TR_1D_model1_SS\jacket_heat_transfer.m

```
% TR_1D_model1_SS\jacket_heat_transfer.m
%
% function [b_HT,bJac_HT,iflag] = jacket_heat_transfer( ...
%    State,epsilon,ProbDim,Reactor,Physical,Grid);
%
% This MATLAB m-file calculates the
% contribution from heat transfer to the
% jacket to the source term and its
% Jacobian in a 1D tubular reactor model.
% A simple heat transfer coefficient expression
% in the limit of large coolant flow rates is
% used in this routine.
%
% INPUT :
% =======
% State          This structure contains the concentration and
%                temperature field data.
% epsilon        This vector has 1's for the interior point
%                equations and 0's for the boundary equations.
% ProbDim        This structure contains the dimensioning
%                parameters for the system.
% Reactor        This structure contains the reactor data
% Physical       This structure contains the physical data
% Grid           This structure contains the grid data
%
% OUTPUT :
% ========
% b_HT           This column vector contains the contribution
%                to the source term from heat transfer to the
%                jacket.
% bJac_HT        This is the Jacobian matrix of b_HT.
%
% Kenneth Beers
% Massachusetts Institute of Technology
% Department of Chemical Engineering
% 7/2/2001
%
% Version as of 7/25/2001


function [b_HT,bJac_HT,iflag] = jacket_heat_transfer( ...
   State,epsilon,ProbDim,Reactor,Physical,Grid);

iflag = 0;

func_name = 'jacket_heat_transfer';
```

```
% This integer flag controls what action to take in
% the case of an error.
i_error = 2;


% Since this is called only from func_calc_b_int, the
% input is not checked.

% First, allocate b_HT and bJac_HT and initialize to zeros.

num_DOF = (ProbDim.num_species+1)*Grid.num_pts;

b_HT = linspace(0,0,num_DOF)';

bJac_HT = spalloc(num_DOF,num_DOF,Grid.num_pts);


% Next, set offset to start of temperature field.

Tpos_offset = ProbDim.num_species*Grid.num_pts;


% Now, exact a list of the interior points from
% the epsilon values for the first field.

list_int = find(epsilon(1:Grid.num_pts) ~= 0);


% To model the heat transfer to the jacket, we use
% a simple heat transfer coefficient model assuming
% large enough coolant flow rates that the coolant
% temperature is constant.  The following factor
% is used to determine the strength of the heat
% transfer.

jacket_strength = 4*Reactor.U_HT/Reactor.dia;


% For every interior point

for count=1:length(list_int)
   ipoint = list_int(count);

   % set the degree of freedom index for the
   % temperature field at this point
   iDOF = Tpos_offset + ipoint;

   % The contribution to the source term from
   % heat transfer to the reactor is now
```

```
    % calculated.
    b_HT(iDOF) = -jacket_strength * ...
       (State.Temp(ipoint) - Reactor.Temp_cool);

    % The contribution to the Jacobian of the
    % source term is also calculated.
    bJac_HT(iDOF,iDOF) = -jacket_strength;

end


iflag = 1;

return;
```