```
function [rhs, rhsjac] = smooth_mkrhs(dfs)
% Computes the rhs of the system maximum smoothness forward curves.
%
% * dfs: Array of discount factors corresponding to the various knot
% points.  The right hand side of the equations does not depend on the
% placement of the knotpoints, so there is no need to specify these. The
% number of knotpoints, however, must be the same. Note that the first
% (implicit) knot point is always at t = 0. The discount factor
% associated with it is 1, and this must *not* be included. The
% knotpoints and the discount factors have to be in the same order
% (time(i) > time(i-1) and dfs(i) corresponding to time(i)).

% Transpose dfs if it is a row vector (but not if it is a column vector!)
if size(dfs, 1) == 1
   dfs = dfs';
end

height = 5 * size(dfs, 1);

rhs = zeros(height, 1);

% Set up the conditions at t = 0. Note that it doesn't matter what type
% of regression we are running when we have to place the values in the
% 'values' array. Only the form of the left-hand side changes (and of
% course the *meaning* of the right-hand side!).
rhs(1:2, 1) = [0 0];

% Compute -log(dfs(i)/dfs(i-1)), where dfs(0) is 1. Then modify the
% coefficient corresponding to the price at the right hand side of the
% interval. Skip the last interval, which is special.
tmpa = log(dfs(1:end - 1));
rhs(7:5:height-3, 1) = [0 ; tmpa(1:end-1)] - tmpa;

% Set up the conditions at t = end. Note that it doesn't matter what type
% of regression we are running when we have to place the values in the
% 'values' array. Only the form of the left-hand side changes (and of
% course the *meaning* of the right-hand side!).

if length(dfs) == 1
  rhs(height-2:height, 1) = [ log(1 ./ dfs(end)) ; ...
                              0 ; ...
                              0 ];
else
  rhs(height-2:height, 1) = [ log(dfs(end-1) ./ dfs(end)) ; ...
                              0 ; ...
                              0 ];
end
```