Economic uncertainty and the cross-section of US stocks

The following script examine whether exposure towards the macroeconomic uncertainty index of Jurado et. al (2015) is priced in the cross-section of US stocks. First, the script follows the approach of Bali et. al (2017) while some of the choices made by the authors is challenged in the end of the script.

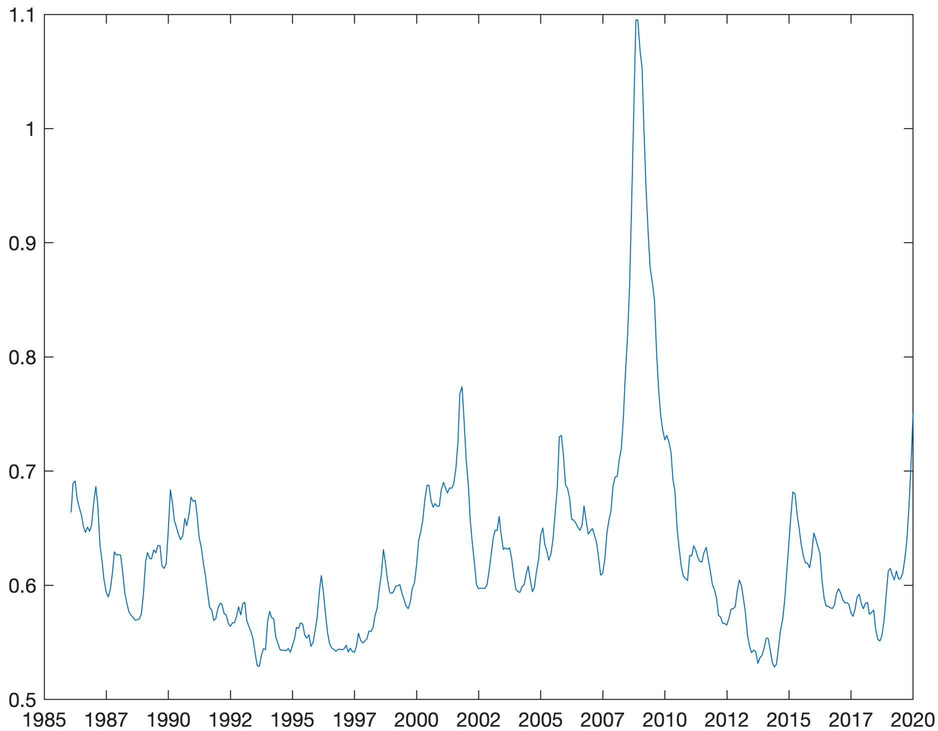
Data

We consider the FF5 model, UMB of Carhart (1997), the liquidity factor of Pastor and Stambaugh (2003), and monthly CRSP returns as in the case of the momentum script. The macroeconomic uncertainty index has a horizon of 1-month in the setting and is available from Sydney Ludvigsons website.

```
% Housekeeping
clear;
clc;


load('ff5.mat')
load('crspDataMonthly.mat')
load('LIQ.mat')
load('UMD.mat')
load('MU1.mat')

% Plot of MU1 over time
figure; plot(vDates, MU1)
hold on
datetick('x', 'yyyy')
```




```
p1(1).Color = colorBrewer(1);
p1(1).LineWidth = 1.2;
```

Estimation

We, next, estimate the exposure towards macroeconomic uncertainty via a linear regression in which we control for the FF5 model, UMD, and LIQ. The regression is estimated using a rolling window with a length of 60 months (5 year

```



for i=60:size(exret,1)
   bv=[ones(60,1), MU1(i-59:i,1), ff5(i-59:i,:), UMD(i-59:i,1), LIQ(i-59:i,1)]\exret(i-

   mu_exposure(i,:)=bv(2,:);

end
```

Univariate portfolio sort

To examine whether MU is priced in the cross-section of stocks, we apply an univariate portfolio based on the stock's exposure towards MU. After the portfolios have been construct, we investigate whether they generate risk premium that cannot be explained by existing models. We will consider the CAPM, FF3, and FF5.

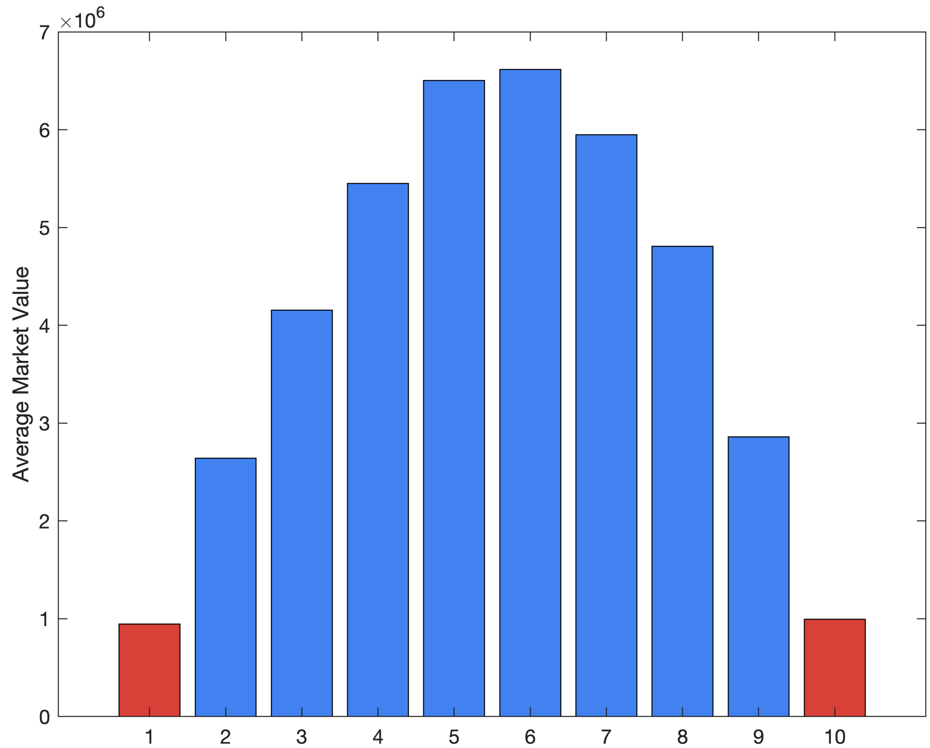
```
prctlVals=[10:10:90];
```

```
mu10fac    = univariateSort(exret,mu_exposure,prctlVals,'EW');


% Constructing average market-value for each portfolio
for i=61:408
    for j=1:10
    meq_port(i,j)=nanmean(meq(i,mu10fac.placement(i,:)==j),2);
    end
end

% Plot of average market value across the ten portfolios

figure;
p1=bar([1:10], nanmean(meq_port(61:end,:)), 'FaceColor',colorBrewer(1));
p1.FaceColor = 'flat';
ylabel('Average Market Value');
p1.CData(1,:) = colorBrewer(2);
p1.CData(10,:) = colorBrewer(2);
```




```
% Test for significant pricing errors

regResults = nwRegress([mu10fac.returns(61:end,:), mu10fac.returns(61:end,10)- mu10fac.
table(1,:)=regResults.bv;
table(2,:)=regResults.tbv;

regResults = nwRegress([mu10fac.returns(61:end,:), mu10fac.returns(61:end,10)- mu10fac.
table(3,:)=regResults.bv(1,:);
```

```
table(4,:)=regResults.tbv(1,:);

regResults = nwRegress([mu10fac.returns(61:end,:), mu10fac.returns(61:end,10)- mu10fac.
table(5,:)=regResults.bv(1,:);
table(6,:)=regResults.tbv(1,:);

regResults = nwRegress([mu10fac.returns(61:end,:), mu10fac.returns(61:end,10)- mu10fac.
table(7,:)=regResults.bv(1,:);
table(8,:)=regResults.tbv(1,:);

table
```

```
table = 8×11
    1.5562    1.3088    1.2539    1.0552    1.0588    1.0905    0.9779    0.9600 ···
    3.3312    3.8637    4.2362    3.9192    4.0654    4.3898    3.7492    3.5634
    0.6297    0.5220    0.5295    0.3715    0.3957    0.4346    0.2969    0.2303
    1.8518    2.3913    2.8721    2.0967    2.5003    2.7930    1.9813    1.6341
    0.5368    0.3751    0.3871    0.2321    0.2617    0.3101    0.1799    0.1244
    2.2372    2.9640    3.8427    2.5004    3.3078    4.5367    2.4430    1.7985
    0.7520    0.3872    0.3676    0.1732    0.1906    0.2411    0.1454    0.1212
    2.6012    2.6804    3.3162    1.9015    2.4089    3.2472    1.8335    1.3650
```

```
% Value weighted sort
mu10fac    = univariateSort(exret,mu_exposure,prctlVals,'VW', meq);

regResults = nwRegress([mu10fac.returns(61:end,:), mu10fac.returns(61:end,10)- mu10fac.
table2(1,:)=regResults.bv;
table2(2,:)=regResults.tbv;

regResults = nwRegress([mu10fac.returns(61:end,:), mu10fac.returns(61:end,10)- mu10fac.
table2(3,:)=regResults.bv(1,:);
table2(4,:)=regResults.tbv(1,:);

regResults = nwRegress([mu10fac.returns(61:end,:), mu10fac.returns(61:end,10)- mu10fac.
table2(5,:)=regResults.bv(1,:);
table2(6,:)=regResults.tbv(1,:);

regResults = nwRegress([mu10fac.returns(61:end,:), mu10fac.returns(61:end,10)- mu10fac.
table2(7,:)=regResults.bv(1,:);
table2(8,:)=regResults.tbv(1,:);
table2
```

```
table2 = 8×11
    1.1224    1.0137    0.7578    0.7715    0.8800    0.7999    0.5793    0.6682 ···
    2.7501    3.1941    2.6889    3.3897    4.0627    3.9405    2.6179    2.9129
    0.1086    0.1850    0.0446    0.1190    0.2566    0.1429   -0.0640   -0.0323
    0.4895    0.8246    0.3631    0.9291    2.3845    1.8686   -1.0453   -0.3645
    0.1144    0.1593   -0.0008    0.0598    0.2182    0.1213   -0.0759   -0.0617
    0.5335    0.7752   -0.0077    0.6396    2.3300    1.6059   -1.2756   -0.7577
    0.3507    0.1379   -0.0694   -0.0796    0.1283    0.0263   -0.1474   -0.1085
    1.5148    0.6619   -0.5770   -1.0352    1.4313    0.3615   -2.6046   -1.2802
```

Change of choice made by Bali et. al (2017)

In the following, we change some of the choices made by Bali et. al (2017). First, we examine what happens when we consider breakpoints calculated using only NYSE stocks. NYSE stocks are generally larger in terms of market capitalization and, thereby, we ensure the portfolio are more homogeneous in terms of market value. Next, we examine the impact of the choice of pricing model when estimating the MU exposures.

```
% NYSE based breakpoints
mu10fac    = univariateSort(exret,mu_exposure,prctlVals,'VW', meq, 'NYSE', exc);


regResults = nwRegress([mu10fac.returns(61:end,:), mu10fac.returns(61:end,10)- mu10fac.
table3(1,:)=regResults.bv;
table3(2,:)=regResults.tbv;

regResults = nwRegress([mu10fac.returns(61:end,:), mu10fac.returns(61:end,10)- mu10fac.
table3(3,:)=regResults.bv(1,:);
table3(4,:)=regResults.tbv(1,:);

regResults = nwRegress([mu10fac.returns(61:end,:), mu10fac.returns(61:end,10)- mu10fac.
table3(5,:)=regResults.bv(1,:);
table3(6,:)=regResults.tbv(1,:);

regResults = nwRegress([mu10fac.returns(61:end,:), mu10fac.returns(61:end,10)- mu10fac.
table3(7,:)=regResults.bv(1,:);
table3(8,:)=regResults.tbv(1,:);

table3
```

```
table3 = 8×11
    0.9570    0.9481    0.7920    0.7734    0.9027    0.8191    0.5588    0.5879 ···
    2.5785    3.4257    3.2102    3.3580    4.3244    3.8917    2.5535    2.5624
    0.0122    0.2130    0.1116    0.1279    0.2789    0.1642   -0.0911   -0.0702
    0.0615    1.5467    0.8303    1.0315    2.6643    1.7664   -1.1368   -0.7748
    0.0129    0.1737    0.0363    0.0726    0.2475    0.1315   -0.0918   -0.1035
    0.0652    1.3740    0.3797    0.7596    2.6434    1.5122   -1.1601   -1.2066
    0.1438    0.1004   -0.0921   -0.0770    0.2049    0.0228   -0.1566   -0.1737
    0.6468    0.8079   -0.8940   -0.8742    2.0986    0.2697   -2.0240   -1.8826
```

```

% Estimating the macroeconomic uncertainty exposure controlling for the
% FF3 model instead

for i=60:size(exret,1)
   bv=[ones(60,1), MU1(i-59:i,1), ff5(i-59:i,1:3)]\exret(i-59:i,:);

   mu_exposure(i,:)=bv(2,:);

end


mu10fac_ff3    = univariateSort(exret,mu_exposure,prctlVals,'EW');


regResults = nwRegress([mu10fac_ff3.returns(61:end,:), mu10fac_ff3.returns(61:end,10)-
table4(1,:)=regResults.bv;
table4(2,:)=regResults.tbv;

regResults = nwRegress([mu10fac_ff3.returns(61:end,:), mu10fac_ff3.returns(61:end,10)-
```

```
table4(3,:)=regResults.bv(1,:);
table4(4,:)=regResults.tbv(1,:);

regResults = nwRegress([mu10fac_ff3.returns(61:end,:), mu10fac_ff3.returns(61:end,10)-
table4(5,:)=regResults.bv(1,:);
table4(6,:)=regResults.tbv(1,:);

regResults = nwRegress([mu10fac_ff3.returns(61:end,:), mu10fac_ff3.returns(61:end,10)-
table4(7,:)=regResults.bv(1,:);
table4(8,:)=regResults.tbv(1,:);

table4
```

```
table4 = 8×11
    1.5051    1.2591    1.1509    1.0414    1.0941    1.0680    0.9798    0.9757 ...
    3.2730    3.7889    3.9217    3.9382    4.3303    4.3074    3.7212    3.4342
    0.5640    0.4870    0.4431    0.3657    0.4446    0.4080    0.2890    0.2404
    1.7250    2.2280    2.3668    2.1212    2.9747    2.7601    1.8940    1.5514
    0.4820    0.3446    0.3015    0.2302    0.3265    0.2846    0.1658    0.1175
    1.9352    2.7631    2.7690    2.5172    4.0311    4.1156    2.3718    1.8425
    0.6710    0.3561    0.2928    0.1931    0.2659    0.2182    0.1285    0.0864
    2.1134    2.4184    2.4034    2.0176    3.3628    3.0598    1.6027    1.2166
```