TR_1D_model1_SS\unstack_state.m

```
% TR_1D_model1_SS\unstack_state.m
%
% function [State,iflag] = ...
%    unstack_state(x_state,num_species,num_pts);
%
% This procedure takes the stacked vector of state
% variables, and extracts the information to the
% concentration and temperature profile arrays.
%
% INPUT :
% =======
% x_state                 REAL(num_DOF= (num_species+1)*num_pts)
%                         this is the 1-D master array of
%                         state variables
% num_species               INT
%                         the number of species in the system
% num_pts                  INT
%                         the number of grid points
%
% OUTPUT :
% ========
% State                 see TR_1D_model1_SS.m for details
%                         data structure containing concentration
%                         and temperature profiles
%
% Kenneth Beers
% Massachusetts Institute of Technology
% Department of Chemical Engineering
% 7/2/2001
%
% Version as of 7/24/2001



function [State,iflag] = ...
   unstack_state(x_state,num_species,num_pts);

iflag = 0;

func_name = 'unstack_state';


% This flag controls what action to take in case of
% an assertion failure.  See the assertion routines
% for further details.
i_error = 2;
```

```
% First, check the input data

% num_species
check_real=1; check_sign=1; check_int=1;
assert_scalar(i_error,num_species,'num_species', ...
   func_name,check_real,check_sign,check_int);

% num_pts
check_real=1; check_sign=1; check_int=1;
assert_scalar(i_error,num_pts,'num_pts', ...
   func_name,check_real,check_sign,check_int);

% calculate total number of degrees of freedom
num_DOF = (num_species+1)*num_pts;

% check characteristics of x_state
dim=num_DOF; check_column=1;
check_real=1; check_sign=0; check_int=0;
assert_vector(i_error,x_state,'x_state', ...
   func_name,dim, ...
  check_real,check_sign,check_int,check_column);


% Allocate and initialize the output state
State.conc = zeros(num_pts,num_species);
State.Temp = linspace(0,0,num_pts)';


%PDL> First, we unstack the concentration
%   profiles

%PDL> Set pos_counter to zero

pos_counter = 0;


%PDL> FOR ispecies FROM 1 TO ProbDim.num_species

for ispecies = 1:num_species


%   PDL> state_data:conc(:,ispecies) =
%       x_state(pos_counter+1:pos_counter+num_pts)

   State.conc(:,ispecies) = ...
      x_state(pos_counter+1:pos_counter+num_pts);

%   PDL> Increment pos_counter by num_pts

   pos_counter = pos_counter + num_pts;
```

%PDL> ENDFOR

end

%PDL> Next, we unstack the temperature profile

%PDL> state_data:Temp =
% x_state(pos_counter+1:pos_counter+num_pts)

State.Temp = x_state(pos_counter+1:pos_counter+num_pts);

iflag = 1;

return;