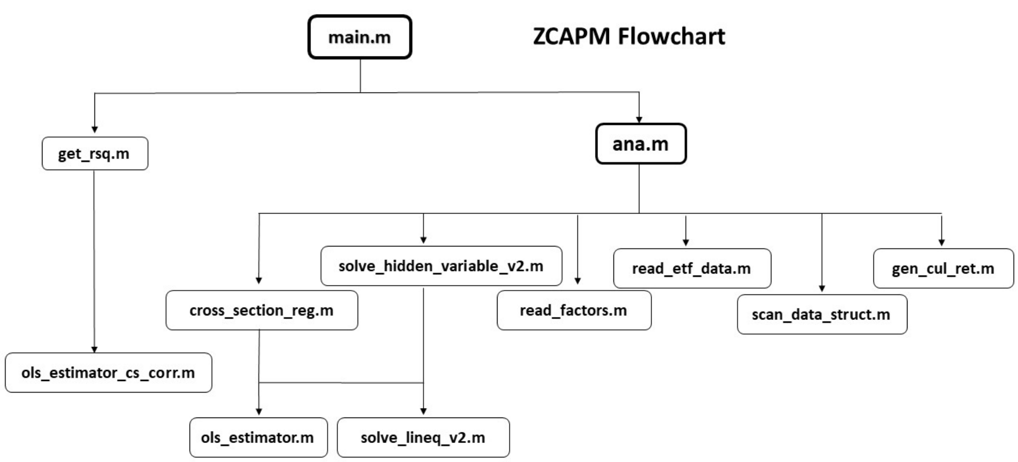
A New Model of Capital Asset Prices: Theory and Evidence

Compendium: Matlab Programs

This compendium provides Matlab programs used in this book for ZCAPM tests. The programs are used to: (1) estimate the empirical ZCAPM, and (2) conduct cross-sectional regression tests of the empirical ZCAPM. The expectation-maximization (EM) algorithm is used to estimate ZCAPM regression model parameters.

If the cross-sectional tests are not needed, you can output the empirical ZCAPM regression results. To do this, uncomment code in the main.m file by removing the percentage symbol: %xlswrite('ts_coeff',ts_coeff). The results will be stored to an excel file called ts_coeff. In this file, the first column is beta (β), second column is zeta (Z), and third column is the probability (p) that Z is positive in sign. To get Z^*, simply compute Z (2p-1). The scale of Z^* is for daily returns. To convert to a monthly return scale, multiply Z^* by 21 as used in our codes (see equation (5.5) in Chapter 5).



main.m

- The main function of the program, which assigns the start date and prediction window, calculates cross-sectional regression estimates for λ_a and λ_{RD} and their *t*-statistics in the second step of the Fama and MacBeth procedure, and outputs results (i.e., the filename is *results*). Run this file to start all codes.
- Called by: n/a
- Calls: ana.m, get_rsq.m
- The % symbol comments out code or wording from being run in the Matlab program. Code can be uncommented by removing this symbol to run some part of the code.

```
function main()

date_start=196501;
%date_start=199001;               %%%% second half for split data results

holding_unit=[1,3,6,9,12,24];   %%%% number of months rolled forward each time

num_roll_window=[6,12];        %%%% number of months used to estimate beta and zeta
for ii=2:2
roll_window_mon=num_roll_window(ii);
for j=1:1
holding_mon=holding_unit(j);
[ts_coeff,cs_coeff,ret_mon,factors_a]...
        =ana(date_start,roll_window_mon,holding_mon);
  %%%%% rolling month approach
dim=size(cs_coeff);
[rsq,beta,t_stat,t1]=get_rsq(ts_coeff,ret_mon,factors_a);  %%%% single regression to get R²

%%%% The following codes calculate the average estimates of λa and λRD, their t-statistics, and
other outputs that appear in the results file %%%%
num_para=(dim(2)-1)/2;
results=NaN(2*num_para+1,2+1);
for i=1:num_para
  results(i,1)=mean(cs_coeff(:,2*i-1));
  results(i,2)=results(i,1)/std(cs_coeff(:,2*i-1))*sqrt(dim(1)-1);
end
%output shanken test
        results(5,3)=t1(1);
        results(6,3)=t1(2);
        results(7,3)=t1(3);
results(num_para+1,1)=rsq;
results(num_para+2,1:2)=[beta(1),t_stat(1)];
```

```
results(num_para+3,1:2)=[beta(2),t_stat(2)];
results(num_para+4,1:2)=[beta(3),t_stat(3)];
xlswrite('results',results);
%xlswrite('ts_coeff',ts_coeff); %%% output time-series regression results for empirical ZCAPM
%xlswrite('cs_coeff',cs_coeff);
%xlswrite('ret_mon',ret_mon)
end
end
```

ana.m

- The workhorse of the program, this subroutine generates coefficients which are then used in both main.m and get_rsq.m to output the final beta, zeta, and *t*-statistic outputs.
- Called by: main.m
- Calls: read_etf_data.m, read_factors.m, scan_data_struct.m, solve_hidden_variable_v2.m, cross_section_reg.m, and gen_cul_ret.m

```
function [ts_coeff,cs_coeff,ret_mon,factors_a]...
   =ana(date_start,roll_window,holding_mon)

[ff25,ind47,own25,bm100,bmop25,bminv25,opinv25]=read_etf_data;
%%%% Read Excel data files containing daily returns of test asset portfolios %%%%

%%%% Choose different test assets, change as your data structure changes %%%%
%%%% Only one stock_ret= should be uncommented at a time %%%%
stock_ret=ff25;
%stock_ret=ind47;
%stock_ret=own25;
%stock_ret=[ff25,ind47(:,2:48)];
%stock_ret=[ff25,ind47(:,2:48),own25(:,2:26)];

%stock_ret=bm100;
%stock_ret=bm100(:,1:21);                       %%%% bottom 20
%stock_ret=[bm100(:,1),bm100(:,22:end)];     %%%%% top 80
%stock_ret=bm100(:,1:51);                       %%%% bottom 50
%stock_ret=[bm100(:,1),bm100(:,52:end)];     %%%% bottom 50

%stock_ret=bmop25;
%stock_ret=bminv25;
%stock_ret=opinv25;

%%%% Split data in half (as robustness check using subperiods) %%%%
%stock_ret=ff25(1:6538,:);
```

```
%stock_ret=ff25(6287:end,:);

%stock_ret=ind47(1:6538,:);
%stock_ret=ind47(6287:end,:);

%stock_ret=own25(1:6538,:);
%stock_ret=own25(6287:end,:);

%stock_ret=[ff25(1:6538,:),ind47(1:6538,2:48)];
%stock_ret=[ff25(6287:end,:),ind47(6287:end,2:48)];

%stock_ret=[ff25(1:6538,:),ind47(1:6538,2:48),own25(1:6538,2:26)];
%stock_ret=[ff25(6287:end,:),ind47(6287:end,2:48),own25(6287:end,2:26)];  %%%% using the
above codes, choose different test assets or their combinations %%%%

dim_ret=size(stock_ret);
factors=read_factors;                                    %%%% read factors from data file
%%%%%%%%%%%%%% Factors for split data (optional) %%%%%%%%%%%%%%%%%%
%factors=factors(1:6538,:);                             %%%% first half
%factors=factors(6287:end,:);                           %%%% second half


%%%% Get row numbers in daily data for each calendar month %%%%
[data_nm,data_str]=scan_data_struct(int32(factors(:,1)/100));
dim=size(data_nm);
for i=1:dim(1)
   if (date_start==data_nm(i,1))
      pos_i=i;
      break;
   end
end

%%%% All factors used for Shanken t-statistic if needed %%%%
factors_a=factors(data_str(pos_i-roll_window,1):data_str(dim(1)-1,2),2:4);

cs_coeff=[];  %%%% empty dataset to store results later
ts_coeff=[];
ret_mon=[];
ts_coeff1=[];

for itr_pd=pos_i:holding_mon:dim(1)
   factors_pr=factors(data_str(itr_pd-roll_window,1):data_str(itr_pd-1,2),2:4); % factors for time
series regression
```

```
   regress_ret=stock_ret(data_str(itr_pd-roll_window,1):data_str(itr_pd-1,2),2:dim_ret(2));
   % return for time series regression

   out_sampl_ret_pr=stock_ret(data_str(itr_pd,1):data_str(itr_pd+holding_mon-
1,2),2:dim_ret(2))...
       -factors(data_str(itr_pd,1):data_str(itr_pd+holding_mon-1,2),4)...
       *ones(1,dim_ret(2)-1); %%% return for out-of-sample month, for cross section regression

   coeff=solve_hidden_variable_v2(regress_ret,factors_pr); %%%time-series ZCAPM regression
   coeff_pr=[coeff(:,1),coeff(:,2).*(2*coeff(:,3)-1)*21];      %%%% 21 trading days per month
ts_coeff_i=[ts_coeff;coeff_pr];                    %%%% output results for time-series regression only
   ts_coeff=ts_coeff_i;
   ts_coeff1_i=[ts_coeff1;coeff_pr];   %%%% time-series results used later in cross-sectional test
   ts_coeff1=ts_coeff1_i;

   out_sample_ret=gen_cul_ret(out_sampl_ret_pr);
   out_sample_mon=out_sample_ret(end,:)/holding_mon;
   %out_sample_mon=((1+out_sample_ret(end,:)/100).^(1/holding_mon)-1)*100;
   cs_coeff_pr=cross_section_reg(out_sample_mon,coeff_pr);%%%% cross-sectional regression
   ret_mon_i=[ret_mon;out_sample_mon(end,:)];
   ret_mon=ret_mon_i;
   cs_coeff_i=[cs_coeff;cs_coeff_pr];
   cs_coeff=cs_coeff_i;
end
```

get_rsq.m

The rolling cross-sectional regression approach does not generate an R^2 value. Here time-series for λ_a and λ_{RD} coefficients and out-of-sample monthly returns are averaged for each stock. One cross-sectional regression is estimated to generate an R^2 value. Coefficients for beta and alpha are stored and output in the main.m file also. To adjust for the cross-correlation problem, ols_estimator_cs_cor is used, in which the Shanken *t*-statistic is computed to adjust estimation errors in betas as well.

- Calculates the R^2 value, a goodness-of-fit measure of the estimated model.
- Called by: main.m
- Calls: ols_estimator_cs_corr.m

```
function [rsq,beta_hat,t_stat,t1]=get_rsq(coeff,ret_mon,factors_a)

mean_ret=mean(ret_mon);
dim=size(ret_mon);
dim_cff=size(coeff);
```

```
mean_cff=zeros(dim(2),dim_cff(2));

for j=1:dim_cff(2)
    for i=1:dim(1)
        mean_cff(:,j)=mean_cff(:,j)+coeff((i-1)*dim(2)+1:i*dim(2),j);
    end
end

mean_cff=mean_cff/dim(1);

X=[ones(dim(2),1),mean_cff];
Y=mean_ret';
factors=factors_a(:,1:2);  %%%% for Shanken t-statistic

[beta_hat,rs,t_stat,t1,u_hat]=ols_estimator_cs_corr(X,Y,ret_mon,coeff,factors);


rsq=rs;
```

ols_estimator_cs_corr.m

- This function takes the data and calculates estimates of beta_hat, rsqr, t_stat, t1, and u_hat after adjusting for cross-correlation in the single regression approach. T1 is the Shanken t-statistics to adjust errors in beta estimations.
- Called by: get_rsq.m
- Calls: n/a

```
function [beta_hat,rsqr,t_stat,t1,u_hat]=ols_estimator_cs_corr(X,Y,ret,coeff,factors)

%%%% OLS estimation %%%%
dim_r=size(ret);
dim=size(Y);
beta_hat = inv(X'*X)*X'*Y;                    %%%% OLS estimation of beta
u_hat = Y - X*beta_hat;                       %%%% estimated residual
s = (u_hat'*u_hat)/(dim(1)-1)*inv(X'*X);  %%%% estimated covariance matrix
se = sqrt(diag(s));

%%%% define variables needed for Shanken test
beta_no_cons=beta_hat(2:end,:);
sig_f=cov(factors)*21;
c=beta_no_cons'*inv(sig_f)*beta_no_cons;
dim_s=size(sig_f);
sig_f_hat=zeros(dim_s(1)+1,dim_s(2)+1);
```

```
sig_f_hat(2:end,2:end)=sig_f;

%%%% variance matrix %%%%
ss=inv(X'*X)*X'*(cov(ret)/dim_r(1))*X*inv(X'*X);  %%%% adjust for correlation
ss=inv(X'*X)*X'*(cov(ret)/dim_r(1))*X*inv(X'*X)*(1+c)+sig_f_hat/dim_f(1); %%%% for
Shanken t-statistic

sse= sqrt(diag(ss));                    %%%% standard errors of beta_hat
sse1= sqrt(diag(ss1));                  %%%% Shanken standard errors of beta_hat


t_stat = beta_hat./sse;                 %%%% t-statistic for beta_hat
t1=beta_hat./sse1;                      %%%% Shanken t-statistic for beta_hat


%p = 2*(1-tcdf(abs(t),dim(1)-1));       %%%% p-value for the t-statistic

y_av=0;
for i=1:dim(1)
   y_av=y_av+Y(i);
end
y_av=y_av/dim(1);

err_y=0;
for i=1:dim(1)
   err_y=err_y+(Y(i)-y_av)^2;
end
rsqr=1.0-(dim(1)-1)/(dim(1)-dim(2))*(u_hat'*u_hat)/err_y;
```

ols_estimator.m

- This function takes the data and calculates estimates of beta_hat, rsqr, t_stat, and u_hat (simple OLS regression).
- Called by: cross_section_reg.m, solve_hidden_variable_v2.m
- Calls: n/a

```
%%%% OLS estimation %%%%
dim=size(Y);
beta_hat = inv(X'*X)*X'*Y;                 %%%% OLS estimation of beta
u_hat = Y - X*beta_hat;                    %%%% estimated residual
s = (u_hat'*u_hat)/(dim(1)-1)*inv(X'*X);   %%%% estimated covariance matrix
se = sqrt(diag(s));                        %%%% standard errors of beta_hat
t_stat = beta_hat./se;                     %%%% t-statistic for beta_hat
%p = 2*(1-tcdf(abs(t),dim(1)-1));          %%%% p-value for the t-statistic
```

```
y_av=0;
for i=1:dim(1)
   y_av=y_av+Y(i);
end
y_av=y_av/dim(1);

err_y=0;
for i=1:dim(1)
   err_y=err_y+(Y(i)-y_av)^2;
end
rsqr=1.0-(dim(1)-1)/(dim(1)-dim(2))*(u_hat'*u_hat)/err_y;
```

cross_section_reg.m

- This function manipulates two data sets within ana.m to create a row of data with 7 values, consisting of beta_hat, t_stat, and rs values. It estimates the cross-sectional OLS regression in the second step of the Fama and MacBeth procedure after reading parameters from the time-series empirical ZCAPM regression.
- Called by: ana.m
- Calls: ols_estimator.m

```
function coeff=cross_section_reg(assets,factors_pr)

dim=size(assets);
dim_f=size(factors_pr);

%%%% Cross-sectional regression  %%%%
coeff=zeros(1,2*dim_f(2)+3);

X=[ones(dim(2),1),factors_pr];
Y=assets';
[beta_hat,rs,t_stat,u_hat]=ols_estimator(X,Y);

for i=1:dim_f(2)+1
   coeff(1,2*i-1)=beta_hat(i);
   coeff(1,2*i)=t_stat(i);
end
coeff(1,2*dim_f(2)+3)=rs;
```

solve_hidden_variable_v2.m

- This function is called in ana.m within a *for loop*. The function selects a slice of data to be processed through this function. Once this data is processed, output is continually stacked in a column in ana.m for final data matrix output to go to main.m. The output is a 3 element row of data, where the first two elements are the beta_hat values, and the third element is a probability element. It utilizes the EM algorithm to estimate the time-series empirical ZCAPM regression.

- Called by: ana.m
- Calls: ols_estimator.m, solve_lineq_v2.m

```
function coeff=solve_hidden_variable_v2(assets,mu_sigma)

dim=size(assets);
factors=mu_sigma(:,1:2);

num_factor=2;
%%%% ZCAPM regression (EM) %%%%
hat_pt=zeros(dim(1),1);
eta_p=zeros(dim(1),1);
eta_n=zeros(dim(1),1);
coeff=zeros(dim(2),num_factor+1);
for j=1:dim(2)
   X=factors(:,1);
   Y=assets(:,j)-mu_sigma(:,3);
   [beta_hat,rs,t_stat,u_hat]=ols_estimator(X,Y);
   Z=factors;
   for kk=1:dim(1)
      if (u_hat(kk)>=0)
         Z(kk,2)=factors(kk,2);
         hat_pt(kk,1)=1;
      else
         Z(kk,2)=-factors(kk,2);
         hat_pt(kk,1)=0;
      end
   end
   [beta_hat,rs,t_stat,u_hat]=ols_estimator(Z,Y);
   p_0=0;
   for kk=1:dim(1)
      if (hat_pt(kk,1)==1)
         p_0=p_0+hat_pt(kk,1);
      end
   end
   p_0=p_0/dim(1);
   sigma_0=mean(u_hat(:,1).*u_hat(:,1));
```

```
    delta=1;
    while (delta>0.001)
        for kk=1:dim(1)
            eta_p(kk,1)=exp(-(Y(kk,1)-beta_hat(1)*factors(kk,1)-beta_hat(2)*factors(kk,2))^2....
                /2/sigma_0);
            eta_n(kk,1)=exp(-(Y(kk,1)-beta_hat(1)*factors(kk,1)+beta_hat(2)*factors(kk,2))^2....
                /2/sigma_0);
            hat_pt(kk,1)=eta_p(kk,1)*p_0/(eta_p(kk,1)*p_0+eta_n(kk,1)*(1-p_0));
        end

        [beta_itr,hat_sigma,p]=solve_lineq_v2(Y,hat_pt,factors);

        diff=[abs((beta_itr(1)-beta_hat(1))/beta_hat(1)),abs((beta_itr(2)-beta_hat(2))/beta_hat(2)),...
            abs((p-p_0)/p_0),abs((hat_sigma-sigma_0)/sigma_0)];
        delta=max(diff);

        p_0=p;
        sigma_0=hat_sigma;
        beta_hat=beta_itr;
    end

    for i=1:num_factor
        coeff(j,i)=beta_hat(i);
    end
    coeff(j,num_factor+1)=p_0;
end
```

read_factors.m

- Loads the data for asset pricing factors input files into the program. Here the data set mu_sigma.xlsx contains value-weighted market mean return and value-weighted market daily sigma (or cross-sectional return dispersion or *RD*). Only sigma is stored in factors and used later. The data set ff_factors_day.xlsx contains the three factors in the Fama and French three-factor model downloaded from Kenneth French's dataset. The market factor and riskless rate are stored in factors and used later. Our data starts from 1960, but our sample period starts from 1964. Therefore, factors only read rows starting from 1005 or 1006 in the original excel file. Users should change this row number based on their own data sources and structure. The data file must be stored within the same folder as Matlab codes -- otherwise, a new path will need to be defined. Variables' names appear in the first row of the data set.
- Called by: ana.m
- Calls: n/a

```
function factors=read_factors()

filename='mu_sigma.xlsx';
data1 = xlsread(filename);

filename='ff_factors_day.xlsx';
data2 = xlsread(filename);

factors=[data1(1005:end,1),data2(1006:end,2),...
   data1(1005:end,3)*100,data2(1006:end,6)];  %%%% combines the RD (sigma) and market
factor together
```

read_etf_data.m

- Loads data set files of daily stock returns for test asset portfolios into the program. The xlsx files starting with ff and ind47 are data downloaded from Kenneth French's website. Like factor data, all Excel data files need to be stored in the same folder as the codes. Users need to redefine which data and which row that Matlab reads based own their own data structure. Variables' names appear in the first row.
- Called by: ana.m
- Calls: n/a

```
function [ff25,ind47,own25,bm100,bmop25,bminv25,opinv25]=read_etf_data()

filename='ff25_day_vw.xlsx';     %%%% 25 size-BM portfolios from Kenneth French website
ff25 = xlsread(filename);
ff25=ff25(1006:end,:);              %%%% row (date) that daily returns begin

filename='ind47_day_vw.xlsx';  %%%% 47 industry portfolios from Kenneth French website
ind47 = xlsread(filename);
ind47=ind47(1006:end,:);

filename='beta_zstar_assets.xlsx';
own25 = xlsread(filename);
own25=[own25(:,1),own25(:,2:end)*100]; %%%% 25 beta-zeta portfolios created by the authors

filename='ff_bm_100.xlsx';
bm100 = xlsread(filename);
bm100=bm100(1006:end,:);

filename='ff_bmop_25.xlsx';
bmop25 = xlsread(filename);
bmop25=bmop25(127:end,:);
```

```
filename='ff_bminv_25.xlsx';
bminv25 = xlsread(filename);
bminv25=bminv25(127:end,:);

filename='ff_opinv_25.xlsx';
opinv25 = xlsread(filename);
opinv25=opinv25(127:end,:);  %%%% other test asset portfolios from Kenneth French website
```

scan_data_struct.m

- Daily data in Matlab only contains rows instead of calendar date. Because our regression uses daily data within calendar time, this function gets the starting and ending row numbers for each corresponding calendar time. (e.g., rows from 1 to 22 in daily data files are days in January 1964)
- Called by: ana.m
- Calls: n/a

```
function [data_nm,data_str]=scan_data_struct(data)

dim=size(data);

data_ini=data(1,1);
data_inf=data_ini;
data_ps=zeros(1000,2);
data_ps(1,1)=1;
count=1;
for i=2:dim(1)
   if (data(i,1)~=data_ini)
      count=count+1;
      data_ps(count-1,2)=i-1;
      data_ps(count,1)=i;
      temp=[data_inf;data(i,1)];
      data_ini=data(i,1);
      data_inf=temp;
   end
end
data_ps(count,2)=dim(1);
data_str=data_ps(1:count,:);
data_nm=data_inf;
```

gen_cul_ret.m

- Compounding daily returns to get monthly returns. For each out-of-sample month, the monthly return is calculated by compounding all daily returns within the month.

- Called by: ana.m
- Calls: n/a

```
function cul_ret=gen_cul_ret(data)

dim=size(data);
cul_ret=zeros(dim(1),dim(2));
for i=1:dim(1)
   if (i>=2)
      cul_ret(i,:)=(1+data(i,:)/100).*(1+cul_ret(i-1,:))-1;
   else
      cul_ret(i,:)=data(i,:)/100;
   end
end
cul_ret=cul_ret*100;
```

solve_lineq_v2.m

- Solves linear equations for the empirical ZCAPM.
- Called by: solve_hidden_variable_v2.m
- Calls: n/a

```
function [hat_beta,hat_sigma,p]=solve_lineq_v2(Y,hat_pt,factors)

dim=size(Y);

a=zeros(2,2);
b=zeros(2,1);
hat_dt=2*hat_pt-1;

a(1,1)=sum(factors(:,1).*factors(:,1));
a(1,2)=sum(hat_dt(:,1).*factors(:,1).*factors(:,2));
a(2,1)=sum(hat_dt(:,1).*factors(:,1).*factors(:,2));
a(2,2)=sum(factors(:,2).*factors(:,2));

b(1,1)=sum(Y(:,1).*factors(:,1));
b(2,1)=sum(hat_dt(:,1).*Y(:,1).*factors(:,2));

hat_beta=linsolve(a,b);
p=mean(hat_pt(:,1));

sigma_t=zeros(dim(1),1);

for i=1:dim(1)
```

```
    sigma_t(i,1)=(Y(i,1)-hat_beta(1,1)*factors(i,1))^2 ...
        -2*(Y(i,1)-hat_beta(1,1)*factors(i,1))* ...
        hat_beta(2,1)*hat_dt(i,1)*factors(i,2)...
        +hat_beta(2,1)^2*factors(i,2)^2;
end
hat_sigma=mean(sigma_t(:,1));
```

Excel data set formating

Test asset portfolios (e.g., the 25 size-BM portfolios from Kenneth French website). Returns are in percentage terms.

Date	SMALL LoBM	ME1 BM2	ME1 BM3	ME1 BM4	SMALL HiBM	ME2 BM1	ME2 BM2
19600104	-0.45	-0.13	1.52	0.59	1.30	0.88	0.22
19600105	3.05	0.30	-0.18	0.29	1.10	0.24	0.17
19600106	-0.50	0.05	-0.05	-0.22	0.31	-0.79	-0.46
19600107	-1.44	-0.24	-0.04	0.25	0.25	-0.58	0.06

Factors in the Fama and French five-factor model (from Kenneth French website). Returns are in percentage terms.

Date	Mkt-RF	SMB	HML	RMW	CMA	RF
19640102	0.60	0.62	0.59	-0.36	0.04	0.013
19640103	0.17	0.22	0.38	-0.31	0.45	0.013
19640106	0.23	0.11	0.02	-0.23	0.4	0.013
19640107	0.04	0.21	0.76	-0.45	0.93	0.013

Factors in the ZCAPM. Here mu is Mkt-RF (or excess average market return over the Treasury rate) and sigma is cross-sectional return dispersion (*RD*). Data are in raw terms.

Date	mu	sigma
19640102	0.0060	0.015651
19640103	0.0017	0.014729
19640106	0.0023	0.014404
19640107	0.0004	0.014697

Excel output in file named Results

Empirical ZCAPM estimation outputs. Each row is a test asset stock portfolio or stock.

beta	zeta	p
1.099748	0.115203	0.42717
1.002184	0.216142	0.476845
0.771063	0.066483	0.67937
0.710273	0.092007	0.501039

Cross-sectional regression outputs. Single regression results are shown also, including Shanken *t*-statistics for this single (one) regression approach.

	lambda	t-stat	Shanken t
alpha	0.759312	3.084977	
market	-0.17832	-0.72595	-0.714536
zeta*	0.488589	4.296504	3.3367293
r-square	0.969062		
alpha(one regression)	0.446998	1.180255	
market(one regression)	0.08308	0.187951	
zeta*(one regression)	1.515916	4.795311	