Momentum portfolios

Table of Contents

Introduction

This Matlab live script provides two classic examples of portfolio sorts in empirical asset pricing. The first example illustrates the construction of an important set of portfolios in empirical asset pricing: *decile-sorted momentum portfolios.* These portfolios have a long history in finance, and we are still strugling to properly understand the sources of momentum profits. In addition, they are well suited to illustrate the implementation of a univariate portfolio sort. This script considers the CRSP sample as the universe of individual stocks and implements a univariate portfolio sort for different choices of breakpoints and return calcualtions. We will use this to discuss the influence of these choices on end-results and overall conclusions. The second example illustrates the construction of the momentum factor from Kenneth R. French's data library following the method outlined in Carhart (1997). As such, this will serve as an example of the indepdendent double sort methodology. We will again use the CRSP sample as our universe of assets. We will show that one can very closely replicate the data available from Kenneth R. French. We follow most of the empirical asset pricing literature and rely on the monthly stock file using all common shares (*SHRCD* 10 and 11) listed on the NYSE, AMEX, and NASDAQ exchanges (*EXCHCD* 1, 2, and 3). This is a standard choice in most empirical asset pricing studies as it eliminates stocks traded on smaller exchanges and ADRs. Alternatively, some papers use the full CRSP tape of common stocks (all exchange codes), but eliminate penny stocks (⊠PRC⊠ < 1$ or 5$) and/or microcap stocks defined as stocks in the bottom two deciles of the market capitalization distribution using NYSE break points. However, throughout, we will focus on the former definition of the universe of eligible stocks.

Loading data

We begin by loading the pre-processed monthly CRSP stock file stored in the *crspDataMonthly.mat* file. This file is created using the *makeCRSP.m* script that takes the full monthly stock file and eliminates stocks that does not conform to the requirements listed above and structures the data in matrix format. The file here contains data from January 1986 to December 2019.You can access the raw files from CRSP directly using our class account.

```
% Housekeeping
clear;
clc;

% Loading CRSP data
load('crspDataMonthly.mat');

% Getting data dimensions
```

```
[nObs,nAss] = size(ret);
```

Univariate portfolio sort

This section considers a univariate portfolio sorting example in which we construct ten portfolios sorted on momentum.

Constructing signals and breakpoints

We begin by constructing the standard 12-month momentum signal, skipping the most recent month to avoid short-term reversal effects originating from market microstructure issues, for each stock in the sample. The signal (sorting variable) for time t can be written as

$$F_{i,t-1} = \prod_{h=0}^{10} (1 + r_{i,t-12+h})$$

where $r_{i,t}$ denotes the simple return on stock i. That is, the momentum signal is computed as the cumulative return from $t-12$ to $t-2$, where we skip the most recent month $(t-1)$ to aviod issues with short-term reversal effects caused by market microstructure issues. This is the standard definition most often used (and is also the one you will find in Kenneth French's data library, check for yourself how close we are to their data).

Having obtained the signals, we turn to a discussion of the construction of breakpoints for the portfolios. We have at least two choices that are common in the literature: 1) use ALL available stocks to compute breakpoints so that all portfolios will feature an equal number of stocks every months or 2) use NYSE stocks to define breakpoints so that all portfolios feature an even number of NYSE stocks every month, but potentially an uneven number of AMEX and NASDAQ stocks. The latter may be desirable when using data over long periods as AMEX and NASDAQ stocks only became available in July 1962 and December 1972, respectively. Moreover, stocks listed on AMEX and NASDAQ are typically smaller than the stocks listed on NYSE. Below we illustrate how to compute breakpoints based on both and later discuss the implications.

In our implementation, we follow Kenneth French and require the following for a stock (besides *SHRCD* 10 and 11 and *EXCHCD* 1, 2, and 3) to be included in a portfolio for month t formed at time $t-1$

1. Portfolios are re-balanced at the end of every month
2. Price at time $t-13$ is not missing
3. Return at time $t-2$ is not missing
4. Market equity (meq) data at time $t-1$ is not missing

```
% Preallocations prior to loop
mom12           = NaN(nObs,nAss);       % Momentum signal

% Looping over observations (starting at 13 due to signal definition)
for iObs = 13:nObs-1

    % Computing cumulative returns over the past 11 months, skipping the most recent mo
    % This corresponds to the definition used in the Kenneth French data
    mom12(iObs,:)  = 100.*(prod(1+ret(iObs-12+1:iObs-1,:)./100)-1);

    % Clean signals for missing returns and market equity observations
    % This corresponds to requirements 2-4 outlined above
    % The function "isnan" located NaN (missing) values and we eliminate the signal ac
    mom12(iObs,isnan(prc(iObs-13+1,:))) = NaN;
    mom12(iObs,isnan(ret(iObs-1,:)))    = NaN;
    mom12(iObs,isnan(meq(iObs,:)))      = NaN;

end
```

Building momentum portfolios

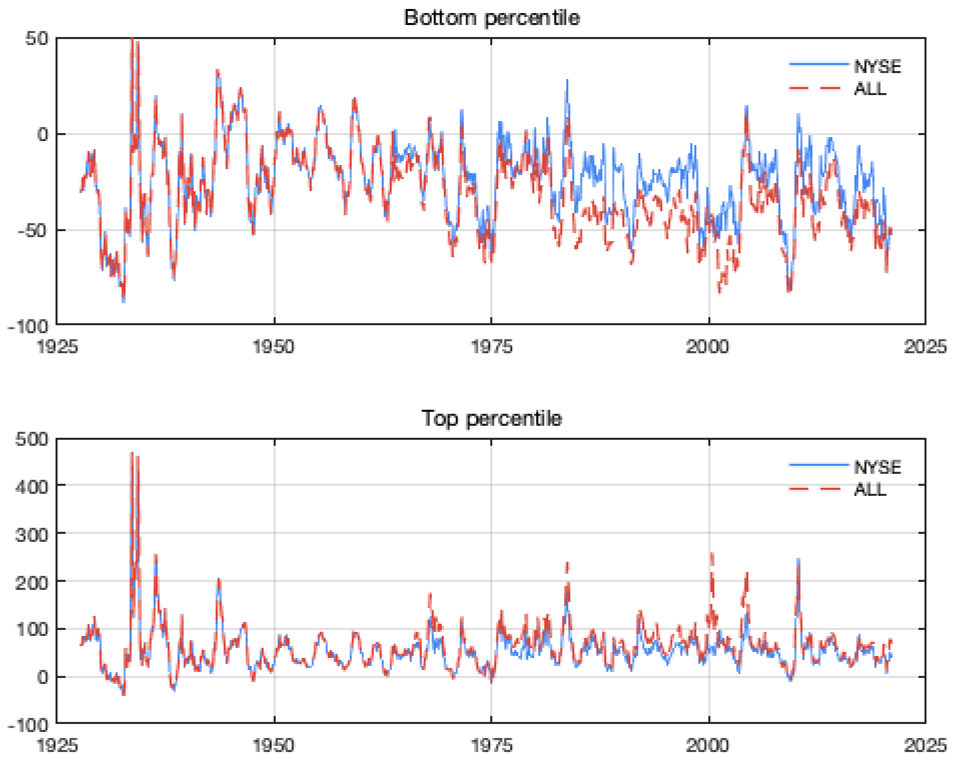
We are now ready to build momentum portfolios. Using the breakpoints, we sort stocks into decile portfolios using, first, the NYSE breakpoints and, second, using breakpoints constructed using ALL available stocks. We will do both equal- and value-weighted portfolios for exposition, but it is normal to only use value-weighted returns. While using equal-weighted returns are simpler, one runs the risk of *overweighting* small and illiquid stocks in the portfolios.

```
% Setting number of portfolios
numPort     = 10;

% Setting percentiles for breakpoints
k           = 1:(numPort-1);
prctlVals   = 100*(k*(1/numPort));

% Building momentum portfolios based on NYSE breakpoints
mom10EW     = univariateSort(ret,mom12,prctlVals,'EW',[],'NYSE',exc);   % Equal-weight
mom10VW     = univariateSort(ret,mom12,prctlVals,'VW',meq,'NYSE',exc);  % Value-weight

% Building momentum portfolios based on ALL breakpoints
mom10EWa    = univariateSort(ret,mom12,prctlVals,'EW',[],'ALL');        % Equal-weight
mom10VWa    = univariateSort(ret,mom12,prctlVals,'VW',meq,'ALL');       % Value-weight
```

We can illustrate the differences in breakpoints by plotting the top and bottom quantiles for each method: ALL and NYSE. We do this below and make the key observation that breakpoints are always more conservative for NYSE stocks.

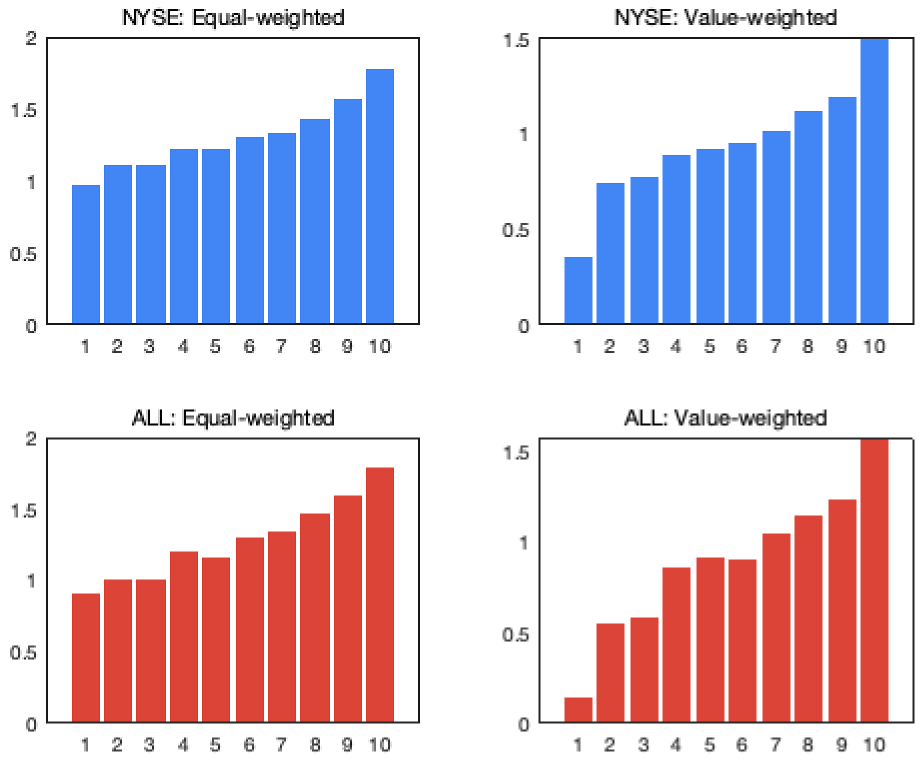
```
% Plotting quantiles for decile sorts
figure;
subplot(2,1,1);
p1 = plot(vDates,[mom10VW.breakpoints(:,1) mom10VWa.breakpoints(:,1)]);
p1(1).Color = colorBrewer(1);
p1(2).Color = colorBrewer(2);
p1(1).LineWidth = 1.2;
p1(2).LineWidth = 1.2;
p1(2).LineStyle = '--';
datetick('x','yyyy');
legend('NYSE','ALL','box','off');
title('Bottom percentile','FontWeight','normal');
grid on
subplot(2,1,2);
p1 = plot(vDates,[mom10VW.breakpoints(:,end) mom10VWa.breakpoints(:,end)]);
p1(1).Color = colorBrewer(1);
p1(2).Color = colorBrewer(2);
p1(1).LineWidth = 1.2;
```

```
p1(2).LineWidth = 1.2;
p1(2).LineStyle = '--';
datetick('x','yyyy');
legend('NYSE','ALL','box','off');
title('Top percentile','FontWeight','normal');
grid on
```



```
% Plotting mean returns for each portfolio
figure;
subplot(2,2,1);
b1 = bar(nanmean(mom10EW.returns));
b1.FaceColor = colorBrewer(1);
b1.EdgeColor = colorBrewer(1);
title('NYSE: Equal-weighted','FontWeight','normal');
subplot(2,2,2);
b1 = bar(nanmean(mom10VW.returns));
b1.FaceColor = colorBrewer(1);
b1.EdgeColor = colorBrewer(1);
title('NYSE: Value-weighted','FontWeight','normal');
subplot(2,2,3);
b1 = bar(nanmean(mom10EWa.returns));
b1.FaceColor = colorBrewer(2);
b1.EdgeColor = colorBrewer(2);
title('ALL: Equal-weighted','FontWeight','normal');
subplot(2,2,4);
b1 = bar(nanmean(mom10VWa.returns));
```

```
b1.FaceColor = colorBrewer(2);
b1.EdgeColor = colorBrewer(2);
title('ALL: Value-weighted','FontWeight','normal');
```

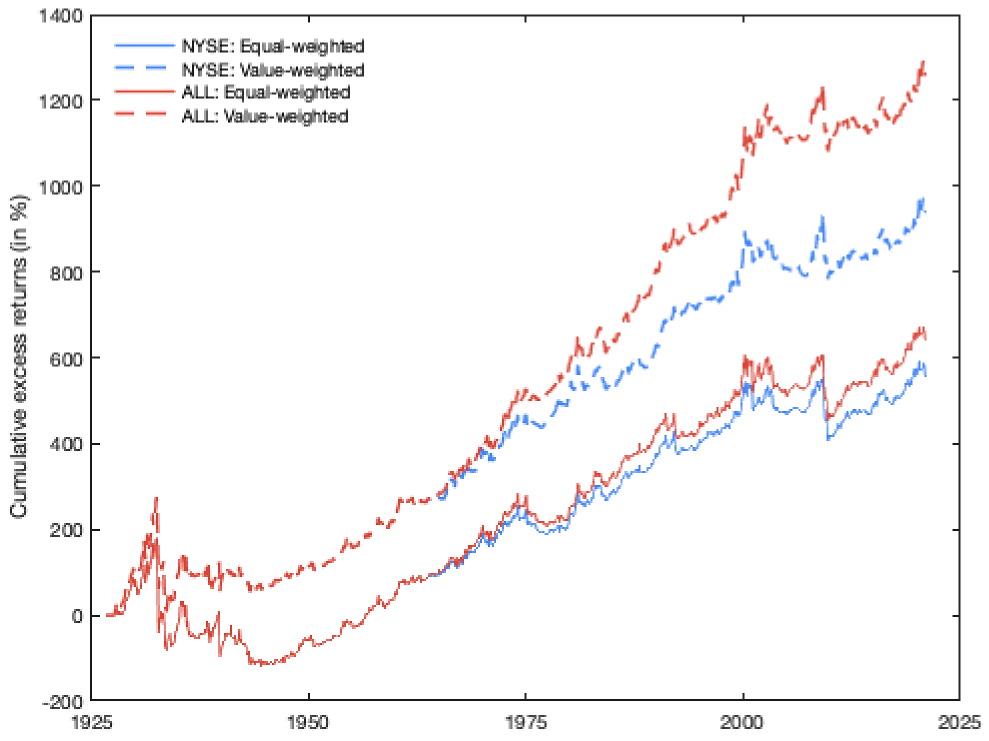


```
% Testing for monotocitic increasing in characteristic
seed=1234; %To replicate the findings
direction=1; %We expect the returns to increase in the characteristic
bootstrap_samples=1000;
block_length=6;

[p]=MR_test_22(mom10VW.returns(14:end,:),bootstrap_samples,direction,block_length,seed
[p(2,:)]=MR_test_22(mom10EW.returns(14:end,:),bootstrap_samples,direction,block_length
[p(3,:)]=MR_test_22(mom10VWa.returns(14:end,:),bootstrap_samples,direction,block_lengt
[p(4,:)]=MR_test_22(mom10EWa.returns(14:end,:),bootstrap_samples,direction,block_lengt

p
```

```
p = 4×2
         0         0
    0.0110    0.0070
    0.0010    0.0010
    0.1370    0.1280
```

```
% Plotting cumulative returns to long-short factors
figure;
p1 = plot(vDates,cumsum([mom10EW.returns(:,10)-mom10EW.returns(:,1) mom10VW.returns(:,
p1(1).Color = colorBrewer(1);
p1(2).Color = colorBrewer(1);
```

```
p1(3).Color = colorBrewer(2);
p1(4).Color = colorBrewer(2);
p1(1).LineWidth = 1.2;
p1(2).LineWidth = 1.4;
p1(3).LineWidth = 1.2;
p1(4).LineWidth = 1.4;
p1(2).LineStyle = '--';
p1(4).LineStyle = '--';
leg = legend('NYSE: Equal-weighted','NYSE: Value-weighted','ALL: Equal-weighted','ALL:
set(leg,'Box','Off','Location','NorthWest');
box on
datetick('x','yyyy');
ylabel('Cumulative excess returns (in %)');
```



Descriptive statistics and risk-adjusted returns

Last, we want to examine returns. We focus here on the NYSE-based, value-weigthed returns. We compute excess portfolio returns by subtracting the risk-free rate from Kenneth French's data library and we compute risk-adjusted excess returns using the Fama and French (1993) three-factor model. That is, we run the regression

$$r_{k,t} - r_{f,t} = \alpha_k + b_{MKT} MKT_t + b_{SMB} SMB_t + b_{HML} HML_t + \varepsilon_{k,t}$$

where we are interested in the sign, magnitude, and significance of α_k and how our momentum returns loads on the three risk factors. Note that the Fama and French (1993) three-factor model is but one applicable asset pricing. More modern models exists such as the Fama and French (2015) five-factor model or the Hou, Xue, and Zhang (2015) q-factor model.

```
% Computing excess portfolio returns
% We skip the first 13 observations as they are lost in the signal creation
mom10VWexcess   = mom10VW.returns(14:end,:) - ff(14:end,end);

% Computing the long-short momentum factor
momFactor       = mom10VWexcess(:,10) - mom10VWexcess(:,1);

% Collecting all data in a matrix
momData         = [mom10VWexcess momFactor];

% Running regression to evaluate significance of mean returns
regRes          = nwRegress(momData,ones(size(momData,1),1),0,6);

% Computing and reporting descriptive statistics
sumStatistic    = [
    regRes.bv.*12                          % Mean excess returns (annualized)
    regRes.tbv                             % t-statistic for mean excess returns
    nanstd(momData).*sqrt(12)              % Standard deviations (annualized)
    skewness(momData)                      % Skewness
    kurtosis(momData)                      % Kurtosis
    regRes.bv./nanstd(momData).*sqrt(12)   % Sharpe ratio (annualized)
]
```

```
sumStatistic = 6×11
    0.9943    5.6329    5.9842    7.3543    7.7178    8.1401    8.8925   10.1244 ···
    0.2762    1.9104    2.4110    3.1287    3.5751    3.8470    4.5026    5.0936
   34.3845   28.4836   24.1854   22.1489   20.5925   20.0149   18.9638   18.4985
    1.7504    1.7786    1.3513    1.4574    1.2750    0.7988    0.1537    0.0205
   18.1350   23.4846   20.3391   20.2240   20.5448   15.6516    9.8157    7.6922
    0.0289    0.1978    0.2474    0.3320    0.3748    0.4067    0.4689    0.5473
```

```
% Computing risk-adjusted excess returns using the Fama-French (1993) model
ffReg           = nwRegress(momData,ff(14:end,2:4),1,3);
output          = [
    ffReg.bv(1,:).*12   % Alpha estiamtes
    ffReg.tbv(1,:)      % t-tstatistics
    ffReg.bv(2,:)       % MKT loadings
    ffReg.tbv(2,:)      % t-tstatistics
    ffReg.bv(3,:)       % SMB loadings
    ffReg.tbv(3,:)      % t-tstatistics
    ffReg.bv(4,:)       % HML loadings
    ffReg.tbv(4,:)      % t-tstatistics
    ffReg.R2vadj'        % Adjusted R-squared
]
```

```
output = 9×11
  -13.2558   -6.4462   -4.2632   -2.1121   -1.1233   -0.4629    1.0641    2.6992 ···
   -8.5532   -5.3161   -4.1391   -2.6425   -1.6516   -0.8131    1.7782    4.3431
    1.4103    1.2645    1.1267    1.0696    1.0143    1.0124    0.9757    0.9489
   27.1496   24.4064   30.7590   33.4397   37.8478   50.2398   50.1747   45.9846
```

```
    0.5029    0.1379   -0.0137   -0.0424   -0.0657   -0.0426   -0.1082   -0.0794
    5.9292    1.9215   -0.1927   -0.9899   -1.6064   -1.5660   -3.4625   -2.7001
    0.4340    0.4150    0.3257    0.2600    0.2280    0.1539    0.0715    0.0028
    4.1285    4.0444    5.0776    6.8245    5.0328    5.6072    2.4348    0.0884
   77.5156   81.8054   84.2619   87.6897   89.6551   92.2985   90.4355   88.5484
```

Independent double sort

This section implement an independent double sort in which we replicate the momentum factor. The momentum factor is constructed from an independent double sort on size and momentum.

Replicating the momentum factor from Kenneth French

We follow the description on the data library of Kenneth French and build six portfolios sorting on size (market equity) and momentum (using the same signal as above). We consider two size groups identified using the median size of NYSE stocks and three momemtum groups identified using the 30th and 70th percentiles of NYSE stocks. This gives us six portfolios of Big (B) and Small (S) stocks for Loser (L), Neutrals (N), and Winners (W). The momentum factor is then constructed from the six portfolios as

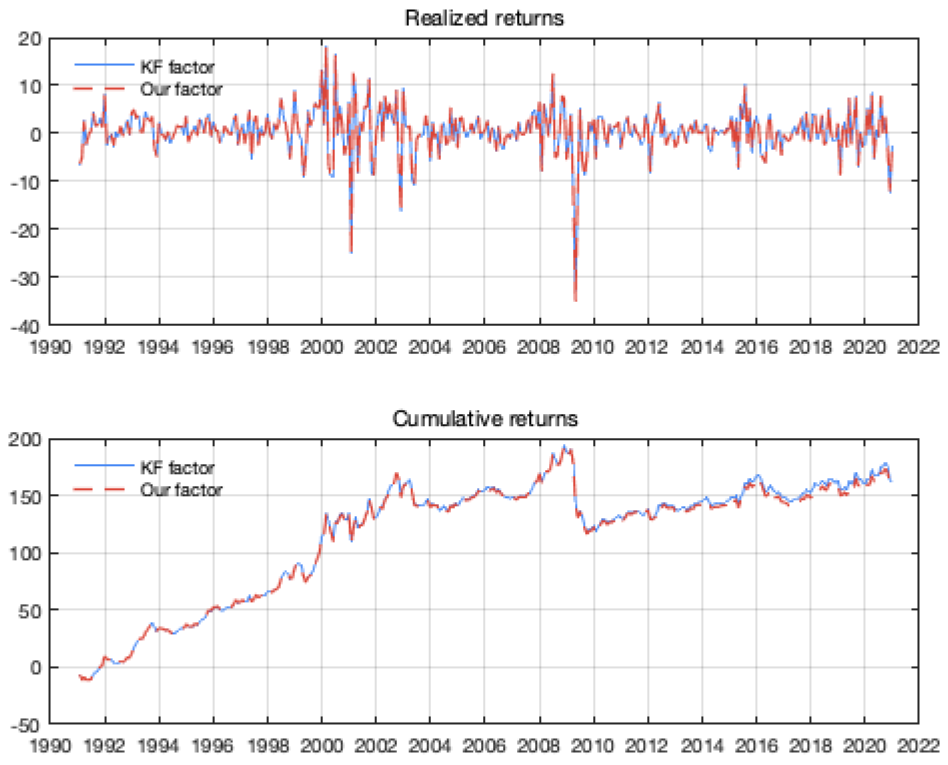
$$MOM = \frac{1}{2}[SW + BW] - \frac{1}{2}[SL + BL]$$

```
% Building six size-momentum portfolios
sizeMom6VW = bivariateSort(ret,meq,mom12,50,[30 70],'Unconditional','VW',meq,'NYSE',ex

% Constructing the momentum factor
momFactor = 0.5.*(sizeMom6VW.returns(:,3) + sizeMom6VW.returns(:,6)) ...
    - 0.5.*(sizeMom6VW.returns(:,1) + sizeMom6VW.returns(:,4));

% Comparing with Kenneth French data
momFF = readmatrix('F-F_Momentum_Factor.CSV','range',[7 2 1142 2]);
momFF(1:13,:) = NaN;

% Note, FF data starts in January 1927 while our data starts in July  1926

% Computing correlation
corrFactors  = corr([momFF(769:end,:) momFactor(775:end,:)],'rows','pairwise')
```

```
corrFactors = 2×2
    1.0000    0.9993
    0.9993    1.0000
```

```

% Plotting cumulative returns
figure;
subplot(2,1,1)
p1 = plot(vDates(775:end,:),[momFF(769:end,:) momFactor(775:end,:)]);
p1(1).Color = colorBrewer(1);
p1(2).Color = colorBrewer(2);
p1(1).LineWidth = 1.0;
p1(2).LineWidth = 1.2;
p1(2).LineStyle = '--';
leg = legend('KF factor','Our factor');
set(leg,'Box','Off','Location','NorthWest');
datetick('x','yyyy');
```

```
grid on
text(vDates(10),-25,['Correlation: ' num2str(corrFactors(1,2))]);
title('Realized returns','FontWeight','Normal');
subplot(2,1,2)
p1 = plot(vDates(775:end,:),cumsum([momFF(769:end,:) momFactor(775:end,:)],'omitnan'))
p1(1).Color = colorBrewer(1);
p1(2).Color = colorBrewer(2);
p1(1).LineWidth = 1.0;
p1(2).LineWidth = 1.2;
p1(2).LineStyle = '--';
leg = legend('KF factor','Our factor');
set(leg,'Box','Off','Location','NorthWest');
datetick('x','yyyy');
grid on
title('Cumulative returns','FontWeight','Normal');
```


Realized returns
KF factor
Our factor
Cumulative returns
KF factor
Our factor