors in potential solution:
##
##             xsol D=xopt-xsol R=D/|xsol|
## omega  0.0001951  -9.755e-05    -0.5000
## alpha1 0.2747000  -1.373e-01    -0.5000
## beta1  0.6514000  -3.257e-01    -0.5000
## shape  3.8260000  -1.797e+00    -0.4697
##
##
## 'optim_proj' check on 4-variable minimization problem.
```

```
##
## Top 4 relative errors in potential solution:
##
##             xsol D=xopt-xsol R=D/|xsol|
## omega  4.411e-06  -2.205e-06       -0.5
## alpha1 5.433e-02  -2.717e-02       -0.5
## beta1  9.353e-01  -4.677e-01       -0.5
## shape  5.607e+00  -2.803e+00       -0.5
##
##
## 'optim_proj' check on 4-variable minimization problem.
##
## Top 4 relative errors in potential solution:
##
##             xsol D=xopt-xsol R=D/|xsol|
## omega  2.398e-06  -1.199e-06       -0.5
## alpha1 8.318e-02  -4.159e-02       -0.5
## beta1  9.158e-01  -4.579e-01       -0.5
## shape  5.368e+00  -2.684e+00       -0.5
##
##
## 'optim_proj' check on 4-variable minimization problem.
##
## Top 4 relative errors in potential solution:
##
##             xsol D=xopt-xsol R=D/|xsol|
## omega  1.213e-06  -6.066e-07       -0.5
## alpha1 4.125e-02  -2.063e-02       -0.5
## beta1  9.577e-01  -4.789e-01       -0.5
## shape  5.705e+00  -2.853e+00       -0.5
##
##
## 'optim_proj' check on 4-variable minimization problem.
##
## Top 4 relative errors in potential solution:
##
##             xsol D=xopt-xsol R=D/|xsol|
## omega  2.816e-06  -1.408e-06       -0.5
## alpha1 5.899e-02  -2.950e-02       -0.5
## beta1  9.400e-01  -4.700e-01       -0.5
## shape  5.844e+00  -2.922e+00       -0.5
##
##
## 'optim_proj' check on 4-variable minimization problem.
##
## Top 4 relative errors in potential solution:
##
##             xsol D=xopt-xsol R=D/|xsol|
## omega  2.345e-06  -1.173e-06    -0.5000
## alpha1 2.097e-02  -1.049e-02    -0.5000
## beta1  9.780e-01  -4.890e-01    -0.5000
## shape  2.894e+00  -8.916e-01    -0.3081
##
##
```

```
## 'optim_proj' check on 4-variable minimization problem.
##
## Top 4 relative errors in potential solution:
##
##             xsol D=xopt-xsol R=D/|xsol|
## omega  4.023e-05  -2.012e-05    -0.5000
## alpha1 1.259e-01  -6.295e-02    -0.5000
## beta1  8.534e-01  -4.267e-01    -0.5000
## shape  3.425e+00  -1.401e+00    -0.4091
##
##
## 'optim_proj' check on 4-variable minimization problem.
##
## Top 4 relative errors in potential solution:
##
##             xsol D=xopt-xsol R=D/|xsol|
## omega  0.0000208  -0.0000104       -0.5
## alpha1 0.1177000  -0.0588700       -0.5
## beta1  0.8794000  -0.4397000       -0.5
## shape  4.4650000  -2.2320000       -0.5
##
##
## 'optim_proj' check on 4-variable minimization problem.
##
## Top 4 relative errors in potential solution:
##
##             xsol D=xopt-xsol R=D/|xsol|
## omega  6.856e-06  -3.428e-06       -0.5
## alpha1 1.008e-01  -5.039e-02       -0.5
## beta1  8.982e-01  -4.491e-01       -0.5
## shape  6.427e+00  -3.213e+00       -0.5
##
##
## 'optim_proj' check on 4-variable minimization problem.
##
## Top 4 relative errors in potential solution:
##
##             xsol D=xopt-xsol R=D/|xsol|
## omega  5.416e-06  -2.708e-06       -0.5
## alpha1 6.622e-02  -3.311e-02       -0.5
## beta1  9.124e-01  -4.562e-01       -0.5
## shape  6.297e+00  -3.148e+00       -0.5
##
##
## 'optim_proj' check on 4-variable minimization problem.
##
## Top 4 relative errors in potential solution:
##
##             xsol D=xopt-xsol R=D/|xsol|
## omega  1.718e-06  -8.591e-07       -0.5
## alpha1 5.245e-02  -2.622e-02       -0.5
## beta1  9.333e-01  -4.666e-01       -0.5
## shape  5.698e+00  -2.849e+00       -0.5
##
```

```
##
## 'optim_proj' check on 4-variable minimization problem.
##
## Top 4 relative errors in potential solution:
##
##             xsol D=xopt-xsol R=D/|xsol|
## omega  2.411e-06  -1.205e-06       -0.5
## alpha1 7.622e-02  -3.811e-02       -0.5
## beta1  9.206e-01  -4.603e-01       -0.5
## shape  5.758e+00  -2.879e+00       -0.5
##
##
## 'optim_proj' check on 4-variable minimization problem.
##
## Top 4 relative errors in potential solution:
##
##             xsol D=xopt-xsol R=D/|xsol|
## omega  5.219e-06  -2.610e-06       -0.5
## alpha1 4.268e-02  -2.134e-02       -0.5
## beta1  9.531e-01  -4.765e-01       -0.5
## shape  7.568e+00   3.784e+00        0.5
##
##
## 'optim_proj' check on 4-variable minimization problem.
##
## Top 4 relative errors in potential solution:
##
##             xsol D=xopt-xsol R=D/|xsol|
## omega  3.318e-06  -1.659e-06       -0.5
## alpha1 2.887e-03  -1.444e-03       -0.5
## beta1  9.726e-01  -4.863e-01       -0.5
## shape  4.430e+00  -2.215e+00       -0.5
##
##
## 'optim_proj' check on 4-variable minimization problem.
##
## Top 4 relative errors in potential solution:
##
##             xsol D=xopt-xsol R=D/|xsol|
## omega  4.226e-06  -2.113e-06       -0.5
## alpha1 1.782e-01  -8.910e-02       -0.5
## beta1  7.798e-01  -3.899e-01       -0.5
## shape  4.255e+00  -2.127e+00       -0.5
##
##
## 'optim_proj' check on 4-variable minimization problem.
##
## Top 4 relative errors in potential solution:
##
##             xsol D=xopt-xsol R=D/|xsol|
## omega  9.911e-07  -4.955e-07       -0.5
## alpha1 6.789e-02  -3.394e-02       -0.5
## beta1  9.218e-01  -4.609e-01       -0.5
## shape  5.938e+00  -2.969e+00       -0.5
```

```
##
##
## 'optim_proj' check on 4-variable minimization problem.
##
## Top 4 relative errors in potential solution:
##
##             xsol D=xopt-xsol R=D/|xsol|
## omega  1.884e-06  -9.422e-07       -0.5
## alpha1 6.381e-02  -3.190e-02       -0.5
## beta1  9.130e-01  -4.565e-01       -0.5
## shape  4.261e+00  -2.131e+00       -0.5
##
##
## 'optim_proj' check on 4-variable minimization problem.
##
## Top 4 relative errors in potential solution:
##
##             xsol D=xopt-xsol R=D/|xsol|
## omega  3.796e-05  -1.898e-05     -0.500
## alpha1 1.578e-01  -7.891e-02     -0.500
## beta1  8.412e-01  -4.206e-01     -0.500
## shape  2.859e+00  -8.520e-01     -0.298
##
##
## 'optim_proj' check on 4-variable minimization problem.
##
## Top 4 relative errors in potential solution:
##
##             xsol D=xopt-xsol R=D/|xsol|
## omega  2.782e-06  -1.391e-06       -0.5
## alpha1 4.939e-02  -2.470e-02       -0.5
## beta1  9.392e-01  -4.696e-01       -0.5
## shape  5.255e+00  -2.627e+00       -0.5
##
##
## 'optim_proj' check on 4-variable minimization problem.
##
## Top 4 relative errors in potential solution:
##
##             xsol D=xopt-xsol R=D/|xsol|
## omega  1.512e-06  -7.558e-07       -0.5
## alpha1 8.251e-02  -4.125e-02       -0.5
## beta1  9.001e-01  -4.500e-01       -0.5
## shape  6.374e+00  -3.187e+00       -0.5
##
##
## 'optim_proj' check on 4-variable minimization problem.
##
## Top 4 relative errors in potential solution:
##
##             xsol D=xopt-xsol R=D/|xsol|
## omega  4.926e-06  -2.463e-06       -0.5
## alpha1 7.571e-02  -3.786e-02       -0.5
## beta1  8.989e-01  -4.494e-01       -0.5
```

```
## shape  4.517e+00  -2.258e+00       -0.5
##
##
## 'optim_proj' check on 4-variable minimization problem.
##
## Top 4 relative errors in potential solution:
##
##             xsol D=xopt-xsol R=D/|xsol|
## omega  2.835e-06  -1.418e-06       -0.5
## alpha1 1.131e-01  -5.657e-02       -0.5
## beta1  8.489e-01  -4.244e-01       -0.5
## shape  5.395e+00  -2.698e+00       -0.5
##
##
## 'optim_proj' check on 4-variable minimization problem.
##
## Top 4 relative errors in potential solution:
##
##             xsol D=xopt-xsol R=D/|xsol|
## omega  2.741e-06  -1.371e-06       -0.5
## alpha1 4.519e-02  -2.260e-02       -0.5
## beta1  9.538e-01  -4.769e-01       -0.5
## shape  4.415e+00  -2.207e+00       -0.5
##
##
## 'optim_proj' check on 4-variable minimization problem.
##
## Top 4 relative errors in potential solution:
##
##             xsol D=xopt-xsol R=D/|xsol|
## omega  1.734e-06  -8.670e-07       -0.5
## alpha1 3.398e-02  -1.699e-02       -0.5
## beta1  9.650e-01  -4.825e-01       -0.5
## shape  4.726e+00  -2.363e+00       -0.5
##
##
## 'optim_proj' check on 4-variable minimization problem.
##
## Top 4 relative errors in potential solution:
##
##             xsol D=xopt-xsol R=D/|xsol|
## omega  8.463e-06  -4.231e-06       -0.5
## alpha1 3.999e-02  -2.000e-02       -0.5
## beta1  9.543e-01  -4.771e-01       -0.5
## shape  5.099e+00  -2.549e+00       -0.5
##
##
## 'optim_proj' check on 4-variable minimization problem.
##
## Top 4 relative errors in potential solution:
##
##             xsol D=xopt-xsol R=D/|xsol|
## omega  4.087e-05  -2.044e-05       -0.5
## alpha1 4.798e-02  -2.399e-02       -0.5
```

```
## beta1  8.560e-01  -4.280e-01       -0.5
## shape  4.805e+00  -2.402e+00       -0.5
##
##
## 'optim_proj' check on 4-variable minimization problem.
##
## Top 4 relative errors in potential solution:
##
##             xsol D=xopt-xsol R=D/|xsol|
## omega  4.269e-05  -2.134e-05       -0.5
## alpha1 7.992e-02  -3.996e-02       -0.5
## beta1  9.007e-01  -4.504e-01       -0.5
## shape  5.521e+00  -2.760e+00       -0.5
##
##
## 'optim_proj' check on 4-variable minimization problem.
##
## Top 4 relative errors in potential solution:
##
##             xsol D=xopt-xsol R=D/|xsol|
## omega  2.505e-05  -1.253e-05       -0.5
## alpha1 4.560e-02  -2.280e-02       -0.5
## beta1  9.357e-01  -4.679e-01       -0.5
## shape  5.965e+00  -2.983e+00       -0.5
##
##
## 'optim_proj' check on 4-variable minimization problem.
##
## Top 4 relative errors in potential solution:
##
##             xsol D=xopt-xsol R=D/|xsol|
## omega  1.356e-06  -6.780e-07       -0.5
## alpha1 4.544e-02  -2.272e-02       -0.5
## beta1  9.469e-01  -4.734e-01       -0.5
## shape  5.263e+00  -2.631e+00       -0.5
##
##
## 'optim_proj' check on 4-variable minimization problem.
##
## Top 4 relative errors in potential solution:
##
##             xsol D=xopt-xsol R=D/|xsol|
## omega  6.371e-06  -3.185e-06       -0.5
## alpha1 8.293e-02  -4.146e-02       -0.5
## beta1  9.133e-01  -4.566e-01       -0.5
## shape  7.111e+00   3.555e+00        0.5
##
##
## 'optim_proj' check on 4-variable minimization problem.
##
## Top 4 relative errors in potential solution:
##
##             xsol D=xopt-xsol R=D/|xsol|
## omega  2.424e-06  -1.212e-06       -0.5
```

```
## alpha1 5.147e-02  -2.574e-02       -0.5
## beta1  9.458e-01  -4.729e-01       -0.5
## shape  6.406e+00  -3.203e+00       -0.5
##
##
## 'optim_proj' check on 4-variable minimization problem.
##
## Top 4 relative errors in potential solution:
##
##             xsol D=xopt-xsol R=D/|xsol|
## omega  1.733e-05  -8.665e-06       -0.5
## alpha1 5.348e-02  -2.674e-02       -0.5
## beta1  9.356e-01  -4.678e-01       -0.5
## shape  7.100e+00   3.550e+00        0.5
##
##
## 'optim_proj' check on 4-variable minimization problem.
##
## Top 4 relative errors in potential solution:
##
##             xsol D=xopt-xsol R=D/|xsol|
## omega  2.469e-05  -1.234e-05       -0.5
## alpha1 4.102e-02  -2.051e-02       -0.5
## beta1  9.443e-01  -4.721e-01       -0.5
## shape  5.460e+00  -2.730e+00       -0.5
##
##
## 'optim_proj' check on 4-variable minimization problem.
##
## Top 4 relative errors in potential solution:
##
##             xsol D=xopt-xsol R=D/|xsol|
## omega  2.376e-06  -1.188e-06       -0.5
## alpha1 4.095e-02  -2.047e-02       -0.5
## beta1  9.581e-01  -4.790e-01       -0.5
## shape  4.541e+00  -2.270e+00       -0.5
##
##
## 'optim_proj' check on 4-variable minimization problem.
##
## Top 4 relative errors in potential solution:
##
##             xsol D=xopt-xsol R=D/|xsol|
## omega  1.184e-05  -5.921e-06       -0.5
## alpha1 6.198e-02  -3.099e-02       -0.5
## beta1  9.294e-01  -4.647e-01       -0.5
## shape  6.998e+00   3.499e+00        0.5
##
##
## 'optim_proj' check on 4-variable minimization problem.
##
## Top 4 relative errors in potential solution:
##
##             xsol D=xopt-xsol R=D/|xsol|
```

```
## omega  7.093e-06  -3.547e-06       -0.5
## alpha1 5.046e-02  -2.523e-02       -0.5
## beta1  9.478e-01  -4.739e-01       -0.5
## shape  5.370e+00  -2.685e+00       -0.5
##
##
## 'optim_proj' check on 4-variable minimization problem.
##
## Top 4 relative errors in potential solution:
##
##             xsol D=xopt-xsol R=D/|xsol|
## omega  1.319e-05  -6.595e-06       -0.5
## alpha1 5.924e-02  -2.962e-02       -0.5
## beta1  9.310e-01  -4.655e-01       -0.5
## shape  5.407e+00  -2.703e+00       -0.5
##
##
## 'optim_proj' check on 4-variable minimization problem.
##
## Top 4 relative errors in potential solution:
##
##             xsol D=xopt-xsol R=D/|xsol|
## omega  0.0001812  -0.0000906    -0.5000
## alpha1 0.2477000  -0.1238000    -0.5000
## beta1  0.7264000  -0.3632000    -0.5000
## shape  3.6090000  -1.5860000    -0.4394
##
##
## 'optim_proj' check on 4-variable minimization problem.
##
## Top 4 relative errors in potential solution:
##
##            xsol D=xopt-xsol R=D/|xsol|
## omega  0.000126  -6.299e-05       -0.5
## alpha1 0.132300  -6.614e-02       -0.5
## beta1  0.807500  -4.038e-01       -0.5
## shape  8.499000   4.250e+00        0.5
##
##
## 'optim_proj' check on 4-variable minimization problem.
##
## Top 4 relative errors in potential solution:
##
##             xsol D=xopt-xsol R=D/|xsol|
## omega  4.765e-05  -2.383e-05    -0.5000
## alpha1 3.763e-02  -1.882e-02    -0.5000
## beta1  9.324e-01  -4.662e-01    -0.5000
## shape  3.943e+00  -1.932e+00    -0.4899
##
##
## 'optim_proj' check on 4-variable minimization problem.
##
## Top 4 relative errors in potential solution:
##
```

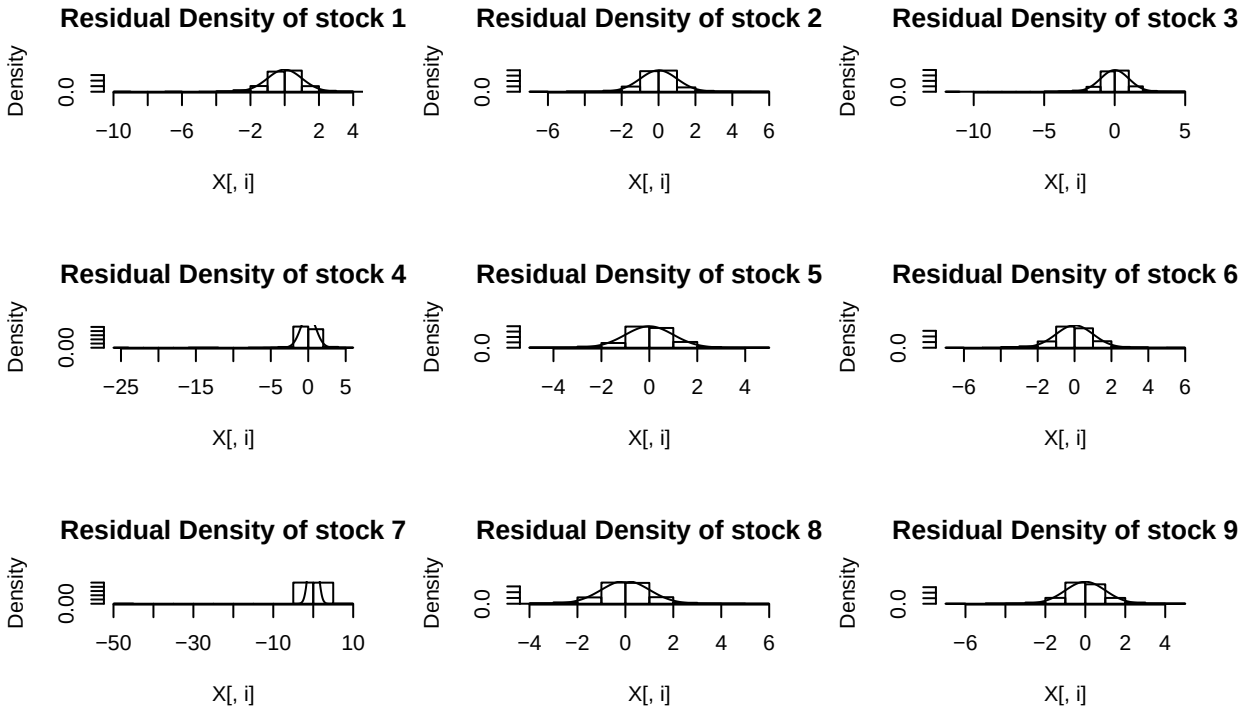
```
##             xsol D=xopt-xsol R=D/|xsol|
## omega  2.536e-05  -1.268e-05       -0.5
## alpha1 5.815e-02  -2.908e-02       -0.5
## beta1  9.251e-01  -4.626e-01       -0.5
## shape  6.291e+00  -3.145e+00       -0.5
##
##
## 'optim_proj' check on 4-variable minimization problem.
##
## Top 4 relative errors in potential solution:
##
##             xsol D=xopt-xsol R=D/|xsol|
## omega  3.904e-06  -1.952e-06       -0.5
## alpha1 4.838e-02  -2.419e-02       -0.5
## beta1  9.489e-01  -4.744e-01       -0.5
## shape  6.858e+00   3.429e+00        0.5
##
##
## 'optim_proj' check on 4-variable minimization problem.
##
## Top 4 relative errors in potential solution:
##
##             xsol D=xopt-xsol R=D/|xsol|
## omega  3.077e-05  -1.538e-05    -0.5000
## alpha1 1.043e-01  -5.217e-02    -0.5000
## beta1  8.722e-01  -4.361e-01    -0.5000
## shape  3.719e+00  -1.709e+00    -0.4596
##
##
## 'optim_proj' check on 4-variable minimization problem.
##
## Top 4 relative errors in potential solution:
##
##             xsol D=xopt-xsol R=D/|xsol|
## omega  4.650e-06  -2.325e-06       -0.5
## alpha1 4.150e-02  -2.075e-02       -0.5
## beta1  9.538e-01  -4.769e-01       -0.5
## shape  4.679e+00  -2.340e+00       -0.5
##
##
## 'optim_proj' check on 4-variable minimization problem.
##
## Top 4 relative errors in potential solution:
##
##             xsol D=xopt-xsol R=D/|xsol|
## omega  9.788e-06  -4.894e-06       -0.5
## alpha1 5.610e-02  -2.805e-02       -0.5
## beta1  9.359e-01  -4.679e-01       -0.5
## shape  9.178e+00   4.589e+00        0.5
##
##
## 'optim_proj' check on 4-variable minimization problem.
##
## Top 4 relative errors in potential solution:
```

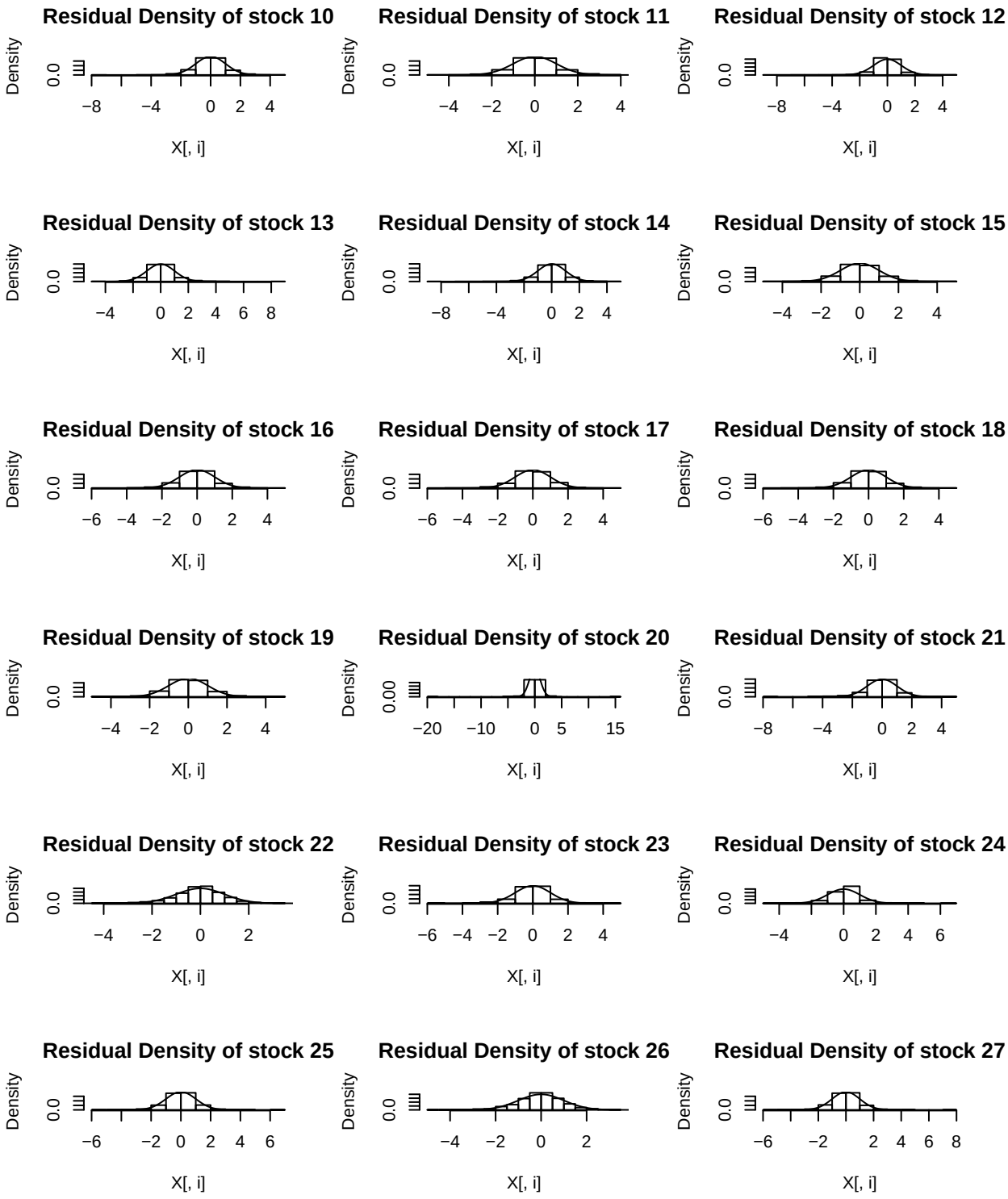
```
##
##             xsol D=xopt-xsol R=D/|xsol|
## omega  6.939e-07  -3.469e-07    -0.5000
## alpha1 4.754e-03  -2.377e-03    -0.5000
## beta1  9.930e-01  -4.965e-01    -0.5000
## shape  3.886e+00  -1.865e+00    -0.4798
##
##
## 'optim_proj' check on 4-variable minimization problem.
##
## Top 4 relative errors in potential solution:
##
##             xsol D=xopt-xsol R=D/|xsol|
## omega  2.564e-05  -1.282e-05    -0.5000
## alpha1 3.093e-02  -1.546e-02    -0.5000
## beta1  9.442e-01  -4.721e-01    -0.5000
## shape  3.271e+00  -1.239e+00    -0.3788
##
##
## 'optim_proj' check on 4-variable minimization problem.
##
## Top 4 relative errors in potential solution:
##
##             xsol D=xopt-xsol R=D/|xsol|
## omega  2.043e-07  -1.022e-07    -0.5000
## alpha1 7.389e-03  -3.695e-03    -0.5000
## beta1  9.916e-01  -4.958e-01    -0.5000
## shape  3.157e+00  -1.132e+00    -0.3586
##
##
## 'optim_proj' check on 4-variable minimization problem.
##
## Top 4 relative errors in potential solution:
##
##             xsol D=xopt-xsol R=D/|xsol|
## omega  1.033e-05  -5.166e-06       -0.5
## alpha1 5.359e-02  -2.679e-02       -0.5
## beta1  8.742e-01  -4.371e-01       -0.5
## shape  4.920e+00  -2.460e+00       -0.5
##
##
## 'optim_proj' check on 4-variable minimization problem.
##
## Top 4 relative errors in potential solution:
##
##             xsol D=xopt-xsol R=D/|xsol|
## omega  2.187e-12  -1.093e-12       -0.5
## alpha1 7.031e-03  -3.516e-03       -0.5
## beta1  9.919e-01  -4.960e-01       -0.5
## shape  4.515e+00  -2.257e+00       -0.5
##
##
## 'optim_proj' check on 4-variable minimization problem.
##
```

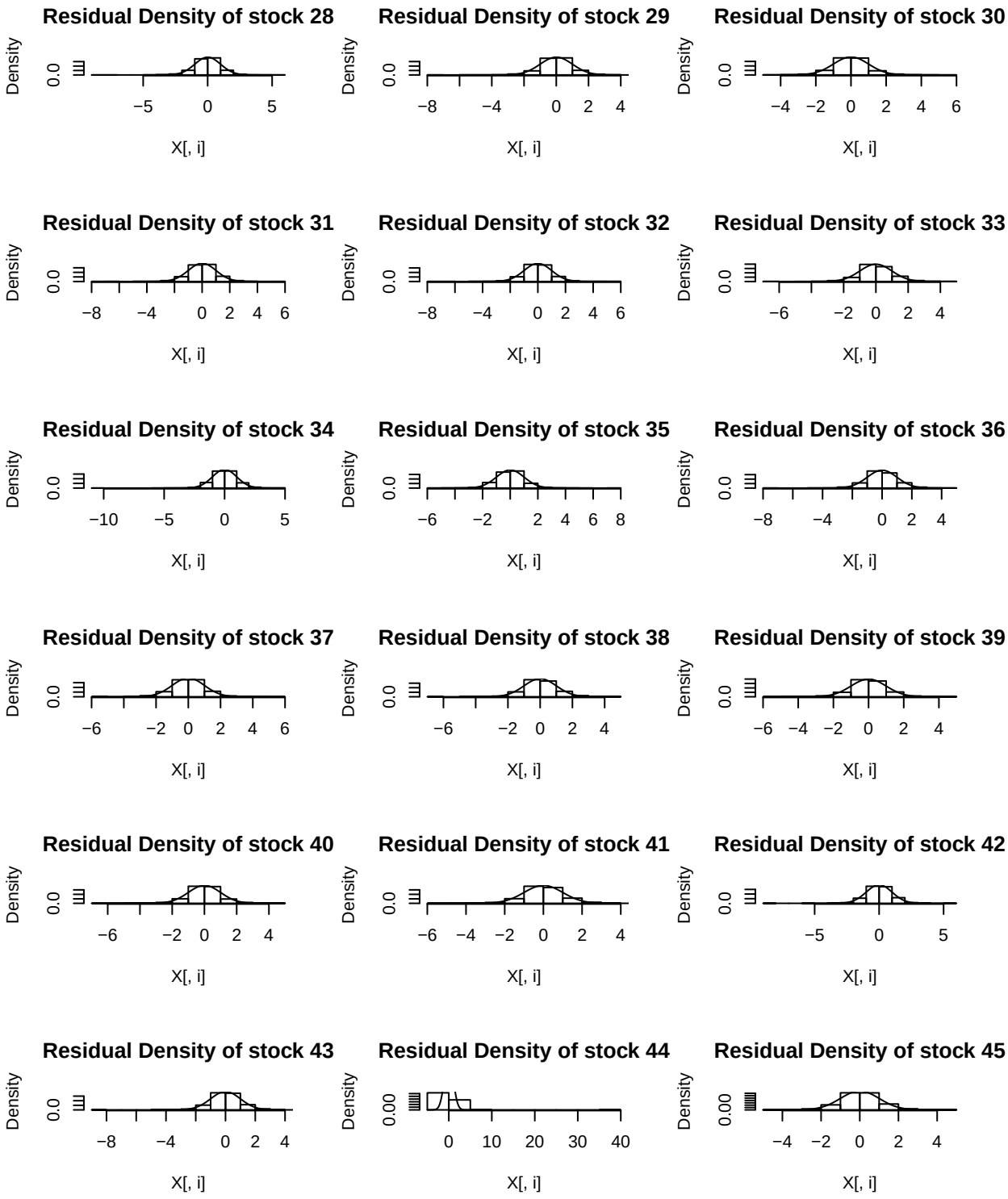
```
## Top 4 relative errors in potential solution:
##
##             xsol D=xopt-xsol R=D/|xsol|
## omega  1.065e-06  -5.324e-07       -0.5
## alpha1 3.304e-02  -1.652e-02       -0.5
## beta1  9.498e-01  -4.749e-01       -0.5
## shape  4.696e+00  -2.348e+00       -0.5
##
##
## 'optim_proj' check on 4-variable minimization problem.
##
## Top 4 relative errors in potential solution:
##
##             xsol D=xopt-xsol R=D/|xsol|
## omega  2.887e-06  -1.443e-06       -0.5
## alpha1 2.263e-02  -1.131e-02       -0.5
## beta1  9.512e-01  -4.756e-01       -0.5
## shape  4.167e+00  -2.083e+00       -0.5
##
##
## 'optim_proj' check on 4-variable minimization problem.
##
## Top 4 relative errors in potential solution:
##
##             xsol D=xopt-xsol R=D/|xsol|
## omega  3.940e-06  -1.970e-06    -0.5000
## alpha1 3.511e-02  -1.756e-02    -0.5000
## beta1  9.261e-01  -4.630e-01    -0.5000
## shape  3.857e+00  -1.851e+00    -0.4798
##
##
## 'optim_proj' check on 4-variable minimization problem.
##
## Top 4 relative errors in potential solution:
##
##             xsol D=xopt-xsol R=D/|xsol|
## omega  9.575e-07  -4.788e-07       -0.5
## alpha1 3.124e-02  -1.562e-02       -0.5
## beta1  9.591e-01  -4.795e-01       -0.5
## shape  5.010e+00  -2.505e+00       -0.5
##
##
## 'optim_proj' check on 4-variable minimization problem.
##
## Top 4 relative errors in potential solution:
##
##             xsol D=xopt-xsol R=D/|xsol|
## omega  1.229e-06  -6.147e-07       -0.5
## alpha1 3.892e-02  -1.946e-02       -0.5
## beta1  9.517e-01  -4.758e-01       -0.5
## shape  5.871e+00  -2.935e+00       -0.5
##
##
## 'optim_proj' check on 4-variable minimization problem.
```

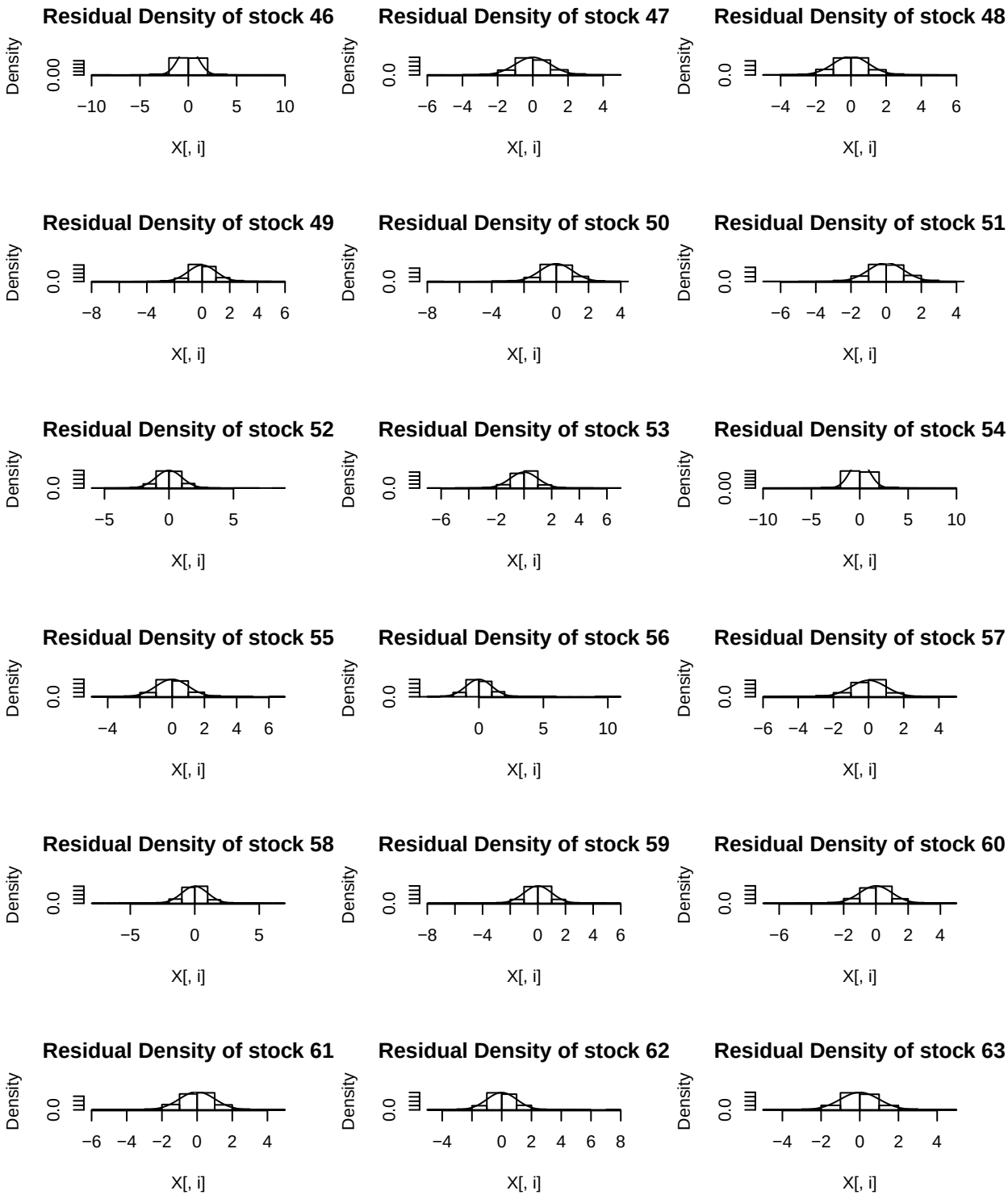
```
##
## Top 4 relative errors in potential solution:
##
##             xsol D=xopt-xsol R=D/|xsol|
## omega  8.351e-07  -4.176e-07       -0.5
## alpha1 2.422e-02  -1.211e-02       -0.5
## beta1  9.727e-01  -4.864e-01       -0.5
## shape  4.543e+00  -2.271e+00       -0.5
##
##
## 'optim_proj' check on 4-variable minimization problem.
##
## Top 4 relative errors in potential solution:
##
##             xsol D=xopt-xsol R=D/|xsol|
## omega  3.805e-06  -1.902e-06       -0.5
## alpha1 9.916e-02  -4.958e-02       -0.5
## beta1  8.863e-01  -4.431e-01       -0.5
## shape  4.834e+00  -2.417e+00       -0.5
```

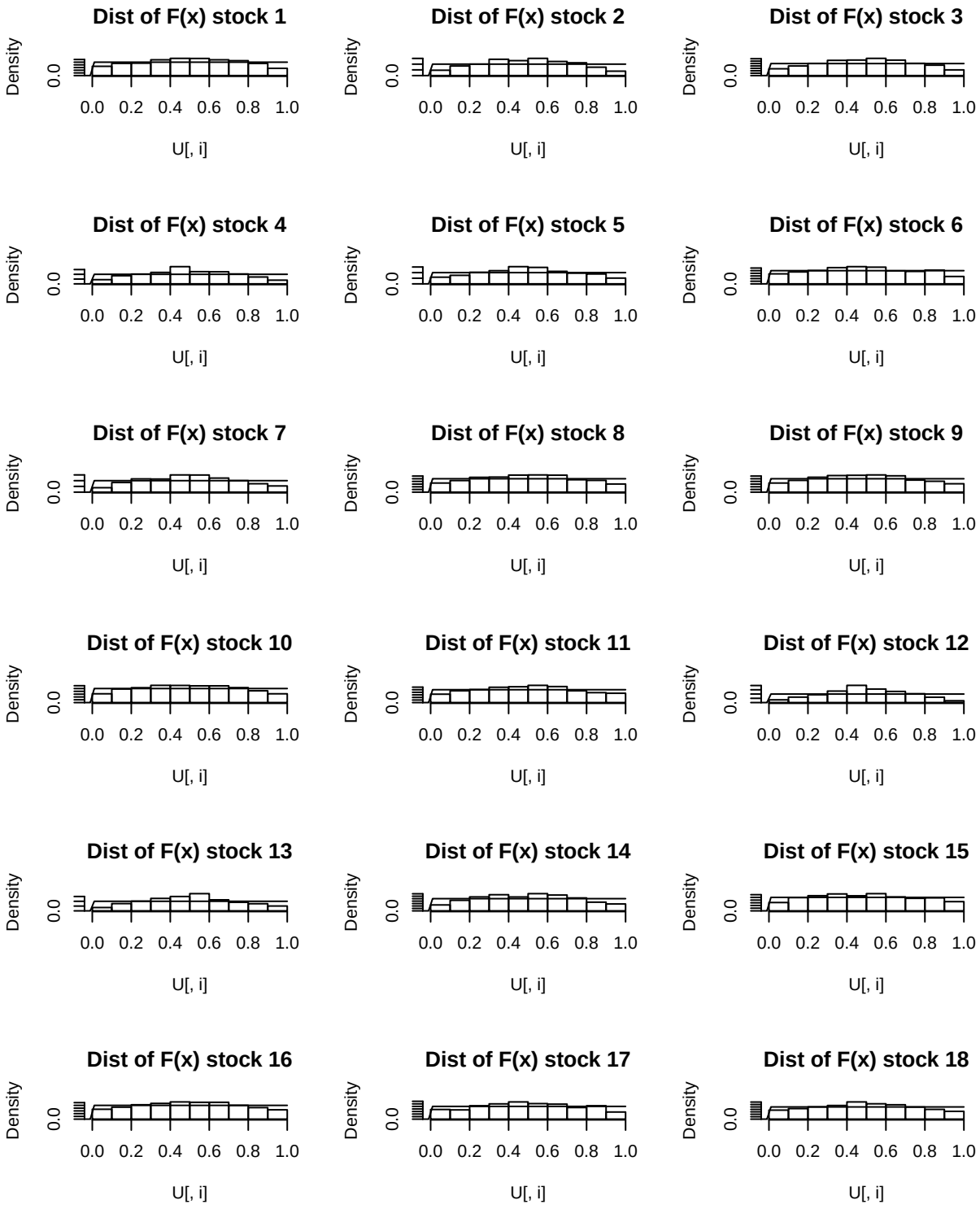
Distribution check for residuals and the transformed residuals

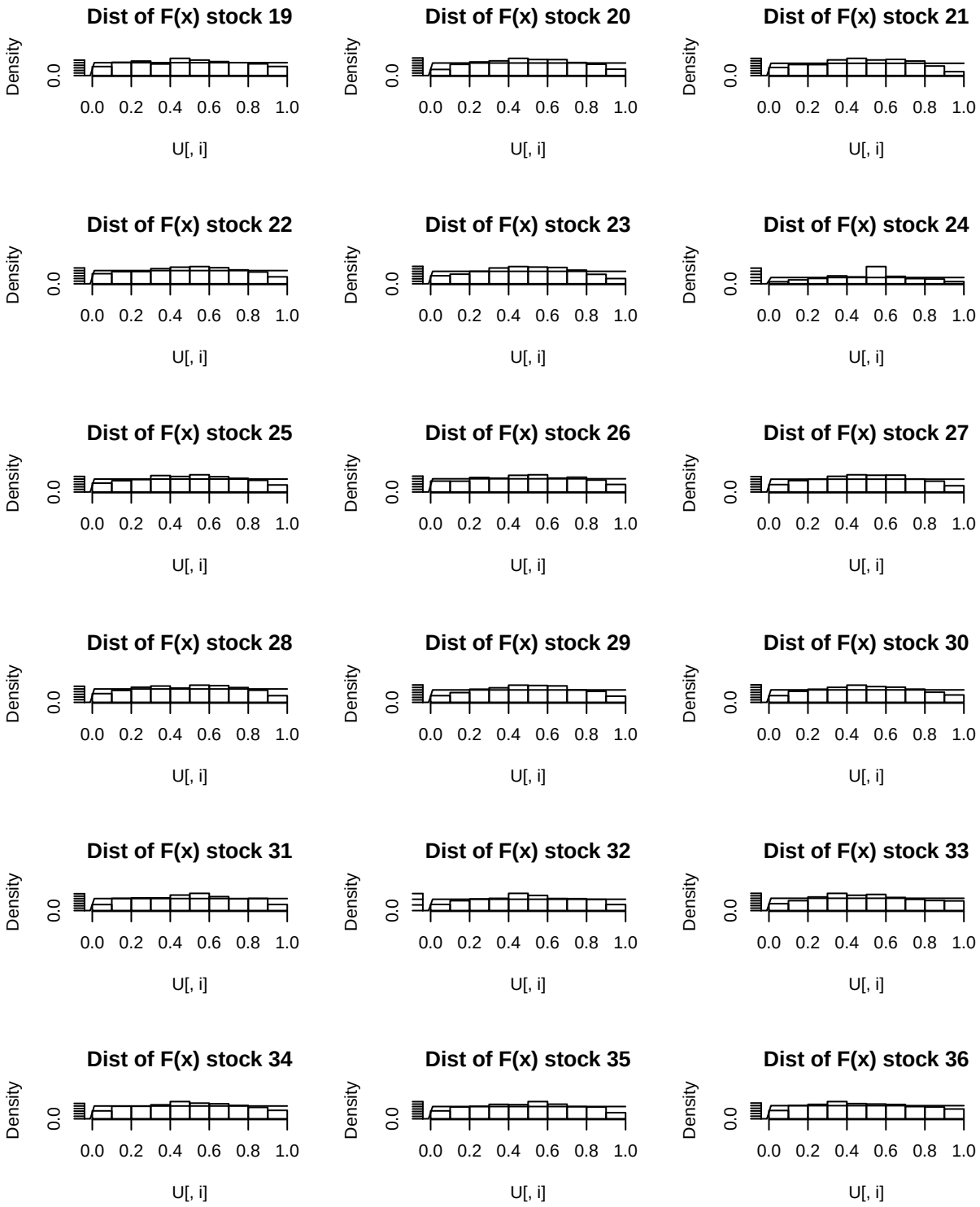


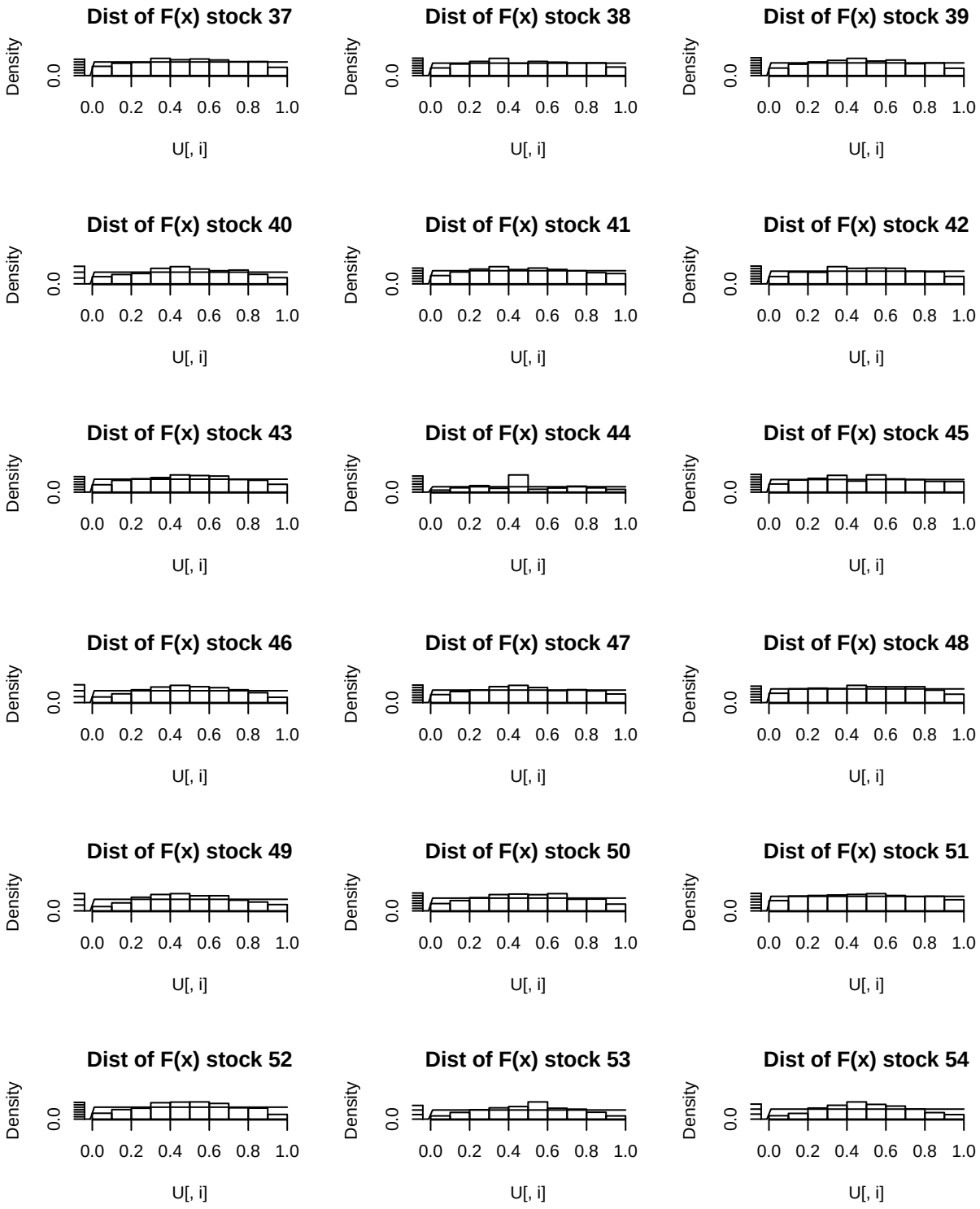

Residual Density of stock 10
Residual Density of stock 11
Residual Density of stock 12
Residual Density of stock 13
Residual Density of stock 14
Residual Density of stock 15
Residual Density of stock 16
Residual Density of stock 17
Residual Density of stock 18
Residual Density of stock 19
Residual Density of stock 20
Residual Density of stock 21
Residual Density of stock 22
Residual Density of stock 23
Residual Density of stock 24
Residual Density of stock 25
Residual Density of stock 26
Residual Density of stock 27
Density
X[, i]

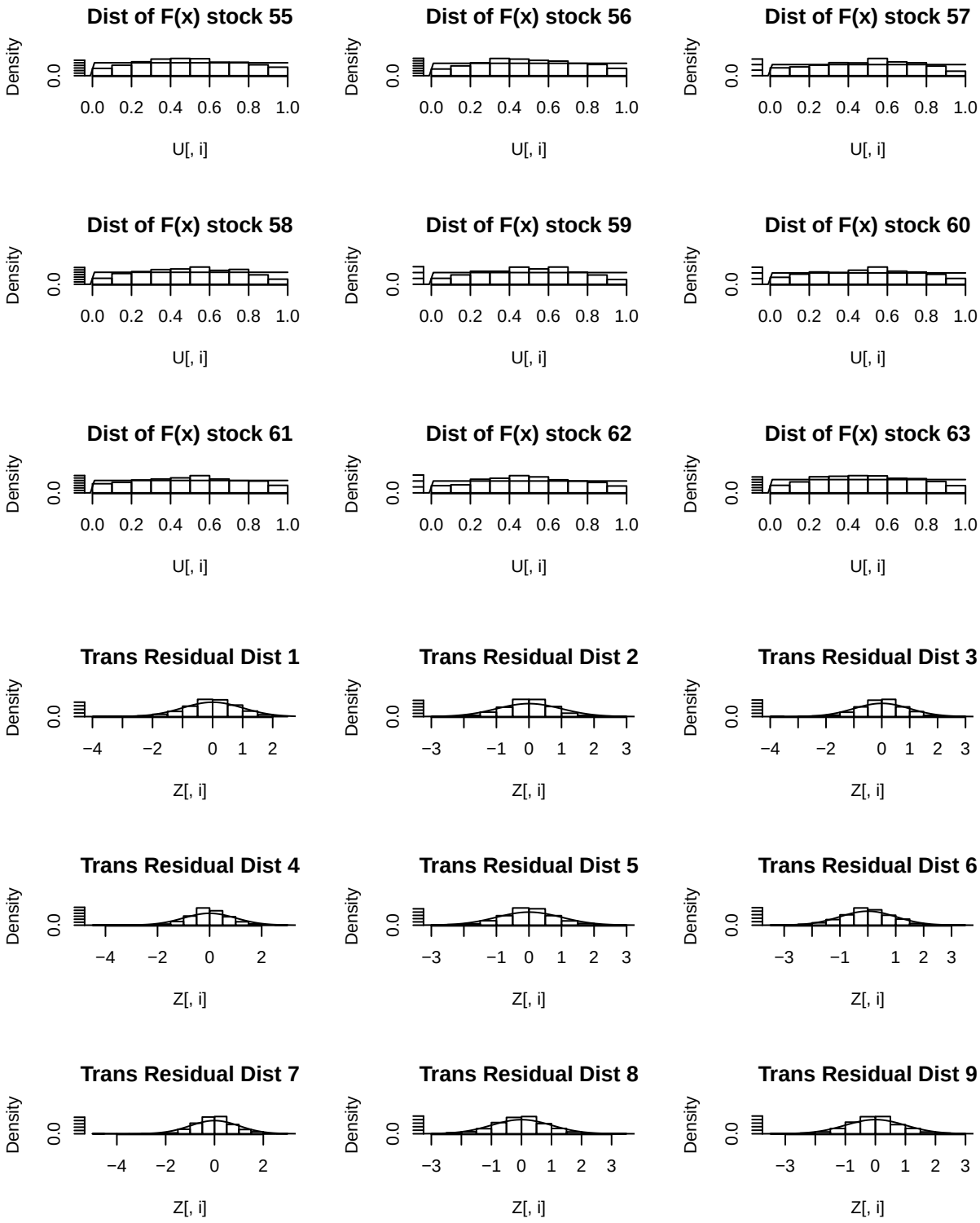

Residual Density of stock 28
Residual Density of stock 29
Residual Density of stock 30
Residual Density of stock 31
Residual Density of stock 32
Residual Density of stock 33
Residual Density of stock 34
Residual Density of stock 35
Residual Density of stock 36
Residual Density of stock 37
Residual Density of stock 38
Residual Density of stock 39
Residual Density of stock 40
Residual Density of stock 41
Residual Density of stock 42
Residual Density of stock 43
Residual Density of stock 44
Residual Density of stock 45
Density
X[, i]

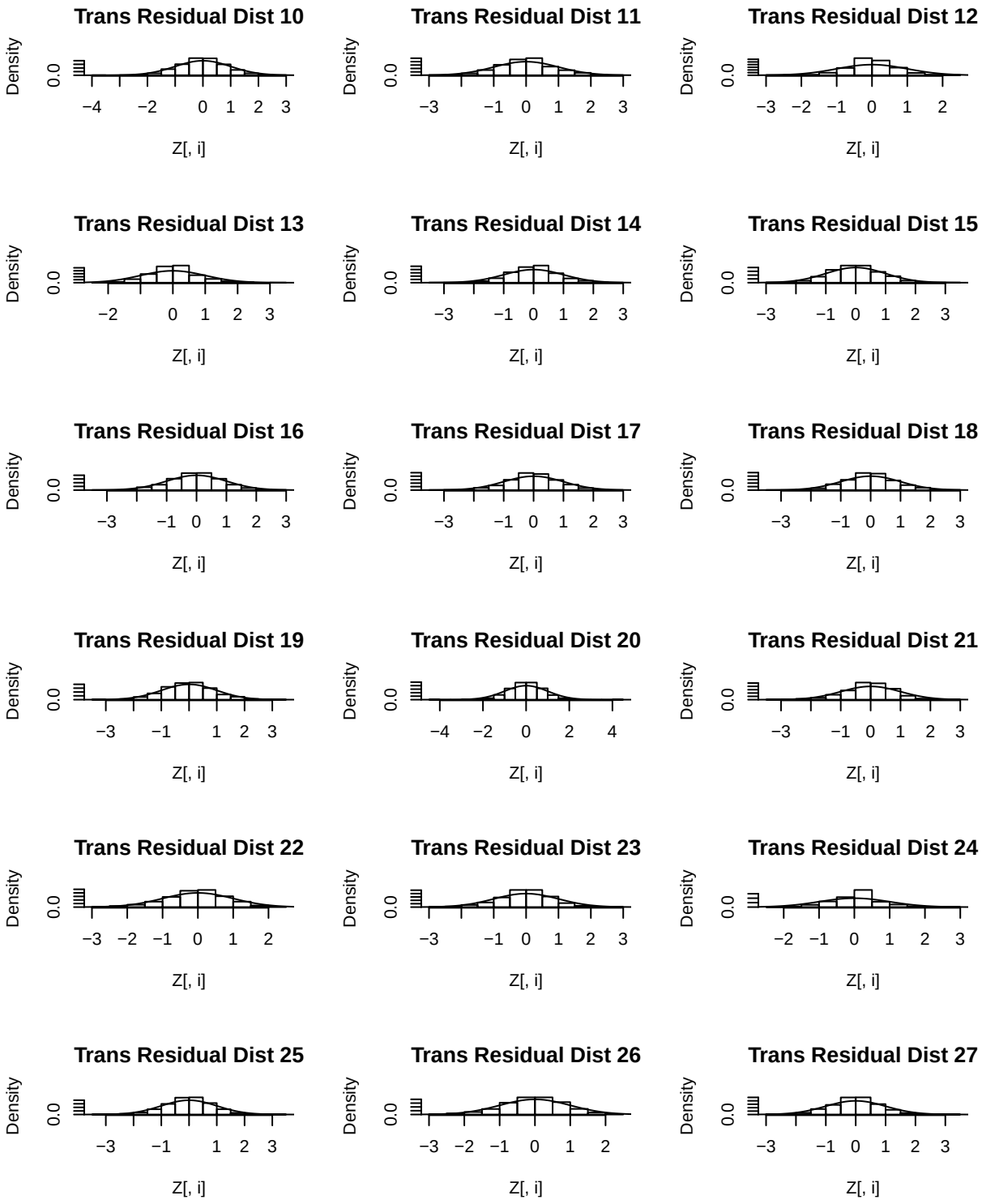

Residual Density of stock 46
Residual Density of stock 47
Residual Density of stock 48
Residual Density of stock 49
Residual Density of stock 50
Residual Density of stock 51
Residual Density of stock 52
Residual Density of stock 53
Residual Density of stock 54
Residual Density of stock 55
Residual Density of stock 56
Residual Density of stock 57
Residual Density of stock 58
Residual Density of stock 59
Residual Density of stock 60
Residual Density of stock 61
Residual Density of stock 62
Residual Density of stock 63
Density
X[, i]

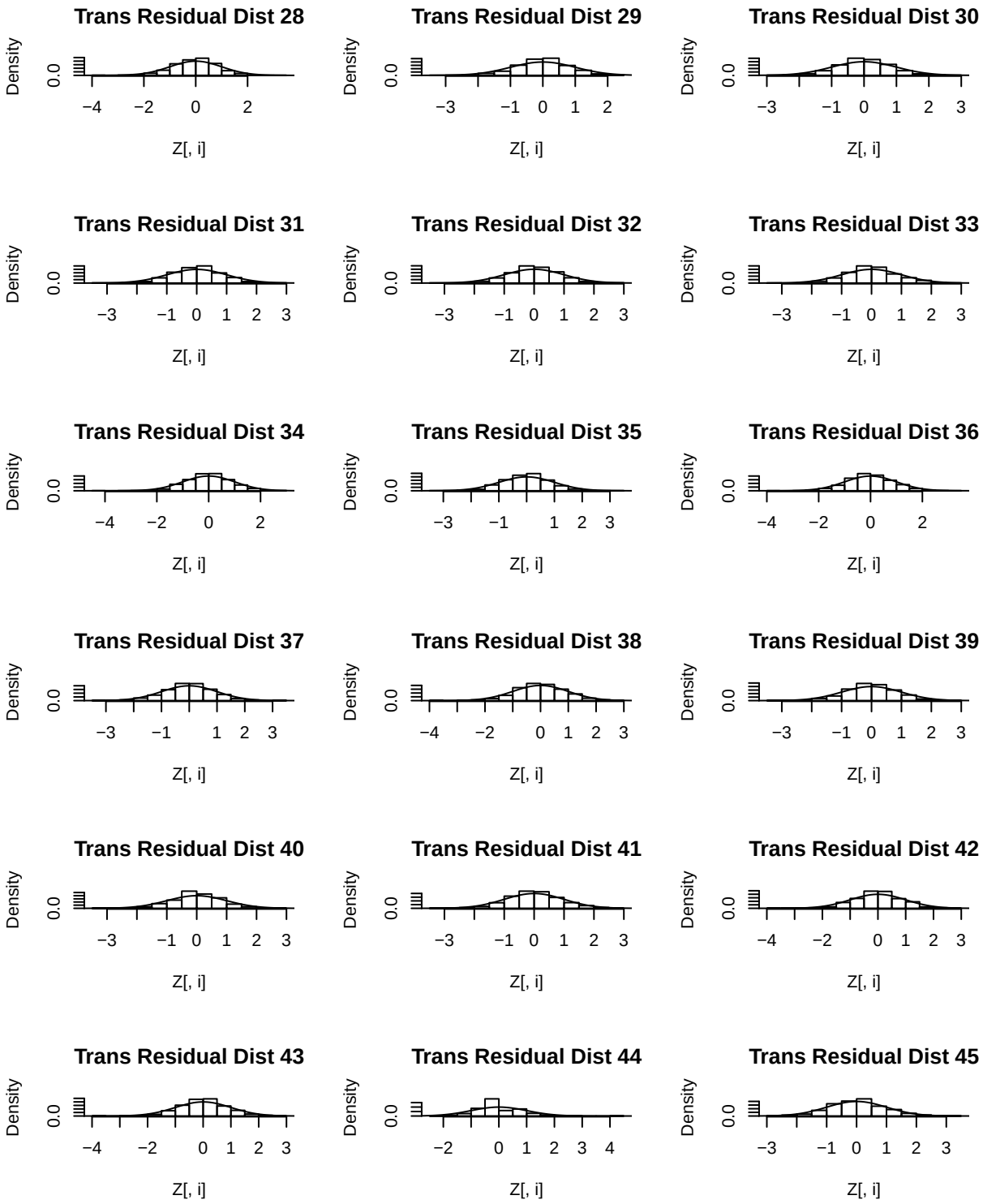

Dist of F(x) stock 1
Dist of F(x) stock 2
Dist of F(x) stock 3
Dist of F(x) stock 4
Dist of F(x) stock 5
Dist of F(x) stock 6
Dist of F(x) stock 7
Dist of F(x) stock 8
Dist of F(x) stock 9
Dist of F(x) stock 10
Dist of F(x) stock 11
Dist of F(x) stock 12
Dist of F(x) stock 13
Dist of F(x) stock 14
Dist of F(x) stock 15
Dist of F(x) stock 16
Dist of F(x) stock 17
Dist of F(x) stock 18
Density
0.0
0.0 0.2 0.4 0.6 0.8 1.0
U[, i]

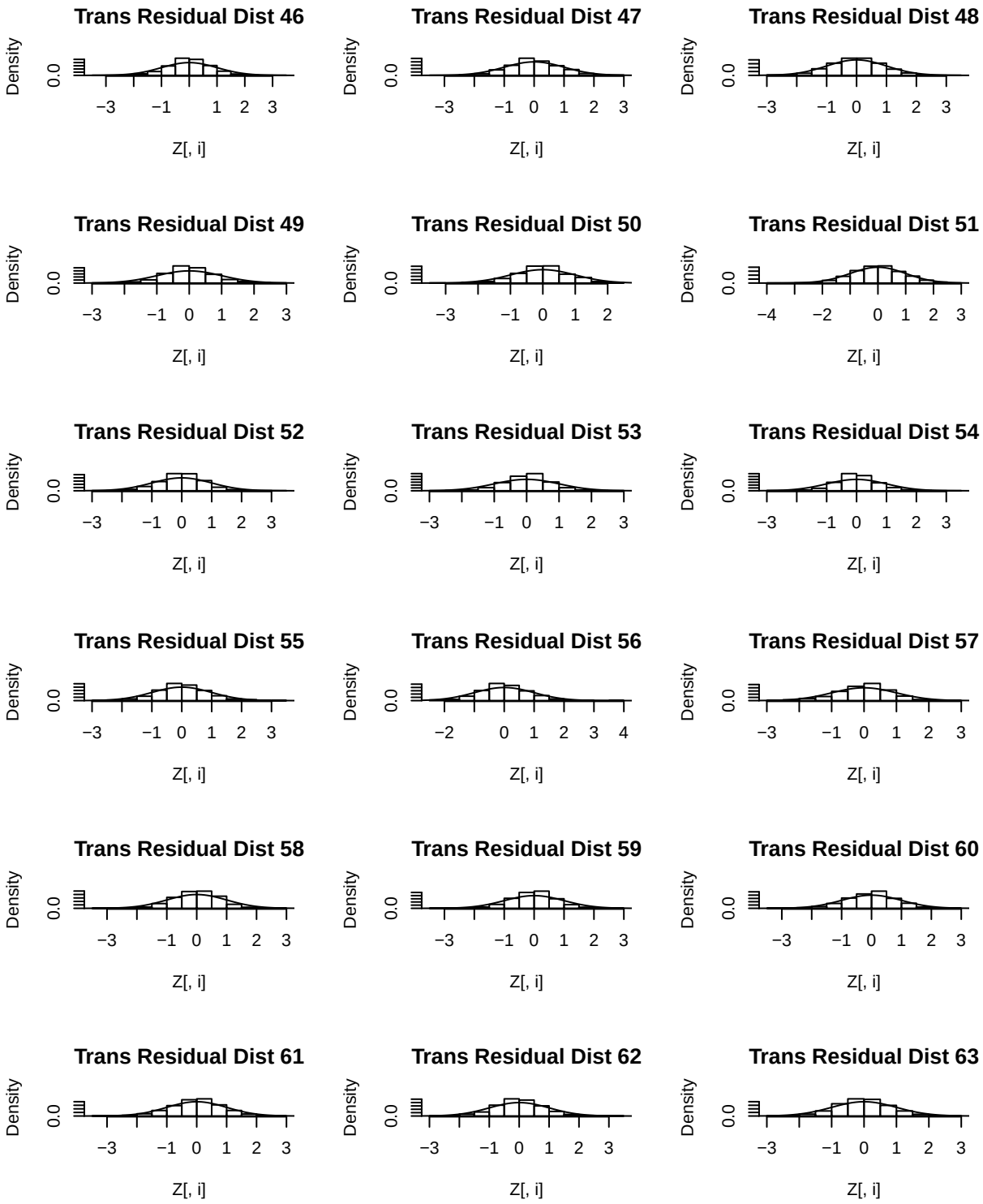

Dist of F(x) stock 19
Dist of F(x) stock 20
Dist of F(x) stock 21
Dist of F(x) stock 22
Dist of F(x) stock 23
Dist of F(x) stock 24
Dist of F(x) stock 25
Dist of F(x) stock 26
Dist of F(x) stock 27
Dist of F(x) stock 28
Dist of F(x) stock 29
Dist of F(x) stock 30
Dist of F(x) stock 31
Dist of F(x) stock 32
Dist of F(x) stock 33
Dist of F(x) stock 34
Dist of F(x) stock 35
Dist of F(x) stock 36
Density
0.0
0.0 0.2 0.4 0.6 0.8 1.0
U[, i]


Dist of F(x) stock 37
Dist of F(x) stock 38
Dist of F(x) stock 39
Dist of F(x) stock 40
Dist of F(x) stock 41
Dist of F(x) stock 42
Dist of F(x) stock 43
Dist of F(x) stock 44
Dist of F(x) stock 45
Dist of F(x) stock 46
Dist of F(x) stock 47
Dist of F(x) stock 48
Dist of F(x) stock 49
Dist of F(x) stock 50
Dist of F(x) stock 51
Dist of F(x) stock 52
Dist of F(x) stock 53
Dist of F(x) stock 54
Density
0.0
0.0 0.2 0.4 0.6 0.8 1.0
U[, i]

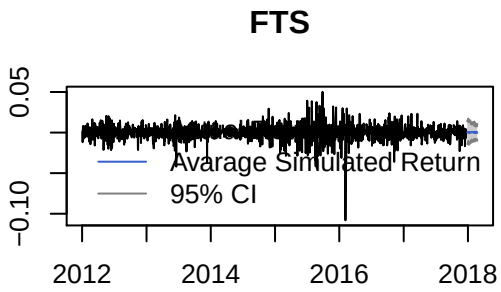

Dist of F(x) stock 55
Dist of F(x) stock 56
Dist of F(x) stock 57
Dist of F(x) stock 58
Dist of F(x) stock 59
Dist of F(x) stock 60
Dist of F(x) stock 61
Dist of F(x) stock 62
Dist of F(x) stock 63
Trans Residual Dist 1
Trans Residual Dist 2
Trans Residual Dist 3
Trans Residual Dist 4
Trans Residual Dist 5
Trans Residual Dist 6
Trans Residual Dist 7
Trans Residual Dist 8
Trans Residual Dist 9
Density
U[, i]
Z[, i]

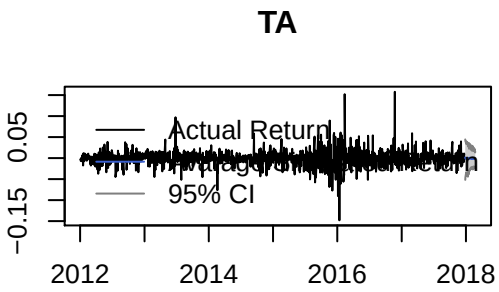

Trans Residual Dist 10
Trans Residual Dist 11
Trans Residual Dist 12
Trans Residual Dist 13
Trans Residual Dist 14
Trans Residual Dist 15
Trans Residual Dist 16
Trans Residual Dist 17
Trans Residual Dist 18
Trans Residual Dist 19
Trans Residual Dist 20
Trans Residual Dist 21
Trans Residual Dist 22
Trans Residual Dist 23
Trans Residual Dist 24
Trans Residual Dist 25
Trans Residual Dist 26
Trans Residual Dist 27
Density
0.0
Z[, i]

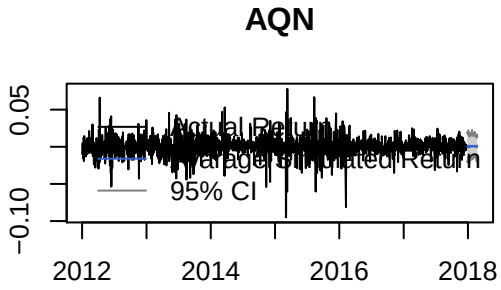

Trans Residual Dist 28
Trans Residual Dist 29
Trans Residual Dist 30
Trans Residual Dist 31
Trans Residual Dist 32
Trans Residual Dist 33
Trans Residual Dist 34
Trans Residual Dist 35
Trans Residual Dist 36
Trans Residual Dist 37
Trans Residual Dist 38
Trans Residual Dist 39
Trans Residual Dist 40
Trans Residual Dist 41
Trans Residual Dist 42
Trans Residual Dist 43
Trans Residual Dist 44
Trans Residual Dist 45
Density
0.0
Z[, i]

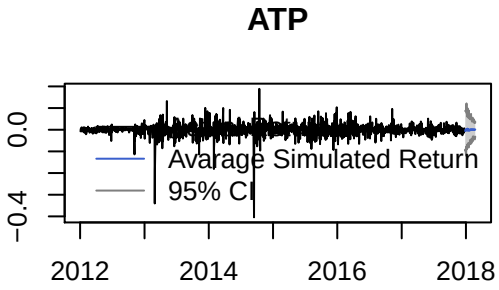


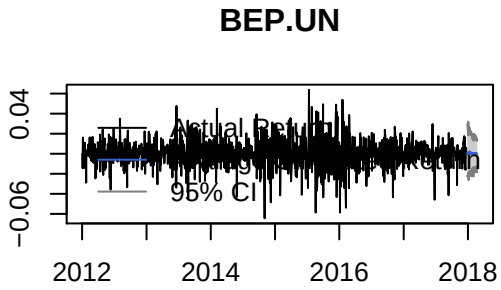

Fitting check and distribution check for the sector-level hierarchical modeling

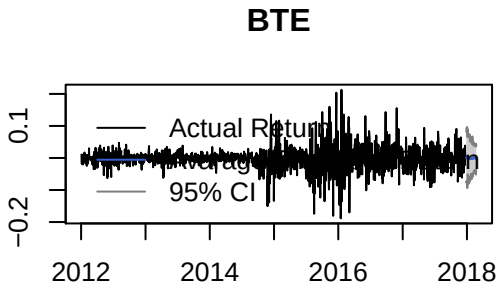

FTS
0.05
−0.10
Avarage Simulated Return
95% CI
2012
2014
2016
2018

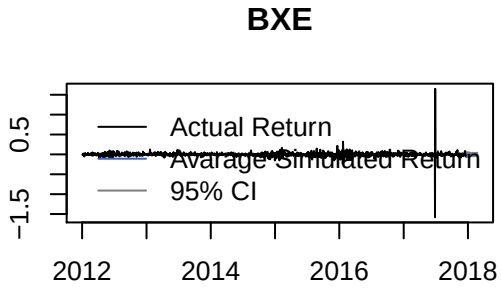

TA
0.05
−0.15
Actual Return
95% CI
2012
2014
2016
2018

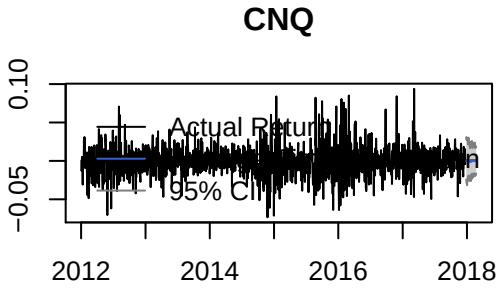

AQN
0.05
−0.10
95% CI
2012
2014
2016
2018

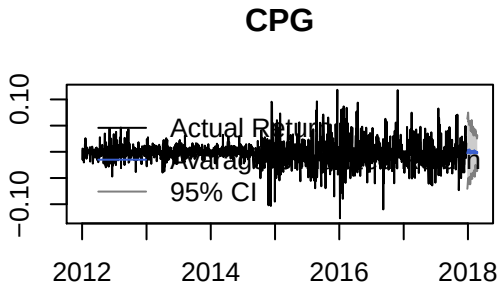

ATP
0.0
−0.4
Avarage Simulated Return
95% CI
2012
2014
2016
2018

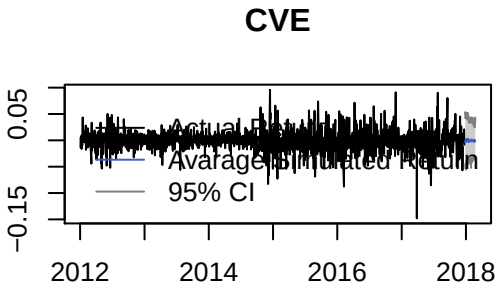

BEP.UN
0.04
−0.06
Actual Return
95% CI
2012
2014
2016
2018

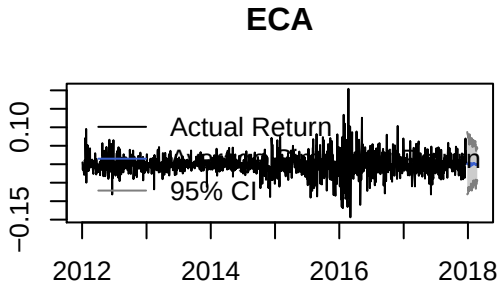

BTE
0.1
−0.2
Actual Return
95% CI
2012
2014
2016
2018

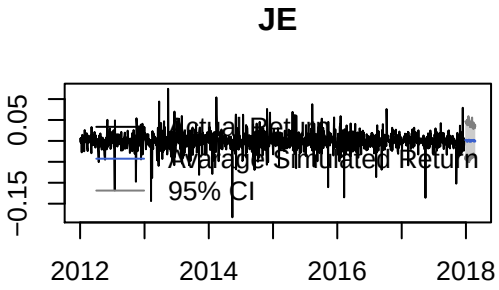

BXE
0.5
−1.5
Actual Return
95% CI
2012
2014
2016
2018

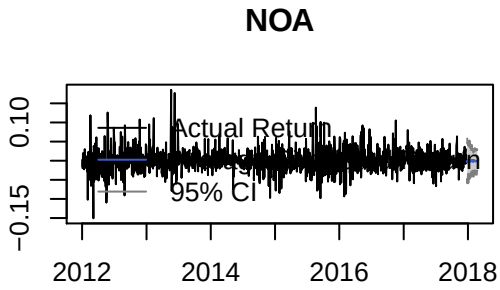

CNQ
0.10
−0.05
Actual Return
95% CI
2012
2014
2016
2018

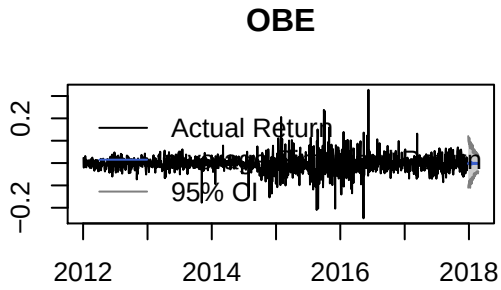

CPG
0.10
−0.10
Actual Return
95% CI
2012
2014
2016
2018

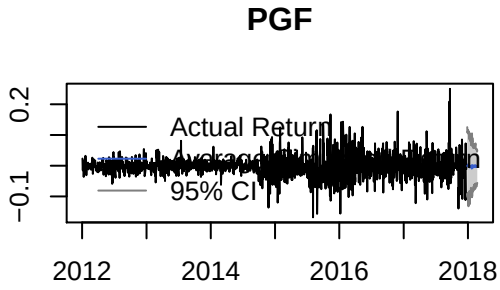

CVE
0.05
−0.15
95% CI
2012
2014
2016
2018

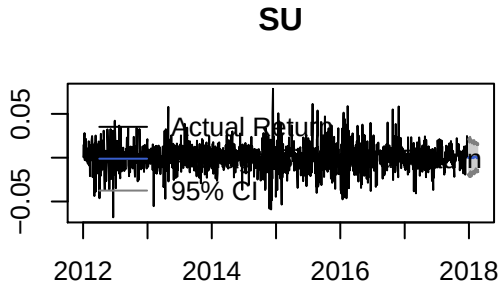

ECA
0.10
−0.15
Actual Return
95% CI
2012
2014
2016
2018

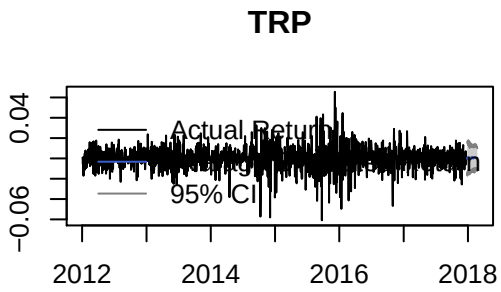

JE
0.05
−0.15
95% CI
2012
2014
2016
2018

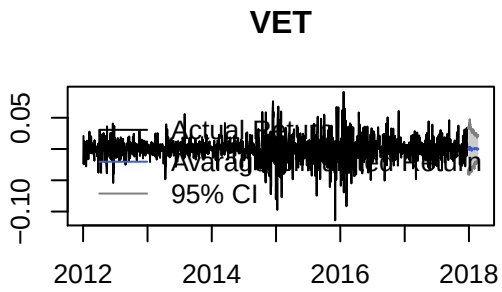

NOA
Actual Return
95% CI
0.10
−0.15
2012
2014
2016
2018

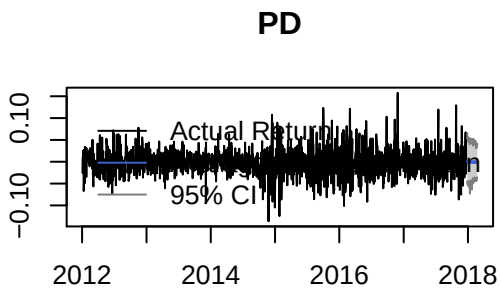

OBE
Actual Return
95% CI
0.2
−0.2
2012
2014
2016
2018

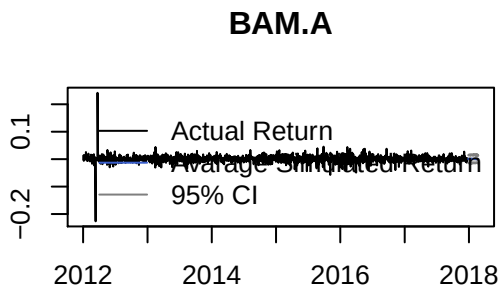

PGF
Actual Return
95% CI
0.2
−0.1
2012
2014
2016
2018

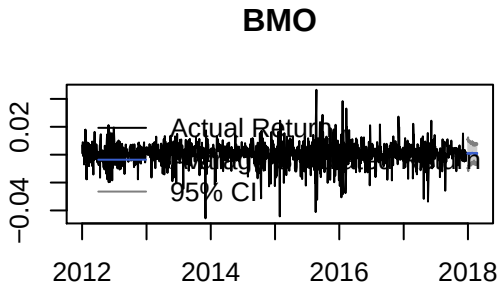

SU
Actual Return
95% CI
0.05
−0.05
2012
2014
2016
2018

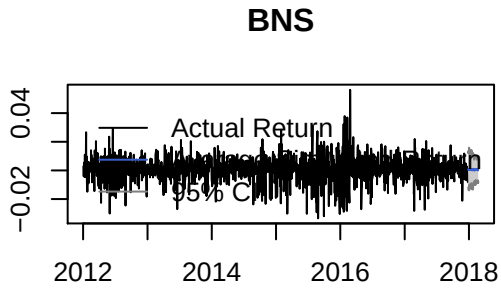

TRP
0.04
−0.06
Actual Return
95% CI
2012
2014
2016
2018

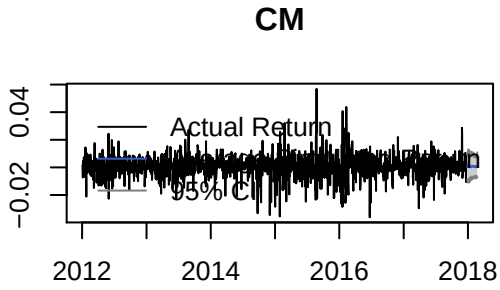

VET
0.05
−0.10
Actual Return
95% CI
2012
2014
2016
2018

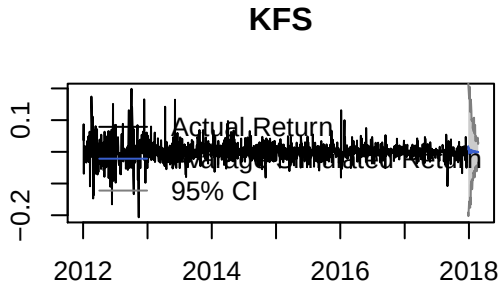

PD
0.10
−0.10
Actual Return
95% CI
2012
2014
2016
2018

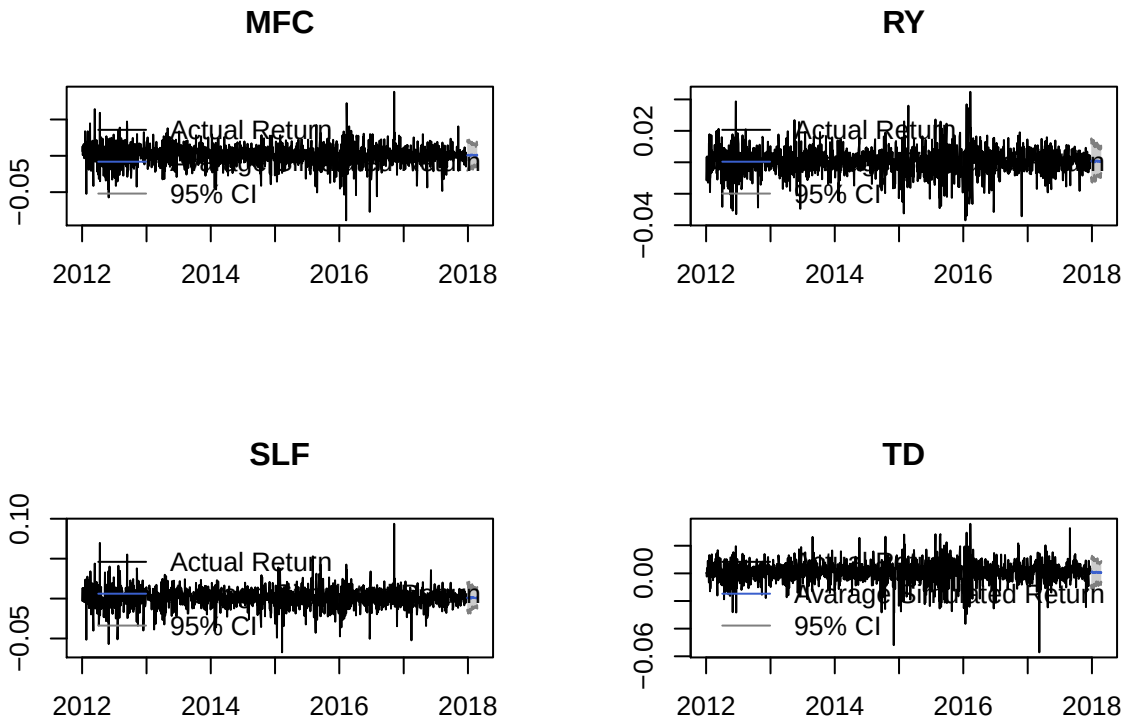

BAM.A
0.1
−0.2
Actual Return
95% CI
2012
2014
2016
2018

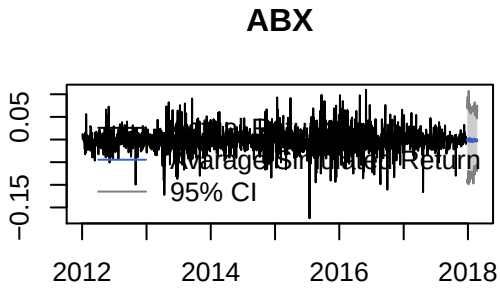

BMO
Actual Return
95% CI
0.02
−0.04
2012
2014
2016
2018

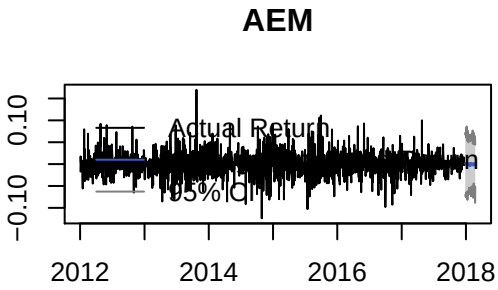

BNS
Actual Return
95% CI
0.04
−0.02
2012
2014
2016
2018

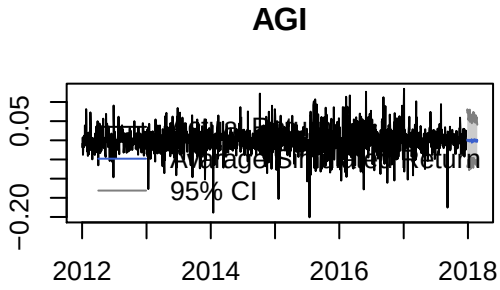

CM
Actual Return
95% CI
0.04
−0.02
2012
2014
2016
2018

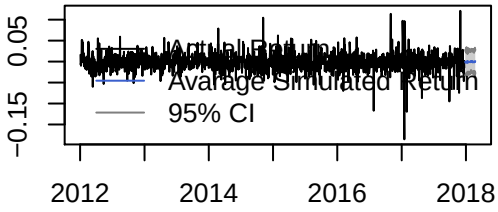

KFS
Actual Return
95% CI
0.1
−0.2
2012
2014
2016
2018

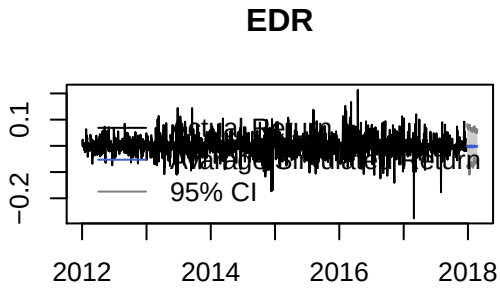

MFC
RY
SLF
TD
Actual Return
95% CI
−0.05
0.02
−0.04
0.10
0.00
−0.06
2012
2014
2016
2018

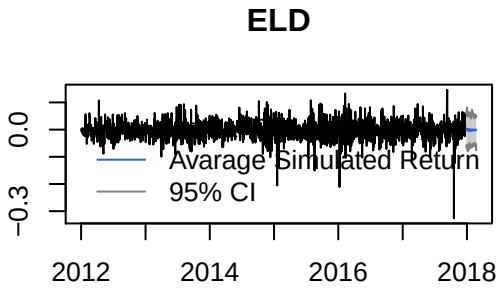

ABX
0.05
−0.15
95% CI
2012
2014
2016
2018

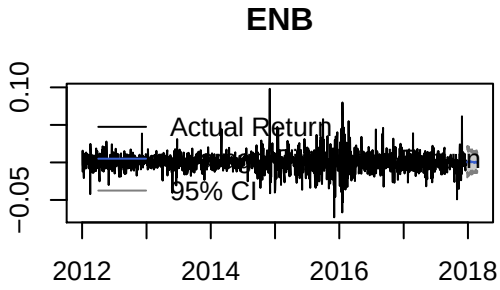

AEM
0.10
−0.10
Actual Return
95% CI
2012
2014
2016
2018

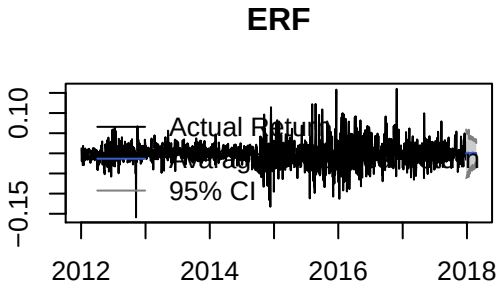

AGI
0.05
−0.20
95% CI
2012
2014
2016
2018

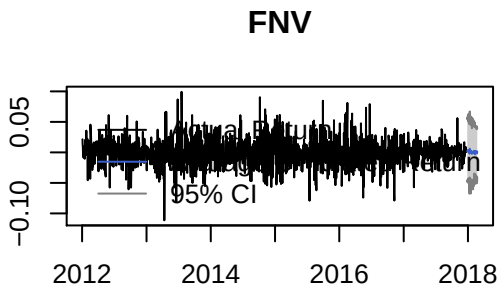

CCO
0.05
−0.15
Avarage Simulated Return
95% CI
2012
2014
2016
2018

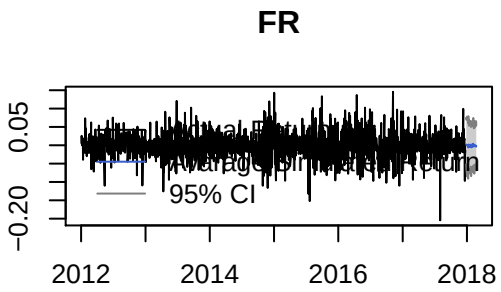

EDR
0.1
−0.2
Actual Return
Avarage Simulated Return
95% CI
2012
2014
2016
2018

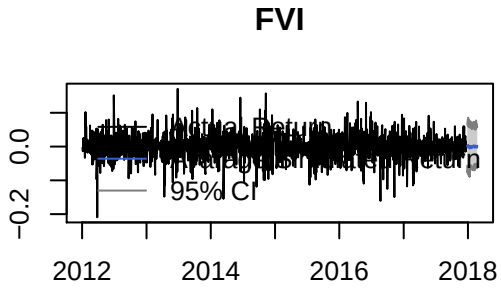

ELD
0.0
−0.3
Avarage Simulated Return
95% CI
2012
2014
2016
2018

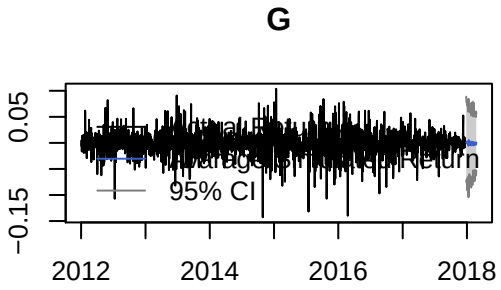

ENB
0.10
−0.05
Actual Return
95% CI
2012
2014
2016
2018

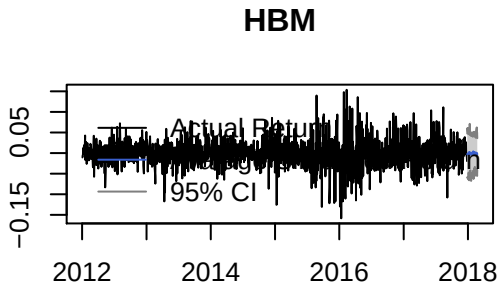

ERF
0.10
−0.15
95% CI
2012
2014
2016
2018

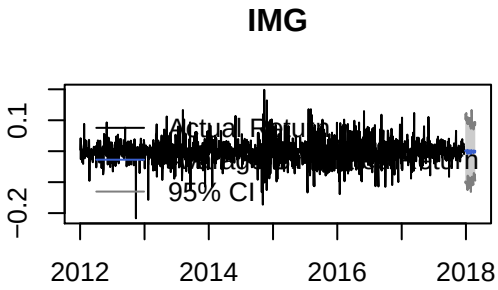

FNV
0.05
−0.10
95% CI
2012
2014
2016
2018

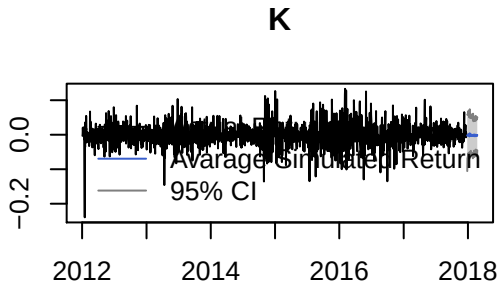

FR
0.05
−0.20
95% CI
2012
2014
2016
2018

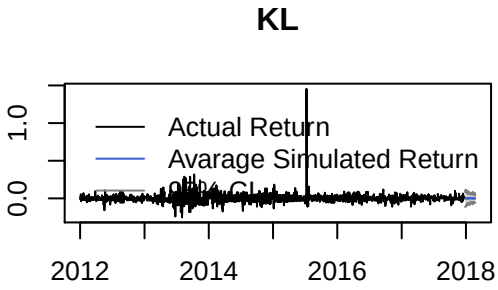

FVI
0.0
−0.2
95% CI
2012
2014
2016
2018

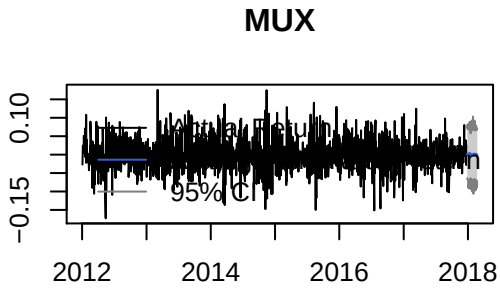

G
0.05
−0.15
95% CI
2012
2014
2016
2018

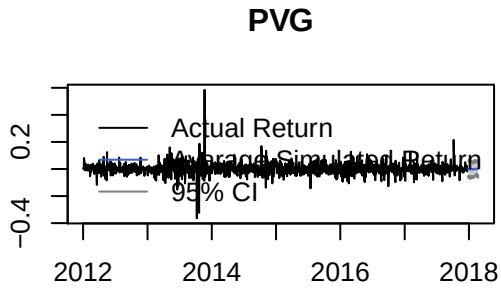

HBM
0.05
−0.15
95% CI
2012
2014
2016
2018

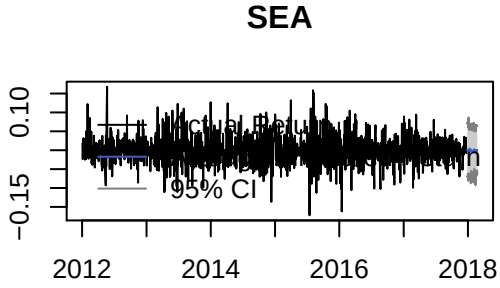

IMG
0.1
−0.2
95% CI
2012
2014
2016
2018

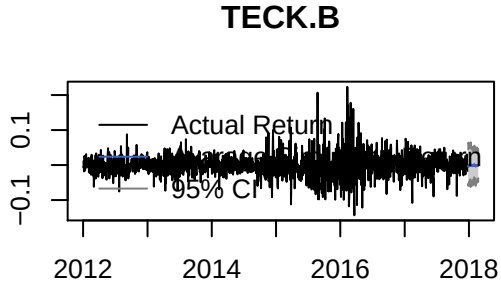

K
0.0
−0.2
95% CI
2012
2014
2016
2018

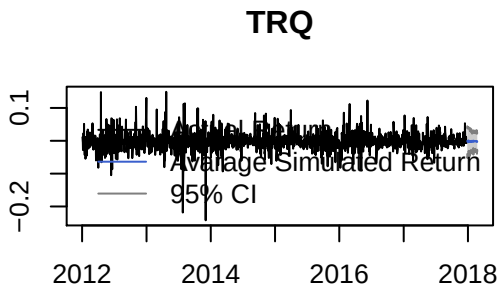

KL
1.0
0.0
Actual Return
Avarage Simulated Return
95% CI
2012
2014
2016
2018

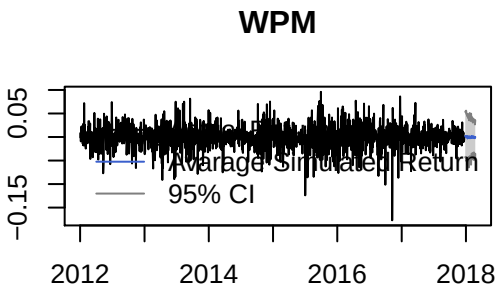

MUX
0.10
−0.15
95% CI
2012
2014
2016
2018

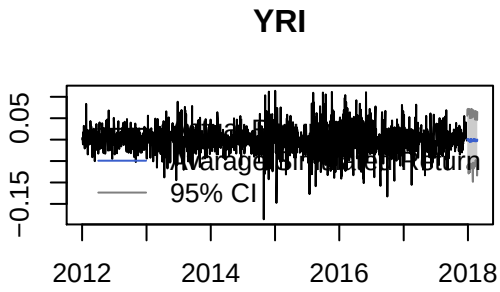

PVG
0.2
−0.4
Actual Return
95% CI
2012
2014
2016
2018

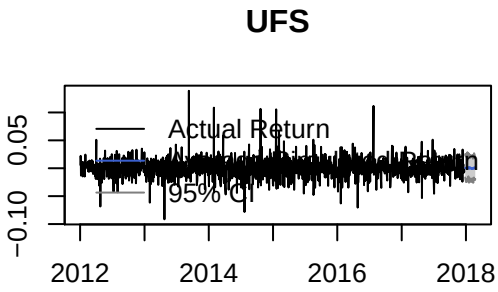

SEA
0.10
−0.15
95% CI
2012
2014
2016
2018

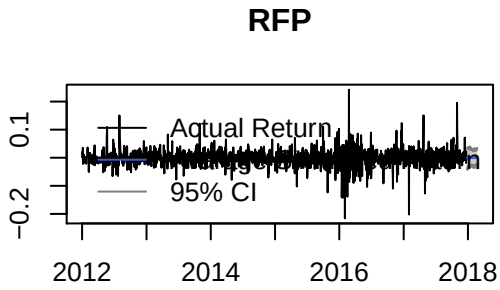

TECK.B
0.1
−0.1
Actual Return
95% CI
2012
2014
2016
2018

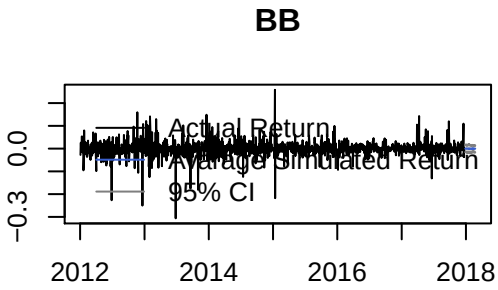

TRQ
0.1
−0.2
Average Simulated Return
95% CI
2012
2014
2016
2018

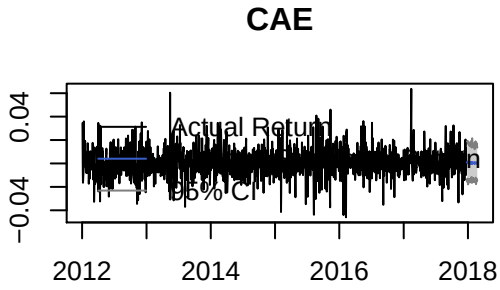

WPM
0.05
−0.15
Average Simulated Return
95% CI
2012
2014
2016
2018

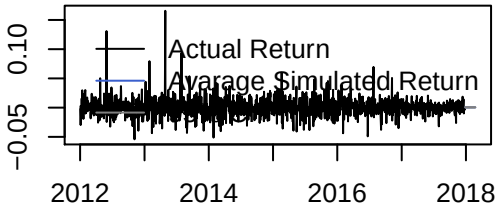

YRI
0.05
−0.15
95% CI
2012
2014
2016
2018

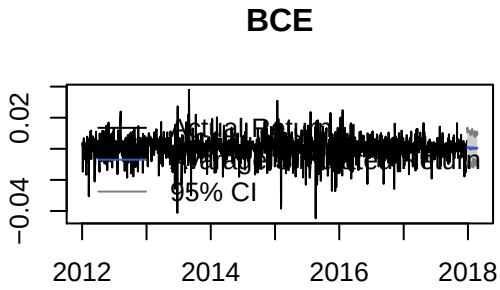

UFS
Actual Return
0.05
−0.10
95% CI
2012
2014
2016
2018

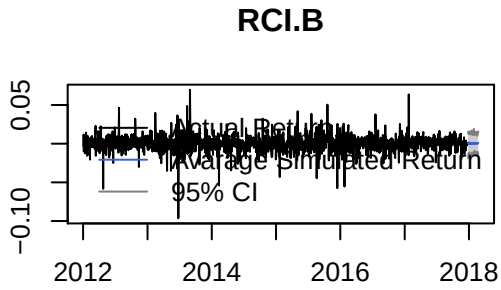

RFP
Actual Return
95% CI
0.1
−0.2
2012
2014
2016
2018

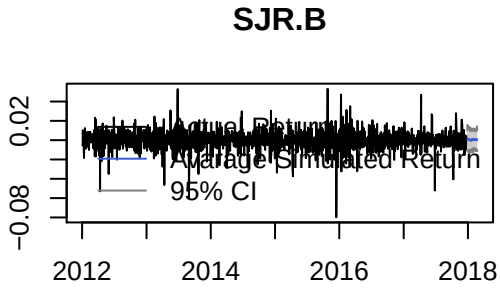

BB
Actual Return
95% CI
0.0
−0.3
2012
2014
2016
2018

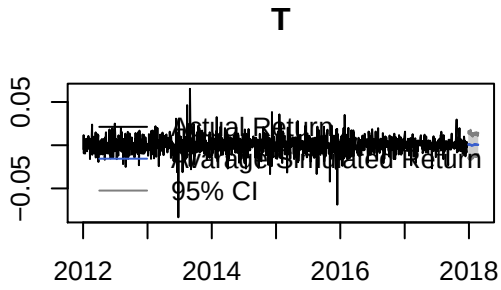

CAE
Actual Return
95% CI
0.04
−0.04
2012
2014
2016
2018

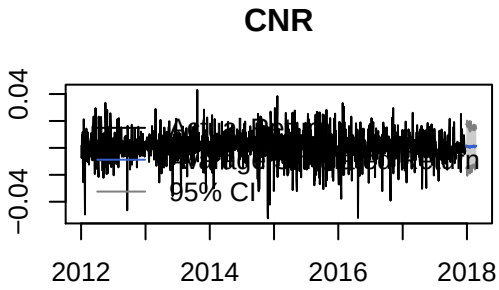

GIBA
Actual Return
0.10
−0.05
2012
2014
2016
2018

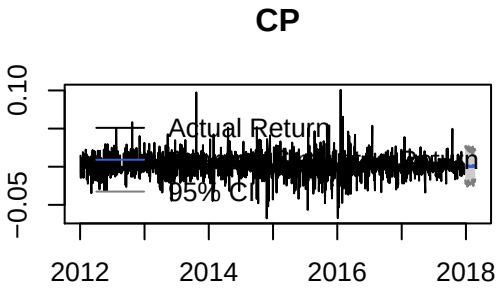

BCE
0.02
−0.04
Actual Return
Average Simulated Return
95% CI
2012
2014
2016
2018

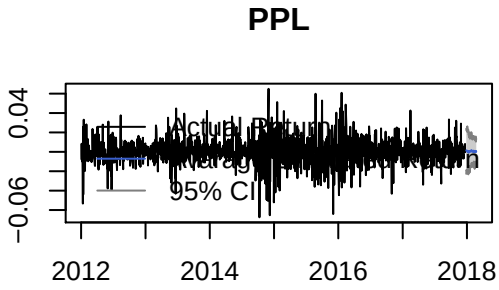

RCI.B
0.05
−0.10
Actual Return
Average Simulated Return
95% CI
2012
2014
2016
2018

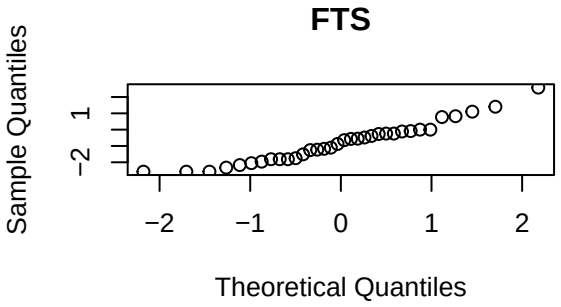

SJR.B
0.02
−0.08
Actual Return
Average Simulated Return
95% CI
2012
2014
2016
2018

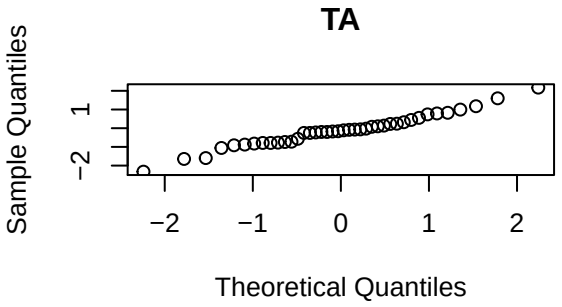

T
0.05
−0.05
Actual Return
Average Simulated Return
95% CI
2012
2014
2016
2018

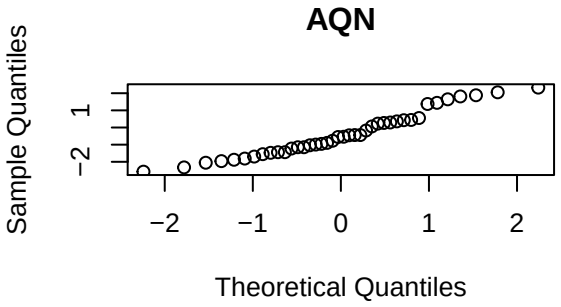

CNR
0.04
−0.04
Actual Return
95% CI
2012
2014
2016
2018

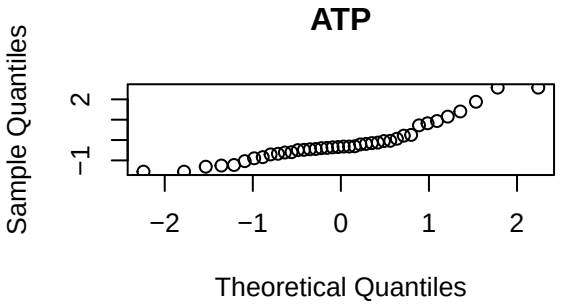

CP
0.10
−0.05
Actual Return
95% CI
2012
2014
2016
2018

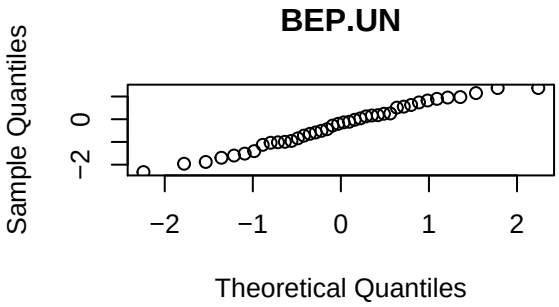

PPL
0.04
−0.06
Actual Return
95% CI
2012
2014
2016
2018

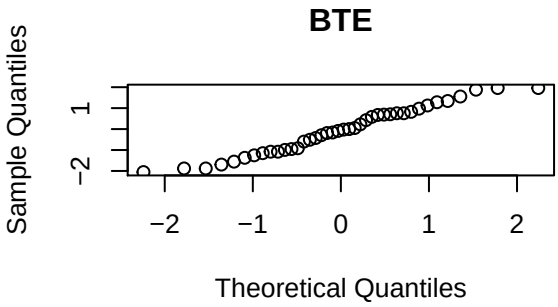

FTS
Sample Quantiles
1
−2
−2
−1
0
1
2
Theoretical Quantiles

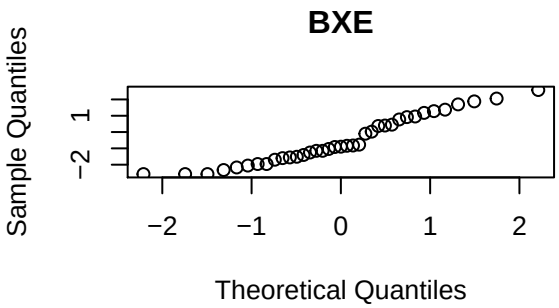

TA
Sample Quantiles
1
−2
−2
−1
0
1
2
Theoretical Quantiles

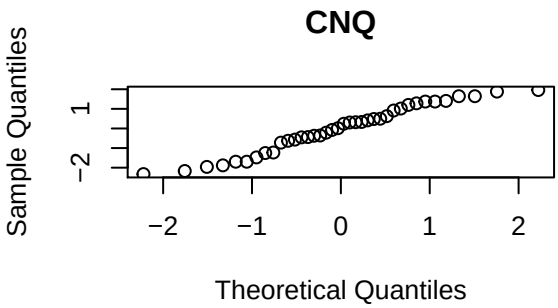

AQN
Sample Quantiles
1
−2
−2
−1
0
1
2
Theoretical Quantiles

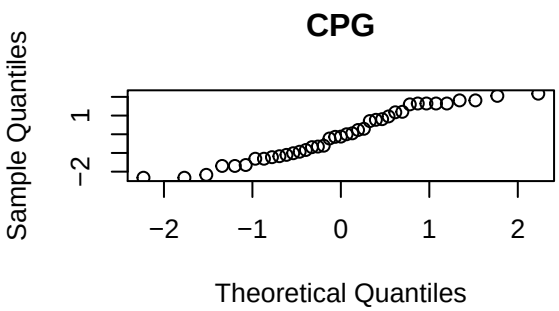

ATP
Sample Quantiles
2
−1
−2
−1
0
1
2
Theoretical Quantiles

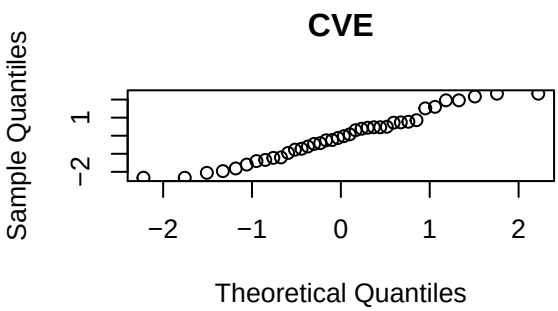

BEP.UN
Sample Quantiles
Theoretical Quantiles

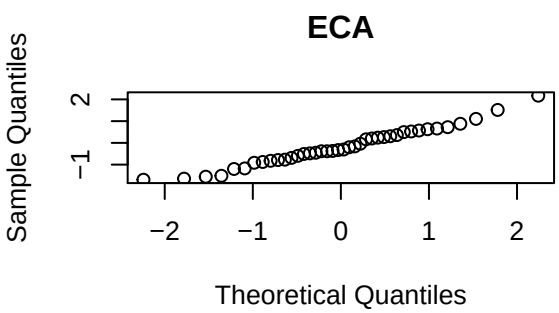

BTE
Sample Quantiles
Theoretical Quantiles

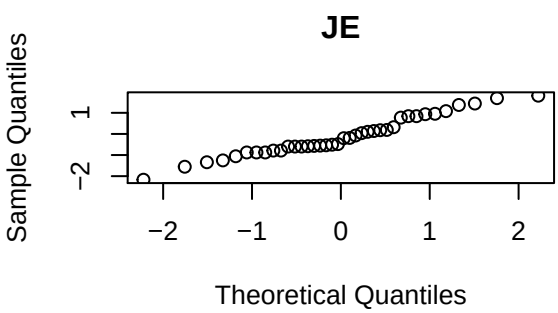

BXE
Sample Quantiles
Theoretical Quantiles

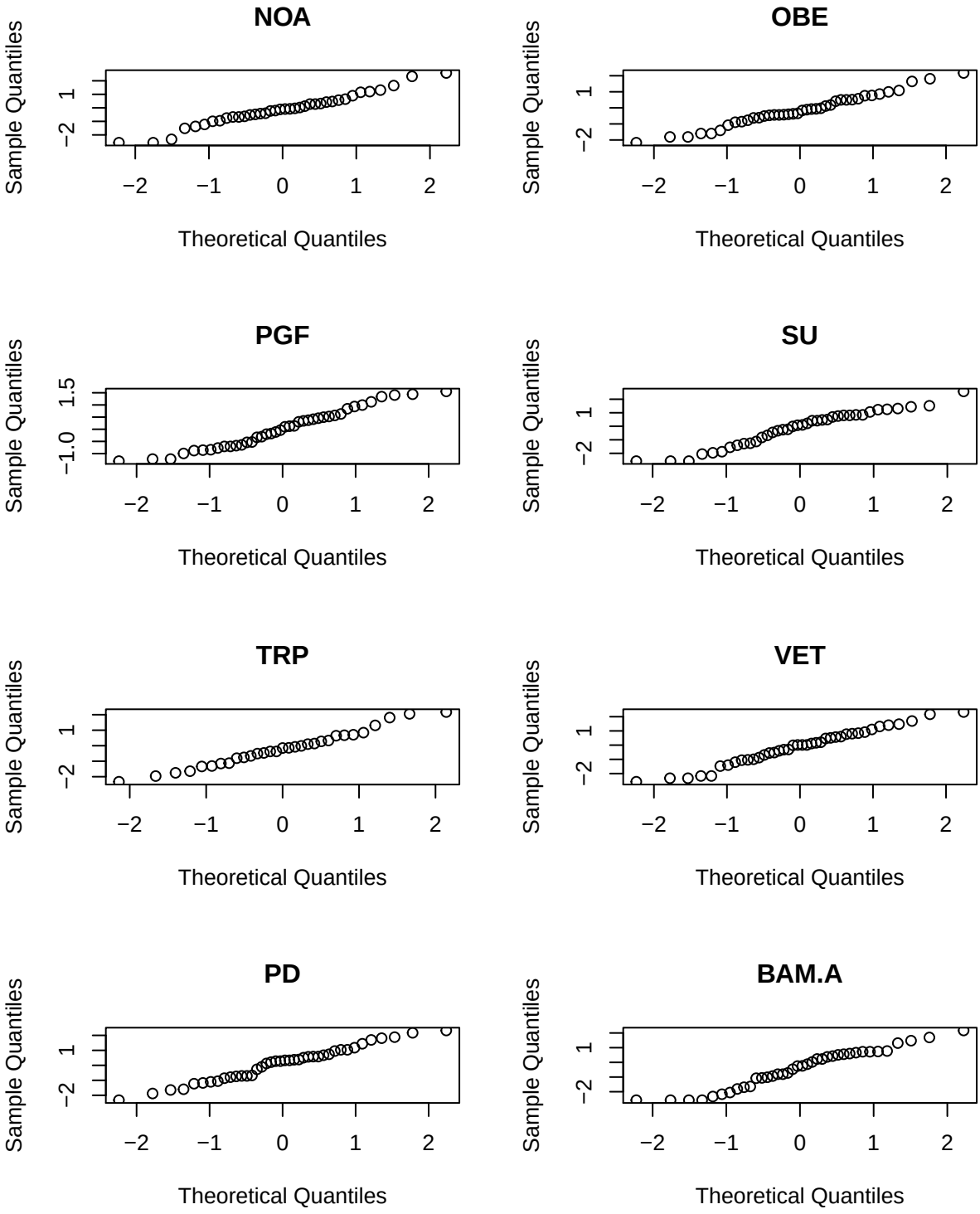

CNQ
Sample Quantiles
Theoretical Quantiles

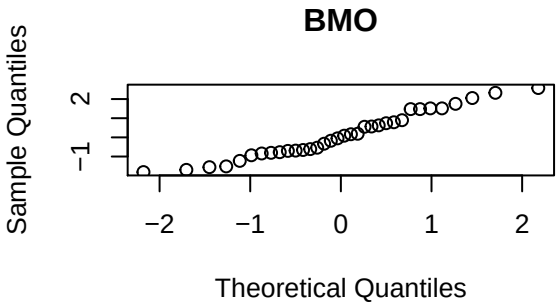

CPG
Sample Quantiles
Theoretical Quantiles

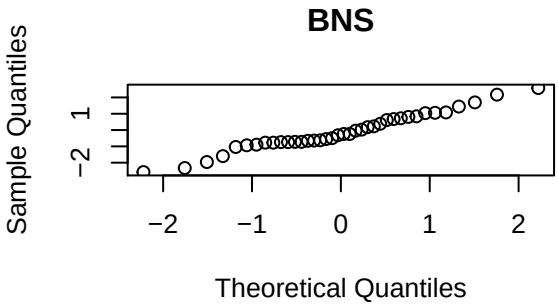

CVE
Sample Quantiles
Theoretical Quantiles

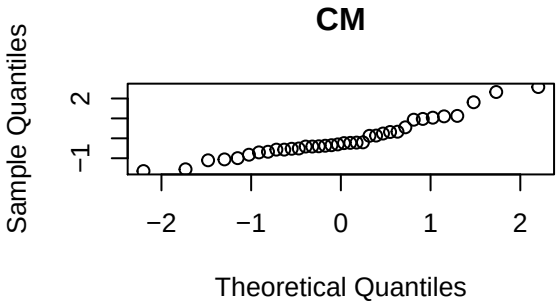

ECA
Sample Quantiles
Theoretical Quantiles

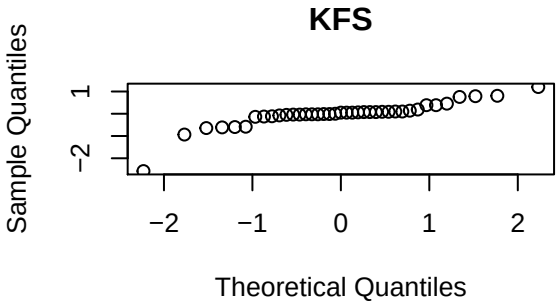

JE
Sample Quantiles
Theoretical Quantiles

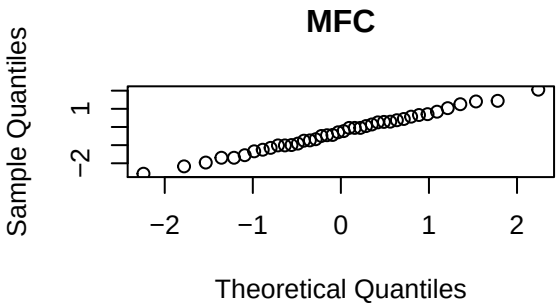

NOA
OBE
PGF
SU
TRP
VET
PD
BAM.A
Sample Quantiles
Theoretical Quantiles

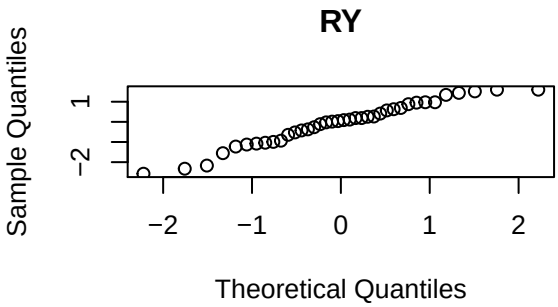

BMO
Sample Quantiles
Theoretical Quantiles

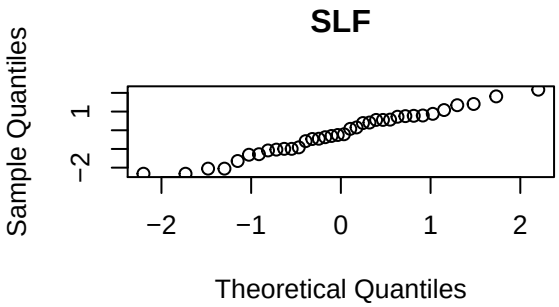

BNS
Sample Quantiles
Theoretical Quantiles

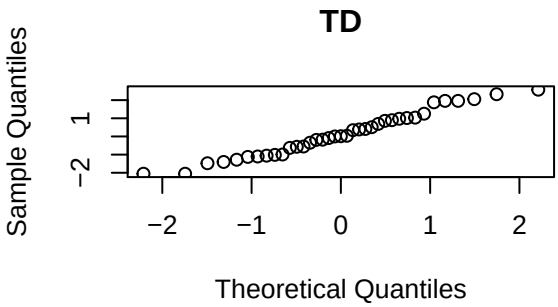

CM
Sample Quantiles
Theoretical Quantiles

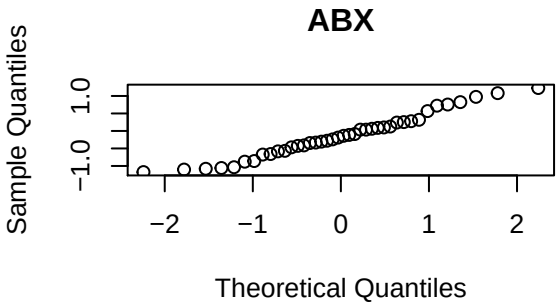

KFS
Sample Quantiles
Theoretical Quantiles

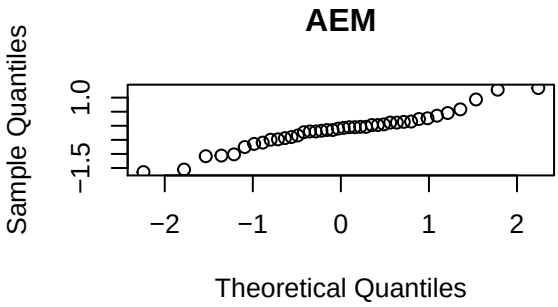

MFC
Sample Quantiles
Theoretical Quantiles

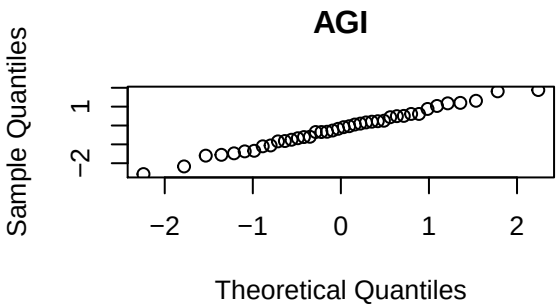

RY
Sample Quantiles
Theoretical Quantiles

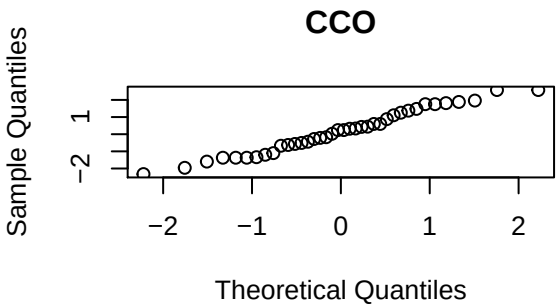

SLF
Sample Quantiles
Theoretical Quantiles

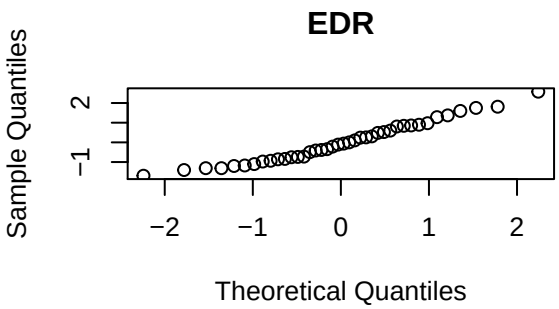

TD
Sample Quantiles
Theoretical Quantiles

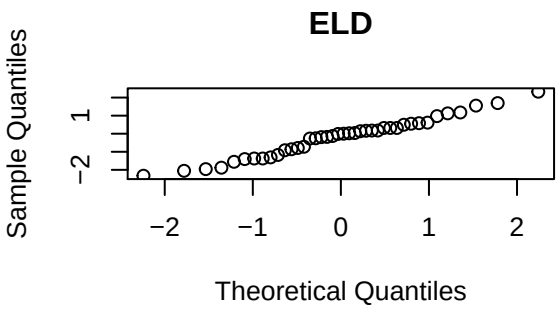

ABX
Sample Quantiles
−1.0
1.0
−2
−1
0
1
2
Theoretical Quantiles

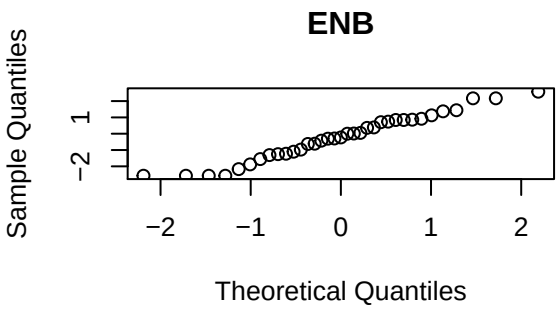

AEM
Sample Quantiles
−1.5
1.0
−2
−1
0
1
2
Theoretical Quantiles

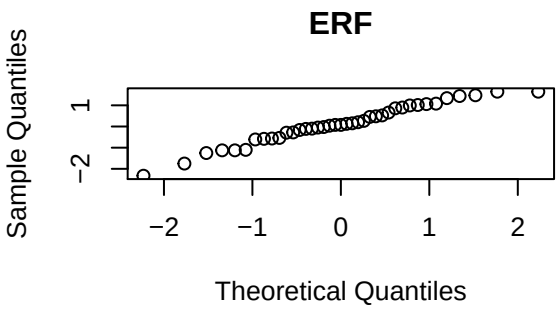

AGI
Sample Quantiles
−2
1
−2
−1
0
1
2
Theoretical Quantiles

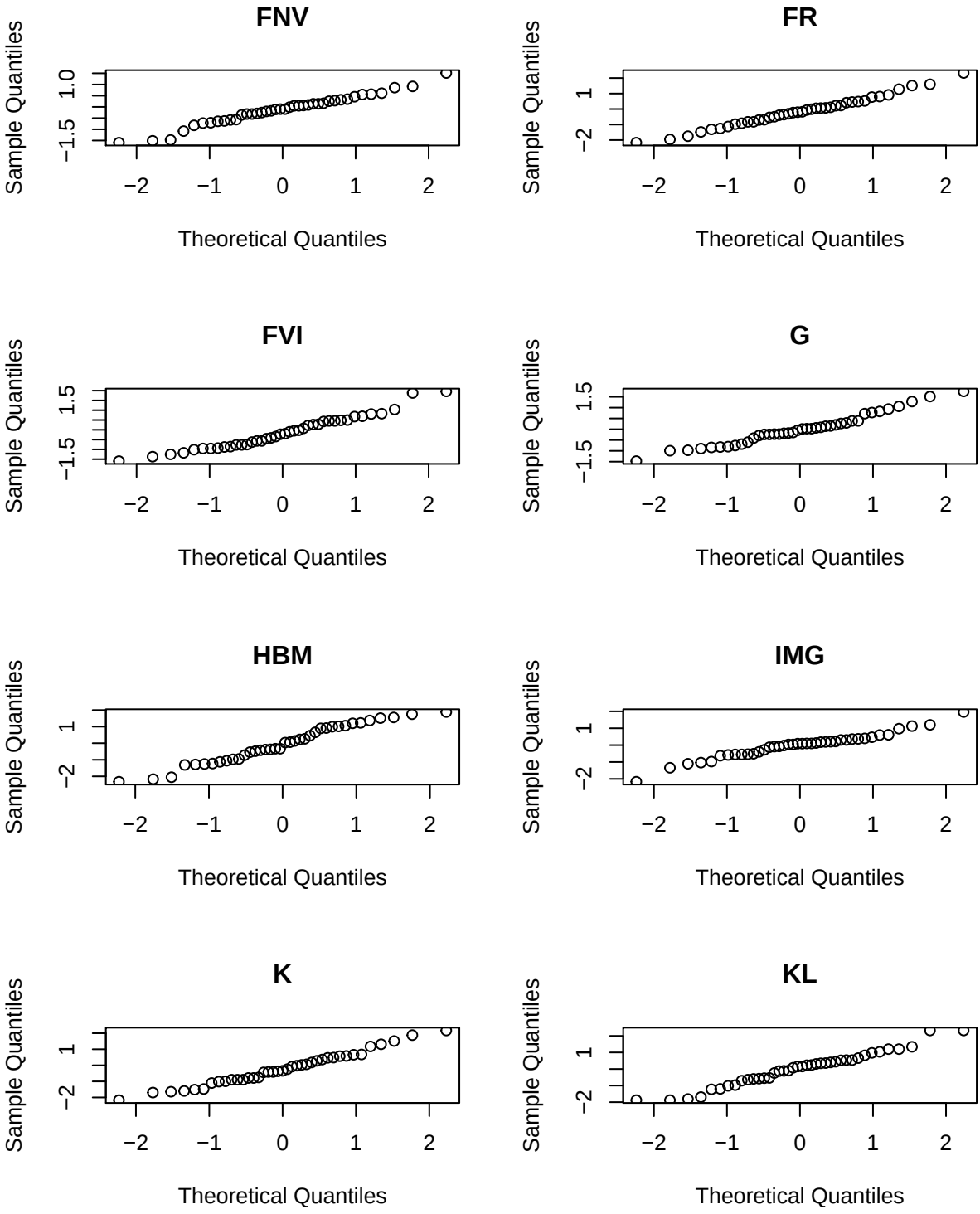

CCO
Sample Quantiles
−2
1
−2
−1
0
1
2
Theoretical Quantiles

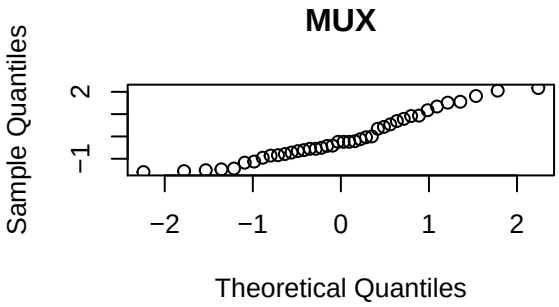

EDR
Sample Quantiles
−1
2
−2
−1
0
1
2
Theoretical Quantiles

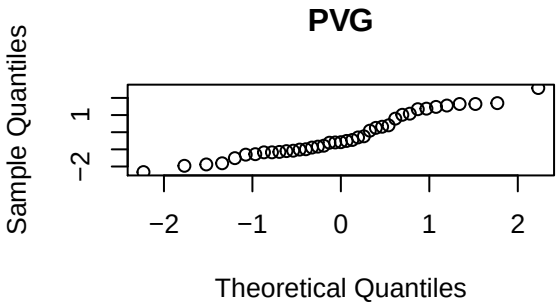

ELD
Sample Quantiles
−2
1
−2
−1
0
1
2
Theoretical Quantiles

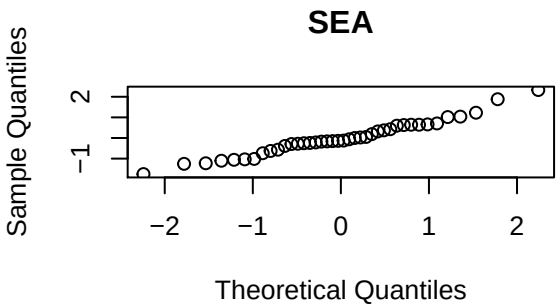

ENB
Sample Quantiles
−2
1
−2
−1
0
1
2
Theoretical Quantiles

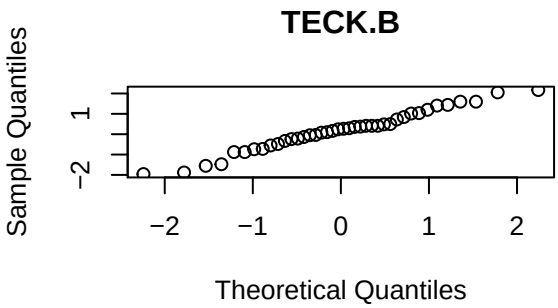

ERF
Sample Quantiles
−2
1
−2
−1
0
1
2
Theoretical Quantiles

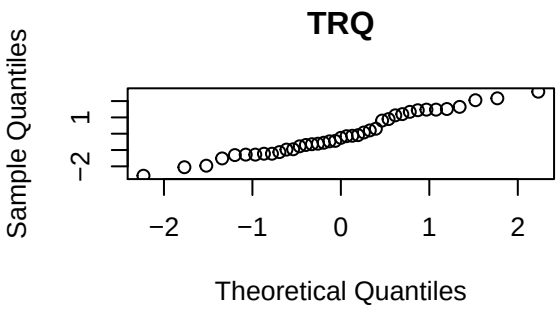

FNV
FR
FVI
G
HBM
IMG
K
KL
Sample Quantiles
Theoretical Quantiles

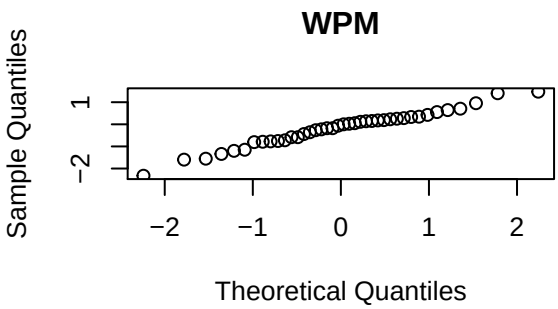

MUX
Sample Quantiles
Theoretical Quantiles

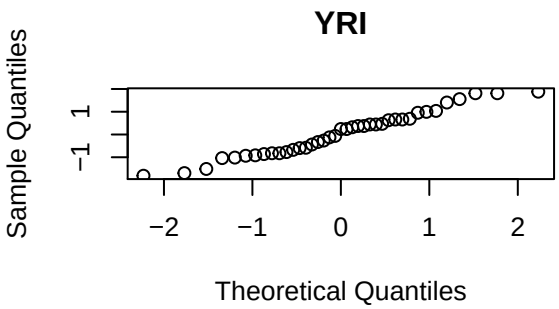

PVG
Sample Quantiles
Theoretical Quantiles

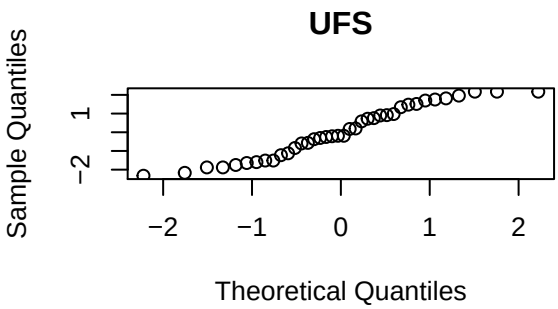

SEA
Sample Quantiles
Theoretical Quantiles

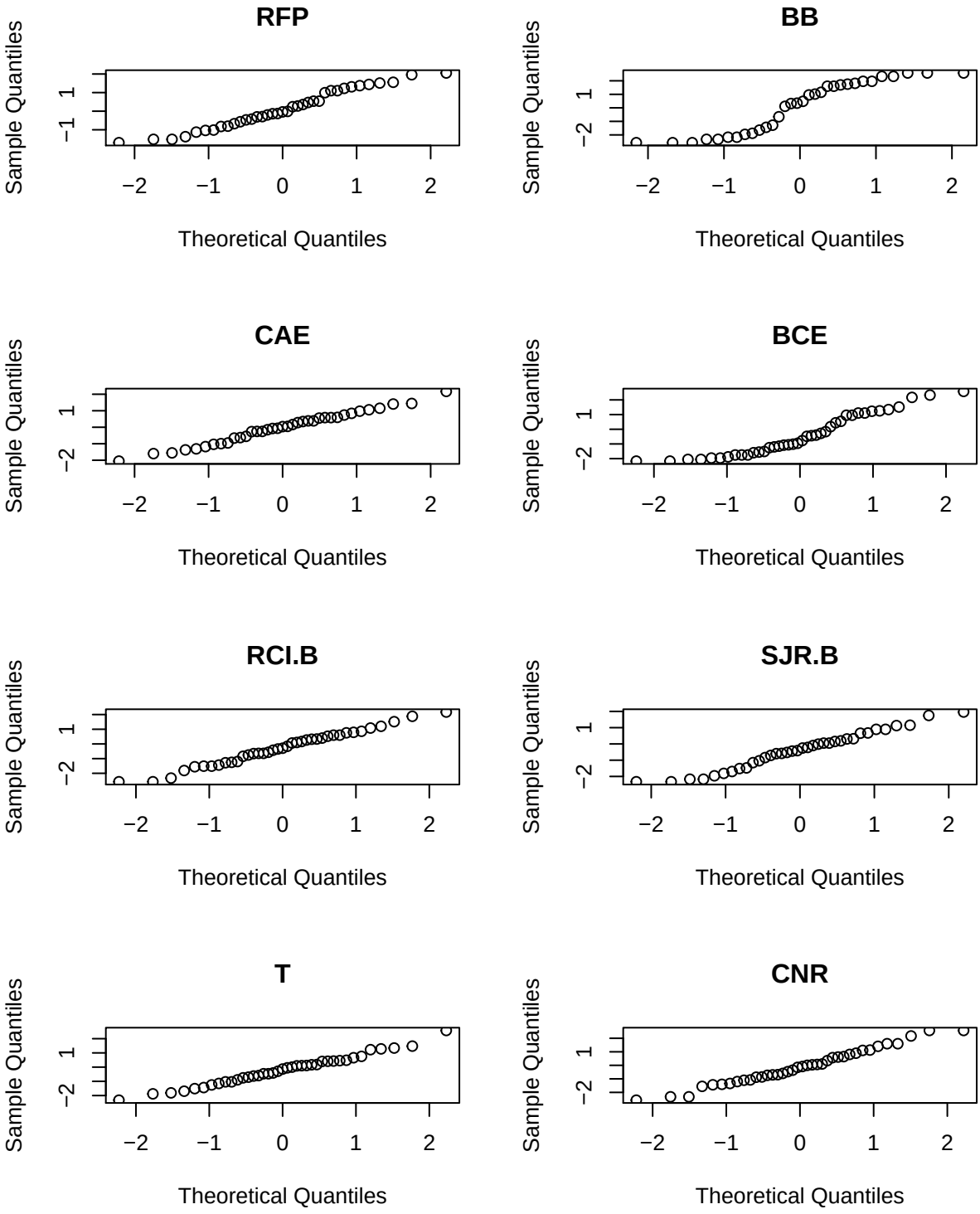

TECK.B
Sample Quantiles
Theoretical Quantiles

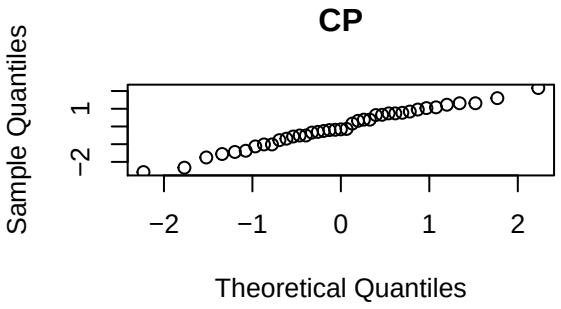

TRQ
Sample Quantiles
Theoretical Quantiles

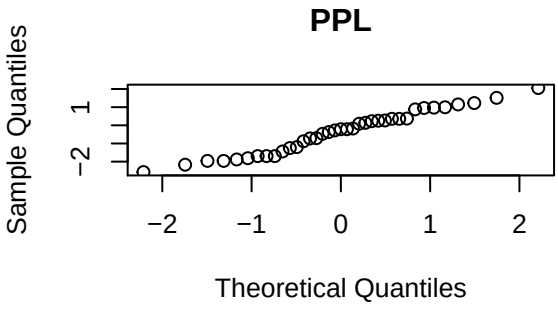

WPM
Sample Quantiles
Theoretical Quantiles


YRI
Sample Quantiles
Theoretical Quantiles


UFS
Sample Quantiles
Theoretical Quantiles


RFP
BB
CAE
BCE
RCI.B
SJR.B
T
CNR
Sample Quantiles
Theoretical Quantiles




shapiro-wilk test result:

| stock | p-val |
|---|---|
| FTS | 0.167106408589198 |
| TA | 0.593638220430627 |
| AQN | 0.107456422966493 |
| ATP | 0.000982758502608562 |
| BEP.UN | 0.536966037347769 |
| BTE | 0.299987418266897 |
| BXE | 0.0485982633379906 |
| CNQ | 0.191241714486934 |
| CPG | 0.053678636724368 |
| CVE | 0.303597147824305 |
| ECA | 0.654442676578152 |
| JE | 0.308376169232095 |
| NOA | 0.601044361106108 |
| OBE | 0.815795234672966 |
| PGF | 0.222849613591416 |
| SU | 0.174703071874289 |
| TRP | 0.806857009697121 |
| VET | 0.649033676359316 |
| PD | 0.649768400621249 |
| BAM.A | 0.124841252354708 |
| BMO | 0.266607932333032 |
| BNS | 0.5237038298593 |
| CM | 0.0258152442560216 |
| KFS | 2.90122574673174e-06 |
| MFC | 0.99874049116472 |
| RY | 0.257612087832972 |

| stock | p-val |
|---|---|
| SLF | 0.610311318105759 |
| TD | 0.463771508091249 |
| ABX | 0.39604067119644 |
| AEM | 0.128035537678782 |
| AGI | 0.986677863263102 |
| CCO | 0.521030045438506 |
| EDR | 0.284114737868201 |
| ELD | 0.607670783313093 |
| ENB | 0.384655524824397 |
| ERF | 0.444946500011169 |
| FNV | 0.609672966247125 |
| FR | 0.982528439251535 |
| FVI | 0.206790371383822 |
| G | 0.447966395714941 |
| HBM | 0.290285965106029 |
| IMG | 0.195540325277308 |
| K | 0.724991321203768 |
| KL | 0.430870439916797 |
| MUX | 0.0574839992782282 |
| PVG | 0.0479301816449936 |
| SEA | 0.436315028791969 |
| TECK.B | 0.693465941668354 |
| TRQ | 0.241181193263507 |
| WPM | 0.582314901781001 |
| YRI | 0.365043786409432 |
| UFS | 0.05345533188289 |
| RFP | 0.298872767495949 |
| BB | 0.0017928369080714 |
| CAE | 0.956533473708186 |
| BCE | 0.0058957437792927 |
| RCI.B | 0.863670552711834 |
| SJR.B | 0.49714049354545 |
| T | 0.930214608484696 |
| CNR | 0.718323812683332 |
| CP | 0.81847060428976 |
| PPL | 0.349046681380663 |

## 5.2 Analysis Code

```
load("stockprice.rda")
numNA <- apply(stockprice, 2, function(l) {
    length(which(is.na(l) == T))
})
plot(numNA, pch = 19, main = "Number of NAs in each stock", ylab = "numbers")
set.seed(2018440)
ind <- sample(1:79, 1)
colnames(stockprice)[ind]
stock <- na.omit(stockprice[, ind])
CAE <- ts(stock, end = c(2018, 4, 1), frequency = 365)
DecomSTL <- stl(CAE, "periodic")
```

```
plot(DecomSTL)
Decom <- stl(CAE, "periodic")
season <- Decom$time.series[, 1]
times <- time(CAE)
fit_season <- loess(season ~ times, span = 0.01, control = loess.control(surface = "direct"))
pred_season <- predict(fit_season, data.frame(times))
trend <- DecomSTL$time.series[, 2]
fitLinear <- lm(trend ~ times)
plot(times, trend, pch = 19, col = "red")
abline(coef(fitLinear), lwd = 2, lty = 2)
legend(2011, 20, legend = c("data", "lm"), col = c("red", "black"), lwd = c(1,
    2), lty = c(NA, 2), pch = c(19, NA))
pred_trend_lm <- predict(fitLinear, newdata = data.frame(times))
library(splines)
fitSmooth <- lm(trend ~ ns(times, df = 10))
pred_trend_smooth <- predict(fitSmooth, newdata = data.frame(x = times))
Xorder <- order(times)
plot(times, trend, pch = 19, col = "red")
lines(times[Xorder], pred_trend_smooth[Xorder], lwd = 2, lty = 2)
legend(2006, 20, legend = c("data", "Smoothing"), col = c("red", "black"), lwd = c(1,
    2), lty = c(NA, 2), pch = c(19, NA))
library(FNN)
fitKNN <- knn.reg(times, y = trend, k = 5)
Xorder <- order(times)
pred_trend_KNN <- fitKNN$pred
plot(times, trend, col = "red", pch = 19)
lines(times[Xorder], pred_trend_KNN[Xorder], lwd = 2, lty = 2)
legend(2011, 20, legend = c("data", "KNN"), col = c("red", "black"), lwd = c(1,
    2), lty = c(NA, 2), pch = c(19, NA))
library(tseries)
res <- DecomSTL$time.series[, 3]
runs.test(factor(sign(res)))
diffRes <- diff(res)
runs.test(factor(sign(diffRes)))
model <- arima(diffRes, c(1, 0, 0), include.mean = F)
set.seed(2018440)
predAR <- function(Rt1 = 0, Rt2 = 0, phi = model$coef, len = length(times),
    sd = sqrt(model$sigma2), bound = NULL) {
    R <- rep(0, len - 2)
    R <- c(Rt1, Rt2, R)
    if (is.null(bound)) {
        for (i in 3:len) {
            R[i] <- R[i - 1] + phi * (R[i - 1] - R[i - 2]) + rnorm(1, 0, sd)
        }
    } else {
        for (i in 3:len) {
            R[i] <- R[i - 1] + phi * (R[i - 1] - R[i - 2]) + rnorm(1, 0, sd)
            if (R[i] > bound[2] || R[i] < bound[1]) {
                while (R[i] > bound[2] || R[i] < bound[1]) {
                  R[i] <- R[i - 1] + phi * (R[i - 1] - R[i - 2]) + rnorm(1,
                    0, sd)
                }
            }
```

```
        }
    }
    R
}
predAR1 <- predAR(Rt1 = 0, Rt2 = 0, phi = model$coef, len = length(times), sd = sqrt(model$sigma2))
Lfun <- function(par = par, Y = Y) {
    sig <- par[1]
    phi <- par[2]
    v <- par[3]
    Yn <- Y[-1]
    Yn_1 <- Y[-length(Y)]
    n <- length(Y)
    L <- n * (log(gamma((v + 1)/2)) - log(gamma(v/2)) - 0.5 * log(v * pi)) -
        n * log(sig) - (v + 1)/2 * sum(log(1 + (Yn - phi * Yn_1)^2/(v * sig^2))) -
        (v + 1)/2 * log(1 + (1 - phi^2) * Y[1]^2/(sig^2 * v))
    L <- -L
    return(L)
}

optL <- optim(c(0.5, 0.5, 3), Lfun, Y = diffRes)
optim_proj(optL$par, fun = function(par) Lfun(par = par, Y = diffRes))

set.seed(2018440)
predLogAR <- function(Rt1 = 0, Rt2 = 0, phi = optL$par[2], len = length(times),
    df = optL$par[3], sigma = optL$par[1], bound = NULL) {
    R <- rep(0, len - 2)
    R <- c(Rt1, Rt2, R)
    if (is.null(bound)) {
        for (i in 3:len) {
            R[i] <- R[i - 1] + phi * (R[i - 1] - R[i - 2]) + sigma * rt(1, df)
        }
    } else {
        for (i in 3:len) {
            R[i] <- R[i - 1] + phi * (R[i - 1] - R[i - 2]) + sigma * rt(1, df)
            if (R[i] > bound[2] || R[i] < bound[1]) {
                while (R[i] > bound[2] || R[i] < bound[1]) {
                  R[i] <- R[i - 1] + phi * (R[i - 1] - R[i - 2]) + sigma * rt(1,
                    df)
                }
            }
        }
    }
    R
}
predLogAR1 <- predLogAR(Rt1 = 0, Rt2 = 0, phi = optL$par[2], len = length(times),
    df = optL$par[3], sigma = optL$par[1])

Rfun <- function(par = par, Y = Y) {
    phiE <- par[1]
    phiR <- par[2]
    sig <- par[3]
    L <- 0
    for (i in 2:length(Y)) {
```

```
        if (Y[i - 1] >= 0) {
            delta <- 1
        } else {
            delta <- 0
        }
        L <- L - log(sig) - (Y[i] - (delta * phiE * Y[i - 1] + (1 - delta) *
            phiR * Y[i - 1]))^2/(2 * sig^2)
    }
    L <- -L
    return(L)
}

optR <- optim(c(1, 1, 1), Rfun, Y = diffRes)
optim_proj(optR$par, fun = function(par) Rfun(par = par, Y = diffRes))
set.seed(2018440)
predRSAR <- function(Rt1 = 0, Rt2 = 0, phiE = optR$par[1], phiR = optR$par[2],
    len = length(times), sigma = optR$par[3], bound = NULL) {
    R <- rep(0, len - 2)
    R <- c(Rt1, Rt2, R)
    if (is.null(bound)) {
        for (i in 3:len) {
            if (R[i - 1] - R[i - 2] >= 0) {
                delta <- 1
            } else {
                delta <- 0
            }
            R[i] <- R[i - 1] + delta * phiE * (R[i - 1] - R[i - 2]) + (1 - delta) *
                phiR * (R[i - 1] - R[i - 2]) + rnorm(1, 0, sigma)
        }
    } else {
        for (i in 3:len) {
            if (R[i - 1] - R[i - 2] >= 0) {
                delta <- 1
            } else {
                delta <- 0
            }
            R[i] <- R[i - 1] + delta * phiE * (R[i - 1] - R[i - 2]) + (1 - delta) *
                phiR * (R[i - 1] - R[i - 2]) + rnorm(1, 0, sigma)
            if (R[i] > bound[2] || R[i] < bound[1]) {
                while (R[i] > bound[2] || R[i] < bound[1]) {
                  if (R[i - 1] - R[i - 2] >= 0) {
                    delta <- 1
                  } else {
                    delta <- 0
                  }
                  R[i] <- R[i - 1] + delta * phiE * (R[i - 1] - R[i - 2]) +
                    (1 - delta) * phiR * (R[i - 1] - R[i - 2]) + rnorm(1, 0,
                    sigma)
                }
            }
        }
    }
    R
```

```
}
predRSAR1 <- predRSAR(Rt1 = 0, Rt2 = 0, phiE = optR$par[1], phiR = optR$par[2],
    len = length(times), sigma = optR$par[3])


lmAR <- pred_season + pred_trend_lm + predAR1
s1 <- sum((CAE - lmAR)^2)/sum((CAE - mean(CAE))^2)
NaturalAR <- pred_season + pred_trend_smooth + predAR1
s2 <- sum((CAE - NaturalAR)^2)/sum((CAE - mean(CAE))^2)
KNNAR <- pred_season + pred_trend_KNN + predAR1
s3 <- sum((CAE - KNNAR)^2)/sum((CAE - mean(CAE))^2)
lmLog <- pred_season + pred_trend_lm + predLogAR1
s4 <- sum((CAE - lmLog)^2)/sum((CAE - mean(CAE))^2)
NaturalLog <- pred_season + pred_trend_smooth + predLogAR1
s5 <- sum((CAE - NaturalLog)^2)/sum((CAE - mean(CAE))^2)
KNNLog <- pred_season + pred_trend_KNN + predLogAR1
s6 <- sum((CAE - KNNLog)^2)/sum((CAE - mean(CAE))^2)
lmRS <- pred_season + pred_trend_lm + predRSAR1
s7 <- sum((CAE - lmRS)^2)/sum((CAE - mean(CAE))^2)
NaturalRS <- pred_season + pred_trend_smooth + predRSAR1
s8 <- sum((CAE - NaturalRS)^2)/sum((CAE - mean(CAE))^2)
KNNRS <- pred_season + pred_trend_KNN + predRSAR1
s9 <- sum((CAE - KNNRS)^2)/sum((CAE - mean(CAE))^2)

plot(CAE, col = "red", type = "l", ylim = extendrange(c(CAE, KNNAR)))
lines(times[Xorder], KNNAR[Xorder])

predAR1 <- predAR(Rt1 = 0, Rt2 = 0, phi = model$coef, len = length(times), sd = sqrt(model$sigma2),
    bound = c(-3, 3))
predLogAR1 <- predLogAR(Rt1 = 0, Rt2 = 0, phi = optL$par[2], len = length(times),
    df = optL$par[3], sigma = optL$par[1], bound = c(-3, 3))
predRSAR1 <- predRSAR(Rt1 = 0, Rt2 = 0, phiE = optR$par[1], phiR = optR$par[2],
    len = length(times), sigma = optR$par[3], bound = c(-3, 3))
lmAR <- pred_season + pred_trend_lm + predAR1
s1 <- sum((CAE - lmAR)^2)/sum((CAE - mean(CAE))^2)
NaturalAR <- pred_season + pred_trend_smooth + predAR1
s2 <- sum((CAE - NaturalAR)^2)/sum((CAE - mean(CAE))^2)
KNNAR <- pred_season + pred_trend_KNN + predAR1
s3 <- sum((CAE - KNNAR)^2)/sum((CAE - mean(CAE))^2)
lmLog <- pred_season + pred_trend_lm + predLogAR1
s4 <- sum((CAE - lmLog)^2)/sum((CAE - mean(CAE))^2)
NaturalLog <- pred_season + pred_trend_smooth + predLogAR1
s5 <- sum((CAE - NaturalLog)^2)/sum((CAE - mean(CAE))^2)
KNNLog <- pred_season + pred_trend_KNN + predLogAR1
s6 <- sum((CAE - KNNLog)^2)/sum((CAE - mean(CAE))^2)
lmRS <- pred_season + pred_trend_lm + predRSAR1
s7 <- sum((CAE - lmRS)^2)/sum((CAE - mean(CAE))^2)
NaturalRS <- pred_season + pred_trend_smooth + predRSAR1
s8 <- sum((CAE - NaturalRS)^2)/sum((CAE - mean(CAE))^2)
KNNRS <- pred_season + pred_trend_KNN + predRSAR1
s9 <- sum((CAE - KNNRS)^2)/sum((CAE - mean(CAE))^2)
# hard margin
lm0 <- pred_season + pred_trend_lm
```

```r
s10 <- sum((CAE - lm0)^2)/sum((CAE - mean(CAE))^2)
Natural0 <- pred_season + pred_trend_smooth
s11 <- sum((CAE - Natural0)^2)/sum((CAE - mean(CAE))^2)
KNN0 <- pred_season + pred_trend_KNN
s12 <- sum((CAE - KNN0)^2)/sum((CAE - mean(CAE))^2)

plot(CAE, col = "red", type = "l", ylim = extendrange(c(CAE, KNNLog)))
lines(times[Xorder], KNNAR[Xorder])
lines(times[Xorder], KNN0[Xorder], col = "blue")
legend(2006, 25, legend = c("data", "KNN bound", "KNN hard margin"), col = c("red",
    "black", "blue"), lty = c(1, 1, 1))
```

```r
library(group88)
library(xts)
library(optimCheck)
load("pricecopy.Rda")

time <- c("2012-01-03", "2017-12-31")
S <- pricecopy[paste0(time, collapse = "/"), ]  # data
S <- na.omit(S)

Y <- getreturns(S, "log")
T <- nrow(Y)
D <- ncol(Y)

garch_basic <- ugarchspec(variance.model = list(model = "sGARCH", garchOrder = c(1,
    1)), mean.model = list(armaOrder = c(0, 0)), distribution.model = "std")

fit.garch <- lapply(Y, function(x) ugarchfit(garch_basic, x))

eps <- as.matrix(do.call(merge, lapply(fit.garch, residuals)))
sigma2 <- as.matrix(do.call(merge, lapply(fit.garch, sigma)))^2


# ----------------------MLE checck rugarch
# package-------------------------------------------
objfun <- function(x) {
    garch.loglik(x[1], x[2], x[3], x[4], eps, sigma2)
}

for (i in 1:D) {
    Xfit <- fit.garch[[i]]@fit$coef[c("omega", "alpha1", "beta1", "shape")]
    oproj <- optim_proj(fun = objfun, xsol = Xfit, maximize = FALSE, xrng = 0.5)

}


#-----------------------test residual distribution-----------------------------------------
X <- as.matrix(do.call(merge, lapply(fit.garch, residuals, standardize = TRUE)))
par(mfrow = c(4, 5))

for (i in 1:D) {
    hist(X[, i], probability = TRUE, main = sprintf("Residual Dist of stock %d",
        i))
```

```
    lines(seq(-5, 5, length = 100), dnorm(seq(-5, 5, length = 100)))
}


U <- sapply(1:D, function(i) U <- pt(X[, i], df = fit.garch[[i]]@fit$coef["shape"]))

for (i in 1:D) {
    hist(U[, i], probability = TRUE, main = sprintf("Residual Dist of stock %d",
        i))
    lines(seq(-1, 1, length = 100), dunif(seq(-1, 1, length = 100)))
}



Z <- qnorm(U)

for (i in 1:D) {
    hist(Z[, i], probability = TRUE, main = sprintf("Trans Residual Dist %d",
        i))
    lines(seq(-5, 5, length = 100), dnorm(seq(-5, 5, length = 100)))
}

par(mfrow = c(1, 1))
```

```
#------------------------sector level hierarchical modelling ----------------------
group <- c(rep(1, 2), rep(2, 17), rep(3, 9), rep(4, 23), rep(5, 2), rep(6, 3),
    rep(7, 4), rep(8, 3))

T <- nrow(Y)
D <- ncol(Y)


S.sector <- sapply(1:length(unique(group)), function(i) {
    groupi <- group == i
    Si <- Z[, groupi]
    rowMeans(Si)
})

row.names(S.sector) <- as.character(row.names(X))


sigma0 <- var(S.sector)
Psi0 <- lapply(1:8, function(x) {
    Zk <- Z[, group == x]
    cor(Zk)
})


L.cur <- -Inf
L.prev <- 0
while (abs(L.cur - L.prev) > 0.1) {
    L.prev <- L.cur

    S.cur <- S.expect(sigma0, Psi0, Z, group)
```

```
    A <- hier.A(S.cur)
    B <- hier.B(S.cur, Z, group)

    sigma0 <- A/T
    Psi0 <- lapply(B, function(x) x/T)

    L.cur <- hier.logLik(S.cur, Z, sigma0, Psi0, group)
}

Vfull <- hier.var(sigma0, Psi0, group)

Var.sector <- lapply(1:length(unique(group)), function(i) {
    ind <- rep(FALSE, ncol(Vfull))
    ind[1:length(group)] <- group == i
    Vfull[ind, ind]

})


#----------------Test mdodel performance----------------------------------------
B <- 200
m <- 40

stock.name <- colnames(S)

Y.sim <- lapply(1:B, function(b) {
    sim <- Garchgc.sim(m, fit.garch, Var.sector, D)
    colnames(sim) <- colnames(Y)
    sim
})

par(mfrow = c(2, 2))

# The visualization code is based on sample code from qrm tutorial
tm <- index(pricecopy)
start <- match(time[1], as.character(tm))
past <- tm[start:(start + T - 1)]
future <- tm[(start + T):(start + T + m - 1)]

for (i in 1:D) {
    Ys <- Y[, i]
    Ys. <- sapply(Y.sim, function(y) y[, i])
    Ys.mean <- rowMeans(Ys.)
    Ys.CI <- apply(Ys., 1, function(x) quantile(x, probs = c(0.025, 0.975)))
    plot(past, Ys, type = "l", xlim = range(c(past, future)), xlab = "", ylab = "",
        main = stock.name[i])
    polygon(c(future, rev(future)), c(Ys.CI[1, ], rev(Ys.CI[2, ])), border = NA,
        col = "grey80")
    lines(future, Ys.mean, col = "royalblue3")
    lines(future, Ys.CI[1, ], col = "grey50")
    lines(future, Ys.CI[2, ], col = "grey50")
    legend("bottomright", bty = "n", lty = rep(1, 3), col = c("black", "royalblue3",
        "grey50"), legend = c("Actual Return", "Average Simulated Return", "95% CI"))
```

```
}



tm <- index(pricecopy)
start <- match(time[1], as.character(tm))
past <- tm[start:(start + T - 1)]
future <- tm[(start + T + 1):(start + T + m + 1)]
S.actual <- as.matrix(pricecopy[future, ])

S.start <- log(S)
S.start <- as.matrix(S.start[nrow(S), ])

S.sim <- lapply(Y.sim, function(y) {
    log.sim <- log(S.actual[-41, ]) + y
    exp(log.sim)
})

pt <- lapply(S.sim, function(s) {
    s <= S.actual[-1, ]
})


pt <- Reduce("+", pt)/200
zt <- qnorm(pt)

par(mfrow = c(3, 3))

stock.name <- colnames(S)

stock.names <- numeric(0)
p.val <- numeric(0)

for (i in 1:D) {
    zz <- zt[, i]
    zz <- zz[!is.infinite(zz)]
    if (length(zz) == 0)
        next
    if (is.na(zz))
        next
    k <- shapiro.test(zz)
    stock.names <- c(stock.names, stock.name[i])
    p.val <- c(p.val, k$p.value)
    qqnorm(zz, main = stock.name[i])
}

library(knitr)
shapiro.result <- matrix(c(stock.names, p.val), ncol = 2)
colnames(shapiro.result) <- c("stock", "p-val")
kable(shapiro.result)
```