Table of Contents

Introduction

This mlx file presents tables and figures of Menkhoff et. al (2011). A well-known violation of the uncovered interest rate parity is that high interest rate currencies tends to appreciate while low interest rate currrencies tend to depreciate. The popular carry strategy takes advantages of the violation by buying high interest rate currencies and short low. Menkhoff et. al (2011) examines whether a measure for global volatility can explain the cross-section of interest rate sorted portfolios. The script will construct the carry portfolios as a univariate portfolio sort by forward discount. Note that under the covered interest rate parity, the interest rate difference should be approximately equal to the forward discount imply that if we assume that the CIP holds, we can sort on forward discounts.

Enjoy the ride!

Load Data

We will consider a standard panel dataset of currencies which consists of 48 currencies spanning from 1984 to 2018. The source of the data is Thomson Reuters and Barclays. The dataset of Barclays has a longer history but especially during the great recession, the dataset is affiliated by errors in the forward rates. We have, therefor, replaced with spot and forward rates with the ones from Thomson Reuters after 1997. All data used for the analysis can be found in Thomson Reuters Eikon which is available to your at the library. I have uploaded a spread sheet that includes tickers for all data series.

The mat files loaded below measures all currencies in prices in foreign unit relative to domestic. Please note that some of the currencies is flipped when you download them from Eikon!

```
% Housekeeping
clear;
clc;
%addpath('functions')

% Loading currency data etc.
% Loading WMR spot and one-month forward exchange rates
load('bbiwmrData.mat');
%load("data/data_carry.mat");
```

Constructing returns and signals

We begin by constructing excess returns and forward discounts of the currrencies. The excess return of a currency is defined as the return of buying a forward at time t and sell the currency at t+1 in excess of the spotrate at time t. More specifically,

$$RX_{t+1,i} = \frac{S_{t+1,i} - F_{t,i}}{S_{t,i}}$$

The excess return can then be split into two parts: spot changes and forward discounts:

$$RX_{t+1,i} = \frac{S_{t+1,i} - S_{t,i}}{S_{t,i}} - \frac{F_{t,i} - S_{t,i}}{S_{t,i}}$$

where we will use the last term as sorting variable.

```
% Daily log changes which will be used to compute the volatility measure
dailyLogSpotChanges = [dateVectorFX(2:end,:) 100.*diff(log(spotRates(:,5:3:end)))];
% Picking out last-day of month exchange rates
spotRates           = spotRates((spotRates(2:end,2) ~= spotRates(1:end-1,2)),:);
forwardRates        = forwardRates((forwardRates(2:end,2) ~= forwardRates(1:end-1,2)),
spotRates           = spotRates(:,[1:3 5:3:end]);
forwardRates        = forwardRates(:,[1:3 5:3:end]);


% Computing excess currency returns and forward discount
currencyPremia          = (spotRates(2:end,4:end)-forwardRates(1:end-1,4:end))./spotRa
forwardDiscount         = (spotRates(:,4:end) - forwardRates(:,4:end))./spotRates(:,4:
dateVector               = spotRates(2:end,1:2);
currencyPremia          = [dateVector currencyPremia.*100];
forwardDiscount         = [dateVector forwardDiscount(2:end,:).*100];

% Construct vector of dates for plots
dates_used=nan(size(dateVector,1),1);
for i=1:size(dateVector,1)
   if  dateVector(i,2)<10
       dates_used(i,1)=datenum([string(dateVector(i,1))+'0'+string(dateVector(i,2))],'
   else
       dates_used(i,1)=datenum([string(dateVector(i,1))+string(dateVector(i,2))],'yyyy
   end

end
```

Constructing FX volatility measure

A large literature examines many different ways of proxying the latent volatility by various realized measures. The measure acceiving[[[]]] most attention in the lituretur is the realized volatility which is given as the square-root of the sum of squared returns. The measure can, however, be noisy across assets due to potential outliers. To reduce the noise in the signal intoduced by outliers, Menkhoff et. al (2011) use the sum of absolut returns instead, such that their FX volatility is given as

$$VOL_{FX,i} = \frac{1}{T}\frac{1}{N}\sum_{t=1}^{T}\sum_{j=1}^{N_i} \mid s_{t,j} \mid,$$

where T is the number of trading days in month i, N is the number of currencies in the cross-section, and s is the daily log spot change.

```
% Preallocations and preliminaries
idx         = 0;
volFX       = NaN(423,3);


% Constructing monthly bid-ask spread measures
for iYear = 1983:2020

    for iMonth = 1:12

        % Setting index and picking out intra-monthly data
        idx         = idx + 1;
        spotData    = dailyLogSpotChanges(dailyLogSpotChanges(:,1) == iYear & dailyLogS
        % Computing global FX volatility as in Menkhoff et al (2012)
        volFX(idx,:)    = [iYear iMonth nanmean(nanmean(abs(spotData),1),2)];

    end

end


% Adapting to the sample size
volFX           = volFX(11:end,3:end);
```

Ang et. al. (2006) find that innovations to the volatility [...]. Menkhoff et. al. find, however, that innovations to FX vol has a significant autocorrelation. They motivate the use of residuals from an AR(1) process instead such that they estimate

$$VOL_{FX,t+1} = \alpha + \beta VOL_{FX,t} + \varepsilon_{t+1}$$

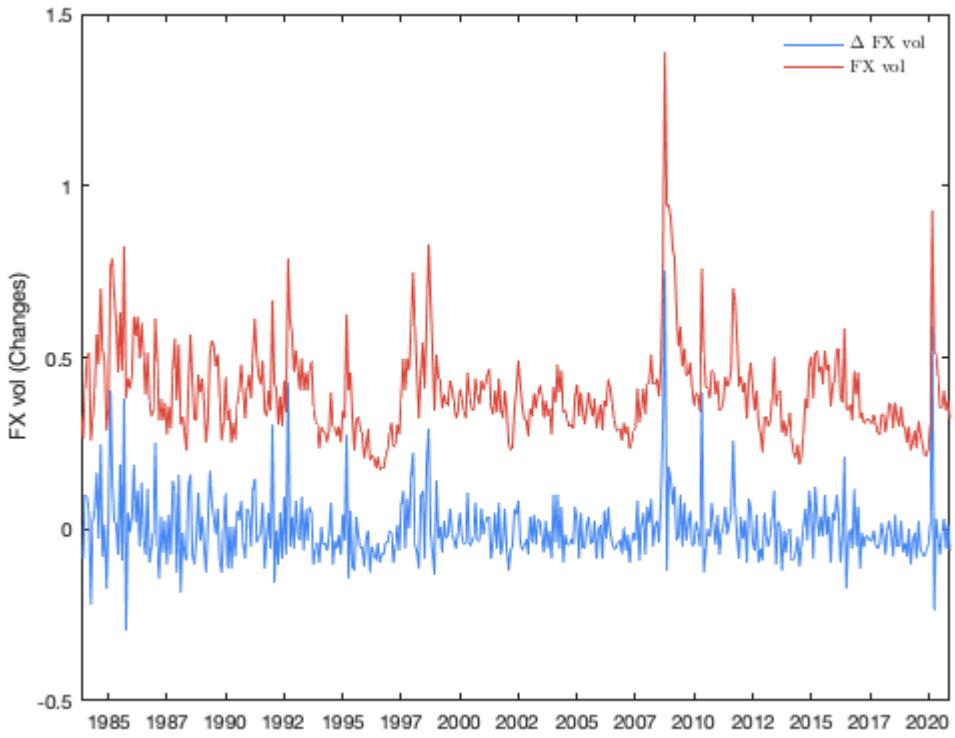
and define the innovations as the estimate of epsilon.

```
% Computing global FX volatility innovations
volARcoef       = [ones(size(volFX,1)-1,1) volFX(1:end-1,1)]\volFX(2:end,1);
```

```
volFXinno       = volFX(2:end,1) - [ones(size(volFX,1)-1,1) volFX(1:end-1,1)]*volARcoe
volFXinno       = [NaN(1,1);volFXinno];
```

The figure belows shows the FX vol and innovation series (as figure 1 in the papir).

```
% Plotting FX vol and FX vol changes (figure 1 lower panel)
figure;
hold on
p1 = plot(dates_used(:,1),volFXinno(:,1));
p2 = plot(dates_used(:,1),volFX(:,1));
hold off
p1.Color = colorBrewer(1);
p2.Color = colorBrewer(2);
datetick('x','yyyy');
ylabel('FX vol (Changes)');
box on
legend('$\Delta$ FX vol','FX vol','Box','Off','interpreter','latex');
axis([-inf inf -0.5 1.5])
```



Portfolio sort

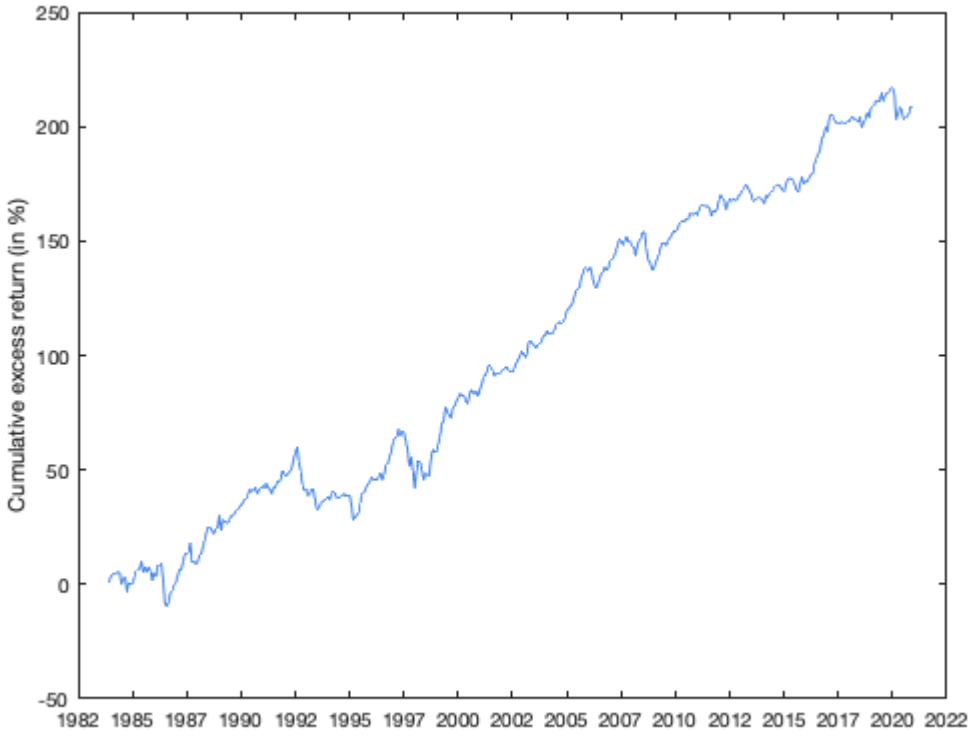
We are now ready to construct Carry portfolios. Due to a relatively low number of currencies in the cross-section, it is standard to examine 5 or 6 portfolios. Menkhoff et. al. consider 5. We do a simply univariarate portfolio with equally spaced portfolios which is equivalent to quantile portfolios. The table below presents describetive statistics for the five portfolios.

```
% Setting number of portfolios
numPort     = 5;
```

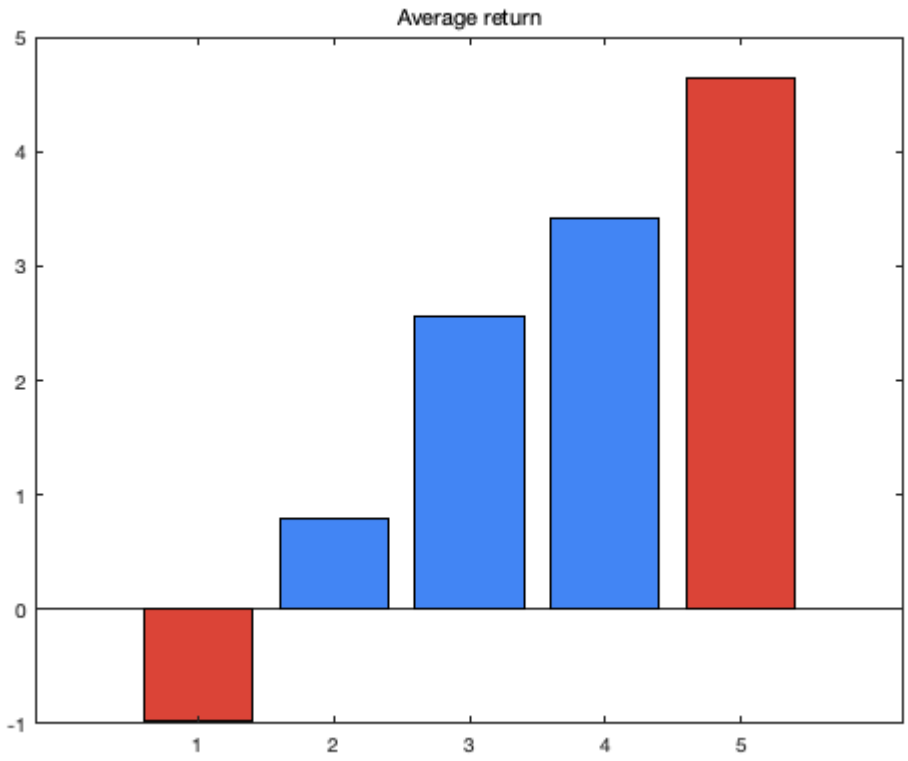
```
% Setting percentiles
prctlVals   = 20:20:80;


out = univariateSort(currencyPremia(:,3:end),forwardDiscount(:,3:end),prctlVals,'EW');


%, "RowNames",{'1', '2','3','4','5','Avg.', 'H/L'}
%out.returns(:,6)=mean(out.returns(:,1:5),2);
out.returns(:,6)=out.returns(:,5)-out.returns(:,1);
table1(1,:)=nanmean(out.returns)*12;
res=linRegress(out.returns(2:end,:), ones(size(out.returns(2:end,:),1),1), 0, 'NW', 6)
table1(2,:)=res.tbv;
%std deviation
table1(3,:)=sqrt(nanvar(out.returns))*sqrt(12);
table1(4,:)=table1(1,:)./table1(3,:);


%table 1 for all countries
table1
```

```
table1 = 4×6
   -0.9768    0.7915    2.5553    3.4102    4.6490    5.6258
   -0.6854    0.6316    1.7017    2.2786    2.5361    3.7356
    7.7709    7.0740    7.8231    8.0925    9.7221    8.6387
   -0.1257    0.1119    0.3266    0.4214    0.4782    0.6512
```

```
% Plotting cumulative returns (figure 1 upper panel)

figure;
hold on
p1 = plot(dates_used(2:end,1),cumsum(out.returns(2:end,6)));
hold off
p1.Color = colorBrewer(1);
datetick('x','yyyy');
ylabel('Cumulative excess return (in %)');
box on
```



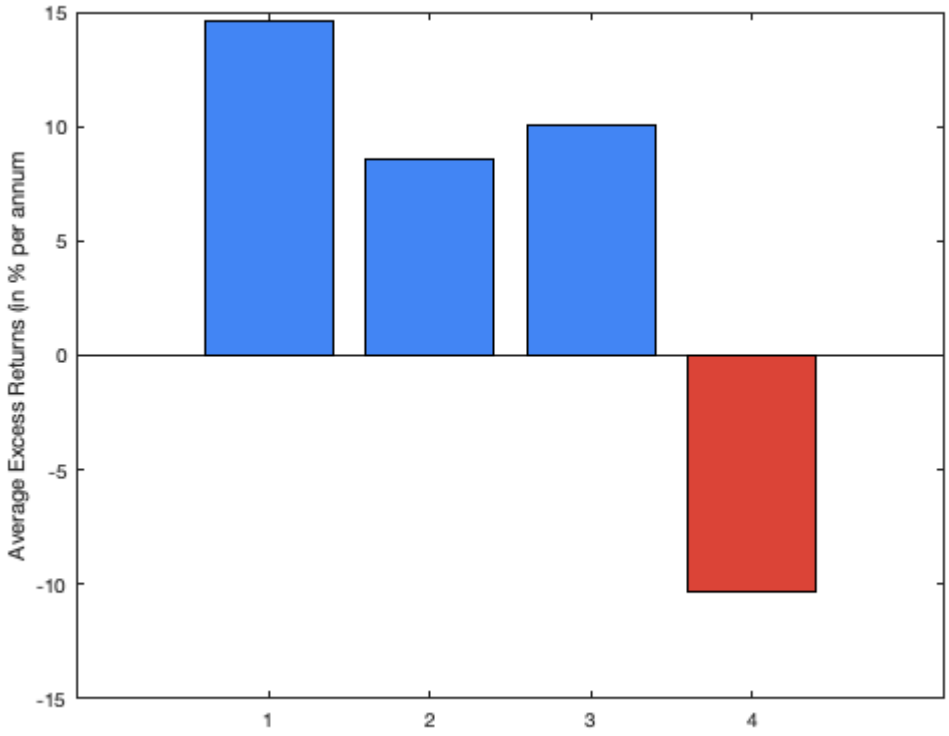

```
% Plotting the average returns

figure;
p1=bar([1:5], nanmean(out.returns(:,1:5))*12, 'FaceColor',colorBrewer(1));
p1.FaceColor = 'flat';
title('Average return','FontWeight','normal');
p1.CData(1,:) = colorBrewer(2);
p1.CData(5,:) = colorBrewer(2);
```



Carry portfolios and volatility

To motivate the relationship between FX vol and the carry strategy, figure 2 in the papir shows annualized returns of the strategy across 4 states of FX vol innovations given as observations below the 25th percentile, between 25th and 50th, between 50th and 75th, and above the 75th percentile. The figure below displays the figure for our dataset.

```
%Motivating figure (figure 2)

figure2(1,:)=mean(out.returns(volFXinno<prctile(volFXinno,25), end));
figure2(2,:)=mean(out.returns(volFXinno>prctile(volFXinno,25) & volFXinno<prctile(volF
figure2(3,:)=mean(out.returns(volFXinno>prctile(volFXinno,50) & volFXinno<prctile(volF
figure2(4,:)=mean(out.returns(volFXinno>prctile(volFXinno,75), end));


figure;
p1=bar([1,2,3,4], figure2*12, 'FaceColor',colorBrewer(1));
p1.FaceColor = 'flat';
ylabel('Average Excess Returns (in % per annum');
p1.CData(4,:) = colorBrewer(2);
box on
```



Asset Pricing Test

To test for whether volatility innovations is priced in the cross-section of carry portfolios, we will specify a linear SDF consisting of the Dollar factor (the portfolio of buying all currencies and short the Dollar) and FX vol innovations:

$$M_t = (1 - b_{Dol}(DOL_t - \mu_{Dol}) - b_{VOL}\Delta\sigma_{FX,t})$$

and examine the coefficent b_{vol}.

Similarly, we can test for whether the risk premium associate with fx vol exposure is significant. The risk premium is given as

$$\lambda = \Sigma b$$

where Σ is the covariance matrix of the DOL factor and FX vol innovations. We compute standard errors of all parameters (incl the risk premia estimates) using the Delta method. We will consider the case in which we ignore estimation variance in the Vol innovations

```
% see table 2
flagAndrews=1;
nLags=4;
para0=[0; -5];
para0=[para0; reshape(cov([nanmean(out.returns(2:end,1:5),2)-mean(nanmean(out.returns(
para0=[para0; nanmean(volFXinno); mean(out.returns(2:end,6))];

output_2=fGMM_FM(para0,out.returns(2:end,1:5),nanmean(out.returns(2:end,1:5),2), volFX
```

```
Parameters after optimization stage:1
```

```
% factor exposure in SDF
output_2.theta(1:2)
```

```
ans = 2×1
   -0.0069
   -4.7057
```

```
%Associated t-stats
output_2.tStat(1:2)
```

```
ans = 2×1
   -0.1831
   -2.3114
```

```
% Risk premia for the two factors
output_2.lambda(1:2)
```

```
ans = 2×1
    0.1528
   -0.0507
```

```
% and again associated t-stats
output_2.lambda_tstat(1:2)
```

```
ans = 2×1
    1.6005
   -3.3719
```

```
% Factor betas
for i=1:5
   res=linRegress(out.returns(2:end,i), [nanmean(out.returns(2:end,1:5),2) volFXinno(2
   table2(2*(i-1)+1,1:3)=res.bv;
   table2(2*(i-1)+1,4)=res.R2v;
   table2(2*(i-1)+2,1:3)=res.tbv;

end
table2
```

```
table2 = 10×4
   -0.2510    0.9758    4.2362   76.8956
   -4.8415   18.2897    7.5929         0
   -0.0879    0.8854    1.0117   77.7165
   -1.7928   19.6324    2.0397         0
    0.0367    1.0141   -0.0083   84.8921
    0.8740   24.1311   -0.0192         0
    0.1043    1.0350   -0.8073   83.9252
    1.9761   24.1127   -1.5243         0
    0.1980    1.0897   -4.4323   70.6838
    2.5149   19.7880   -4.2835         0
```

```
```

Factor mimicking portfolio

Given that FX vol innovations is not traded in the financial markets, it would be interesting to do the analysis using only financial assets. To do so, we need to construct a portfolio that mimicks the behaviour of FX vol inno. We construct the factor mimicking portfolio by first regressing the following:

$$\Delta\sigma_{FX,t} = a + bR_t + \eta_t$$

where R_t is a vector of the excess returns of the five carry portfolios. Next, the factor mimiking portfolio is a portfolio with the weights of b in the five carry portfolios

$$r_{fm,t} = \hat{b}' R_t$$

```
% Find the relation between vol innovations and the five portfolios

res=linRegress(volFXinno(2:end)*12, out.returns(2:end, 1:5), 1, 'NW', 12); %see equati
portfolio_exposures=res.bv(2:end) %compare with eq uation below 7
```

```
portfolio_exposures = 5×1
    0.2516
   -0.0727
   -0.0804
   -0.0468
   -0.1341
```

```
% Construct the factor mimicking portfolio using the fitted values of the
% regression
r_fm=out.returns(2:end, 1:5)*res.bv(2:end);
average_factor_mimicking=mean(r_fm(1:end,:)) %-0.107% in Menkhoff paper
```

```
average_factor_mimicking = -0.1076
```

Asset Pricing Test 2.0 (with factor mimicking portfolio)

We can now test whether the factor mimicking portfolio is priced in the cross-section of Carry portfolios

```
para0=[0; -5];
para0=[para0; reshape(cov([out.returns(2:end,end-1)-mean(out.returns(2:end,end-1)), vo
para0=[para0; mean(r_fm); mean(out.returns(2:end,6))];

output_fm=fGMM_FM(para0,out.returns(2:end,1:5),out.returns(2:end,6), r_fm(1:end),flagA
```

```
Parameters after optimization stage:1
```

```
output_fm.theta(1:2)
```

```
ans = 2×1
    0.0050
   -0.3872
```

```
output_fm.tStat(1:2)
```

```
ans = 2×1
    0.0825
   -1.0690
```

```
% Risk premia for the two factors
output_fm.lambda(1:2)
```

```
ans = 2×1
    0.4558
   -0.0983
```

```
output_fm.lambda_tstat(1:2)
```

```
ans = 2×1
    3.4448
   -3.1007
```

Vol exposure based portfolios

Last, we can examine whether exposure towards vol innovations are priced. To do that we estimate a risk exposures using a 36 month rolling window and use a univariate sort to examine the research question. s

```
% calculation of vol betas
betas=nan(size(currencyPremia(:,3:end)));
for i=37:size(currencyPremia,1)
   for j=1:size(currencyPremia,2)-2
      res=linRegress(currencyPremia(i-35:i, j+2), [volFXinno(i-35:i)], 1, 'Skip');
      betas(i,j)=res.bv(2);
   end

end


out2 = univariateSort(currencyPremia(37:end,3:end),betas(37:end,1:end),prctlVals,'EW')


out2.returns(:,6)=out2.returns(:,1)-out2.returns(:,5);
table4(1,:)=nanmean(out2.returns)*12;
res=linRegress(out2.returns(2:end,:), ones(size(out2.returns(2:end,:),1),1), 0, 'NW', 
table4(2,:)=res.tbv
```

```
table4 = 2×6
    2.0024    2.0265    2.0290    1.2360    0.7933    1.2091
    1.1206    1.3231    1.6971    0.8949    0.5722    0.7824
```

```
table4(3,:)=sqrt(nanvar(out2.returns)*12);
table4(4,:)=table4(1,:)./table4(3,:);

table4
```

```
table4 = 4×6
    2.0024    2.0265    2.0290    1.2360    0.7933    1.2091
    1.1206    1.3231    1.6971    0.8949    0.5722    0.7824
    9.7385    8.2830    7.0135    6.9750    7.3651    9.1854
    0.2056    0.2447    0.2893    0.1772    0.1077    0.1316
```