MIE376 Project

Fady Shoukry, Bryan Wan
998918783, 999383885

April 10, 2014

Abstract

In this project, we develop a robust variation of the mean absolute deviation model for portfolio optimization. The model is dervied and then implmented in MATLAB® and it is tested using 6 months of data and then compared to the non-robust version over the last 2 months. The results have shown that, on average, the optimal portfolio generated by the robust MAD model performed better than the non-robust version. There was also a positive correlation between the increase in mean portfolio return and the robustness of the model.

1 Introduction

Financial portfolio optimization is one of the most commonly used applications of mathematical programming. The original model developed by Markowitz, the Mean Variance Optimization model (MVO), is a quadratic program which aims to minimize the total variance of a portfolio return as a measure of its risk. Given the difficulties associated with parameter estimation and solving this large quadratic program, the Mean Absolute Deviation model was formulated by Konno and Yamazaki. This model aims to minimize the expected value of the absolute difference between the actual returns and the mean return for an asset. This version requires considerably less parameter estimation and it can also be formulated as a linear program, making it more computationally tractable. However, this model does not eliminate the need for parameter estimation which naturally leads to uncertainty. This suggests that a robust version of the model, a version incorporating this uncertainty could lead to more accurate portfolios.

2 Model Derivation

2.1 Formulation of the linear non-robust MAD program

The mean absolute deviation (MAD) optimization model uses the sum of the absolute difference between the actual and expected rates of return of the assets in a portfolio as a measure of the portfolio's risk. Thus the goal of the model becomes minimizing this sum. Mathematically, the model is represented as follows

$$
\begin{aligned}
\text{minimize} \ & \sum_{t=1}^{T} \Big| \sum_{i=1}^{n} (r_{it} - \mu_i) x_i \Big| \\
\text{subject to} \ & \sum_{i=1}^{n} x_i \mu_i \geq R \\
& \sum_{i=1}^{n} x_i = 1 \\
& x_i \text{ is unrestricted, } i = 1, \ldots, n
\end{aligned}
$$

This model can be reformulated into a linear program by minimizing the auxiliary variable y_t such that the following holds

$$
\begin{aligned}
y_t \geq \Big| \sum_{i=1}^{n} (r_{it} - \mu_i) x_i \Big| \iff & \ y_t \geq - \sum_{i=1}^{n} (r_{it} - \mu_i) x_i \text{ AND } y_t \geq \sum_{i=1}^{n} (r_{it} - \mu_i) x_i \\
\iff & \ y_t \geq \sum_{i=1}^{n} (r_{it} - \mu_i) x_i \geq 0 \text{ AND } y_t - \sum_{i=1}^{n} (r_{it} - \mu_i) x_i \geq 0
\end{aligned}
$$

As a result, the MAD model can be represented as follows

$$
\begin{aligned}
\text{minimize} \ & \sum_{t=1}^{T} y_t \\
\text{subject to} \ & y_t + \sum_{i=1}^{n} (r_{it} - \mu_i) x_i \geq 0 \\
& y_t - \sum_{i=1}^{n} (r_{it} - \mu_i) x_i \geq 0 \\
& \sum_{i=1}^{n} x_i \mu_i \geq R \\
& \sum_{i=1}^{n} x_i = 1 \\
& x_i \text{ is unrestricted, } i = 1, \ldots, n
\end{aligned}
$$

2.2 Formulating the Robust Counterpart

2.2.1 Derivation of the uncertainty set

In order to formulate the robust counterpart, an uncertainty set is required. As previously mentioned, the MAD model bases its measure of risk on the deviation of the actual returns from the mean return. The mean return is approximated by the sample average over a given period of time,

$$
\mu_i = \frac{1}{T} \sum_{t=1}^{T} r_{it}
$$

As a result, there is an uncertainty associated with the approximate value of the mean. This uncertainty can be estimated as the margin of error (MOE) associated with the sample mean. Given the initial assumption that the return vector $\mathbf{r} = (r_1, \cdots, r_n)^\top \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, it follows that the each $r_i \sim \mathcal{N}(\mu_i, \sigma_i^2)$. To incorporate the uncertainty in the model, let $\hat{\mu}_i$ be the estimated sample mean return for asset i. it follows that

$$\frac{\hat{\mu}_i - \mu_i}{\hat{\sigma}_i/\sqrt{T}} \sim t(T-1) \text{ where } \hat{\sigma}_i = \sqrt{\frac{1}{T-1}\sum_{t=1}^{T}(r_{it} - \hat{\mu}_i)}$$

Thus a $(1-\alpha)$ confidence interval for the mean can be constructed as follows

$$\hat{\mu}_i - t_{\frac{\alpha}{2}}\frac{\hat{\sigma}_i}{\sqrt{T}} \leq \mu_i \leq \hat{\mu}_i + t_{\frac{\alpha}{2}}\frac{\hat{\sigma}_i}{\sqrt{T}}$$

This confidence interval will be used as the following uncertainty set

$$B_i = \left\{\mu_i : \mu_i = \hat{\mu}_i + \zeta_i t_{\frac{\alpha}{2}}\frac{\hat{\sigma}_i}{\sqrt{T}}, \forall \zeta_i \in [-1, 1]\right\}$$

The value of α can be adjusted to control the robustness of the model: smaller values of α will generate a more robust model.

2.2.2 Formulation of the RC constraints

Given the uncertainty set B_i defined above, the first constraint in the original MAD model can be adjusted as follows

$$\begin{aligned}
& y_t + \sum_{i=1}^{n}(r_{it} - \mu_i)x_i \geq 0 \\
\Longrightarrow\ & y_t + \sum_{i=1}^{n}(r_{it} - \hat{\mu}_i + \zeta_i t_{\frac{\alpha}{2}}\frac{\hat{\sigma}_i}{\sqrt{T}})x_i \geq 0 \\
\Longrightarrow\ & y_t + \sum_{i=1}^{n}(r_{it} - \hat{\mu}_i)x_i \geq \sum_{i=1}^{n}\zeta_i t_{\frac{\alpha}{2}}\frac{\hat{\sigma}_i}{\sqrt{T}}x_i, \forall \zeta_i \in [-1, 1] \\
\Longrightarrow\ & y_t + \sum_{i=1}^{n}(r_{it} - \hat{\mu}_i)x_i \geq \max_{-1\leq\zeta_i\leq 1}\left(\sum_{i=1}^{n}\zeta_i t_{\frac{\alpha}{2}}\frac{\hat{\sigma}_i}{\sqrt{T}}x_i\right) = \sum_{i=1}^{n}|t_{\frac{\alpha}{2}}\frac{\hat{\sigma}_i}{\sqrt{T}}x_i|
\end{aligned}$$

The absolute value term $|t_{\frac{\alpha}{2}}\frac{\hat{\sigma}_i}{\sqrt{T}}x_i|$ can be removed by substituting the following 2 inequalities

$$\begin{aligned}
y_t + \sum_{i=1}^{n}(r_{it} - \hat{\mu}_i)x_i &\geq \sum_{i=1}^{n}t_{\frac{\alpha}{2}}\frac{\hat{\sigma}_i}{\sqrt{T}}x_i \\
y_t + \sum_{i=1}^{n}(r_{it} - \hat{\mu}_i)x_i &\geq -\sum_{i=1}^{n}\zeta_i t_{\frac{\alpha}{2}}\frac{\hat{\sigma}_i}{\sqrt{T}}x_i
\end{aligned}$$

Using the same procedure outlined above and with some algebraic manipulations, the RC of MAD model can be formulated as follows

$$
\begin{aligned}
\text{minimize} \quad & \sum_{t=1}^{T} y_t \\
\text{subject to} \quad & y_t + \sum_{i=1}^{n} (r_{it} - \hat{\mu}_i - t_{\frac{\alpha}{2}} \frac{\hat{\sigma}_i}{\sqrt{T}}) x_i \geq 0 \\
& y_t + \sum_{i=1}^{n} (r_{it} - \hat{\mu}_i + t_{\frac{\alpha}{2}} \frac{\hat{\sigma}_i}{\sqrt{T}}) x_i \geq 0 \\
& y_t - \sum_{i=1}^{n} (r_{it} - \hat{\mu}_i - t_{\frac{\alpha}{2}} \frac{\hat{\sigma}_i}{\sqrt{T}}) x_i \geq 0 \\
& y_t - \sum_{i=1}^{n} (r_{it} - \hat{\mu}_i + t_{\frac{\alpha}{2}} \frac{\hat{\sigma}_i}{\sqrt{T}}) x_i \geq 0 \\
& \sum_{i=1}^{n} (\mu_i - t_{\frac{\alpha}{2}} \frac{\hat{\sigma}_i}{\sqrt{T}}) x_i \geq R \\
& \sum_{i=1}^{n} (\mu_i + t_{\frac{\alpha}{2}} \frac{\hat{\sigma}_i}{\sqrt{T}}) x_i \geq R \\
& \sum_{i=1}^{n} x_i = 1 \\
& x_i \text{ is unrestricted, } i = 1, \ldots, n
\end{aligned}
$$

Remarks

It's important to note that the generated uncertainty set or confidence interval above assumes the mean follows the t-distribution. This is true for the arithmetic mean of normal any random variable. However, in financial optimization, the geometric mean is a lot more commonly used. In this case the uncertainty set would have to be adjusted with the proper distribution. The subsequent derivation, however, is completely applicable.

3 Analysis of the Robust model

The model derived above was implemented in MATLAB® (code is shown in Appendix A). The daily adjusted closing prices for 10 stocks was obtained from Yahoo Finance® for the period of August 3rd, 2013 to April 4th, 2014. This was used to calculate the daily rates of return over that period. The first 6 months of data was used to estimate the parameters for both the robust and non-robust models. An optimal solution was found for both models for a range of minimum expected return values. The confidence level was set at 95% for the robust model. The optimal objective function values for both models were plotted for the range of return and are shown in Figure 1 below. As expected, the optimal objective function value of the more constrained robust model is consistently higher than the non-robust model. It is also clear that the difference increases as the rate of return increases.

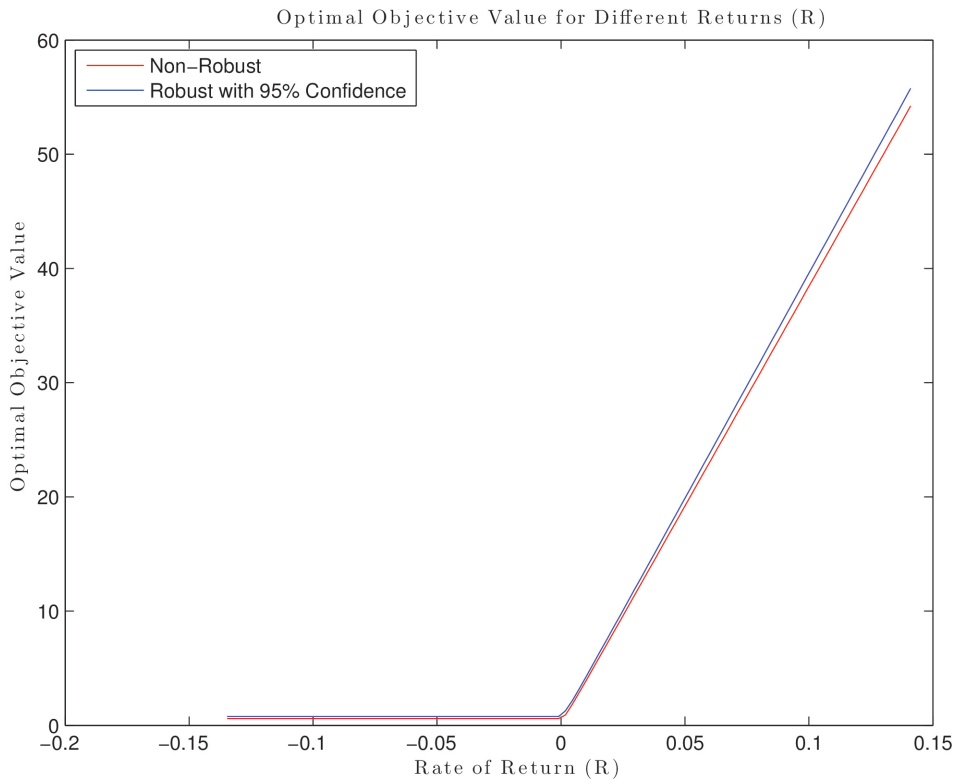


Figure 1: The optimal Objective function value plotted for different values of required minimum return

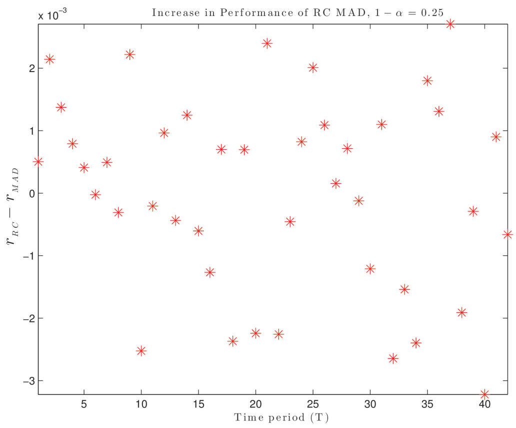
4 Testing the performance of the Robust model

4.1 Procedure

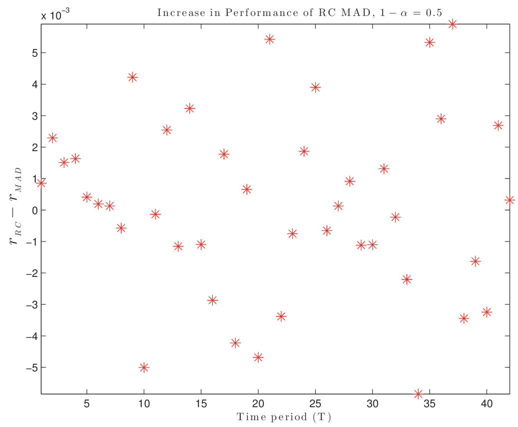
Using the first 6 months of data, the robust model was optimized with a reasonable minimum required return rate (in this case we used the mean of all returns in the data) for 4 different levels of confidence, 25%, 50%, 75% and 95%. The original MAD was also optimized using the same required minimum return. The performance of the resulting portfolios was then compared over the last 2 months of the data to see which portfolio made a better return. The difference between the rate of return of the robust and the non-robust portfolios was computed and plotted for the 4 confidence levels. The plots are shown in Figures 1-4 below.

4.2 Results

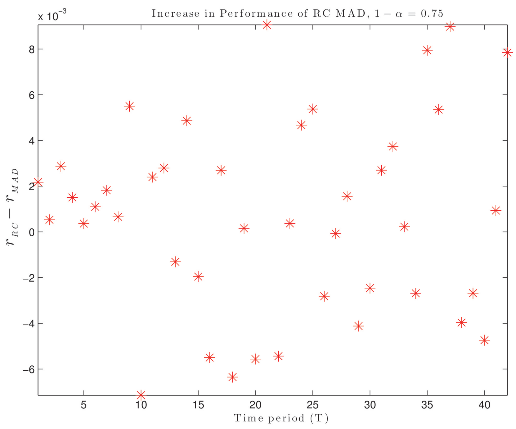
As shown in the Figures 1(a)-(d)below, the performance of the robust portfolio relative to the non-robust portfolio appears to be quite random. This is expected given the random nature of the financial markets.



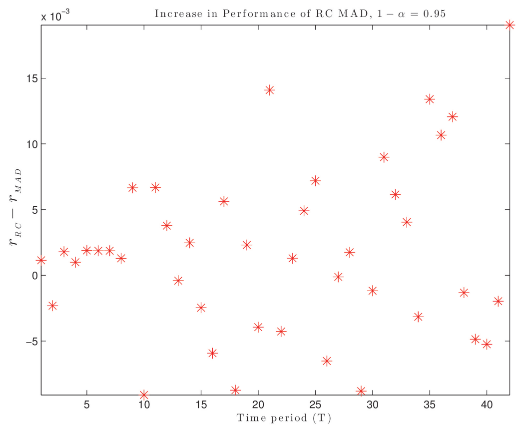
(a) 25% Confidence



(b) 50% Confidence



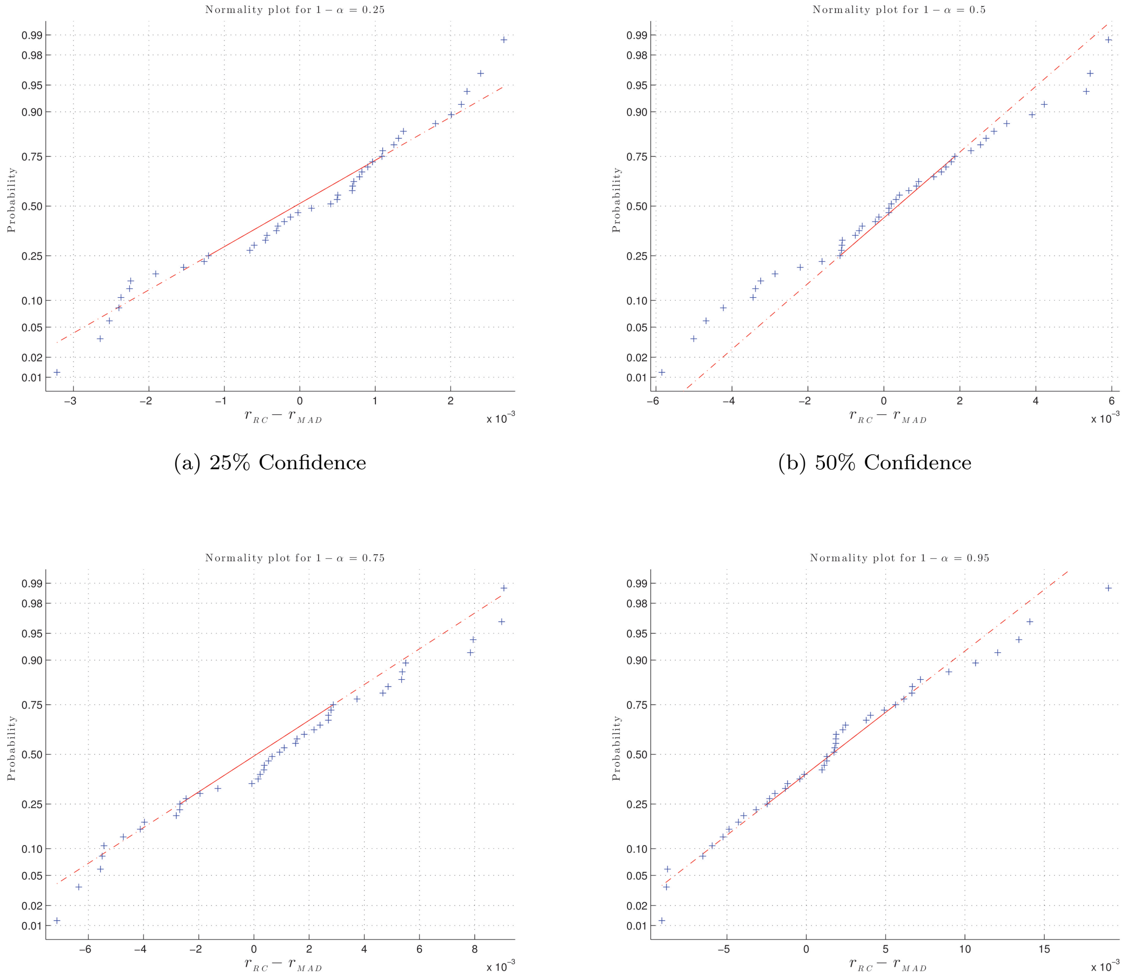
(c) 75% Confidence



(d) 95% Confidence

Figure 1. The plots of the difference between the returns of the robust and non-robust portfolios for different levels of confidence

To analyse the nature of this randomness, the probability of the data was plotted and compared against a normal distribution as shown in Figures 2(a)-(d) below. The data appears to fit a normal distribution very well confirming that the difference between the performance of the 2 portfolios most likely follows a normal distribution as well.



(a) 25% Confidence

(b) 50% Confidence

(c) 75% Confidence

(d) 95% Confidence

Figure 2. Normality plots for the different confidence levels

Since the difference appears to be normally distributed, the mean difference for each confidence level was computed and the values are shown in Table 1 below. The means show a clear consistent trend of increase in the strictly non-negative difference (in favour of the robust portfolio) as the robustness of the model increases. This implies that a more robust MAD model will, on average, perform better than a non-robust MAD model.

1 - α	0.25	0.50	0.75	0.95
mean difference	0.0000	0.0002	0.0007	0.0017

Table 1. The mean difference between the return of the robust and non-robust portfolios for different confidence levels

5 Conclusion

We have derived a robust formulation of the MAD financial optimization model by incorporating the uncertainty associated with the model parameters estimation , namely the mean rate of return. The performance of the robust MAD relative to the MAD model appeared random. The difference between the robust portfolio return and the non-robust return followed a normal distribution, the mean of which was strictly non-negative and increasing as the robustness of the model increased. While the results might not be conclusive due to the small size of the sample set, they appear to favour the non-robust model.

Appendix A: MATLAB® Code

RCMAD.m

```
function [x, fval] = RCMAD(r_it, confidence, R)
%RCMAD implements a robust version of the Mean Absolute Deviation
% (MAD) financial optimization model.

    % Get the size of the data
    % N is the number of assets
    % T is the number of periods
    [T, N] = size(r_it);

    %% Calculating the required model parameters

    % Confidence level
    alpha = 1-confidence;

    % Calculate mu using geometric mean
    mu = geomean(1 + r_it) - 1;

    % Calculate the sample standard deviation for each asset
    sigma = std(r_it);

    % Calculate the standard error in mean for a (1-alpha) confidence
interval
    s_e = tinv(alpha/2, T-1)*sigma./sqrt(T);

    %% Formulate the model

    % Cost function
    f = [ones(T, 1); zeros(N, 1)];

    % Inequality Constraints
    % The following are repitition of my and s_e needed for
    % the construction of the A Matrix of inequality constraints
    mu_rep = repmat(mu, T, 1);
    s_e_rep = repmat(s_e, T, 1);

    A = [-eye(T), -r_it + mu_rep + s_e_rep;
         -eye(T), -r_it + mu_rep - s_e_rep;
         -eye(T),  r_it - mu_rep - s_e_rep;
         -eye(T),  r_it - mu_rep + s_e_rep;
         zeros(1, T), -mu - s_e;
         zeros(1, T), -mu + s_e];

    b = [zeros(T*4, 1); -R; -R];
    % Euqality constraint
    % sum(xi) = 1
    Aeq = [zeros(1, T), ones(1, N)];
    beq = 1;

    % Lower and Upper bounds
    lb = [zeros(T, 1); -inf*ones(N, 1)];
```

```
    %% Optimize the model

    %set the options
    options.Display = 'off';

    [x, fval] = linprog(f, A, b, Aeq, beq, lb, [], 0, options);
    x = x(T+1 : T+N);

end
```

MAD.m

```
function [x, fval] = MAD(r_it, R)
%MAD implements the Mean Absolute Deviation (MAD) financial
optimization model.

    % Get the size of the data
    % N is the number of assets
    % T is the number of periods
    [T, n] = size(r_it);
    T;

    %% Calculating the required model parameters

    % Calculate mu using geometric mean
    mu = geomean(1 + r_it) - 1;

    %% MAD by linprog
    c = [zeros(n,1); ones(T,1); ones(T,1)];% [x_i; y_t; z_t]
    Aeq = [r_it-repmat(mu,T,1) -eye(T) eye(T);
        ones(1,n) zeros(1,2*T);];% the constraint coefficient of MAD
    beq=[zeros(T,1); 1;];
    lb =[ones(n, 1)*-inf; zeros(T+T,1)];         % the lower bound of
the variables

    A=-[mu zeros(1,2*T);];
    b=-R;

    %% Optimize the model

    % Set the options
    options.Display = 'off';

    [x, fval] = linprog(c, A, b, Aeq, beq, lb, [], 0, options);
    x = x(1:n);


end
```

testing.m

```
% This script tests the robust and non-robust formulations of the MAD
model

clear
clc

%% Fetching Required Data

connect = yahoo; %the source
assets = {'GOOGL', 'AAPL', 'ALTR', 'FB', 'YHOO', 'GS', 'TXN', 'IBM',
'SSNLF', 'MSIQX'};

% Get the data from Yahoo
% The modelling data
for i = 1:length(assets);
    tmp = fetch(connect, assets{i}, 'adj close', 'Aug 3 2013', 'Feb 3
2014', 'd');
    model_data(:, i) = tmp(:, 2);
end
% Testing data
for i = 1:length(assets);
    tmp = fetch(connect, assets{i}, 'adj close', 'Feb 4 2014', 'April 4
2014', 'd');
    testing_data(:, i) = tmp(:, 2);
end

% Re-order the data from oldest to newest
model_data = flipud(model_data);
testing_data = flipud(testing_data);

% Calculate the per-period return rates for both data sets
model_r_it = (model_data(2:end,:)./model_data(1:end-1,:))-1;
testing_r_it = (testing_data(2:end,:)./testing_data(1:end-1,:))-1;

% Get the size of the data
% N is the number of assets
% T is the number of periods
[T, N] = size(model_r_it);

% Compute the range of R values for comparison
R = linspace(min(model_r_it(:)), max(model_r_it(:)), 100);


%% Get the optimum solution for both models for a range of R values
% For a confidence level of 50%
% initialize the result vectors for speed
x_RMAD = zeros(N, length(R));
x_MAD = zeros(N, length(R));
fval_RMAD = zeros(1, length(R));
fval_MAD = zeros(1, length(R));

% For a confidence level of 95%
for i = 1:length(R)
```

```
    [x_RMAD(:,i), fval_RMAD(i)] = RCMAD(model_r_it, 0.95, R(i));
    [x_MAD(:,i), fval_MAD(i)] = MAD(model_r_it, R(i));
end

%% Plot the 2 sets of data
figure
plot(R, fval_MAD, 'r', ...
     R, fval_RMAD, 'b');
% Make the graph pretty
title('Optimal Objective Value for Different Returns (R)', ...
       'interpreter', 'latex');
ylabel('Optimal Objective Value', ...
       'interpreter', 'latex');
xlabel('Rate of Return (R)', ...
       'interpreter', 'latex');
legend('Non-Robust', 'Robust with 95% Confidence', ...
        'Location', 'NorthWest');

%% Find the optimal portfolios for different values of alpha for a
given R
R = mean2(model_r_it); % Use the mean value of all returns as your R
confidence = [0.25, 0.5, 0.75, 0.95];

for a = 1:length(confidence)
    [x_RMAD_test(:,a), fval_RMAD_test(a)] = RCMAD(model_r_it,
confidence(a), R);
end

[x_MAD_test, fval_MAD_test] = MAD(model_r_it, R); % find the optimal
portfolio with MAD

%% Compare performance over the last 2 months of data
% Compute the return of RMAD portfolios for all values of confidence
% over all the time periods in the last 2 months
RMAD_returns = (testing_r_it*x_RMAD_test)';
MAD_returns = (testing_r_it*x_MAD_test)';

% Compute the difference between the RMAD and MAD returns
RMAD_performance = RMAD_returns - repmat(MAD_returns, 4, 1);

%% Plot the difference for the 4 values of confidence
x = 1:42;
for i = 1:4
    figure
    plot(x, RMAD_performance(i, :), '*r', 'MarkerSize', 10);
    title(['Increase in Performance of RC MAD, $1- \alpha$ = '
num2str(confidence(i))], ...
           'interpreter', 'latex');
    ylabel('$r_{_{RC}} - r_{_{MAD}}$', ...
           'interpreter', 'latex', 'FontSize', 15);
    xlabel('Time period (T)', ...
           'interpreter', 'latex');
    axis tight;
end

%% Plot probability plots to check for normality
for i = 1:4
```

```
    figure
    normplot(RMAD_performance(i,:));
    title(['Normality plot for $1- \alpha$ = ' num2str(confidence(i))],
...
             'interpreter', 'latex');
    xlabel('$r_{_{RC}} - r_{_{MAD}}$', ...
            'interpreter', 'latex', 'FontSize', 15);
    ylabel('Probability', 'interpreter', 'latex');
end
%% Compute the mean performance increase over the last 2 months for
each alpha
mean_performance = mean(RMAD_performance');
```

Instructions to run the files

The main script is 'testing.m', to execute all the code used in the project, simply run this script.

To test the individual models, you can do the following for each of the 2 models:

Robust MAD:

Call the function ***RCMAD(r_it, confidence, R)*** with the following parameters:

r_it is a T x N matrix containing the returns of N assets for T periods of time.

confidence is the (1 – alpha) level such that 0 ≤ confidence ≤ 1.

R is the minimum required return rate as a decimal point.

MAD:

Call the function ***MAD(r_it, R)*** with the following parameters:

r_it is a T x N matrix containing the returns of N assets for T periods of time.

R is the minimum required return rate as a decimal point.