```
function [Q,lambda,its,Delta] = EFS(FF,astar,k)

% Uses as inputs a data matrix FF with t rows of data points and n columns
% of variables. You specify a targetted fraction (astar) of the sum of squared
% average mistakes relative to the sum of squared averages in the range [0,1]
% with variables explained by k < n components of the original variables.  The
% output is a matrix Q normalized so that Q'Q is I with k columns containing n
% weights (portfolio shares)in the variables (test asset returns).

%=========================================================================
%Program with "Efficient Factor Selection: Explaining Risk and Mean Returns Jointly"
%   by Ron Balvers and Adam Stivers.
%=========================================================================

Sigma=cov(FF); %covariance matrix for the n asset returns
mu=mean(FF)'; %vector of mean returns over the t periods of the n assets
lambda=0; %lambda is the lagrangian multiplier constraint for the pricing
           %errors constraint.  Initial value is zero.
eps=1.0e-12; %tolerance level for deviations from the target fraction of
           %squared pricing errors.
diff=-5; %the experimental change in the lagrangian multiplier
a=1.01; %initial level for the fraction of pricing errors
its=0; %iterations counter

%Loop to converge to true lambda (and Q) contingent on choice of astar
while (a-astar)^2>eps && its<1000 %continue revisions until pricing errors
    %are within tolerance or the number of iterations becomes too large
V=Sigma+lambda*(mu*mu'); %V is the matrix for which eigenvectors are found
[Q,Delta]=eigs(V,k); %'eigs' easier than 'eig' since it automatically provide Qk
abar=a; %abar accounts for the lagged iteration of a.
a=(mu'*mu - (mu'*(Q*Q')*mu))/(mu'*mu); %defines fraction of pricing errors.
if (a-astar)*(a-abar)>0   %if a is getting worse reverse the change in lambda.
    diff=-diff/2; %now reverse change in lambda but at half the pace.
end
lambda=lambda+diff;  %update lambda in the direction that brings pricing
               %errors closer to target
its=its+1;
end
lambda=lambda-diff; %corrected for the unneccessary change in lambda in
               %the final iteration.

end
```