APPENDIX. MATLAB-CODE FOR THE PAIRS ANALYSIS

```
delete pairs_fin.out; diary pairs_fin.out;
clc; clear all;
warning('off','MATLAB:dispatcher:InexactMatch');

disp('*********************************************************************************');
disp(' PAIRS_FIN.M: Pairs-long-short study using Finnish stock market data 1987-2008.  ');
disp('                                                                                 ');
disp(' Notes:  - use 12 month analysis to calculate distances between all stocks       ');
disp('         - use 6 month period to calculate the return for short-long position    ');
disp('           for the best pairs                                                   ');
disp('         - similar to Gatev, Goetzmann, and Rouwenhorst (RFS, 2006) with the     ');
disp('           exception that we roll the analysis forward by Six_mo days in each    ');
disp('           step, i.e. no parallel analysis                                      ');
disp('                                                                                 ');
disp(' (c) JPB & MV, 01.06.2012                                                       ');
disp('*********************************************************************************');
disp(' ');
disp(['Date and time ', datestr(now)]);
disp(' ');
% ----------------------------------------------------
% Load in data
% ----------------------------------------------------

testing = 0;

Periods = xlsread('Periods.xls','a2:d45');   % read number of days in Jan-Jun, Jul-Dec, Jan-Dec
periods; see file for details; needs to manually set

load ..\data2008\dpri8708;
load ..\data2008\dret8708;
load ..\data2008\dvol8708;

Pt = dpri8708;                               % closing prices at time t
Rt = dret8708;                               % continuously compounded returns
Vt = dvol8708;                               % trading volumes (# of stocks traded)

load c:\windata\data08\d_f.txt;              % daily risk factor etc. data

Rm = d_f(:,1);                               % continuously compounded rate for the market
Rf = d_f(:,13);                              % continuously compounded risk-free rate

Rt = exp(Rt)-1;                              % cont. compounded returns into percentage returns
Rm = exp(Rm)-1;
Rf = exp(Rf)-1;

% ----------------------------------------------------
% Set default values
% ----------------------------------------------------

[T,N]=size(Rt);                              % T = number of periods (days)
                                             % N = number of assets in the whole period

daylag    = 1;                               % set whether (1, dafault) or not (0) signals take one
day to go into effect (cf. Table 1A vs. 1B)
wi_update = 1;                               % set whether (1, default) or not (0) intra-pair weights
are updated when pair is open; if not: weights stay at 50-50%

Years = 2008 - 1987 + 1;

no_pairs = 5;

trading_cost = 0.002;                        % choose a percentage cost for opening and closing pairs
(works just with one day lag)

trade_req = 0;                               % set whether (0) or not (2) positive trading volume is
required for opening/closing a pair
```

```
s1218 = 1;                                  % listing req. (look ahead): 12+6 months (=1)
                                            % NOTE: =0 does not work as the program does attempt to
form a pair if price drops to zero

disp(['Daylag (Yes=1) =    ', num2str(daylag)]);
disp(['wi_update (Yes=1) = ', num2str(wi_update)]);
disp(' ');

% ------------------------------------------------------
% Reset several variables used for desc statistics
% ------------------------------------------------------

Av_price_dev = zeros(T-sum(Periods(1:2,1)),no_pairs*2);     % 12 months are w/o price deviations

periods_with_open_pair = 0;                 % number of periods with pairs opened
periods_without_openpa = 0;                 % number of periods without pairs opened
pairs_number = 0;pair_open = 0;             %
days_open = zeros(no_pairs*300,1);          % measures number of days each pair open; bad
programming, but we do not know how many pairs we get
no_pairs_opened = zeros(Years*2-2,no_pairs); % measures number of times pairs opened in each pair per
6 month period
counter = 1;

% ------------------------------------------------------
% Main part of the program starts here
% ------------------------------------------------------

disp(' ');
disp('*********************************************************************************');
disp('**                                PAIRS ANALYSIS                                **');
disp('*********************************************************************************');
disp(' ');

big_loop = 1;

i = 1;

while big_loop <= (Years * 2 - 2);          % BIG LOOP STARTS HERE; take 6 month steps

Twe_mo = Periods(big_loop,4);               % picks up # of days in 12 months
Six_mo = Periods(big_loop+2,1);             % picks up # of days in following 6 months

% ------------------------------------------------------
% Create price index IPt by setting first Pt>0 to 1
% ------------------------------------------------------

IPt = zeros(Twe_mo+Six_mo,N);               % IPt = Indexed Price at time t

for j=1:N
m = 0;
for i2=1:(Twe_mo+Six_mo)
        if Pt(i+i2-1,j)>0 & m == 0
                IPt(i2,j) = 1; m=1;
        elseif Pt(i+i2-1,j)>0 & m == 1
                IPt(i2,j) = IPt(i2-1,j) * (1 + Rt(i+i2-1,j));
        end
end
end

listed1 = IPt(1,:) > 0;                     % listed at the beginning (1xN vector of 0/1s)
listed2 = IPt(Twe_mo+Six_mo*(s1218==1),:)>0; % listed at the end: 12/18 months from now (1xN vector
of 0/1s)
listed  = listed1 .* listed2;               % listed throughout the 12+6 month period (1xN vector of
0/1s)

no_listed = sum(listed);
index_listed = find(listed>0);              % find columns that should be included


% ------------------------------------------------------
% Add filters (if needed)
```

```
% ------------------------------------------------

% e.g. remove if liquidity below value X, the second listed stock series etc.

% ------------------------------------------------
% Desc stat of the price series
% ------------------------------------------------

no_comp = sum((IPt > 0)')';

disp(['Period                                ',num2str(big_loop)]);
disp(['Time series mean no of stock series ',num2str(mean(no_comp))]);
disp(['Max number of stock series          ',num2str(max(no_comp))]);
disp(['Min number of stock series          ',num2str(min(no_comp))]);

% ------------------------------------------------
% Calc SSEs
% ------------------------------------------------

SSE = zeros(no_listed,no_listed);

for j=1:(no_listed-1)                                % calculate SSE for all pairs
for k=(j+1):no_listed                                % can prob. be speeded up by better programming

       SSE(index_listed(j),index_listed(k)) = sum( ( IPt(1:Twe_mo,index_listed(j)) -
IPt(1:Twe_mo,index_listed(k)) ).^2 );

end
end

% ------------------------------------------------
% Find min SSEs
% ------------------------------------------------

max_SSE = max(max(SSE)) + 1;
min_SSE    = zeros(no_pairs,1);
min_SSE_ro = zeros(1,no_pairs);
min_SSE_co = zeros(1,no_pairs);

for ii=1:no_pairs                                    % find NO_PAIRS minimum SSEs

       t_SSE = max_SSE;                              % temp_SSE

       for j=1:(no_listed-1)
           for k=(j+1):no_listed
              if SSE(j,k) > 0 & SSE(j,k) < t_SSE
                     t_SSE = SSE(j,k);               % a new minimum is found
              end
           end
       end

       if t_SSE == max_SSE
              disp('ERROR---------------');
       end

       [ro,co] = find(SSE == t_SSE);                 % row and column of the minimum
                                                     % If there is a risk that the same asset is twice
                                                     % in the database, add ro=ro(1) and co=co(1)
       min_SSE(ii,1) = SSE(ro,co);
       min_SSE_ro(ii) = ro;                          % column of the 1st stock in a pair
       min_SSE_co(ii) = co;                          % column of the 2nd stock in a pair

       %SSE(ro,co) = max_SSE;                        % prevent re-selection
       SSE(ro,:) = max_SSE;                          % prevent re-selection
       SSE(:,ro) = max_SSE;                          % prevent re-selection
       SSE(co,:) = max_SSE;                          % prevent re-selection
       SSE(:,co) = max_SSE;                          % prevent re-selection

end                                                  % of ii

disp([min_SSE_ro;min_SSE_co]);
```

```
% ------------------------------------------------------
% Calculate returns during the 6 month period
% ------------------------------------------------------

count_temp = counter;

for p=1:no_pairs

        counter = count_temp; pairs_opened = 0; new_pairs_opened = 0; lag = 0;

        std_limit = std(IPt(1:Twe_mo,min_SSE_ro(p))-IPt(1:+Twe_mo,min_SSE_co(p)));

        for j=(i+Twe_mo):(i+Twe_mo+Six_mo-1)                % portfolio period

        if daylag == 0                                      % w/o one day delay

                if pairs_opened == -1                       % pairs opened: long 1st, short 2nd stock
                        Rpair(counter,p) = +Rt(j,min_SSE_ro(p)) .* wi(1)
                                         - Rt(j,min_SSE_co(p)) .* wi(2);
                        if wi_update == 1
                                wi(1)=wi(1)*(1+Rt(j,min_SSE_ro(p)));
                                wi(2)=wi(2)*(1+Rt(j,min_SSE_co(p)));
                        end
                elseif pairs_opened == 1                    % pairs opened: short 1st, long 2nd stock
                        Rpair(counter,p) = -Rt(j,min_SSE_ro(p)) .* wi(1)
                                         + Rt(j,min_SSE_co(p)) .* wi(2);
                        if wi_update == 1
                                wi(1)=wi(1)*(1+Rt(j,min_SSE_ro(p)));
                                wi(2)=wi(2)*(1+Rt(j,min_SSE_co(p)));
                        end
                else
                        Rpair(counter,p) = 0*Rf(j,1);       % closed (this code not necessary)
                end

                if ((pairs_opened == +1 & (IPt(j-i+1,min_SSE_ro(p))-IPt(j-i+1,min_SSE_co(p))) <= 0)
                 | (pairs_opened == -1 & (IPt(j-i+1,min_SSE_ro(p))-IPt(j-i+1,min_SSE_co(p))) >= 0)) &
                   ((trade_req + (Vt(j,min_SSE_ro(p))>0) + (Vt(j,min_SSE_co(p))>0)) > 1)
                        pairs_opened = 0;                   % close pairs: prices cross
                        Av_price_dev(counter,no_pairs+p) = 1; % add a marker for closing; used to calc
                                                              length of the "open-period"

                elseif (+IPt(j-i+1,min_SSE_ro(p))-IPt(j-i+1,min_SSE_co(p)) > 2*std_limit) & ...
                       ((trade_req + (Vt(j,min_SSE_ro(p))>0) + (Vt(j,min_SSE_co(p))>0)) > 1)
                       if pairs_opened == 0                 % record dev (and time) at open
                              Av_price_dev(counter,p) = 2*(+IPt(j-i+1,min_SSE_ro(p))
                               - IPt(j-i+1,min_SSE_co(p)))/(IPt(j-i+1,min_SSE_ro(p))
                               + IPt(j-i+1,min_SSE_co(p)));
                       end

                       pairs_opened = 1;                    % open pairs
                       wi = ones(1,2);

                elseif (-IPt(j-i+1,min_SSE_ro(p))+IPt(j-i+1,min_SSE_co(p)) > 2*std_limit) & ...
                       ((trade_req + (Vt(j,min_SSE_ro(p))>0) + (Vt(j,min_SSE_co(p))>0)) > 1)
                       if pairs_opened == 0                 % record dev (and time) at open

                              Av_price_dev(counter,p) = 2*(-IPt(j-i+1,min_SSE_ro(p))
                               + IPt(j-i+1,min_SSE_co(p)))/(IPt(j-i+1,min_SSE_ro(p))
                               + IPt(j-i+1,min_SSE_co(p)));
                       end

                       pairs_opened = -1;                   % open pairs
                       wi = ones(1,2);
                end
                counter = counter + 1;

        elseif daylag == 1                                  % w/one day delay

                if pairs_opened == -1                       % pairs opened: long 1st, short 2nd stock
                       Rpair(counter,p) = (+Rt(j,min_SSE_ro(p)) * wi(1) - Rt(j,min_SSE_co(p))
```

```
                                * wi(2)) - (lag==2)*trading_cost;
                if wi_update == 1
                        wi(1)=wi(1)*(1+Rt(j,min_SSE_ro(p)));
                        wi(2)=wi(2)*(1+Rt(j,min_SSE_co(p)));
                end

        elseif pairs_opened == 1                    % pairs opened: short 1st, long 2nd stock
                Rpair(counter,p) = (-Rt(j,min_SSE_ro(p)) * wi(1)
                        + Rt(j,min_SSE_co(p)) * wi(2)) - (lag==2)*trading_cost;
                if wi_update == 1
                        wi(1)=wi(1)*(1+Rt(j,min_SSE_ro(p)));
                        wi(2)=wi(2)*(1+Rt(j,min_SSE_co(p)));
                end

        else
                Rpair(counter,p) = 0*Rf(j,1);       % closed (this code not necessary)
        end

        pairs_opened = new_pairs_opened;

        if (pairs_opened == +1 & (IPt(j-i+1,min_SSE_ro(p))
                                -IPt(j-i+1,min_SSE_co(p))) <= 0 | ...
           pairs_opened == -1 & (IPt(j-i+1,min_SSE_ro(p))
                                -IPt(j-i+1,min_SSE_co(p))) >= 0) & ...
           ((trade_req + (Vt(j,min_SSE_ro(p))>0) + (Vt(j,min_SSE_co(p))>0)) > 1)
                new_pairs_opened = 0;               % close prices: prices cross
                lag = 0;
                Av_price_dev(counter+1,no_pairs+p) = 1;      % see above, marker
                if wi_update == 1
                        Rpair(counter,p) = Rpair(counter,p) - trading_cost;
                end
        elseif (+IPt(j-i+1,min_SSE_ro(p))-IPt(j-i+1,min_SSE_co(p)) > 2*std_limit) & ...
                ((trade_req + (Vt(j,min_SSE_ro(p))>0) + (Vt(j,min_SSE_co(p))>0)) > 1)
                new_pairs_opened = 1;               % open pairs
                lag = lag + 1;
        elseif (-IPt(j-i+1,min_SSE_ro(p))+IPt(j-i+1,min_SSE_co(p)) > 2*std_limit) & ...
                ((trade_req + (Vt(j,min_SSE_ro(p))>0) + (Vt(j,min_SSE_co(p))>0)) > 1)
                new_pairs_opened = -1;              % open pairs
                lag = lag + 1;
        end

        if new_pairs_opened == +1 & lag == 1

                Av_price_dev(counter,p) = 2*(+IPt(j-i+1,min_SSE_ro(p))
                 - IPt(j-i+1,min_SSE_co(p)))/(IPt(j-i+1,min_SSE_ro(p))
                 + IPt(j-i+1,min_SSE_co(p)));
                lag = lag + 1;
                wi = ones(1,2);
        elseif new_pairs_opened == -1 & lag == 1

                Av_price_dev(counter,p) = 2*(-IPt(j-i+1,min_SSE_ro(p))
                 + IPt(j-i+1,min_SSE_co(p)))/(IPt(j-i+1,min_SSE_ro(p))
                 + IPt(j-i+1,min_SSE_co(p)));
                lag = lag + 1;
                wi = ones(1,2);
        end

        counter = counter + 1;
    end                                             % endif of one day lag (0/1)

end                                                 % end for pairs-period days (j)

                                                    % if a pair open at the end, add a sign
                                                      that closed
                                                    % otherwise no additional steps
                                                      necessary, as closing just gets last
                                                      day's return
    if pairs_opened ~= 0
            Av_price_dev(counter-1,no_pairs+p) = 1;
    end
```

```
end                                                     % end for portfolio (p)


% ------------------------------------------------------------
% Calculate portfolio returns (ew, vw) out of percentage Rpair
% ------------------------------------------------------------

Rpair_temp = Rpair(counter-Six_mo:counter-1,:);

%
% eq-weighted average on committed cap.; weights reset to "one" (or any equal weight) after each day
%

Rp_ew_cc(counter-Six_mo:counter-1,:) = sum(Rpair_temp')' / no_pairs;

%
% vw-weighted, committed cap.; weights "restart" every 6 month period; as in GGR
% (each portfolio gets 1 dollar at the beginning)
%

wi = ones(1,no_pairs);
wi = [wi;cumprod(1+Rpair_temp)];
wi = wi(1:Six_mo,:);

Rp_vw_cc(counter-Six_mo:counter-1,:) = (sum((wi .* Rpair_temp)'))' ./ sum(wi')';


%
% vw-weighted, fully invested; weights "restart" from 1 every time a new pair is opened;
% as in GGR (capital divided between open portfolios)
%

pa_open = zeros(Six_mo,no_pairs);                       % indicator for days when pairs open

for i2=1:no_pairs
        pa_opened_temp = 0;temp_lag = 0;
        for i1=1:Six_mo

                if pa_opened_temp == 1 & daylag == 0    % opening period not included, closing
                                                          included
                        pa_open(i1,i2) = 1;
                        days_open(pairs_number,1) = days_open(pairs_number,1) + 1;
                end

                if pa_opened_temp == 1 & daylag == 1 & temp_lag == 1
                        pa_open(i1,i2) = 1;
                        days_open(pairs_number,1) = days_open(pairs_number,1) + 1;
                end

                if pa_opened_temp == 1 & daylag == 1 & temp_lag == 0
                        temp_lag = 1;
                end

                if Av_price_dev(counter-1-Six_mo+i1,i2) ~= 0
                        pa_opened_temp = 1;
                        pairs_number = pairs_number + 1;
                end

                if Av_price_dev(counter-1-Six_mo+i1,no_pairs+i2) ~= 0
                        pa_opened_temp = 0;
                        temp_lag = 0;
                end

        end
end

wi2 = wi .* pa_open;

for i2=1:Six_mo                                         % takes care a situation where no
                                                          pairs are open
        if sum(pa_open(i2,:)') == 0
```

```
               wi2(i2,1:no_pairs) = 0.2 * ones(1,no_pairs);
               pa_open(i2,1:no_pairs) = ones(1,no_pairs);
       end
end

Rp_vw_fi(counter-Six_mo:counter-1,:) = (sum((wi2 .* Rpair_temp)'))' ./ sum(wi2')';

%
% eq-weighted, average on fully invested; portfolio returns scaled by the number of pairs that
% are open; weights reset to "one" (or any equal weight) after each day
%

Rp_ew_fi(counter-Six_mo:counter-1,:) = sum(Rpair_temp')' ./ sum(pa_open')';


% record for desc stat, used later

for i2=1:no_pairs
               if sum(Av_price_dev(counter-Six_mo:counter-1,i2) ~= 0) ~= 0
                       periods_with_open_pair = periods_with_open_pair + 1;
                       no_pairs_opened(big_loop,i2) = no_pairs_opened(big_loop,p)
                                       + sum(Av_price_dev(counter-Six_mo:counter-1,i2) ~= 0);
               else
                       periods_without_openpa = periods_without_openpa + 1;
               end
end

i = i + Periods(big_loop,1);

big_loop = big_loop + 1;

end                                          % end main loop


% ------------------------------------------------
% ************* Analysis of the results **************
% ------------------------------------------------

% 3.1: basic pairs analysis
% - Table 1: Descriptive analysis
% - we have done this for top 5 with (Table 1A) and without one day waiting analysis (Table 1B)
% - top 20 or top 101-200 or all are probably not possible since N is limited in Finland --> at most
I think we can analyze the results using top 10 pairs
% - however, the program is generic so that we can test with different N_pairs and time periods for
trading/formation

T_Rp = size(Rp_ew_cc,1);
T_Rm = size(Rm,1);
Rm   = Rm((1+T_Rm-T_Rp):T_Rm,1);             % market
Rf   = Rf((1+T_Rm-T_Rp):T_Rm,1);             % risk-free rate
RmRf = Rm - Rf;                              % excess market

disp('*******************************************************************************');
disp('**                            Descriptive statistics                          **');
disp('**                               PAIRS / FINLAND                              **');
disp('*******************************************************************************');

disp(' ');
disp('Number of observation, T = ');
T_Rp
disp(' ');

disp(' ');
disp('Desc stat for percentage returns of EW, comm capital and fully invested VW strategies, market,
and risk-free asset');
disp('                           Rp_ew      Rp_vw_cc    Rp_vw_fi    Rm       Rf       ');
desc=stat2([Rp_ew_cc Rp_ew_fi Rp_vw_cc Rp_vw_fi Rm Rf],1,5,3);

disp(' ');
disp('Standard errors for the mean and associated t-test statistic values');
disp(' ');
```

```
desc(2,:)/sqrt(T_Rp);
desc(1,:)./(desc(2,:)/sqrt(T_Rp));

disp(' ');
disp('Share of negative excess returns');
disp(sum([Rp_ew_cc Rp_ew_fi Rp_vw_cc Rp_vw_fi Rm Rf]<0)/size(Rf,1));
disp(' ');

disp('*********************************************************************************');
disp('**                            Trading statistics                               **');
disp('**                              PAIRS / FINLAND                                **');
disp('*********************************************************************************');
disp(' ');

disp('Average price deviation for opening pairs');
disp(sum(sum(abs(Av_price_dev(:,1:no_pairs)))')./sum((sum(abs(Av_price_dev(:,1:no_pairs)) >.0))'));

disp('Average price deviation for each opening pairs');
disp(sum(abs(Av_price_dev(:,1:no_pairs)))./(sum(abs(Av_price_dev(:,1:no_pairs)) >.0)));

no_pairs_open = sum(sum(no_pairs_opened)');;

disp('Number of pairs opened for each pair');
x = sum(abs(Av_price_dev(:,1:no_pairs)) > 0);
disp(x);

disp('Total number of pairs opened');
disp(no_pairs_open);

disp('Total number of 6 month trading periods');
disp(Years*2-2);

disp('Total number of pairs opened / number of 6 month periods = av. no of pairs opened each 6 mo
period');
disp(no_pairs_open/(Years*2-2));

disp('Total number of 6 month periods when at least one pair was opened');
disp(periods_with_open_pair);

disp('Total number of 6 month periods when no pairs were opened');
disp(periods_without_openpa);

disp('% of periods with open pair(s)');
disp(periods_with_open_pair/(periods_with_open_pair+periods_without_openpa));

disp('Average number of pairs traded in months when at least one pair opened');
disp(no_pairs_open/periods_with_open_pair);


disp('Average number of round-trip trades per pair');
disp(mean(reshape(no_pairs_opened,no_pairs*(big_loop-1),1)));

disp('Standard deviation of round-trips per pair');
disp(std(reshape(no_pairs_opened,no_pairs*(big_loop-1),1)));

days_open = days_open(1:no_pairs_open,1);

disp('Average time pairs are open in days');
disp(mean(days_open));

disp('Median time pairs are open in days');
disp(median(days_open));

disp('Average time pairs are open in months');
disp(mean(days_open)/21);

disp('Standard deviation of time open per pair in days');
disp(std(days_open));

disp('Standard deviation of time open per pair in months');
disp(std(days_open/21));
```

```
disp('*********************************************************************************');
disp('**                                  Figures                                    **');
disp('**                               PAIRS / FINLAND                               **');
disp('*********************************************************************************');
disp(' ');

% Cumulative excess return of top 5 pairs and HEX-index

plot(cumprod(1+[Rp_ew_cc Rp_ew_fi Rp_vw_cc Rp_vw_fi Rm]));
legend('Rp_ew_cc','Rp_ew_fi','Rp_vw_cc','Rp_vw_fi','Rm','Location','Northwest');

result = [Rp_ew_cc Rp_ew_fi Rp_vw_cc Rp_vw_fi Rm Rf RmRf];
save result.txt result -ascii;

diary off;
```