```
% hpf.m

function g=hpf(y,lam)

%  program to apply the "Hodrick-Prescott" filter to time series
%  using a matrix inversion technique. The data is assumed to be
%  organized into a matrix "y" with the rows of y
%  being the observations and the columns being the series.

%  The value of the smoothing parameter is the second argument.

%  HP filtering involves the solution of the difference equation:

%  y(t) =  lam*g(t+2) - 4*lam*g(t+1) + (1+6*lam)g(t) -4*lam*g(t-1) +lam*g(t-2)

%  subject to the two initial conditions:

%  g(1) - 2*g(0) + g(-1) = 0;
%  g(2) - 2*g(1) + g(0)  = 0;

%  Note that these initial conditions imply that:

%  y(1) = lam*g(3) -2*lam*g(2) +(1+lam)*g(1)
%  y(2) = lam*g(4) -4*lam*g(3) +(1+5*lam)*g(2) -2*lam*g(1)

%  and two terminal conditions

%  y(T-1) =-2*lam*g(T) +(1+5*lam)*g(T-1) -4*lam*g(T-2) + lam*g(T-1)
%  y(T)   = (1+lam) *g(T)  -2*lam*g(T-1) + lam*g(T-2)


% If there are a smaller number of observations than series, then
% there is most likely a mistake: convert the vector/matrix of inputs:

oy = size(y);
ny = max(oy);  % length of time series

if (oy(1)<oy(2))
 y=y';
end

disp('Computing Hodrick-Prescott Filtered Time Series with Matrix Inversion')
disp('Growth Component is Returned as g')

%  Strategy: Structure difference equation as a matrix equation:
%               M g = y
%            and then invert M.

M  = zeros(ny,ny);

d1=ones(ny-2,1);
d1=d1*lam;
```

```
d2=ones(ny-1,1);
d2=d2*(-4*lam);
d2(1)=-2*lam;
d2(ny-1)=-2*lam;

d3=ones(ny,1);
d3=d3*(1+6*lam);
d3(1)=1+lam;
d3(2)=1+5*lam;
d3(ny-1)=1+5*lam;
d3(ny)=1+lam;

M = diag(d1,2)+diag(d2,1)+diag(d3)+diag(d2,-1)+diag(d1,-2);

g = inv(M)*y;

% convert if necessary

if (oy(1)<oy(2))
 g=g';
end
```