Constructing the BAB Factor

```
clc;
clear all;
close all;
```

Load and clean market retuns and risk-free rate

```
% Market returns and risk-free rate
mktrf = readtable ("mktret_and_rf_daily_fs.xlsx");
mktrf(10035:end,:) = []; % remove 2020
mktrf(4636,:) = []; % remove June 30th 1998 (NaN)
mktdates = table2array(mktrf(:,1));
mktrf = removevars(mktrf, 'date');
mktrfarray = table2array (mktrf);
rf_before = mktrfarray (:, 5);


% Risk-free rate interpolation
dates = 1:length(rf_before);
interpol_rf = fillmissing (rf_before (:, 1), "spline" ,'SamplePoints',
dates);
mktrfarray (:, 5) = interpol_rf;


% plot interpolation points of rf
interpoints = interpol_rf == rf_before (:, 1);
interpoints = ~interpoints;
interpointz = zeros (length(rf_before), 1);
for i = 1:length(rf_before)
    if interpoints (i) == 1
       interpointz (i) = interpol_rf (i);
    end
end
for i = 1:length(rf_before)
    if interpointz (i) == 0;
       interpointz (i) = NaN;
    end
end


plot (rf_before (:, 1));
hold on
plot (interpointz, 'o');


% Log market returns
mkt = mktrfarray (:, 2);
mkt_log = log (mkt + 1);


% Log rf
rf_log = log (interpol_rf +1);
```

```
% Log market excess ret
mkt_x = mkt_log-rf_log;
```

Load and clean stock returns

```
% stock returns
OSEret      = readtable   ("OSErets_80_19.xlsx");
OSEdates    = table2array (OSEret (:, 1));
OSEret      = removevars  (OSEret, 'Var1');
OSErets     = table2array (OSEret (:, 1:end));
OSErets_log = log (OSErets + 1);
OSErets_x   = OSErets_log - rf_log;
```

Standard Deviation - stock returns

```
% vector which takes on value 0 if NaN and 1 otherwise
OSE_NaNs = isnan (OSErets_x);
OSE_NaNs = (~OSE_NaNs);


svol = NaN (size(OSErets_x));
for i = 251:length(OSErets_x)
   for j = 1:width(OSEret)
       if sum (OSE_NaNs (i-250:i-1, j)) >= 120
          svol (i, j) = nanstd (OSErets_x (i-250:i-1, j));
       end
   end
end
```

Standard Deviation - market returns

```
mktvol_rw = NaN (size (mkt_x));
for i = 251:length(mkt_x)
    mktvol_rw (i) = nanstd (mkt_x (i-250:i-1));
end
```

Correlation between stock returns and market returns

```
% Overlapping 3 day log returns
OSE_threeday = NaN (size(OSErets_x));
for i = 3:length(OSErets_x);
    for j = 1:width (OSEret);
        OSE_threeday (i,j) = (OSErets_x(i,j)+OSErets_x(i-
1,j)+OSErets_x(i-2,j));
    end
end


mkt_threeday = NaN (size(OSErets_x));
for i = 3:length(OSErets_x);
        mkt_threeday (i,1) = (mkt_x(i,1)+mkt_x(i-1,1)+mkt_x(i-2,1));
end


stckmktcorr = NaN (size(OSErets_x));
for i = 1251:length(OSErets_x)
```

```
    for j = 1:width(OSEret);
        if sum (OSE_NaNs (i-1250:i-1, j)) >= 750;
           stckmktcorr (i, j) = corr (mkt_threeday (i-1250:i-1, 1),
OSE_threeday (i-1250:i-1, j), "rows","complete");
        end
    end
end
```

Ex-ante Betas

```
% beta = (corr (s, m)*sigma(s))/sigma(m)
betas = NaN (size(OSErets));
for i = 1:length(OSErets)
    for j = 1:width(OSEret)
    betas (i, j) = (stckmktcorr (i, j)*svol (i, j))/mktvol_rw (i);
        if stckmktcorr(i, j) == NaN
           betas (i, j) == NaN
        end
    end
end


% shrinking betas to the cross-sectional mean
betas = betas*0.6 + 0.4;
```

Ranking betas

```
% rank betas and returns at the end of each month
betasranked   = betas;
returnsranked = OSErets;


newmonth = zeros (length(OSErets),1);
for i = 2:length(OSErets);
    if month(OSEdates(i)) == month(OSEdates(i-1));
        newmonth(i) = 0;
    else
        newmonth(i) = 1;
    end
end
newmonth(1) = 1;


for i = 1:length(OSErets);
    if newmonth(i) == 1;
        [betasranked(i,:),order]=sort(betasranked(i,:));
        betaobs = sum(~isnan(betasranked(i,:)));
        returnsranked(i,:)=returnsranked(i,order);
        returnsranked(i,betaobs+1:end)=NaN;
    else
        returnsranked(i,:)=returnsranked(i,order);
        returnsranked(i,betaobs+1:end)=NaN;
    end
end
```

```
monthlybetas = NaN (size(OSErets));
for i = 2:length(OSErets);
    for j = 1:width(OSEret);
        if newmonth(i) == 1;
            monthlybetas(i,:) = betasranked(i,:);
        else
            monthlybetas(i,j) = monthlybetas(i-1,j);
        end
    end
end
```

Number of betas in each decile

```
betaobs1 = zeros (length(OSErets), 1); % total number of betas for each
time t
for i = 1:length (OSErets)
    betaobs1 (i) = sum (~isnan (monthlybetas (i, :)));
end


fractions = betaobs1 ./ 2; % number of betas per decile


betasinalldec = zeros (length(OSErets), 2);
for i = 1:length(OSErets)
    betasinalldec (i, :) = floor(fractions (i));
end


for i = 1:length(OSErets)
    additional (i) = 2 * (fractions (i) - betasinalldec (i, 1));
end
additional = round (additional)';


for i = 1:length(monthlybetas)
    if additional(i) == 1
        monthlybetas  (i, betasinalldec (i, 1) + 1) = NaN;
        returnsranked (i, betasinalldec (i, 1) + 1) = NaN;
    end
end
```

Decile start & end

```
decstart = ones (length(OSErets), 2);
for i = 1:length (OSErets);
        decstart (i, 2) = decstart (i, 1) + betasinalldec (i, 1) +
additional (i);
end


decend = zeros (length(OSErets), 2);
for i = 1:length(OSErets);
        decend (i,1) = sum (betasinalldec (i,1));
        decend (i,2) = sum (betasinalldec (i,1:2))+additional(i);
end
```

```
% weight betas in each decile
rank1 = zeros (size(OSErets));
for i = 751:length(OSErets);
    for j = 1:2;
        rank1 (i,decstart(i,j):decend(i,j)) = betaobs1(i)-
additional(i);
    end
end


for i = 1:length(OSErets)
    for j = 2:width(OSEret)
        if rank1 (i,j) == 0
            rank1 (i, j) = rank1 (i, j-1);
        else
            rank1 (i, j) = rank1 (i, j-1)-1;
        end
    end
end


ranks = NaN (size(OSErets));
for i = 1:length(OSErets)
    for j = 1:2;
        ranks (i,decstart(i,j):decend(i,j)) = rank1
(i,decstart(i,j):decend(i,j));
    end
end


zhat = NaN (length(OSErets),1);
for i = 1:length(OSErets);
        zhat (i) = nanmean(ranks(i,:));
end


z_zhat = NaN (size(OSErets));
for i = 1:length(OSErets);
    for j = 1:width(OSEret);
        z_zhat (i,j) = abs(ranks(i,j)-zhat(i));
    end
end


k = NaN (length(OSErets),1);
for i = 1:length(OSErets);
        k (i) = 2/nansum(z_zhat(i,:));
end


betaweights = NaN (size(OSErets));
for i = 1:length(OSErets);
```

```
    for j = 1:width(OSEret);
        betaweights (i,j) = k(i)*z_zhat(i,j);
    end
end
```

Weighted betas

```
betaweighted = NaN (size(OSErets));
for i = 1:length(OSErets);
    for j = 1:width(OSEret);
        betaweighted (i,j) = betaweights (i,j)*monthlybetas(i,j);
    end
end


% weighted average beta in each decile
betameans = NaN (length(OSErets), 2);
for i = 1:length(OSErets);
    for j = 1:2;
        betameans (i,j) = sum
(betaweighted(i,decstart(i,j):decend(i,j)));
    end
end


% Subtracting rf
returnsranked_rf = returnsranked - interpol_rf;
returnsranked_rf (isnan(returnsranked_rf)) = 0;% SETTING NaNs to zero
```

Mont ret

```
retrank = returnsranked;
retrank (isnan(retrank)) = 0;
retrank = retrank+1;
for i = 2:length(betaweights); %Takes a few min to run
    for j = 1:width(OSEret);
        for k = 1:2;
            if newmonth(i,1) == 1
                betaweights(i,j) = betaweights(i,j);
            else
                betaweights(i,j) = betaweights(i-1,j)*retrank(i-1,j);
                betaweights(i,decstart(i,k):decend(i,k)) =
betaweights(i,decstart(i,k):decend(i,k))/sum(betaweights(i,decstart(i,k
):decend(i,k)));
            end
        end
    end
end


bw_returns = zeros (size(returnsranked_rf));
for i = 2:length(returnsranked_rf)
    for j = 1:width(OSEret)
        bw_returns(i,j) = returnsranked_rf(i,j)*betaweights(i,j);
    end
```

```
end


bw_betadecret = zeros (length(OSErets), 2);
for i = 1:length(OSErets)
    for j = 1:2
        bw_betadecret (i, j) = sum (bw_returns
(i,decstart(i,j):decend(i,j)));
    end
end


levbetarets = zeros (size(bw_betadecret));
for i = 1:length(bw_betadecret);
    for j = 1:2;
        levbetarets (i,j) = bw_betadecret (i, j)/betameans(i,j);
    end
end
bw_betadecret = levbetarets;


% Monthly
betadecret_plus1 = bw_betadecret + 1;
cumbetadec = NaN (length(OSErets), 2);
for i = 2:length(OSErets)
    for j = 1:2
        if newmonth (i) == 1
            cumbetadec (i, j) = betadecret_plus1 (i, j);
        else
            cumbetadec (i, j) = cumbetadec (i-1,
j)*(betadecret_plus1(i,j));
        end
    end
end


monthlybetaret = NaN (length(OSErets), 2);
for i = 1:length(OSErets)-1
    for j = 1:2
        if newmonth (i+1) == 1;
            monthlybetaret (i, j) = cumbetadec (i, j);
        end
    end
end
monthlybetaret (length(monthlybetaret), :) = cumbetadec (end, :);
monthlybetaret = monthlybetaret - 1;


monthlybetaret = rmmissing (monthlybetaret); % Starter Jan 1985
BaB = monthlybetaret;
```

Betting-against-beta table

```
t = (datetime(1985,1,31):calmonths(1):datetime(2019,12,31)).';
t.Format='MMM-yyyy';
```

```
BaBtable = array2table(NaN(length(BaB),2));
t = cellstr( datestr(t));
BaBtable.Properties.RowNames = t;
BaBtable (:,1:2) = array2table(BaB);
BaBtable.Properties.VariableNames = {'LowBeta','HighBeta', };
```

IN-SAMPLE EW MULTIFACTOR STRATEGY

```
clc;
clear all;
warning('off')
```

LOAD AND CLEAN FACTOR PORTFOLIOS AND MARKET RETURNS

```
EW_btm_table = readtable ('ew_monthly_btm_fs.xlsx');
EW_mom_table = readtable ('ew_monthly_mom_fs.xlsx');
EW_siz_table = readtable ('ew_monthly_siz_fs.xlsx');
BW_bet_table = load      ("BettingAgainstBeta.mat");
BW_bet_table = BW_bet_table.BaBtable;
market = readtable ("mktret_rf_monthly_fs.xlsx");
mktret = table2array (market);
rf = mktret (61:312, end);
%
% In-sample period Jan 1985 - Dec 2005
btm = table2array (EW_btm_table (49:300, 2:end));
mom = table2array (EW_mom_table (61:312, 2:end));
siz = table2array (EW_siz_table (61:312, 2:end));
bet = table2array (BW_bet_table (1 :252, :    ));
mkt = mktret (61:312, 4);


% Quintile returns
HML = btm (:, 10) + btm (:, 9) - btm (:, 1) - btm (:,  2);
WML = mom (:, 10) + mom (:, 9) - mom (:, 1) - mom (:,  2);
SMB = siz (:,  1) + siz (:, 2) - siz (:, 9) - siz (:, 10);
BAB = bet (:, 1) - bet (:, 2);


% Return matrix
retmat = [mkt, HML, WML, SMB, BAB];
HMLcs = cumsum (HML);
WMLcs = cumsum (WML);
SMBcs = cumsum (SMB);
BABcs = cumsum (BAB);
mktcs = cumsum (mkt);


plot (HMLcs);
hold on;
plot (WMLcs);
plot (SMBcs);
plot (BABcs);
plot (mktcs);
% plot (MFMcs);
hold off;
title ('Cumulative Returns')
legend ('HML', 'WML', 'SMB', 'BAB', 'mkt', 'MFM')
```

Descriptive Statistics

```
% Correlations Matrix
corrmat = corr (retmat);
corrmat = round (corrmat, 4);
```

```
corrtbl = array2table (corrmat);
corrtbl.Properties.VariableNames = {'mkt', 'HML', 'WML', 'SMB', 'BAB'};
corrtbl.Properties.RowNames = {'mkt', 'HML', 'WML', 'SMB', 'BAB'};
disp (corrtbl)


% Returns, StDev, SR
returns = mean (retmat) * 12;
excess_rets = retmat - rf;
excess_means = mean (excess_rets) * 12;
stdev = std (retmat) * sqrt(12);
SR = excess_means ./ stdev;


% tabulate results
TD = round([returns; excess_means; stdev; SR;], 3);
Tbl = table(TD(:,1),TD(:,2),TD(:,3),TD(:,4), TD(:, 5));
Tbl.Properties.VariableNames = {'Mkt','HML','WML','SMB', 'BAB'};
Tbl.Properties.RowNames =  {'Ret','Ex_ret','StDev','SR'};
disp(Tbl)
```

Equal-weighted static multifactor model

```
n = 4;
MFM = 1/n * (HML + WML + SMB + BAB);
MFMcomp = [HML, WML, SMB, BAB, MFM];
M_returns = mean (MFMcomp) * 12;
M_exret = MFMcomp - rf;
M_exmean = mean (M_exret) * 12;
M_std = std (MFMcomp) * sqrt(12);
M_SR = M_exmean ./ M_std;


TD = round([M_returns; M_exmean; M_std; M_SR;], 3);
Tbl = table(TD(:,1),TD(:,2),TD(:,3),TD(:,4), TD(:, 5));
Tbl.Properties.VariableNames = {'HML','WML','SMB', 'BAB', 'MFM'};
Tbl.Properties.RowNames =  {'Ret','Ex_ret','StDev','SR'};
disp(Tbl)
```

DIFFERENT MEAN-VARIANCE COMBINATIONS IN-SAMPLE

```
facret = retmat (:, 2:5);
lb = 0; % no short sales
ub = 0.35; % change for different upper-bounds
```

Mean-Variance 1-year Rolling Window

```
s = 12;
l = 252;


% Return Vector
% 1985-2005: 1-year rolling window
ret_m_rebal = NaN (l, 4);
ex_ret_m_rebal = NaN (l, 4);
rf_m_rw_rebal  = NaN (l, 1);
for i = s:l
    for j = 1:4
        ret_m_rebal (i, j) = mean (facret (i-11:i, j)) * 12;
        rf_m_rw_rebal (i) = mean (rf (i-11:i))*12;
        ex_ret_m_rebal (i, j) = ret_m_rebal(i,j)-rf_m_rw_rebal(i);
    end
end


% Covariance Matrix & mean-variance optimization
mv_W_rebal = NaN (252,4);
mv_std_rebal = NaN (252,1);
mv_mu_rebal = NaN (252,1);
mv_exmu_rebal = NaN (252,1);
mv_SR_rebal = NaN (252,1);
for i = s:l
    cov_mat_rw = cov(facret (i-11:i, 1:4) - rf (i-11:i, 1))*12;
    trgt_mean_rw = min (ret_m_rebal (i, :)):0.05:max(ret_m_rebal (i,
:))+0.02;
    [W_wo, trgt_std_wo, n_trgt_mean_wo] = port_meanvar (ret_m_rebal
(i,:), cov_mat_rw, trgt_mean_rw, [], lb, ub);
    %
    SR = n_trgt_mean_wo./trgt_std_wo;
    mv_SR_rebal (i) = max (SR);
    mv_i = find (mv_SR_rebal(i) == SR);
    mv_W_rebal(i,:) = W_wo (:, mv_i);
    mv_std_rebal(i) = trgt_std_wo (mv_i);
    mv_mu_rebal(i) = n_trgt_mean_wo (mv_i);
    mv_exmu_rebal(i) = n_trgt_mean_wo (mv_i)-rf_m_rw_rebal (i);
end


returns_rw_rebal = NaN (252,4);
for i = 2:length(returns_rw_rebal);
    for j = 1:4
    returns_rw_rebal (i,j) = mv_W_rebal(i-1,j)*facret(i,j);
    end
end
```

```
for i = 1:length(returns_rw_rebal);
    agg_ret_rw1 (i, 1) = sum (returns_rw_rebal (i, :));
end


exret_rw1 = agg_ret_rw1-rf;


% Performance measures
mean_rw1 = nanmean (exret_rw1(61:end))*12
std_rw1 = nanstd (exret_rw1(61:end))*sqrt(12)
SR_rw1 = mean_rw1/std_rw1
arit_mean_rw1 = nanmean (agg_ret_rw1(61:end))*12;
skew_rw1 = skewness (agg_ret_rw1(61:end))
kurt_rw1 = kurtosis (agg_ret_rw1(61:end))
```

Mean-Variance Recursive Window

```
% Return Vector
% 1985-2005: Recursive window
ret_m_rec_rebal = NaN (l, 4);
ex_ret_rec_rebal = NaN (l, 4);
rf_m_rec_rebal  = NaN (l, 1);
for i = s:l
    for j = 1:4
        ret_m_rec_rebal (i, j) = mean (facret (1:i, j)) * 12;
        rf_m_rec_rebal (i) = mean (rf (1:i))*12;
        ex_ret_rec_rebal (i, j) = ret_m_rec_rebal(i,j)-
rf_m_rec_rebal(i);
    end
end


% Covariance Matrix & mean-variance optimization
mv_W_rec_rebal = NaN (252,4);
mv_std_rec_rebal = NaN (252,1);
mv_mu_rec_rebal = NaN (252,1);
mv_exmu_rec_rebal = NaN (252,1);
mv_SR_rec_rebal = NaN (252,1);
for i = s:l
    cov_mat_rec = cov(facret (1:i, 1:4) - rf (1:i, 1))*12;
    trgt_mean_rec = min (ret_m_rec_rebal (i,
:)):0.05:max(ret_m_rec_rebal (i, :))+0.02;
    [W_wo, trgt_std_wo, n_trgt_mean_wo] = port_meanvar (ret_m_rec_rebal
(i,:), cov_mat_rec, trgt_mean_rec, [], lb, ub);
    %
    SR_rec_rebal = n_trgt_mean_wo./trgt_std_wo;
    mv_SR_rec_rebal (i) = max (SR_rec_rebal);
    mv_i = find (mv_SR_rec_rebal(i) == SR_rec_rebal);
    mv_W_rec_rebal(i,:) = W_wo (:, mv_i);
    mv_std_rec_rebal(i) = trgt_std_wo (mv_i);
    mv_mu_rec_rebal(i) = n_trgt_mean_wo (mv_i);
    mv_exmu_rec_rebal(i) = n_trgt_mean_wo (mv_i)-rf_m_rec_rebal (i);
```

```
end


returns_rec_rebal = NaN (252,4);
for i = 2:length(returns_rec_rebal);
    for j = 1:4
    returns_rec_rebal (i,j) = mv_W_rec_rebal(i-1,j)*facret(i,j);
    end
end


for i = 1:length(returns_rec_rebal);
    agg_ret_rec (i, 1) = sum (returns_rec_rebal (i, :));
end


exret_rec = agg_ret_rec-rf;


% Performance measures
mean_rec = nanmean (exret_rec(61:end))*12
std_rec = nanstd (exret_rec(61:end))*sqrt(12)
SR_rec = mean_rec/std_rec
arit_mean_rec = nanmean (agg_ret_rec(61:end))*12
skew_rec = skewness (agg_ret_rec(61:end))
kurt_rec = kurtosis (agg_ret_rec(61:end))
```

Mean-Variance 3-year Rolling Window

```
s = 36;
l = 252;


% Return Vector
% 1985-2005: 3-year rolling window
ret_m_rebal = NaN (l, 4);
ex_ret_m_rebal = NaN (l, 4);
rf_m_rw_rebal  = NaN (l, 1);
for i = s:l
    for j = 1:4
        ret_m_rebal (i, j) = mean (facret (i-35:i, j)) * 12;
        rf_m_rw_rebal (i) = mean (rf (i-35:i))*12;
        ex_ret_m_rebal (i, j) = ret_m_rebal(i,j)-rf_m_rw_rebal(i);
    end
end


% Covariance Matrix & mean-variance optimization
mv_W_rebal = NaN (l,4);
mv_std_rebal = NaN (l,1);
mv_mu_rebal = NaN (l,1);
mv_exmu_rebal = NaN (l,1);
mv_SR_rebal = NaN (l,1);
for i = s:l
    cov_mat_rw = cov(facret (i-35:i, 1:4) - rf (i-35:i, 1))*12;
```

```
    trgt_mean_rw = min (ret_m_rebal (i, :)):0.05:max(ret_m_rebal (i,
:))+0.02;
    [W_wo, trgt_std_wo, n_trgt_mean_wo] = port_meanvar (ret_m_rebal
(i,:), cov_mat_rw, trgt_mean_rw, [], lb, ub);
    %
    SR = n_trgt_mean_wo./trgt_std_wo;
    mv_SR_rebal (i) = max (SR);
    mv_i = find (mv_SR_rebal(i) == SR);
    mv_W_rebal(i,:) = W_wo (:, mv_i);
    mv_std_rebal(i) = trgt_std_wo (mv_i);
    mv_mu_rebal(i) = n_trgt_mean_wo (mv_i);
    mv_exmu_rebal(i) = n_trgt_mean_wo (mv_i)-rf_m_rw_rebal (i);
end


returns_rw_rebal = NaN (l,4);
for i = 2:length(returns_rw_rebal);
    for j = 1:4
    returns_rw_rebal (i,j) = mv_W_rebal(i-1,j)*facret(i,j);
    end
end


for i = 1:length(returns_rw_rebal);
    agg_ret_rw3 (i, 1) = sum (returns_rw_rebal (i, :));
end


exret_rw3 = agg_ret_rw3-rf;


% Performance measures
mean_rw3 = nanmean (exret_rw3(61:end))*12
std_rw3 = nanstd (exret_rw3(61:end))*sqrt(12)
SR_rw3 = mean_rw3/std_rw3
arit_mean_rw3 = nanmean (agg_ret_rw3(61:end))*12;
skew_rw3 = skewness (agg_ret_rw3(61:end))
kurt_rw3 = kurtosis (agg_ret_rw3(61:end))
```

Mean-Variance 5-year Rolling Window

```
s = 60;
l = 252;


% Return Vector
% 1985-2005: 5-year rolling window
ret_m_rebal = NaN (l, 4);
ex_ret_m_rebal = NaN (l, 4);
rf_m_rw_rebal  = NaN (l, 1);
for i = s:l
    for j = 1:4
        ret_m_rebal (i, j) = mean (facret (i-59:i, j)) * 12;
        rf_m_rw_rebal (i) = mean (rf (i-59:i))*12;
        ex_ret_m_rebal (i, j) = ret_m_rebal(i,j)-rf_m_rw_rebal(i);
```

```
    end
end


% Covariance Matrix & mean-variance optimization
mv_W_rebal = NaN (252,4);
mv_std_rebal = NaN (252,1);
mv_mu_rebal = NaN (252,1);
mv_exmu_rebal = NaN (252,1);
mv_SR_rebal = NaN (252,1);
for i = s:l
    cov_mat_rw = cov(facret (i-59:i, 1:4) - rf (i-59:i, 1))*12;
    trgt_mean_rw = min (ret_m_rebal (i, :)):0.05:max(ret_m_rebal (i,
:))+0.02;
    [W_wo, trgt_std_wo, n_trgt_mean_wo] = port_meanvar (ret_m_rebal
(i,:), cov_mat_rw, trgt_mean_rw, [], lb, ub);
    %
    SR = n_trgt_mean_wo./trgt_std_wo;
    mv_SR_rebal (i) = max (SR);
    mv_i = find (mv_SR_rebal(i) == SR);
    mv_W_rebal(i,:) = W_wo (:, mv_i);
    mv_std_rebal(i) = trgt_std_wo (mv_i);
    mv_mu_rebal(i) = n_trgt_mean_wo (mv_i);
    mv_exmu_rebal(i) = n_trgt_mean_wo (mv_i)-rf_m_rw_rebal (i);
end


returns_rw_rebal = NaN (252,4);
for i = 2:length(returns_rw_rebal);
    for j = 1:4
    returns_rw_rebal (i,j) = mv_W_rebal(i-1,j)*facret(i,j);
    end
end


for i = 1:length(returns_rw_rebal);
    agg_ret_rw5 (i, 1) = sum (returns_rw_rebal (i, :));
end


exret_rw5 = agg_ret_rw5-rf;


% Performance measures
mean_rw5 = nanmean (exret_rw5(61:end))*12
std_rw5 = nanstd (exret_rw5(61:end))*sqrt(12)
SR_rw5 = mean_rw5/std_rw5
arit_mean_rw5 = nanmean (agg_ret_rw5(61:end))*12;
skew_rw5 = skewness (agg_ret_rw5(61:end))
kurt_rw5 = kurtosis (agg_ret_rw5(61:end))
```

MULTIVARIATE TIME SERIES ANALYSIS
(VAR & Granger Causality)

```
clear all;
clc;
warning('off')
```

Monthly market returns

```
mktreturns = readtable ("mktret_rf_monthly_fs.xlsx");
mktret = table2array (mktreturns);
mktret (313:end, :) = [];
rf = mktret (:, end);
```

Factor deciles - monthly returns

```
% Load and Clean Factor Portfolios
EW_btm_table = readtable ('ew_monthly_btm_fs.xlsx');
EW_mom_table = readtable ('ew_monthly_mom_fs.xlsx');
EW_siz_table = readtable ('ew_monthly_siz_fs.xlsx');
BW_bet_table = load      ("BettingAgainstBeta.mat");
BW_bet_table = BW_bet_table.BaBtable;


EW_btm_table (301:end, :) = [];
EW_mom_table (313:end, :) = [];
EW_siz_table (313:end, :) = [];
BW_bet_table (253:end, :) = [];


btm = table2array (EW_btm_table); btm = btm (:, 2:11);
mom = table2array (EW_mom_table); mom = mom (:, 2:11);
siz = table2array (EW_siz_table); siz = siz (:, 2:11);
bet = table2array (BW_bet_table);


HML = btm (:, 10) + btm (:, 9) - btm (:,  1) - btm (:, 2);
WML = mom (:, 10) + mom (:, 9) - mom (:,  1) - mom (:, 2);
SMB = siz (:,  1) + siz (:, 2) - siz (:, 10) - siz (:, 9);
BAB = bet (:,  1) - bet (:, 2);
```

Market volatility

```
daily_mkt_rets = readtable ("mktret_daily_fs.xlsx");
daily_mkt_rets (6523:end, :) = [];
mkt_dates_daily = daily_mkt_rets (:, 1);
mkt_dates_daily = table2array (mkt_dates_daily);
daily_mkt_rets = table2array (daily_mkt_rets (:, 3));


mkt_dates_daily = datetime (mkt_dates_daily, 'ConvertFrom',
'yyyymmdd');
```

```
newmonth = zeros (length(mkt_dates_daily),1);
for i = 2:length(mkt_dates_daily);
    if month(mkt_dates_daily(i)) == month(mkt_dates_daily(i-1));
        newmonth(i) = 0;
    else newmonth(i) = 1;
    end
end
newmonth (1,1) = 1;


nodays = (1:length (mkt_dates_daily))';
for i = 1:length (nodays)
    if newmonth (i) == 0
        nodays (i) = NaN;
    end
end
nodays = rmmissing (nodays);
nodays (end+1, 1) = length(newmonth)+1;


mktvol_month = zeros (length (nodays), 1);
for i = 2:(length(mktvol_month))
    mktvol_month (i) = nanstd (daily_mkt_rets (nodays(i-1):(nodays(i)-
1), 1));
end
mktvol_month (1,:) = [];


mktvol = mktvol_month (:, 1);
logdeltavol = NaN (length (mktvol), 1);
for i = 2:length(mktvol)
    logdeltavol (i) = log (mktvol(i)/mktvol(i-1));
end


vol_80_05 = logdeltavol ; % SMB and WML
vol_81_05 = logdeltavol (13:end, 1); % HML
vol_85_05 = logdeltavol (61:end, 1); % BAB
```

Composite Leading Indicators

```
CLI = readtable ("CLI_indicators.xlsx");
CLI (314:end, :) = [];
CLI_ = table2array (CLI(:,2));
CLI_80_05 = CLI_ ( 2:end, 1); % SMB and WML
CLI_81_05 = CLI_ (14:end, 1); % HML
CLI_85_05 = CLI_ (62:end, 1); % BAB
```

Market Liquidity

```
liq_data = readtable ("monthly_liq_data.xlsx");
liq_data (313:end, :) = [];
liqdata = table2array(liq_data (:, 2));
logdeltaliq = NaN (length (liqdata), 1);
for i = 2:length(liqdata)
```

```
    logdeltaliq (i) = log (liqdata(i)/liqdata(i-1));
end


liq_80_05 = logdeltaliq ; % SMB and WML
liq_81_05 = logdeltaliq (13:end, 1); % HML
liq_85_05 = logdeltaliq (61:end, 1); % BAB
```

Oil Prices

```
oiltbl = readtable ('Brent_CO1_Comdty.xlsx');
oilpr = table2array(oiltbl (:, 2));
oilpr (212:end) = [];


logoil = NaN (size (oilpr));
for i = 2:length(oilpr)
    logoil (i) = log (oilpr (i)/oilpr(i-1));
end
logoil = rmmissing (logoil);


HML_88 = HML (91 :end, 1); HML_88 = log (HML_88 + 1); % 31.07.1988
SMB_88 = SMB (103:end, 1); SMB_88 = log (SMB_88 + 1);
WML_88 = WML (103:end, 1); WML_88 = log (WML_88 + 1);
BAB_88 = BAB (43 :end, 1); BAB_88 = log (BAB_88 + 1);
```

Risk-free rate

```
logrf = log (rf + 1);
logrf_80 = logrf;
logrf_81 = logrf (13:end, 1);
logrf_85 = logrf (61:end, 1);
```

Var(k) models

```
logHML = log (HML + 1);
logSMB = log (SMB + 1);
logWML = log (WML + 1);
logBAB = log (BAB + 1);
```

VAR(k) estimation

```
HMLlog = array2table (logHML);
WMLlog = array2table (logWML);
SMBlog = array2table (logSMB);
BABlog = array2table (logBAB);


HML88 = array2table (HML_88);
SMB88 = array2table (SMB_88);
WML88 = array2table (WML_88);
BAB88 = array2table (BAB_88);


CLI80 = array2table (CLI_80_05);
```

```
CLI81 = array2table (CLI_81_05);
CLI85 = array2table (CLI_85_05);


liq80 = array2table (liq_80_05);
liq81 = array2table (liq_81_05);
liq85 = array2table (liq_85_05);


vol80 = array2table (vol_80_05);
vol81 = array2table (vol_81_05);
vol85 = array2table (vol_85_05);


rf80 = array2table (logrf_80);
rf81 = array2table (logrf_81);
rf85 = array2table (logrf_85);


logoil = array2table (logoil);


indic_SMB = [SMBlog, CLI80, liq80, vol80, rf80];
indic_WML = [WMLlog, CLI80, liq80, vol80, rf80];
indic_HML = [HMLlog, CLI81, liq81, vol81, rf81];
indic_BAB = [BABlog, CLI85, liq85, vol85, rf85];


indic_HMLoil = [HML88, logoil];
indic_SMBoil = [SMB88, logoil];
indic_WMLoil = [WML88, logoil];
indic_BABoil = [BAB88, logoil];


max_lags = 12;
```

HML

```
HML_vol = indic_HML(:, {'logHML', 'vol_81_05'});
HMLvol_output = varorder (HML_vol, max_lags);
optlag_HMLvol = HMLvol_output.hqor;


HML_CLI = indic_HML(:, {'logHML', 'CLI_81_05'});
HMLCLI_output = varorder (HML_CLI, max_lags);
optlag_HMLCLI = HMLCLI_output.hqor;


HML_liq = indic_HML(:, {'logHML', 'liq_81_05'});
HMLliq_output = varorder (HML_liq, max_lags);
optlag_HMLliq = HMLliq_output.hqor;


HML_rf = indic_HML(:, {'logHML', 'logrf_81'});
HMLrf_output = varorder (HML_rf, max_lags);
```

```
optlag_HMLrf = HMLrf_output.hqor;


HML_oil = indic_HMLoil(:, {'HML_88', 'logoil'});
HMLoil_output = varorder (HML_oil, max_lags);
optlag_HMLoil = HMLoil_output.hqor;
```

WML

```
WML_vol = indic_WML(:, {'logWML', 'vol_80_05'});
WMLvol_output = varorder (WML_vol, max_lags);
optlag_WMLvol = WMLvol_output.hqor;


WML_CLI = indic_WML(:, {'logWML', 'CLI_80_05'});
WMLCLI_output = varorder (WML_CLI, max_lags);
optlag_WMLCLI = WMLCLI_output.hqor;


WML_liq = indic_WML(:, {'logWML', 'liq_80_05'});
WMLliq_output = varorder (WML_liq, max_lags);
optlag_WMLliq = WMLliq_output.hqor;


WML_rf = indic_WML(:, {'logWML', 'logrf_80'});
WMLrf_output = varorder (WML_rf, max_lags);
optlag_WMLrf = WMLrf_output.hqor;


WML_oil = indic_WMLoil(:, {'WML_88', 'logoil'});
WMLoil_output = varorder (WML_oil, max_lags);
optlag_WMLoil = WMLoil_output.hqor;
```

SMB

```
SMB_vol = indic_SMB(:, {'logSMB', 'vol_80_05'});
SMBvol_output = varorder (SMB_vol, max_lags);
optlag_SMBvol = SMBvol_output.hqor;


SMB_CLI = indic_SMB(:, {'logSMB', 'CLI_80_05'});
SMBCLI_output = varorder (SMB_CLI, max_lags);
optlag_SMBCLI = SMBCLI_output.hqor;


SMB_liq = indic_SMB(:, {'logSMB', 'liq_80_05'});
SMBliq_output = varorder (SMB_liq, max_lags);
optlag_SMBliq = SMBliq_output.hqor;


SMB_rf = indic_SMB(:, {'logSMB', 'logrf_80'});
SMBrf_output = varorder (SMB_rf, max_lags);
optlag_SMBrf = SMBrf_output.hqor;


SMB_oil = indic_SMBoil(:, {'SMB_88', 'logoil'});
```

```
SMBoil_output = varorder (SMB_oil, max_lags);
optlag_SMBoil = SMBoil_output.hqor;
```

BAB

```
BAB_vol = indic_BAB(:, {'logBAB', 'vol_85_05'});
BABvol_output = varorder (BAB_vol, max_lags);
optlag_BABvol = BABvol_output.hqor;


BAB_CLI = indic_BAB(:, {'logBAB', 'CLI_85_05'});
BABCLI_output = varorder (BAB_CLI, max_lags);
optlag_BABCLI = BABCLI_output.hqor;


BAB_liq = indic_BAB(:, {'logBAB', 'liq_85_05'});
BABliq_output = varorder (BAB_liq, max_lags);
optlag_BABliq = BABliq_output.hqor;


BAB_rf = indic_BAB(:, {'logBAB', 'logrf_85'});
BABrf_output = varorder (BAB_rf, max_lags);
optlag_BABrf = BABrf_output.hqor;


BAB_oil = indic_BABoil(:, {'BAB_88', 'logoil'});
BABoil_output = varorder (BAB_oil, max_lags);
optlag_BABoil = BABoil_output.hqor;
```

Granger Causality Tests

HML

```
var_HML = table ();
var_HML.HML_t   = indic_HML.logHML;
var_HML.HML_t_1 = lagmatrix (indic_HML.logHML, 1);
var_HML.HML_t_2 = lagmatrix (indic_HML.logHML, 2);
var_HML.HML_t_3 = lagmatrix (indic_HML.logHML, 3);
var_HML.HML_t_4 = lagmatrix (indic_HML.logHML, 4);


var_HML.vol_t_1 = lagmatrix (indic_HML.vol_81_05, 1);
var_HML.vol_t_2 = lagmatrix (indic_HML.vol_81_05, 2);


var_HML.CLI_t_1 = lagmatrix (indic_HML.CLI_81_05, 1);
var_HML.CLI_t_2 = lagmatrix (indic_HML.CLI_81_05, 2);
var_HML.CLI_t_3 = lagmatrix (indic_HML.CLI_81_05, 3);
var_HML.CLI_t_4 = lagmatrix (indic_HML.CLI_81_05, 4);


var_HML.liq_t_1 = lagmatrix (indic_HML.liq_81_05, 1);
```

```
var_HML.rf_t_1  = lagmatrix (indic_HML.logrf_81, 1);
var_HML.rf_t_2  = lagmatrix (indic_HML.logrf_81, 2);


gc_HMLvol = fitlm (var_HML, 'HML_t ~ HML_t_1 + vol_t_1 + HML_t_2 +
vol_t_2')
gc_HMLCLI = fitlm (var_HML, 'HML_t ~ HML_t_2 + CLI_t_2 + HML_t_3 +
CLI_t_3+HML_t_4 + CLI_t_4')
gc_HMLliq = fitlm (var_HML, 'HML_t ~ HML_t_1 + liq_t_1')
gc_HMLrf  = fitlm (var_HML, 'HML_t ~ HML_t_1 + rf_t_1 + HML_t_2 +
rf_t_2')


gc_HML = fitlm (var_HML, 'HML_t ~ vol_t_1 ')
anova(gc_HML,'summary')
finv (0.90, 10, 288)
```

WML

```
var_WML = table ();
var_WML.WML_t   = indic_WML.logWML;
var_WML.WML_t_1 = lagmatrix (indic_WML.logWML, 1);
var_WML.WML_t_2 = lagmatrix (indic_WML.logWML, 2);
var_WML.WML_t_3 = lagmatrix (indic_WML.logWML, 3);
var_WML.WML_t_4 = lagmatrix (indic_WML.logWML, 4);
var_WML.WML_t_5 = lagmatrix (indic_WML.logWML, 5);


var_WML.vol_t_1 = lagmatrix (indic_WML.vol_80_05, 1);
var_WML.vol_t_2 = lagmatrix (indic_WML.vol_80_05, 2);


var_WML.CLI_t_1 = lagmatrix (indic_WML.CLI_80_05, 1);
var_WML.CLI_t_2 = lagmatrix (indic_WML.CLI_80_05, 2);
var_WML.CLI_t_3 = lagmatrix (indic_WML.CLI_80_05, 3);
var_WML.CLI_t_4 = lagmatrix (indic_WML.CLI_80_05, 4);
var_WML.CLI_t_5 = lagmatrix (indic_WML.CLI_80_05, 5);


var_WML.liq_t_1 = lagmatrix (indic_WML.liq_80_05, 1);


var_WML.rf_t_1  = lagmatrix (indic_WML.logrf_80, 1);
var_WML.rf_t_2  = lagmatrix (indic_WML.logrf_80, 2);


gc_WMLvol = fitlm (var_WML, 'WML_t ~ WML_t_1 + vol_t_1 + WML_t_2 +
vol_t_2')
gc_WMLCLI = fitlm (var_WML, 'WML_t ~ WML_t_2 + CLI_t_2 + WML_t_3 +
CLI_t_3 + WML_t_4 + CLI_t_4 + WML_t_5 + CLI_t_5')
gc_WMLliq = fitlm (var_WML, 'WML_t ~ WML_t_1 + liq_t_1')
gc_WMLrf  = fitlm (var_WML, 'WML_t ~ WML_t_1 + rf_t_1 + WML_t_2 +
rf_t_2')
```

```
gc_WML = fitlm (var_WML, 'WML_t ~  CLI_t_2 + CLI_t_3+ CLI_t_4
+CLI_t_5')
anova(gc_WML,'summary')
finv (0.90, 10, 298)
```

SMB

```
var_SMB = table ();
var_SMB.SMB_t   = indic_SMB.logSMB;
var_SMB.SMB_t_1 = lagmatrix (indic_SMB.logSMB, 1);
var_SMB.SMB_t_2 = lagmatrix (indic_SMB.logSMB, 2);
var_SMB.SMB_t_3 = lagmatrix (indic_SMB.logSMB, 3);
var_SMB.SMB_t_4 = lagmatrix (indic_SMB.logSMB, 4);


var_SMB.vol_t_1 = lagmatrix (indic_SMB.vol_80_05, 1);
var_SMB.vol_t_2 = lagmatrix (indic_SMB.vol_80_05, 2);


var_SMB.CLI_t_1 = lagmatrix (indic_SMB.CLI_80_05, 1);
var_SMB.CLI_t_2 = lagmatrix (indic_SMB.CLI_80_05, 2);
var_SMB.CLI_t_3 = lagmatrix (indic_SMB.CLI_80_05, 3);
var_SMB.CLI_t_4 = lagmatrix (indic_SMB.CLI_80_05, 4);


var_SMB.liq_t_1 = lagmatrix (indic_SMB.liq_80_05, 1);


var_SMB.rf_t_1  = lagmatrix (indic_SMB.logrf_80, 1);
var_SMB.rf_t_2  = lagmatrix (indic_SMB.logrf_80, 2);
var_SMB.rf_t_3  = lagmatrix (indic_SMB.logrf_80, 3);
var_SMB.rf_t_4  = lagmatrix (indic_SMB.logrf_80, 4);


gc_SMBvol = fitlm (var_SMB, 'SMB_t ~ SMB_t_1 + vol_t_1 + SMB_t_2 +
vol_t_2')
gc_SMBCLI = fitlm (var_SMB, 'SMB_t ~ SMB_t_2 + CLI_t_2 + SMB_t_3 +
CLI_t_3 + SMB_t_4 + CLI_t_4')
gc_SMBliq = fitlm (var_SMB, 'SMB_t ~ SMB_t_1 + liq_t_1')
gc_SMBrf  = fitlm (var_SMB, 'SMB_t ~ SMB_t_1 + rf_t_1 + SMB_t_2 +
rf_t_2 + SMB_t_3 + rf_t_3 + SMB_t_4 + rf_t_4')


gc_SMB = fitlm (var_SMB, 'SMB_t ~  liq_t_1')
anova(gc_SMB,'summary')
finv (0.90, 13, 298)
```

BAB

```
var_BAB = table ();
var_BAB.BAB_t   = indic_BAB.logBAB;
var_BAB.BAB_t_1 = lagmatrix (indic_BAB.logBAB, 1);
var_BAB.BAB_t_2 = lagmatrix (indic_BAB.logBAB, 2);
var_BAB.BAB_t_3 = lagmatrix (indic_BAB.logBAB, 3);
var_BAB.BAB_t_4 = lagmatrix (indic_BAB.logBAB, 4);
```

```
var_BAB.vol_t_1 = lagmatrix (indic_BAB.vol_85_05, 1);
var_BAB.vol_t_2 = lagmatrix (indic_BAB.vol_85_05, 2);


var_BAB.CLI_t_1 = lagmatrix (indic_BAB.CLI_85_05, 1);
var_BAB.CLI_t_2 = lagmatrix (indic_BAB.CLI_85_05, 2);
var_BAB.CLI_t_3 = lagmatrix (indic_BAB.CLI_85_05, 3);
var_BAB.CLI_t_4 = lagmatrix (indic_BAB.CLI_85_05, 4);


var_BAB.liq_t_1 = lagmatrix (indic_BAB.liq_85_05, 1);
var_BAB.liq_t_2 = lagmatrix (indic_BAB.liq_85_05, 2);
var_BAB.liq_t_3 = lagmatrix (indic_BAB.liq_85_05, 3);
var_BAB.liq_t_4 = lagmatrix (indic_BAB.liq_85_05, 4);


var_BAB.rf_t_1  = lagmatrix (indic_BAB.logrf_85, 1);
var_BAB.rf_t_2  = lagmatrix (indic_BAB.logrf_85, 2);
var_BAB.rf_t_3  = lagmatrix (indic_BAB.logrf_85, 3);


gc_BABvol = fitlm (var_BAB, 'BAB_t ~ BAB_t_1 + vol_t_1 + BAB_t_2 +
vol_t_2')
gc_BABCLI = fitlm (var_BAB, 'BAB_t ~ BAB_t_2 + CLI_t_2 + BAB_t_3 +
CLI_t_3 + BAB_t_4 + CLI_t_4 ')
gc_BABliq = fitlm (var_BAB, 'BAB_t ~ BAB_t_1 + liq_t_1 + BAB_t_2 +
liq_t_2 + BAB_t_3 + liq_t_3 + BAB_t_4 + liq_t_4')
gc_BABrf  = fitlm (var_BAB, 'BAB_t ~ BAB_t_1 + rf_t_1 + BAB_t_2 +
rf_t_2 + BAB_t_3 + rf_t_3')


gc_BAB = fitlm (var_BAB, 'BAB_t ~ CLI_t_2 + CLI_t_3 + CLI_t_4 + liq_t_2
+ liq_t_3 ')
anova(gc_BAB,'summary')
finv (0.90, 11, 236)
```

IN-SAMPLE FACTOR TIMING STRATEGIES

```
clear all;
clc;
warning('off');
```

Monthly market returns

```
mktreturns = readtable ("mktret_rf_monthly_fs.xlsx");
mktret = table2array (mktreturns);
mktret (313:end, :) = [];
rf = mktret (:, end);rf (1:60, :) = [];
```

Factor deciles - monthly returns

```
% Load and Clean Factor Portfolios
indic_HML = load ("HML_indic.mat");  indic_HML = indic_HML.var_HML;
indic_HML (1:48, :) = [];
indic_WML = load ("WML_indic.mat");  indic_WML = indic_WML.var_WML;
indic_WML (1:60, :) = [];
indic_SMB = load ("SMB_indic.mat");  indic_SMB = indic_SMB.var_SMB;
indic_SMB (1:60, :) = [];
indic_BAB = load ("BAB_indic.mat");  indic_BAB = indic_BAB.var_BAB;
facret    = load ("facret.mat");     facret = facret.facret;
```

Rolling window 1 year, 40% UB

```
lb = 0;    % no short sales
ub = 0.4;  % no leverage


s = 12;
l = 252;
HML_coeff = NaN (l+1, 2);
for i = s:l;
    indic_HMLrw = indic_HML (i-11:i, :);
    HML_reg = fitlm (indic_HMLrw, 'HML_t ~ vol_t_1');
    HML_coeff (i+1, :) = table2array (HML_reg.Coefficients (1:2, 1))';
end


WML_coeff = NaN (l+1, 5);
for i = s:l;
    indic_WMLrw = indic_WML (i-11:i, :);
    WML_reg = fitlm (indic_WMLrw,  'WML_t ~  CLI_t_2 + CLI_t_3 +
CLI_t_4 + CLI_t_5');
    WML_coeff (i+1, :) = table2array (WML_reg.Coefficients (1:5, 1))';
end


SMB_coeff = NaN (l+1, 2);
for i = s:l;
    indic_SMBrw = indic_SMB (i-11:i, :);
    SMB_reg = fitlm (indic_SMBrw,  'SMB_t ~ liq_t_1');
    SMB_coeff (i+1, :) = table2array (SMB_reg.Coefficients (1:2, 1))';
end
```

```
BAB_coeff = NaN (l+1, 6);
for i = s:l;
    indic_BABrw = indic_BAB (i-11:i, :);
    BAB_reg = fitlm (indic_BABrw, 'BAB_t ~ CLI_t_2 + CLI_t_3 + CLI_t_4
+ liq_t_2 + liq_t_3');
    BAB_coeff (i+1, :) = table2array (BAB_reg.Coefficients (1:6, 1))';
end


HML_coeff (end, :) = []; WML_coeff (end, :) = []; SMB_coeff (end, :) =
[]; BAB_coeff (end, :) = [];
```

Returns forcasted by indicators

```
HML_estret = HML_coeff (:, 1) + HML_coeff (:, 2).*indic_HML.vol_t_1;
WML_estret = WML_coeff (:, 1) + WML_coeff (:, 2).*indic_WML.CLI_t_2 +
WML_coeff (:, 3).*indic_WML.CLI_t_3 + WML_coeff (:,
4).*indic_WML.CLI_t_4 + WML_coeff (:, 5).*indic_WML.CLI_t_5;
SMB_estret = SMB_coeff (:, 1) + SMB_coeff (:, 2).*indic_SMB.liq_t_1;
BAB_estret = BAB_coeff (:, 1) + BAB_coeff (:, 2).*indic_BAB.CLI_t_2 +
BAB_coeff (:, 3).*indic_BAB.CLI_t_3 + BAB_coeff (:,
4).*indic_BAB.CLI_t_4 + BAB_coeff (:, 5).*indic_BAB.liq_t_2 + BAB_coeff
(:, 6).*indic_BAB.liq_t_3;
```

Mean-Variance Optimization

```
estrets = [HML_estret, WML_estret, SMB_estret, BAB_estret];
ret_m   = estrets * 12;
rf_m_rw = rf .*12;


% Covariance Matrix & mean-variance optimization
mv_W    = NaN (l,4);
mv_std  = NaN (l,1);
mv_mu   = NaN (l,1);
mv_exmu = NaN (l,1);
mv_SR   = NaN (l,1);
for i = s+1:l
    cov_mat_rw = cov (facret (i-12:i-1, 1:4) - rf (i-12:i-1, 1))*12;
    trgt_mean_rw = min (ret_m):0.05:max (ret_m);
    [W_wo, trgt_std_wo, n_trgt_mean_wo] = port_meanvar (ret_m (i,:),
cov_mat_rw, trgt_mean_rw, [], lb, ub);
    SR = n_trgt_mean_wo./trgt_std_wo;
    %
    mv_SR (i)   = max (SR);
    mv_i        = find (mv_SR(i) == SR);
    mv_W (i, :) = W_wo (:, mv_i);
    mv_std (i)  = trgt_std_wo (mv_i);
    mv_mu (i)   = n_trgt_mean_wo (mv_i);
    mv_exmu (i) = n_trgt_mean_wo (mv_i)-rf_m_rw (i);
end
```

```
returns = mv_W.*facret;
for i = 1:length(returns);
    agg_ret40 (i, 1) = sum (returns (i, :));
end


exret_rw40 = agg_ret40-rf;


% Performance measures
mean_factim_rw40 = nanmean (exret_rw40(61:end))*12
std_factim_rw40 = nanstd (exret_rw40(61:end))*sqrt(12)
SR_factim_rw40 = mean_factim_rw40/std_factim_rw40
arit_mean_factim_rw40 = nanmean (agg_ret40(61:end))*12;
skew_fac_rw40 = skewness (agg_ret40(61:end))
kurt_fac_rw40 = kurtosis (agg_ret40(61:end))
```

Recursive window 1 year, 40% UB

```
s = 12;
l = 252;
HML_coeff = NaN (l+1, 2);
for i = s:l;
    indic_HMLrw = indic_HML (1:i, :);
    HML_reg = fitlm (indic_HMLrw, 'HML_t ~ vol_t_1');
    HML_coeff (i+1, :) = table2array (HML_reg.Coefficients (1:2, 1))';
end


WML_coeff = NaN (l+1, 5);
for i = s:l;
    indic_WMLrw = indic_WML (1:i, :);
    WML_reg = fitlm (indic_WMLrw,  'WML_t ~  CLI_t_2 + CLI_t_3 +
CLI_t_4 + CLI_t_5');
    WML_coeff (i+1, :) = table2array (WML_reg.Coefficients (1:5, 1))';
end


SMB_coeff = NaN (l+1, 2);
for i = s:l;
    indic_SMBrw = indic_SMB (1:i, :);
    SMB_reg = fitlm (indic_SMBrw,  'SMB_t ~ liq_t_1');
    SMB_coeff (i+1, :) = table2array (SMB_reg.Coefficients (1:2, 1))';
end


BAB_coeff = NaN (l+1, 6);
for i = s:l;
    indic_BABrw = indic_BAB (1:i, :);
    BAB_reg = fitlm (indic_BABrw, 'BAB_t ~ CLI_t_2 + CLI_t_3 + CLI_t_4
+ liq_t_2 + liq_t_3');
    BAB_coeff (i+1, :) = table2array (BAB_reg.Coefficients (1:6, 1))';
end
```

```
HML_coeff (end, :) = []; WML_coeff (end, :) = []; SMB_coeff (end, :) =
[]; BAB_coeff (end, :) = [];
```

Returns forcasted by indicators

```
HML_estret = HML_coeff (:, 1) + HML_coeff (:, 2).*indic_HML.vol_t_1;
WML_estret = WML_coeff (:, 1) + WML_coeff (:, 2).*indic_WML.CLI_t_2 +
WML_coeff (:, 3).*indic_WML.CLI_t_3 + WML_coeff (:,
4).*indic_WML.CLI_t_4 + WML_coeff (:, 5).*indic_WML.CLI_t_5;
SMB_estret = SMB_coeff (:, 1) + SMB_coeff (:, 2).*indic_SMB.liq_t_1;
BAB_estret = BAB_coeff (:, 1) + BAB_coeff (:, 2).*indic_BAB.CLI_t_2 +
BAB_coeff (:, 3).*indic_BAB.CLI_t_3 + BAB_coeff (:,
4).*indic_BAB.CLI_t_4 + BAB_coeff (:, 5).*indic_BAB.liq_t_2 + BAB_coeff
(:, 6).*indic_BAB.liq_t_3;
```

Mean-Variance Optimization

```
estrets = [HML_estret, WML_estret, SMB_estret, BAB_estret];
ret_m   = estrets * 12;
rf_m_rw = rf .*12;


% Covariance Matrix & mean-variance optimization
mv_W    = NaN (l,4);
mv_std  = NaN (l,1);
mv_mu   = NaN (l,1);
mv_exmu = NaN (l,1);
mv_SR   = NaN (l,1);
for i = s+1:l
    cov_mat_rw = cov (facret (1:i-1, 1:4) - rf (1:i-1, 1))*12;
    trgt_mean_rw = min (ret_m):0.05:max (ret_m);
    [W_wo, trgt_std_wo, n_trgt_mean_wo] = port_meanvar (ret_m (i,:),
cov_mat_rw, trgt_mean_rw, [], lb, ub);
    SR = n_trgt_mean_wo./trgt_std_wo;
    %
    mv_SR (i)   = max (SR);
    mv_i        = find (mv_SR(i) == SR);
    mv_W (i, :) = W_wo (:, mv_i);
    mv_std (i)  = trgt_std_wo (mv_i);
    mv_mu (i)   = n_trgt_mean_wo (mv_i);
    mv_exmu (i) = n_trgt_mean_wo (mv_i)-rf_m_rw (i);
end


returns = mv_W.*facret;
for i = 1:length(returns);
    agg_ret_rec1_40 (i, 1) = sum (returns (i, :));
end


exret_rec1_40 = agg_ret_rec1_40-rf;


% Performance measures
mean_factim_rec1_40 = nanmean (exret_rec1_40(61:end))*12
```

```
std_factim_rec1_40 = nanstd (exret_rec1_40(61:end))*sqrt(12)
SR_factim_rec40 = mean_factim_rec1_40/std_factim_rec1_40
arit_mean_factim_rec1_40 = nanmean (agg_ret_rec1_40(61:end))*12;
skew_fac_rec1_40 = skewness (agg_ret_rec1_40(61:end))
kurt_fac_rec1_40 = kurtosis (agg_ret_rec1_40(61:end))
```

Rolling window 3 year, 40% UB

```
s = 36;
l = 252;
HML_coeff = NaN (l+1, 2);
for i = s:l;
    indic_HMLrw = indic_HML (i-35:i, :);
    HML_reg = fitlm (indic_HMLrw, 'HML_t ~ vol_t_1');
    HML_coeff (i+1, :) = table2array (HML_reg.Coefficients (1:2, 1))';
end


WML_coeff = NaN (l+1, 5);
for i = s:l;
    indic_WMLrw = indic_WML (i-35:i, :);
    WML_reg = fitlm (indic_WMLrw,  'WML_t ~  CLI_t_2 + CLI_t_3 +
CLI_t_4 + CLI_t_5');
    WML_coeff (i+1, :) = table2array (WML_reg.Coefficients (1:5, 1))';
end


SMB_coeff = NaN (l+1, 2);
for i = s:l;
    indic_SMBrw = indic_SMB (i-35:i, :);
    SMB_reg = fitlm (indic_SMBrw,  'SMB_t ~ liq_t_1');
    SMB_coeff (i+1, :) = table2array (SMB_reg.Coefficients (1:2, 1))';
end


BAB_coeff = NaN (l+1, 6);
for i = s:l;
    indic_BABrw = indic_BAB (i-35:i, :);
    BAB_reg = fitlm (indic_BABrw, 'BAB_t ~ CLI_t_2 + CLI_t_3 + CLI_t_4
+ liq_t_2 + liq_t_3');
    BAB_coeff (i+1, :) = table2array (BAB_reg.Coefficients (1:6, 1))';
end


HML_coeff (end, :) = []; WML_coeff (end, :) = []; SMB_coeff (end, :) =
[]; BAB_coeff (end, :) = [];
```

Returns forcasted by indicators

```
HML_estret = HML_coeff (:, 1) + HML_coeff (:, 2).*indic_HML.vol_t_1;
WML_estret = WML_coeff (:, 1) + WML_coeff (:, 2).*indic_WML.CLI_t_2 +
WML_coeff (:, 3).*indic_WML.CLI_t_3 + WML_coeff (:,
4).*indic_WML.CLI_t_4 + WML_coeff (:, 5).*indic_WML.CLI_t_5;
SMB_estret = SMB_coeff (:, 1) + SMB_coeff (:, 2).*indic_SMB.liq_t_1;
```

```
BAB_estret = BAB_coeff (:, 1) + BAB_coeff (:, 2).*indic_BAB.CLI_t_2 +
BAB_coeff (:, 3).*indic_BAB.CLI_t_3 + BAB_coeff (:,
4).*indic_BAB.CLI_t_4 + BAB_coeff (:, 5).*indic_BAB.liq_t_2 + BAB_coeff
(:, 6).*indic_BAB.liq_t_3;
```

Mean-Variance Optimization

```
estrets = [HML_estret, WML_estret, SMB_estret, BAB_estret];
ret_m   = estrets * 12;
rf_m_rw = rf .*12;


% Covariance Matrix & mean-variance optimization
mv_W    = NaN (l,4);
mv_std  = NaN (l,1);
mv_mu   = NaN (l,1);
mv_exmu = NaN (l,1);
mv_SR   = NaN (l,1);
for i = s+1:l
    cov_mat_rw = cov (facret (i-36:i-1, 1:4) - rf (i-36:i-1, 1))*12;
    trgt_mean_rw = min (ret_m):0.05:max (ret_m);
    [W_wo, trgt_std_wo, n_trgt_mean_wo] = port_meanvar (ret_m (i,:),
cov_mat_rw, trgt_mean_rw, [], lb, ub);
    SR = n_trgt_mean_wo./trgt_std_wo;
    %
    mv_SR (i)   = max (SR);
    mv_i        = find (mv_SR(i) == SR);
    mv_W (i, :) = W_wo (:, mv_i);
    mv_std (i)  = trgt_std_wo (mv_i);
    mv_mu (i)   = n_trgt_mean_wo (mv_i);
    mv_exmu (i) = n_trgt_mean_wo (mv_i)-rf_m_rw (i);
end


returns = mv_W.*facret;
for i = 1:length(returns);
    agg_ret_rw3_40 (i, 1) = sum (returns (i, :));
end


exret_rw3_40 = agg_ret_rw3_40-rf;


% Performance measures
mean_factim_rw3_40 = nanmean (exret_rw3_40(61:end))*12
std_factim_rw3_40 = nanstd (exret_rw3_40(61:end))*sqrt(12)
SR_factim_rw3_40 = mean_factim_rw3_40/std_factim_rw3_40
arit_mean_factim_rw3_40 = nanmean (agg_ret_rw3_40(61:end))*12;
skew_fac_rw3_40 = skewness (agg_ret_rw3_40(61:end))
kurt_fac_rw3_40 = kurtosis (agg_ret_rw3_40(61:end))
```

Rolling window 5 year, 40% UB

```
s = 60;
l = 252;
```

```
HML_coeff = NaN (l+1, 2);
for i = s:l;
    indic_HMLrw = indic_HML (i-59:i, :);
    HML_reg = fitlm (indic_HMLrw, 'HML_t ~ vol_t_1');
    HML_coeff (i+1, :) = table2array (HML_reg.Coefficients (1:2, 1))';
end


WML_coeff = NaN (l+1, 5);
for i = s:l;
    indic_WMLrw = indic_WML (i-59:i, :);
    WML_reg = fitlm (indic_WMLrw,  'WML_t ~  CLI_t_2 + CLI_t_3 +
CLI_t_4 + CLI_t_5');
    WML_coeff (i+1, :) = table2array (WML_reg.Coefficients (1:5, 1))';
end


SMB_coeff = NaN (l+1, 2);
for i = s:l;
    indic_SMBrw = indic_SMB (i-59:i, :);
    SMB_reg = fitlm (indic_SMBrw,  'SMB_t ~ liq_t_1');
    SMB_coeff (i+1, :) = table2array (SMB_reg.Coefficients (1:2, 1))';
end


BAB_coeff = NaN (l+1, 6);
for i = s:l;
    indic_BABrw = indic_BAB (i-59:i, :);
    BAB_reg = fitlm (indic_BABrw, 'BAB_t ~ CLI_t_2 + CLI_t_3 + CLI_t_4
+ liq_t_2 + liq_t_3');
    BAB_coeff (i+1, :) = table2array (BAB_reg.Coefficients (1:6, 1))';
end


HML_coeff (end, :) = []; WML_coeff (end, :) = []; SMB_coeff (end, :) =
[]; BAB_coeff (end, :) = [];
```

Returns forcasted by indicators

```
HML_estret = HML_coeff (:, 1) + HML_coeff (:, 2).*indic_HML.vol_t_1;
WML_estret = WML_coeff (:, 1) + WML_coeff (:, 2).*indic_WML.CLI_t_2 +
WML_coeff (:, 3).*indic_WML.CLI_t_3 + WML_coeff (:,
4).*indic_WML.CLI_t_4 + WML_coeff (:, 5).*indic_WML.CLI_t_5;
SMB_estret = SMB_coeff (:, 1) + SMB_coeff (:, 2).*indic_SMB.liq_t_1;
BAB_estret = BAB_coeff (:, 1) + BAB_coeff (:, 2).*indic_BAB.CLI_t_2 +
BAB_coeff (:, 3).*indic_BAB.CLI_t_3 + BAB_coeff (:,
4).*indic_BAB.CLI_t_4 + BAB_coeff (:, 5).*indic_BAB.liq_t_2 + BAB_coeff
(:, 6).*indic_BAB.liq_t_3;
```

Mean-Variance Optimization

```
estrets = [HML_estret, WML_estret, SMB_estret, BAB_estret];
ret_m   = estrets * 12;
rf_m_rw = rf .*12;
```

```
% Covariance Matrix & mean-variance optimization
mv_W    = NaN (l,4);
mv_std  = NaN (l,1);
mv_mu   = NaN (l,1);
mv_exmu = NaN (l,1);
mv_SR   = NaN (l,1);
for i = s+1:l
    cov_mat_rw = cov (facret (i-60:i-1, 1:4) - rf (i-60:i-1, 1))*12;
    trgt_mean_rw = min (ret_m):0.05:max (ret_m);
    [W_wo, trgt_std_wo, n_trgt_mean_wo] = port_meanvar (ret_m (i,:),
cov_mat_rw, trgt_mean_rw, [], lb, ub);
    SR = n_trgt_mean_wo./trgt_std_wo;
    %
    mv_SR (i)   = max (SR);
    mv_i        = find (mv_SR(i) == SR);
    mv_W (i, :) = W_wo (:, mv_i);
    mv_std (i)  = trgt_std_wo (mv_i);
    mv_mu (i)   = n_trgt_mean_wo (mv_i);
    mv_exmu (i) = n_trgt_mean_wo (mv_i)-rf_m_rw (i);
end


returns = mv_W.*facret;
for i = 1:length(returns);
    agg_ret_rw5_40 (i, 1) = sum (returns (i, :));
end


exret_rw5_40 = agg_ret_rw5_40-rf;


% Performance measures
mean_factim_rw5_40 = nanmean (exret_rw5_40(61:end))*12
std_factim_rw5_40 = nanstd (exret_rw5_40(61:end))*sqrt(12)
SR_factim_rw5_40 = mean_factim_rw5_40/std_factim_rw5_40
arit_mean_factim_rw5_40 = nanmean (agg_ret_rw5_40(61:end))*12;
skew_fac_rw5_40 = skewness (agg_ret_rw5_40(61:end))
kurt_fac_rw5_40 = kurtosis (agg_ret_rw5_40(61:end))
```

IN-SAMPLE WEIGHTS

```
clear all;
clc;
load ("IS-HML-WML-SMB-BAB-OSEAX-EW-MV-FT-rf.mat")
rets = tbl;


W = mv_W (61:end,:);
W = mv_W_rebal(61:end,:);




startDate = datenum('01-31-1990');
endDate   = datenum('12-31-2005');
xData     = linspace(startDate,endDate,192);


%%HML
figure;
yyaxis left
bar (xData, W(:,1),'FaceColor',[0.6 0.6 0.6])
yyaxis right
plot (xData, cumsum(rets(:,2)), 'LineWidth',2.3, "Color",[0 0.4470
0.7410])
datetick ('x', 'yy')
legend   ('Weight','HML', 'location', 'northwest')


%% WML
figure;
yyaxis left
bar (xData, W(:,2),'FaceColor',[0.6 0.6 0.6])
yyaxis right
plot (xData, cumsum(rets(:,3)), 'LineWidth',2.3, "Color",[0.8500 0.3250
0.0980])
datetick ('x', 'yy')
legend   ('Weight','WML', 'location', 'northwest')


%% SMB
figure;
yyaxis left
bar (xData, W(:,3),'FaceColor',[0.6 0.6 0.6])
yyaxis right
plot (xData, cumsum(rets(:,4)), 'LineWidth',2.3, "Color",[0.9290 0.6940
0.1250])
datetick ('x', 'yy')
legend   ('Weight','SMB', 'location', 'northwest')


%BAB
figure;
```

```
yyaxis left
bar (xData, W(:,4),'FaceColor',[0.6 0.6 0.6])
yyaxis right
plot (xData, cumsum(rets(:,5)), 'LineWidth',2.3, "Color",[0.4940 0.1840
0.5560])
datetick ('x', 'yy')
legend   ('Weight','BAB', 'location', 'northwest')
```

IN-SAMPLE SUBSAMPLES

```
excess = NaN (192,4);
for i = 1:192
    for j = 6:9
        excess(i,j-5) = tbl (i,j)-tbl(i,10);
    end
end
% 4 periods
subsam1     = tbl (1:48,    6:9);
subsam2     = tbl (49:96,   6:9);
subsam3     = tbl (97:144,  6:9);
subsam4     = tbl (145:end, 6:9);
subsample1 = excess (1:48,    :);
subsample2 = excess (49:96,   :);
subsample3 = excess (97:144,  :);
subsample4 = excess (145:end, :);


rf1 = mean (tbl(1:48,10))*12;
rf2 = mean (tbl(49:96,10))*12;
rf3 = mean (tbl(97:144,10))*12;
rf4 = mean (tbl(145:end,10))*12;


figure
plot (tbl(:,10))
```

Mean, excess mean, standard deviation, Sharpe ratio

```
means1     = mean (subsam1)*12;
means2     = mean (subsam2)*12;
means3     = mean (subsam3)*12;
means4     = mean (subsam4)*12;
exmeans1     = mean (subsample1)*12;
exmeans2     = mean (subsample2)*12;
exmeans3     = mean (subsample3)*12;
exmeans4     = mean (subsample4)*12;
stds1      = std (subsample1)*sqrt(12);
stds2      = std (subsample2)*sqrt(12);
stds3      = std (subsample3)*sqrt(12);
stds4      = std (subsample4)*sqrt(12);
SRs1       = exmeans1./stds1
SRs2       = exmeans2./stds2
SRs3       = exmeans3./stds3
SRs4       = exmeans4./stds4
```

M2

```
for i = 1:4
    M2_1(i) = (stds1(1,1)/stds1(1,i))*exmeans1(1,i)+rf1;
end
for i = 1:4
```

```
    M2_2(i) = (stds2(1,1)/stds2(1,i))*exmeans2(1,i)+rf2;
end
for i = 1:4
    M2_3(i) = (stds3(1,1)/stds3(1,i))*exmeans3(1,i)+rf3;
end
for i = 1:4
    M2_4(i) = (stds4(1,1)/stds4(1,i))*exmeans4(1,i)+rf4;
end
```

Skewness & Kurtosis

```
skew1 = skewness(subsam1);
skew2 = skewness(subsam2);
skew3 = skewness(subsam3);
skew4 = skewness(subsam4);


kurt1 = kurtosis(subsam1);
kurt2 = kurtosis(subsam2);
kurt3 = kurtosis(subsam3);
kurt4 = kurtosis(subsam4);
```

Information Ratio

```
for i = 1:4
IR1 (i) = (means1(1,i)-means1(1,1))/(std(subsam1(:,i)-
subsam1(:,1))*sqrt(12));
end
for i = 1:4
IR2 (i) = (means2(1,i)-means2(1,1))/(std(subsam2(:,i)-
subsam2(:,1))*sqrt(12));
end
for i = 1:4
IR3 (i) = (means3(1,i)-means3(1,1))/(std(subsam3(:,i)-
subsam3(:,1))*sqrt(12));
end
for i = 1:4
IR4 (i) = (means4(1,i)-means4(1,1))/(std(subsam4(:,i)-
subsam4(:,1))*sqrt(12));
end
```

```
SS1cs = cumsum (subsam1 (:, 1:4));
figure;
startDate = datenum('01-31-1990');
endDate   = datenum('12-31-1993');
xData     = linspace(startDate,endDate,48);
plot     (xData, SS1cs, 'LineWidth', 1.7);
datetick ('x', 'yyyy')
legend   ('OSEAX', 'EW', 'MV', 'FT', 'location', 'northwest')
ylabel   ('cumret');


SS2cs = cumsum (subsam2 (:, 1:4));
figure;
```

```
startDate = datenum('01-31-1994');
endDate   = datenum('12-31-1997');
xData     = linspace(startDate,endDate,48);
plot     (xData, SS2cs, 'LineWidth', 1.7);
datetick ('x', 'yyyy')
ylabel   ('cumret');


SS3cs = cumsum (subsam3 (:, 1:4));
figure;
startDate = datenum('01-31-1998');
endDate   = datenum('12-31-2001');
xData     = linspace(startDate,endDate,48);
plot     (xData, SS3cs, 'LineWidth', 1.7);
datetick ('x', 'yyyy')
ylabel   ('cumret');


SS4cs = cumsum (subsam4 (:, 1:4));
figure;
startDate = datenum('01-31-2002');
endDate   = datenum('12-31-2005');
xData     = linspace(startDate,endDate,48);
plot     (xData, SS4cs, 'LineWidth', 1.7);
datetick ('x', 'yyyy')
ylabel   ('cumret');
```

OUT-OF-SAMPLE EW MULTIFACTOR STRATEGY

```
clc;
clear all;
warning('off')
```

LOAD AND CLEAN FACTOR PORTFOLIOS AND MARKET RETURNS

```
EW_btm_table = readtable ('ew_monthly_btm_fs.xlsx');
EW_mom_table = readtable ('ew_monthly_mom_fs.xlsx');
EW_siz_table = readtable ('ew_monthly_siz_fs.xlsx');
BW_bet_table = load      ("BettingAgainstBeta.mat");
BW_bet_table = BW_bet_table.BaBtable;
market = readtable ("mktret_rf_monthly_fs.xlsx");
mktret = table2array (market);
rf = mktret (313:480, end);
%
% In-sample period Jan 2006 - Dec 2019
btm = table2array (EW_btm_table (301:468, 2:end));
mom = table2array (EW_mom_table (313:480, 2:end));
siz = table2array (EW_siz_table (313:480, 2:end));
bet = table2array (BW_bet_table (253:end, :    ));
mkt = mktret (313:480, 4);


% Quintile returns
HML = btm (:, 10) + btm (:, 9) - btm (:, 1) - btm (:,  2);
WML = mom (:, 10) + mom (:, 9) - mom (:, 1) - mom (:,  2);
SMB = siz (:,  1) + siz (:, 2) - siz (:, 9) - siz (:, 10);
BAB = bet (:, 1) - bet (:, 2);


% Return matrix
retmat = [mkt, HML, WML, SMB, BAB];
HMLcs = cumsum (HML);
WMLcs = cumsum (WML);
SMBcs = cumsum (SMB);
BABcs = cumsum (BAB);
mktcs = cumsum (mkt);


plot (HMLcs);
hold on;
plot (WMLcs);
plot (SMBcs);
plot (BABcs);
plot (mktcs);
% plot (MFMcs);
hold off;
title ('Cumulative Returns')
legend ('HML', 'WML', 'SMB', 'BAB', 'mkt', 'MFM')
```

Descriptive Statistics

```
% Correlations Matrix
corrmat = corr (retmat);
corrmat = round (corrmat, 4);
```

```
corrtbl = array2table (corrmat);
corrtbl.Properties.VariableNames = {'mkt', 'HML', 'WML', 'SMB', 'BAB'};
corrtbl.Properties.RowNames = {'mkt', 'HML', 'WML', 'SMB', 'BAB'};
disp (corrtbl)


% Returns, StDev, SR
returns = mean (retmat) * 12;
excess_rets = retmat - rf;
excess_means = mean (excess_rets) * 12;
stdev = std (retmat) * sqrt(12);
SR = excess_means ./ stdev;


% tabulate results
TD = round([returns; excess_means; stdev; SR;], 3);
Tbl = table(TD(:,1),TD(:,2),TD(:,3),TD(:,4), TD(:, 5));
Tbl.Properties.VariableNames = {'Mkt','HML','WML','SMB', 'BAB'};
Tbl.Properties.RowNames =  {'Ret','Ex_ret','StDev','SR'};
disp(Tbl)
```

Equal-weighted static multifactor model

```
n = 4;
MFM = 1/n * (HML + WML + SMB + BAB);
MFMcomp = [HML, WML, SMB, BAB, MFM];
M_returns = mean (MFMcomp) * 12;
M_exret = MFMcomp - rf;
M_exmean = mean (M_exret) * 12;
M_std = std (MFMcomp) * sqrt(12);
M_SR = M_exmean ./ M_std;


TD = round([M_returns; M_exmean; M_std; M_SR;], 3);
Tbl = table(TD(:,1),TD(:,2),TD(:,3),TD(:,4), TD(:, 5));
Tbl.Properties.VariableNames = {'HML','WML','SMB', 'BAB', 'MFM'};
Tbl.Properties.RowNames =  {'Ret','Ex_ret','StDev','SR'};
disp(Tbl)
```

OUT-OF-SAMPLE MEAN-VARIANCE PORTFOLIO CONSTRUCTION

```
clear all;
clc;
warning('off');
```

Monthly market returns

```
mktreturns = readtable ("mktret_rf_monthly_fs.xlsx");
mktret = table2array (mktreturns);
mktret (1:276, :) = [];mktret (205:end, :) = [];
rf = mktret (:, end);
```

Factor deciles - monthly returns

```
EW_btm_table = readtable ('ew_monthly_btm_fs.xlsx');
EW_mom_table = readtable ('ew_monthly_mom_fs.xlsx');
EW_siz_table = readtable ('ew_monthly_siz_fs.xlsx');
BW_bet_table = load      ("BettingAgainstBeta.mat");
BW_bet_table = BW_bet_table.BaBtable;


% Out-of-sample period Jan 2005 - Dec 2019
btm = table2array (EW_btm_table (265:468, 2:end));
mom = table2array (EW_mom_table (277:480, 2:end));
siz = table2array (EW_siz_table (277:480, 2:end));
bet = table2array (BW_bet_table (217:end, :    ));


% Quintile returns
HML = btm (:, 10) + btm (:, 9) - btm (:, 1) - btm (:,  2);
WML = mom (:, 10) + mom (:, 9) - mom (:, 1) - mom (:,  2);
SMB = siz (:,  1) + siz (:, 2) - siz (:, 9) - siz (:, 10);
BAB = bet (:, 1)  - bet (:, 2);


% Return matrix
facret = [HML, WML, SMB, BAB];
```

40% UB

```
lb = 0; % no short sales
ub = 0.40; %  no leverage
```

Mean-Variance Rolling Window (Monthly rebalancing)

```
s = 36;
l = 204;


% Return Vector
% 1985-2005: 5-year rolling window
ret_m_rebal = NaN (l, 4);
ex_ret_m_rebal = NaN (l, 4);
```

```
rf_m_rw_rebal  = NaN (l, 1);
for i = s:l
    for j = 1:4
        ret_m_rebal (i, j) = mean (facret (i-35:i, j)) * 12;
        rf_m_rw_rebal (i) = mean (rf (i-35:i))*12;
        ex_ret_m_rebal (i, j) = ret_m_rebal(i,j)-rf_m_rw_rebal(i);
    end
end


% Covariance Matrix
% Ex rets
mv_W_rebal = NaN (l,4);
mv_std_rebal = NaN (l,1);
mv_mu_rebal = NaN (l,1);
mv_exmu_rebal = NaN (l,1);
mv_SR_rebal = NaN (l,1);
for i = s:l
    cov_mat_rw = cov(facret (i-35:i, 1:4) - rf (i-35:i, 1))*12;
    trgt_mean_rw = min (ret_m_rebal (i, :)):0.05:max(ret_m_rebal (i,
:))+0.02;
    [W_wo, trgt_std_wo, n_trgt_mean_wo] = port_meanvar (ret_m_rebal
(i,:), cov_mat_rw, trgt_mean_rw, [], lb, ub);
    %
    SR = n_trgt_mean_wo./trgt_std_wo;
    mv_SR_rebal (i) = max (SR);
    mv_i = find (mv_SR_rebal(i) == SR);
    mv_W_rebal(i,:) = W_wo (:, mv_i);
    mv_std_rebal(i) = trgt_std_wo (mv_i);
    mv_mu_rebal(i) = n_trgt_mean_wo (mv_i);
    mv_exmu_rebal(i) = n_trgt_mean_wo (mv_i)-rf_m_rw_rebal (i);
end


returns_rw_rebal = NaN (l,4);
for i = 2:length(returns_rw_rebal);
    for j = 1:4
    returns_rw_rebal (i,j) = mv_W_rebal(i-1,j)*facret(i,j);
    end
end


for i = 1:length(returns_rw_rebal);
    agg_ret_mv (i, 1) = sum (returns_rw_rebal (i, :));
end


exret_mv = agg_ret_mv-rf;


mean_mv = nanmean (exret_mv)*12
std_mv = nanstd (exret_mv)*sqrt(12)
SR_mv = mean_mv/std_mv
```

```
arit_mean_mv = nanmean (agg_ret_mv)*12;
skew_mv = skewness (agg_ret_mv)
kurt_mv = kurtosis (agg_ret_mv)
pos_mv  = sum (agg_ret_mv>0);
neg_mv  = sum (agg_ret_mv<0);
```

OUT-OF-SAMPLE FACTOR TIMING PORTFOLIO CONSTRUCTION

```
clear all;
clc;
warning('off');
```

Monthly market returns

```
mktreturns = readtable ("mktret_rf_monthly_fs.xlsx");
mktret = table2array (mktreturns);
mktret (1:300, :) = [];mktret (181:end, :) = [];
rf = mktret (:, end);
```

Factor deciles - monthly returns

```
% Load and Clean Factor Portfolios
indic  = load ("indic1985_2019.mat"); indic = indic.indic_lags;
indic(1:240,:) = [];
facret = load ("facret1985_2019.mat"); facret = facret.facret;
facret(1:240,:) = [];
```

40% UB

```
lb = 0;     % no short sales
ub = 0.4;  % no leverage
```

Rolling Regressions

```
s = 12;
l = 180;
HML_coeff = NaN (l+1, 2);
for i = s:l;
    indic_HMLrw = indic (i-11:i, :);
    HML_reg = fitlm (indic_HMLrw, 'HML_t ~ vol_t_1');
    HML_coeff (i+1, :) = table2array (HML_reg.Coefficients (1:2, 1))';
end


WML_coeff = NaN (l+1, 5);
for i = s:l;
    indic_WMLrw = indic (i-11:i, :);
    WML_reg = fitlm (indic_WMLrw,  'WML_t ~  CLI_t_2 + CLI_t_3 +
CLI_t_4 + CLI_t_5');
    WML_coeff (i+1, :) = table2array (WML_reg.Coefficients (1:5, 1))';
end


SMB_coeff = NaN (l+1, 2);
for i = s:l;
    indic_SMBrw = indic (i-11:i, :);
    SMB_reg = fitlm (indic_SMBrw,  'SMB_t ~ liq_t_1');
    SMB_coeff (i+1, :) = table2array (SMB_reg.Coefficients (1:2, 1))';
end
```

```
BAB_coeff = NaN (l+1, 6);
for i = s:l;
    indic_BABrw = indic (i-11:i, :);
    BAB_reg = fitlm (indic_BABrw, 'BAB_t ~ CLI_t_2 + CLI_t_3 + CLI_t_4
+ liq_t_2 + liq_t_3');
    BAB_coeff (i+1, :) = table2array (BAB_reg.Coefficients (1:6, 1))';
end


HML_coeff (end, :) = []; WML_coeff (end, :) = []; SMB_coeff (end, :) =
[]; BAB_coeff (end, :) = [];
```

HML returns forcasted on indicators

```
HML_estret = HML_coeff (:, 1) + HML_coeff (:, 2).*indic.vol_t_1;
WML_estret = WML_coeff (:, 1) + WML_coeff (:, 2).*indic.CLI_t_2 +
WML_coeff (:, 3).*indic.CLI_t_3 + WML_coeff (:, 4).*indic.CLI_t_4 +
WML_coeff (:, 5).*indic.CLI_t_5;
SMB_estret = SMB_coeff (:, 1) + SMB_coeff (:, 2).*indic.liq_t_1;
BAB_estret = BAB_coeff (:, 1) + BAB_coeff (:, 2).*indic.CLI_t_2 +
BAB_coeff (:, 3).*indic.CLI_t_3 + BAB_coeff (:, 4).*indic.CLI_t_4 +
BAB_coeff (:, 5).*indic.liq_t_2 + BAB_coeff (:, 6).*indic.liq_t_3;
```

Mean-Variance Optimization

```
estrets = [HML_estret, WML_estret, SMB_estret, BAB_estret];
ret_m   = estrets * 12;
rf_m_rw = rf .*12;


% Covariance Matrix
% Ex rets
mv_W    = NaN (l,4);
mv_std  = NaN (l,1);
mv_mu   = NaN (l,1);
mv_exmu = NaN (l,1);
mv_SR   = NaN (l,1);
for i = s+1:l
    cov_mat_rw = cov (facret (i-12:i-1, 1:4) - rf (i-12:i-1, 1))*12;
    trgt_mean_rw = min (ret_m):0.05:max (ret_m);
    [W_wo, trgt_std_wo, n_trgt_mean_wo] = port_meanvar (ret_m (i,:),
cov_mat_rw, trgt_mean_rw, [], lb, ub);
    SR = n_trgt_mean_wo./trgt_std_wo;
    %
    mv_SR (i)   = max (SR);
    mv_i        = find (mv_SR(i) == SR);
    mv_W (i, :) = W_wo (:, mv_i);
    mv_std (i)  = trgt_std_wo (mv_i);
    mv_mu (i)   = n_trgt_mean_wo (mv_i);
    mv_exmu (i) = n_trgt_mean_wo (mv_i)-rf_m_rw (i);
end
```

```
returns = mv_W.*facret;
for i = 1:length(returns);
    agg_ret_ft (i, 1) = sum (returns (i, :));
end


exret_ft = agg_ret_ft-rf;


arit_mean_factim = nanmean (agg_ret_ft)*12;
skew_fac = skewness (agg_ret_ft)
kurt_fac = kurtosis (agg_ret_ft)
pos_fac  = sum (agg_ret_ft>0);
neg_fac  = sum (agg_ret_ft<0);


mean_factim = nanmean (exret_ft)*12
std_factim = nanstd (exret_ft)*sqrt(12)
SR_factim = mean_factim/std_factim
```

OUT-OF-SAMPLE WEIGHTS

```
clear all;
clc;
load ("OS-HML-WML-SMB-BAB-OSEAC-EW-MV-FT-RF.mat");


W = mv_W (13:end,:);
W = mv_W_rebal(37:end,:);




startDate = datenum('01-31-2006');
endDate   = datenum('12-31-2019');
xData     = linspace(startDate,endDate,168);


%%HML
figure;
yyaxis left
bar (xData, W(:,1),'FaceColor',[0.6 0.6 0.6])
yyaxis right
plot (xData, cumsum(rets(:,2)), 'LineWidth',2.3, "Color",[0 0.4470
0.7410])
datetick ('x', 'yy')
legend   ('Weight','HML', 'location', 'northwest')


%% WML
figure;
yyaxis left
bar (xData, W(:,2),'FaceColor',[0.6 0.6 0.6])
yyaxis right
plot (xData, cumsum(rets(:,3)), 'LineWidth',2.3, "Color",[0.8500 0.3250
0.0980])
datetick ('x', 'yy')
legend   ('Weight','WML', 'location', 'northwest')


%% SMB
figure;
yyaxis left
bar (xData, W(:,3),'FaceColor',[0.6 0.6 0.6])
yyaxis right
plot (xData, cumsum(rets(:,4)), 'LineWidth',2.3, "Color",[0.9290 0.6940
0.1250])
datetick ('x', 'yy')
legend   ('Weight','SMB', 'location', 'northwest')


%BAB
figure;
yyaxis left
```

```
bar (xData, W(:,4),'FaceColor',[0.6 0.6 0.6])
yyaxis right
plot (xData, cumsum(rets(:,5)), 'LineWidth',2.3, "Color",[0.4940 0.1840
0.5560])
datetick ('x', 'yy')
legend   ('Weight','BAB', 'location', 'northwest')
```

OUT-OF-SAMPLE SUBSAMPLES

```
clear all;
clc;
load ("OS-HML-WML-SMB-BAB-OSEAC-EW-MV-FT-RF.mat");


excess = NaN (168,4);
for i = 1:168
    for j = 6:9
        excess(i,j-5) = rets (i,j)-rets(i,10);
    end
end
```

```
% out-of-sample
ret = NaN (168,2);
ret (:,1) = excess (:,4);
ret (:,2) = excess (:,3);


% after the financial crisis
ret1 = NaN (130,2);
ret1 (:,1) = excess (39:end,4);
ret1 (:,2) = excess (39:end,3);
```

```
% Financial crisis
subsamFC = rets (39:42,    6:9);
subsampFC_ex = excess(39:42, :);
meanFC = mean(subsamFC)*4
exmeansFC     = mean (subsampFC_ex)*12;
stdsFC      = std (subsampFC_ex)*sqrt(12);
SRsFC       = exmeansFC./stdsFC


% After the financial crisis
subsamAF = rets (39:end,    6:9);
subsampAF_ex = excess(39:end, :);
meanAF = mean(subsamAF)*12
exmeansAF     = mean (subsampAF_ex)*12;
stdsAF      = std (subsampAF_ex)*sqrt(12);
SRsAF       = exmeansAF./stdsAF


% % 4 sampleperiods - 3.5 years
subsam1 = rets (1:42,    6:9);
subsam2 = rets (43:84,    6:9);
subsam3 = rets (85:126,    6:9);
subsam4 = rets (127:end,    6:9);
means1 = mean(subsam1)*12
means2 = mean(subsam2)*12
means3 = mean(subsam3)*12
means4 = mean(subsam4)*12
```

```
subsamp1_ex = excess(1:42, :);
subsamp2_ex = excess(43:84, :);
subsamp3_ex = excess(85:126, :);
subsamp4_ex = excess(127:end, :);


exmeans1     = mean (subsamp1_ex)*12;
exmeans2     = mean (subsamp2_ex)*12;
exmeans3     = mean (subsamp3_ex)*12;
exmeans4     = mean (subsamp4_ex)*12;
stds1      = std (subsamp1_ex)*sqrt(12);
stds2      = std (subsamp2_ex)*sqrt(12);
stds3      = std (subsamp3_ex)*sqrt(12);
stds4      = std (subsamp4_ex)*sqrt(12);
SRs1       = exmeans1./stds1
SRs2       = exmeans2./stds2
SRs3       = exmeans3./stds3
SRs4       = exmeans4./stds4
```

```
SS1cs = cumsum (subsam1 (:, 1:4));
figure;
startDate = datenum('01-31-2006');
endDate   = datenum('06-30-2009');
xData     = linspace(startDate,endDate,42);
plot     (xData, SS1cs, 'LineWidth', 1.7);
datetick ('x', 'yyyy')
legend   ('OSEAX', 'EW', 'MV', 'FT', 'location', 'northwest')
ylabel   ('cumret');
xlim ([startDate,endDate]);




SS2cs = cumsum (subsam2 (:, 1:4));
figure;
startDate = datenum('07-31-2009');
endDate   = datenum('12-31-2012');
xData     = linspace(startDate,endDate,42);
plot     (xData, SS2cs, 'LineWidth', 1.7);
datetick ('x', 'yyyy')
ylabel   ('cumret');
xlim ([startDate,endDate]);


SS3cs = cumsum (subsam3 (:, 1:4));
figure;
startDate = datenum('01-31-2013');
endDate   = datenum('06-30-2016');
xData     = linspace(startDate,endDate,42);
plot     (xData, SS3cs, 'LineWidth', 1.7);
datetick ('x', 'yyyy')
```

```
ylabel   ('cumret');
xlim ([startDate,endDate]);


SS4cs = cumsum (subsam4 (:, 1:4));
figure;
startDate = datenum('07-31-2016');
endDate   = datenum('12-31-2019');
xData     = linspace(startDate,endDate,42);
plot     (xData, SS4cs, 'LineWidth', 1.7);
datetick ('x', 'yyyy')
ylabel   ('cumret');
xlim ([startDate,endDate]);
```

OUT-OF-SAMPLE HAC INFERENCE

```
clear all;
clc;
load ("OS-HML-WML-SMB-BAB-OSEAC-EW-MV-FT-RF.mat");


excess = NaN (168,8);
for i = 1:168
        for j = 2:9
            excess(i,j-1) = rets (i,j)-rets(i,10);
        end
end
```

EW vs SF

```
% EW & HML
ret = NaN (168,2);
ret (:,1) = excess (:,6);
ret (:,2) = excess (:,1);


[se_EW_HML,pval_EW_HML,sepw_EW_HML,pvalpw_EW_HML] = sharpeHAC(ret)


% EW & WML
ret = NaN (168,2);
ret (:,1) = excess (:,6);
ret (:,2) = excess (:,2);


[se_EW_WML,pval_EW_WML,sepw_EW_WML,pvalpw_EW_WML] = sharpeHAC(ret)


% EW & SMB
ret = NaN (168,2);
ret (:,1) = excess (:,6);
ret (:,2) = excess (:,3);


[se_EW_SMB,pval_EW_SMB,sepw_EW_SMB,pvalpw_EW_SMB] = sharpeHAC(ret)


% EW & BAB
ret = NaN (168,2);
ret (:,1) = excess (:,6);
ret (:,2) = excess (:,4);


[se_EW_BAB,pval_EW_BAB,sepw_EW_BAB,pvalpw_EW_BAB] = sharpeHAC(ret)
```

MV vs. EW

```
ret = NaN (168,2);
ret (:,1) = excess (:,7);
ret (:,2) = excess (:,6);
```

```
[se_MV_EW,pval_MV_EW,sepw_MV_EW,pvalpw_MV_EW] = sharpeHAC(ret)
```

FT vs. MV

```
ret = NaN (168,2);
ret (:,1) = excess (:,8);
ret (:,2) = excess (:,7);


[se_FT_MV,pval_FT_MV,sepw_FT_MV,pvalpw_FT_MV] = sharpeHAC(ret)
```