Three-pass Methodology

Table of Contents

Introduction

The aim of this Live Script is to illustrate the workings of the three-pass methodology of Giglio and Xiu (2021) for estimating risk premia on observable risk factors.

We will focus on several risk factors, both traded and nontraded, and we will in particular see how that a wrong and unintuitive negative risk premia on the market factor typically obtained via Fama-Macbeth regressions (recall the example in Asset Pricing) can be corrected for using this new technique.

The script utilizes parts of some code provided by the authors, some of it pertains to an earlier version of their paper. I have provided you with a function called fThreePass(). This is a modified version of their code. This conducts the steps in the three-pass methodology for estimating the risk premia, including for comparison a standard Fama-Macbeth estimations as well. The original code from the authors can be obtained from Stefano Giglio's website at https://sites.google.com/view/stefanogiglio/.

Loading data

We use the a data set that was applied in a previous version of the working paper of Giglio and Xiu (2021). The working paper can be found here: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=2988748. The data runs from 192 to 2015. As test assets we use 202 equity portfolios sorted on various characteristics (see the description in the Online Appendix of Giglio and Xiu (2019) or my slides).

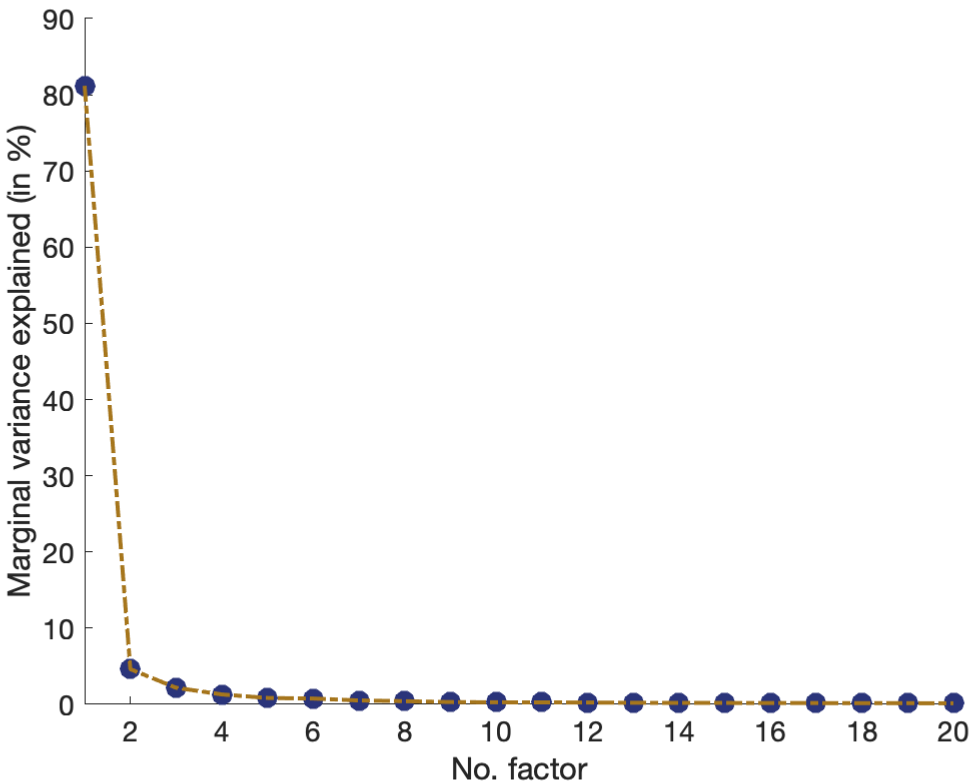
```
% Housekeeping
clear
clc

% Loading test assets from mat file
load('test_portfolios_giglio.mat');

% Define returns
R=Test_assets./100;

% Remove missings
nonmissingobs  = find(~isnan(sum(R,2)));
R              = R(nonmissingobs,:);
load('rf');
R=R-rf./100;
load('factors.mat')
```

```
% We transpose it in order to have it functioning with the fThreePass() function
R = R';
```

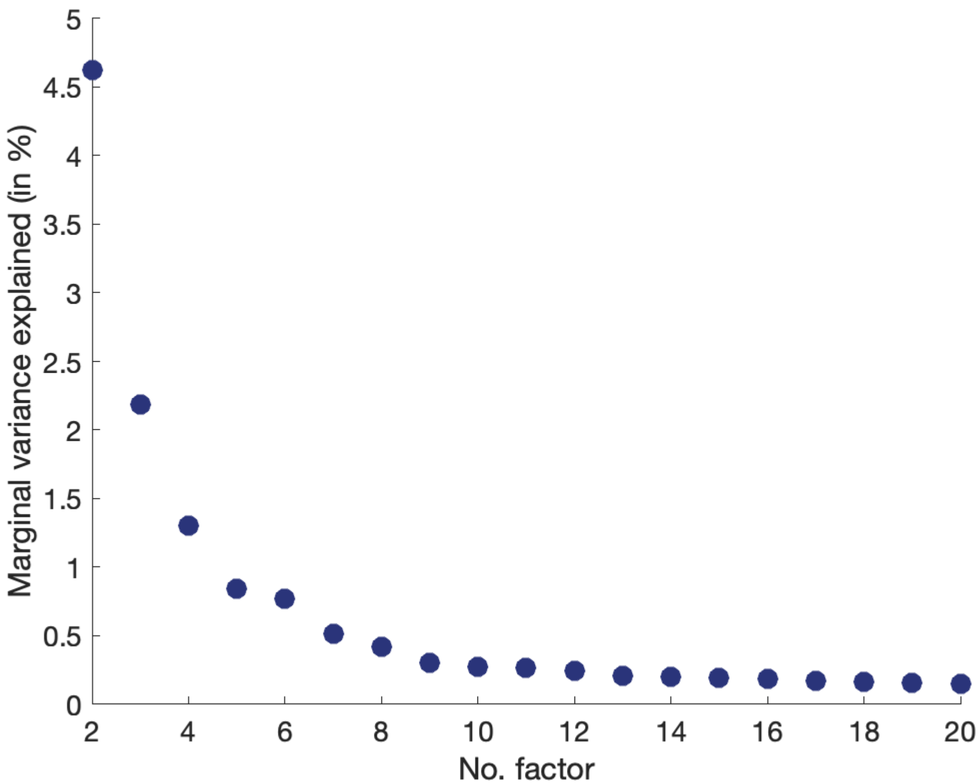
Inspect choice of $\hat{p}$ (# of PCs)

To show the workings of PCA, let us first estimate the PCs and inspect which number of factors to use. I have made a function that computes the estimated optimal no. of factors called fEstP().

```
% Some preliminaries
[N,T]    = size(R);
Rtbar    = R - repmat(mean(R,2),1,T);
Rbar     = mean(Rtbar,2);

% We estimate the PCA components and the amount of variance explained by each componen
[~,score,~,~,explained,~] = pca(Rtbar');

% Plot explained variance for inspection (of the first 20 factors here)
% Set maximum no. for examination
pInspMax   = 20;
fig1 = figure(1);
scatter(1:pInspMax,explained(1:pInspMax),100,'filled','MarkerEdgeColor',[42/255 52/255
'MarkerFaceColor',[42/255 52/255 122/255])
hold on
plot(1:pInspMax,explained(1:pInspMax),'LineStyle','-.','Color',[166/255 118/255 29/255
hold off
set(gca,'FontSize',14)
ylabel('Marginal variance explained (in %)')
xlabel('No. factor')
xlim([1 pInspMax])
```




```
% Zoom in
fig2 = figure(2);
scatter(2:pInspMax,explained(2:pInspMax),100,'filled','MarkerEdgeColor',[42/255 52/255
'MarkerFaceColor',[42/255 52/255 122/255])
set(gca,'FontSize',14)
ylabel('Marginal variance explained (in %)')
xlabel('No. factor')
xlim([2 pInspMax])
```




```
% We can also estimate the optimal no. of factors via the formula provided by Giglio an
pHat   = fEstP(Rtbar,pInspMax)
```

```
pHat = 4
```

Estimating risk premia on the market

We will focus on the choice of no. of factorts $\widehat{p}$, yet conduct robustness checks. As Giglio and Xiu (2021) point out, our choice of factors should be done so that we do not include too few. As such, one should conduct robustness in the direction of *more* factors. As our evidence above suggests that $\widehat{p} = 4$, we will conduct robustness using from 4 up to 10 factors.

```
% We will do robustness over various choices of phat (no. of PCA components), here fron
pMin = pHat; pMax = 10;

% No. of lags in HAC estimator used for the covariance matrix
nLags    = 3; % their choice in the paper
```

We will now estimate the risk premia on the market.

```
% Pick out the market factor from the file 'factors', using info in 'factorslist'
whichG   = 'rmrf';
g        = factors.(whichG)./100;
G        = g(nonmissingobs)';
```

```
% Run three-pass estimator
output  = fThreePass(R,G,pMax,nLags);
```

Inspecting output

Now we gather some relevant output and inspect it.

```
% Cross-sectional R2 using using only the PCs (for different choice of no. of factors)
output.R2F(pMin:pMax)
```

```
ans = 1×7
    0.6424    0.6425    0.6431    0.6608    0.6747    0.6786    0.6828
```

```
% Standard Fama-Macbeth output without controls (including a constant)
coef   = output.GammaFM(1:end)
```

```
coef = 2×1
    0.0082
   -0.0010
```

```
se     = sqrt(output.avarhatFM)
```

```
se = 2×1
    0.0020
    0.0027
```

```
tstat  = coef./se
```

```
tstat = 2×1
    4.2117
   -0.3601
```

```
% Three-pass estimates at optimal pHat
threePassCoef    = output.Gammahat(:,pHat)
```

```
threePassCoef = 2×1
    0.0023
    0.0036
```

```
threePassSe      = sqrt(output.avarhat(:,pHat))
```

```
threePassSe = 2×1
    0.0008
    0.0019
```

```
threePassTstat   = threePassCoef./threePassSe
```

```
threePassTstat = 2×1
    2.7514
    1.8676
```

```
% Mean of the market risk factor
mean(G)
```

```
ans = 0.0057
```

We may treat results using $\widehat{p} = 4$ as our main results, yet we should consider some robustness. This can for instance be done by examining the estimate risk premia and t-stats:

```
threePassCoefRobust    = output.Gammahat(:,pMin:pMax)
```

```
threePassCoefRobust = 2×7
    0.0023    0.0022    0.0019    0.0036    0.0043    0.0049    0.0044
    0.0036    0.0037    0.0040    0.0023    0.0017    0.0011    0.0016
```

```
threePassSeRobust      = sqrt(output.avarhat(:,pMin:pMax));
threePassTstatRobust   = threePassCoefRobust./threePassSeRobust
```

```
threePassTstatRobust = 2×7
    2.7514    2.1917    1.6967    2.9688    3.5505    3.8714    3.4429
    1.8676    1.8401    1.9257    1.0823    0.7808    0.5176    0.7300
```

```
% Note that without controls (or too little), we still get the negative (and intuitive
threePassCoefRobust    = output.Gammahat(2,1:3)
```

```
threePassCoefRobust = 1×3
    0.0004   -0.0046   -0.0026
```

Other interesting output is the R_g^2 that measures the strength of the fator. We obtain this, using $\hat{p} = 4$

```
output.R2G(pHat)
```

```
ans = 0.9859
```

... indicating substantial strength of the factor. We can also compute the test statistic $\widehat{W}$ for testing $\mathbb{H}_0 : \eta = 0$

```
output.What
```

```
ans = 1×10
10^5 ×
    0.0370    0.1090    0.2492    0.3163    0.6321    0.8378    0.8369    0.9770 ···
```

```
%Evaluate in Chi2 with phat degrees of freedom
etaPvals   = 1- chi2cdf(output.What,1:pMax)
```

```
etaPvals = 1×10
     0     0     0     0     0     0     0     0     0     0
```

... with a strong rejection of the market risk factor being a so-called weak factor (given the PCs are pervasive/strong).

Estimating risk premia on Fama-French factors

We will now estimate the risk premia on Fama-French factors SMB and HML, in excess to the market risk premia.

```
% Pick out the market factor from the file 'factors', using info in 'factorslist'
whichG  = 'ff3';
g       = factors.(whichG)./100; % this has three factors (market, SMB and HML) We now
G       = g(nonmissingobs,:)';

% Run three-pass estimator
output  = fThreePass(R,G,pMax,nLags);

% Standard Fama-Macbeth output without controls (including a constant)
```

```
coef    = output.GammaFM(1:end)
```

```
coef = 4×1
    0.0113
   -0.0050
    0.0020
    0.0015
```

```
se      = sqrt(output.avarhatFM)
```

```
se = 4×1
    0.0016
    0.0023
    0.0012
    0.0012
```

```
tstat   = coef./se
```

```
tstat = 4×1
    7.0962
   -2.1512
    1.6517
    1.1773
```

```
% Three-pass estimates at optimal pHat
threePassCoef    = output.Gammahat(:,pHat)
```

```
threePassCoef = 4×1
    0.0023
    0.0036
    0.0027
    0.0014
```

```
threePassSe      = sqrt(output.avarhat(:,pHat))
```

```
threePassSe = 4×1
    0.0008
    0.0019
    0.0012
    0.0011
```

```
threePassTstat   = threePassCoef./threePassSe
```

```
threePassTstat = 4×1
    2.7514
    1.8676
    2.1484
    1.3262
```

```
% Mean of the risk factors
mean(G,2)
```

```
ans = 3×1
    0.0057
    0.0023
    0.0025
```

... which is fairly close to the estimated one. We may treat results using $\widehat{p} = 4$ as our main results, yet we should consider some robustness. This can for instance be done by examining the estimate risk premia and t-stats:

```
threePassCoefRobust    = output.Gammahat(:,pMin:pMax)
```

```
threePassCoefRobust = 4×7
    0.0023    0.0022    0.0019    0.0036    0.0043    0.0049    0.0044
    0.0036    0.0037    0.0040    0.0023    0.0017    0.0011    0.0016
    0.0027    0.0027    0.0026    0.0027    0.0025    0.0026    0.0026
    0.0014    0.0014    0.0015    0.0012    0.0019    0.0019    0.0019
```

```
threePassSeRobust      = sqrt(output.avarhat(:,pMin:pMax));
threePassTstatRobust   = threePassCoefRobust./threePassSeRobust
```

```
threePassTstatRobust = 4×7
    2.7514    2.1917    1.6967    2.9688    3.5505    3.8714    3.4429
    1.8676    1.8401    1.9257    1.0823    0.7808    0.5176    0.7300
    2.1484    2.1610    2.1359    2.1647    2.0602    2.0932    2.1238
    1.3262    1.3320    1.3891    1.1324    1.5136    1.5174    1.5251
```

Other interesting output is the R_g^2 that measures the strength of the fator. We obtain this, using $\widehat{p} = 4$

```
output.R2G(:,pHat)
```

```
ans = 3×1
    0.9859
    0.9512
    0.6614
```

```
output.What
```

```
ans = 3×10
10^5 ×
    0.0370    0.1090    0.2492    0.3163    0.6321    0.8378    0.8369    0.9770 ···
    0.0057    0.0112    0.1045    0.1335    0.2064    0.2763    0.2849    0.3283
    0.0003    0.0007    0.0291    0.0366    0.0592    0.0937    0.0953    0.2299
```

```
%Evaluate in Chi2 with phat degrees of freedom
etaPvalsrmrf   = 1- chi2cdf(output.What(1,:),1:pMax)
```

```
etaPvalsrmrf = 1×10
     0     0     0     0     0     0     0     0     0     0
```

```
etaPvalsSMB   = 1- chi2cdf(output.What(2,:),1:pMax)
```

```
etaPvalsSMB = 1×10
     0     0     0     0     0     0     0     0     0     0
```

```
etaPvalsHML   = 1- chi2cdf(output.What(3,:),1:pMax)
```

```
etaPvalsHML = 1×10
10^-7 ×
    0.2345    0.0000         0         0         0         0         0         0 ···
```

Estimating risk premia on Industrial Production

We will now estimate the risk premia on Indistrial production growth (specifically the residuals from an AR(1) fitted to industrial production growth - to get shocks/innovations to the factor that are "unpredictable")

```
% Pick out the market factor from the file 'factors', using info in 'factorslist'
whichG  = 'ip';
g       = factors.(whichG);
```

```
G       = g(nonmissingobs,:)';

% Run three-pass estimator
output  = fThreePass(R,G,pMax,nLags);

% Standard Fama-Macbeth output without controls (including a constant)
coef   = output.GammaFM(1:end)
```

```
coef = 2×1
    0.0078
    0.2256
```

```
se     = sqrt(output.avarhatFM)
```

```
se = 2×1
    0.0017
    0.1527
```

```
tstat  = coef./se
```

```
tstat = 2×1
    4.6363
    1.4771
```

```
% Three-pass estimates at optimal pHat
threePassCoef    = output.Gammahat(:,pHat)
```

```
threePassCoef = 2×1
    0.0023
    0.0064
```

```
threePassSe      = sqrt(output.avarhat(:,pHat))
```

```
threePassSe = 2×1
    0.0008
    0.0115
```

```
threePassTstat   = threePassCoef./threePassSe
```

```
threePassTstat = 2×1
    2.7514
    0.5526
```

... which is fairly close to the estimated one. We may treat results using $\widehat{p} = 4$ as our main results, yet we should consider some robustness. This can for instance be done by examining the estimate risk premia and t-stats:

```
threePassCoefRobust    = output.Gammahat(:,pMin:pMax)
```

```
threePassCoefRobust = 2×7
    0.0023    0.0022    0.0019    0.0036    0.0043    0.0049    0.0044
    0.0064    0.0063    0.0065    0.0095    0.0162    0.0139    0.0142
```

```
threePassSeRobust      = sqrt(output.avarhat(:,pMin:pMax));
threePassTstatRobust   = threePassCoefRobust./threePassSeRobust
```

```
threePassTstatRobust = 2×7
    2.7514    2.1917    1.6967    2.9688    3.5505    3.8714    3.4429
```

```
    0.5526    0.5461    0.5829    0.7378    1.0315    0.8841    0.8848
```

Other interesting output is the R_g^2 that measures the strength of the factor. We obtain this, using $\widehat{p} = 4$

```
output.R2G(:,pHat)
```

```
ans = 0.0039
```

```
output.What
```

```
ans = 1×10
    0.4950    0.7965    0.8028    0.8934    0.8963    4.4921    5.1833    7.8266 ···
```

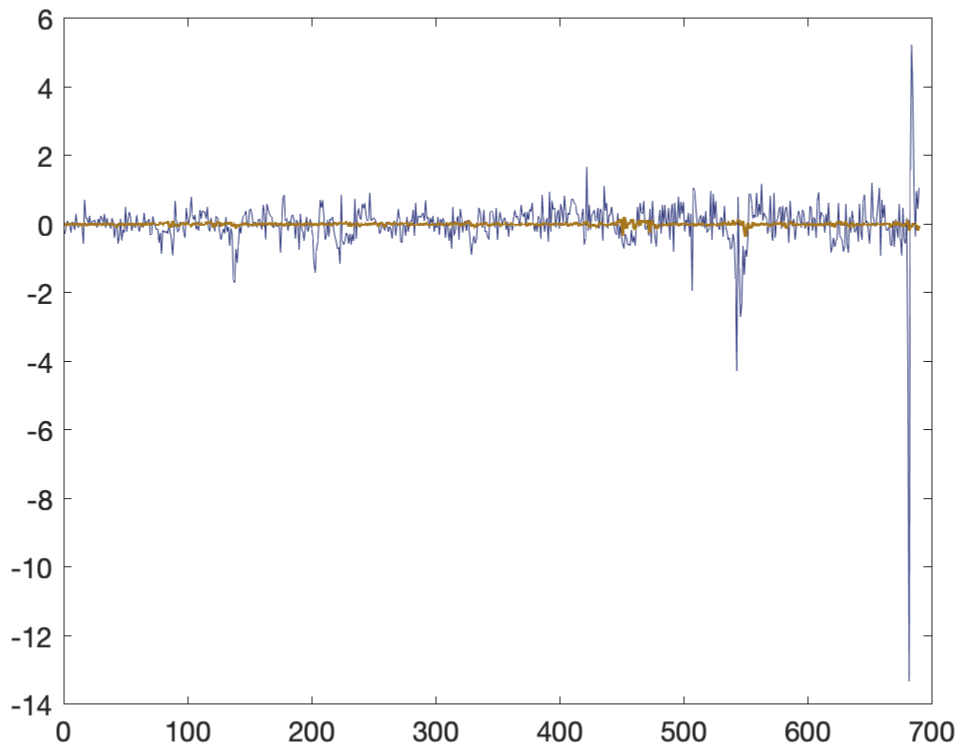
```
%Evaluate in Chi2 with phat degrees of freedom
etaPvals   = 1- chi2cdf(output.What(1,:),1:pMax)
```

```
etaPvals = 1×10
    0.4817    0.6715    0.8488    0.9255    0.9705    0.6104    0.6376    0.4506 ···
```

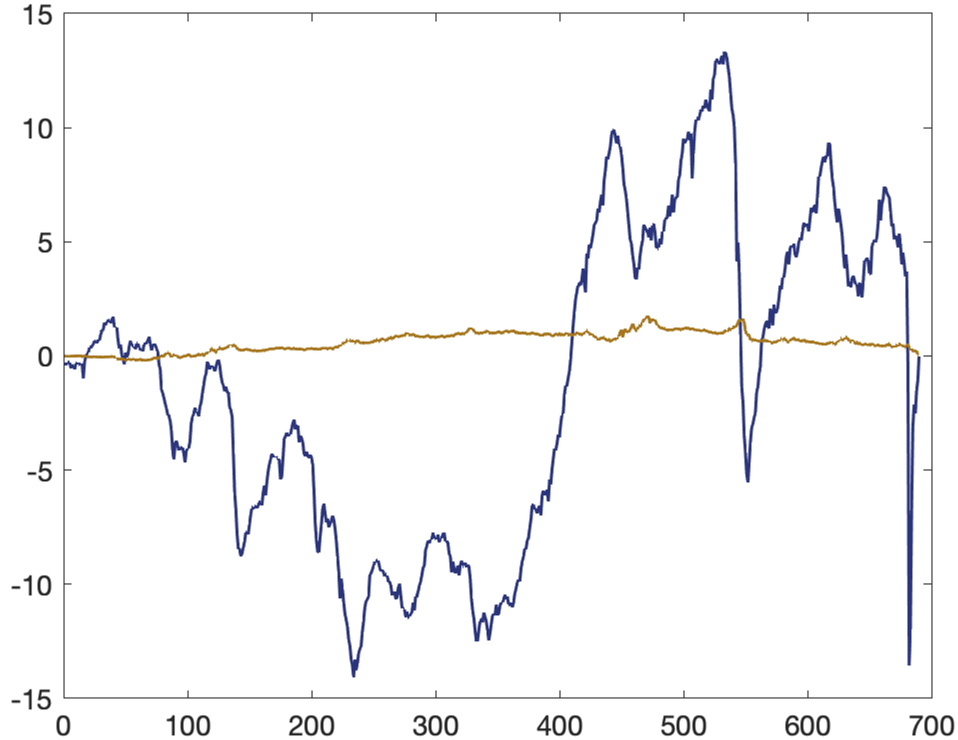
Removing measurement errors from the factor

It is clear that the industrial production growth factor may be subject to a large degree of measurement error. The methodology is effectively taking that into account when estimating the risk premia, but it may be usefull seeing the effect in the time series itself. Note that the fitted values of g_t, i.e. $\widehat{g}_t$ is the de-noised factors. Those may be obtained directly as an output of the procedure as per:

```
% Obtain de-noised factor
gDeNoised = output.gthat;

% Pick out the one for optimal no. of factors, phat
gDeNoisedPhat  = gDeNoised();

% Name the original factor
gOriginal      = G - mean(G);

% One figure just shows the two series together
fig3 = figure(3);
plot(gOriginal,'color',[42/255 52/255 122/255])
set(gca,'FontSize',14)
hold on
plot(gDeNoisedPhat(1,:,pHat),'color',[166/255 118/255 29/255],'LineWidth',1.5)
hold off
```




```
% Another highlights more the effect by cumulating, like Giglio and Xiu
fig4 = figure(4);
plot(cumsum(gOriginal),'color',[42/255 52/255 122/255],'LineWidth',1.5)
set(gca,'FontSize',14)
hold on
plot(cumsum(gDeNoisedPhat(1,:,pHat)),'color',[166/255 118/255 29/255],'LineWidth',1.5)
hold off
```


15
10
5
0
-5
-10
-15
0
100
200
300
400
500
600
700