F-Chart Software

No. 36, Spring 2019

# EESy Solutions

Engineering Equation Solver Newsletter

## Inside this issue:

## Welcome

This is the 36th issue of EESy Solutions, a newsletter that provides news, tips, and other updates for users of the Engineering Equation Solver software. This issue discusses some new Macro commands that make it easy to communicate with MATLAB.

EES has been a commercially available for more than two decades. If you have missed any of the previous issues of EESy Solutions, they can be downloaded from www.fchart.com.

## Macros

Macros are instructions that can be written in a script form to control the execution of EES or, as discussed in this article, the execution of other programs. Macro commands can instruct EES to do almost anything that a user can manually accomplish using the menus or shortcut keys. You can make a plot or solve a Parametric Table, for example.

Macros are just ASCII files but they are most conveniently developed using the Macro Window in EES. Macros have been covered in previous editions of this Newsletter. In this edition we will look specifically at a set of Macro Commands that allow EES to execute commands in MATLAB and exchange information with the program. This capability allows the combined power of EES and MATLAB to be brought to bear on a problem. In addition to the many toolboxes that are available with MATLAB, the program is extremely good at dealing with very large amounts of data or carrying out numerical simulations with a large number of nodes.

## Opening MATLAB from EES

The **MATLAB.Open** Macro command will start MATLAB and open the Command Window.  This command is ignored if MATLAB has been previously opened in this session and not closed with the MATLAB.Quit command.  It is necessary to use the MATLAB.Open command before proceeding with any of the other MATLAB Macro commands in EES.

## Moving Data from EES to MATLAB

The **MATLAB.SetVariable** Macro command can be used to set the value of a variable in MATLAB.  The MATLAB variable can be set to the value of any existing variable within the EES environment.

```
S_E=2
MATLAB.SetVariable(S_M, S_E)
```

Sets the value of the MATLAB Variable S_M to the value of the EES variable S_E.  The MATLAB.SetVariable command also works with strings and arrays.

```
A_E[1..3] =[1,2,3]
MATLAB.SetVariable(A_M, A_E[1..3])
```

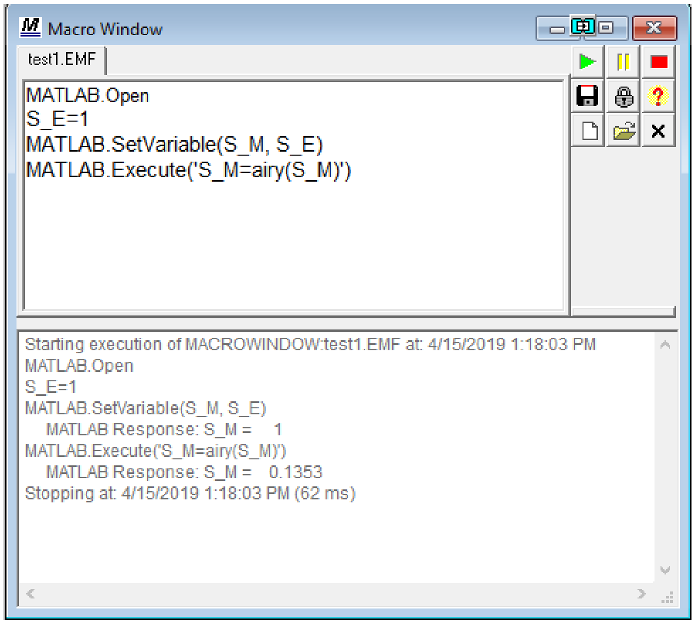
Sets the value of the MATLAB vector A_M to the value of the EES array A_E.

## Executing MATLAB Commands

The **MATLAB.Execute** Macro command can be used to execute any command in the MATLAB Command Window.  This can include direct operations on data provided to MATLAB via the MATLAB.SetVariable Command or other operations, like opening files, reading data, running functions, etc.

```
S_E=1
MATLAB.SetVariable(S_M, S_E)
MATLAB.Execute('S_M=airy(S_M)')
```

Determines the value of the Airy function of the quantity S_E.



```
MATLAB.Open
S_E=1
MATLAB.SetVariable(S_M, S_E)
MATLAB.Execute('S_M=airy(S_M)')
```

```
Starting execution of MACROWINDOW:test1.EMF at: 4/15/2019 1:18:03 PM
MATLAB.Open
S_E=1
MATLAB.SetVariable(S_M, S_E)
   MATLAB Response: S_M =     1
MATLAB.Execute('S_M=airy(S_M)')
   MATLAB Response: S_M =    0.1353
Stopping at: 4/15/2019 1:18:03 PM (62 ms)
```

## Moving Data from MATLAB to EES

The **MATLAB.GetVariable** Macro command will retrieve a variable in the MATLAB workspace.

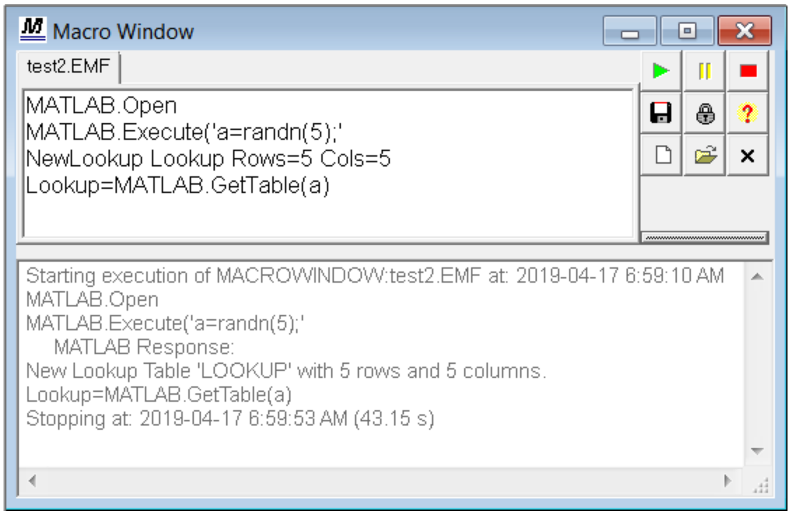
```
MATLAB.Execute(‘a=airy(1)’)
X=MATLAB.GetVariable(a)
```

Assigns the EES variable X to the MATLAB variable a. The MATLAB variable can be a scalar variable as well as a vector or array.

```
MATLAB.Execute(‘a=[1 2 3]’)
X[1..3]=MATLAB.GetVariable(a)
```

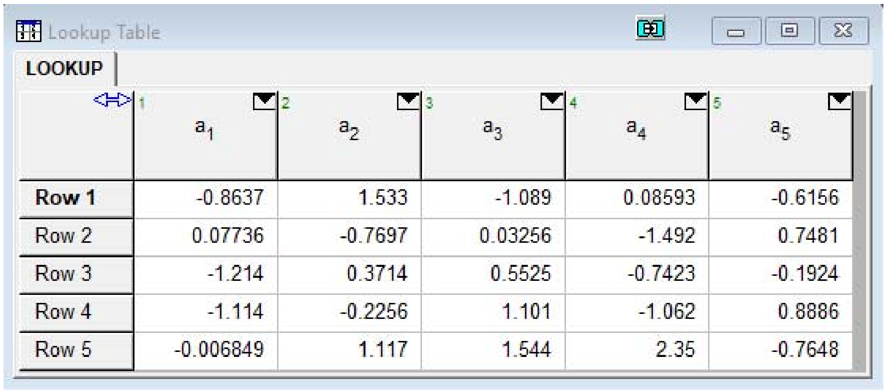
The **MATLAB.GetTable** Macro command retrieves a vector or matrix in the MATLAB workspace and copies it to a Lookup Table in EES.

```
MATLAB.Execute(‘a=randn(5)’)
NewLookup Lookup Rows=5 Cols=5
Lookup=MATLAB.GetTable(a)
```



```
MATLAB.Open
MATLAB.Execute('a=randn(5);'
NewLookup Lookup Rows=5 Cols=5
Lookup=MATLAB.GetTable(a)
```

```
Starting execution of MACROWINDOW:test2.EMF at: 2019-04-17 6:59:10 AM
MATLAB.Open
MATLAB.Execute('a=randn(5);'
    MATLAB Response:
New Lookup Table 'LOOKUP' with 5 rows and 5 columns.
Lookup=MATLAB.GetTable(a)
Stopping at: 2019-04-17 6:59:53 AM (43.15 s)
```



|  | 1<br>$a_1$ | 2<br>$a_2$ | 3<br>$a_3$ | 4<br>$a_4$ | 5<br>$a_5$ |
|---|---|---|---|---|---|
| **Row 1** | -0.8637 | 1.533 | -1.089 | 0.08593 | -0.6156 |
| Row 2 | 0.07736 | -0.7697 | 0.03256 | -1.492 | 0.7481 |
| Row 3 | -1.214 | 0.3714 | 0.5525 | -0.7423 | -0.1924 |
| Row 4 | -1.114 | -0.2256 | 1.101 | -1.062 | 0.8886 |
| Row 5 | -0.006849 | 1.117 | 1.544 | 2.35 | -0.7648 |

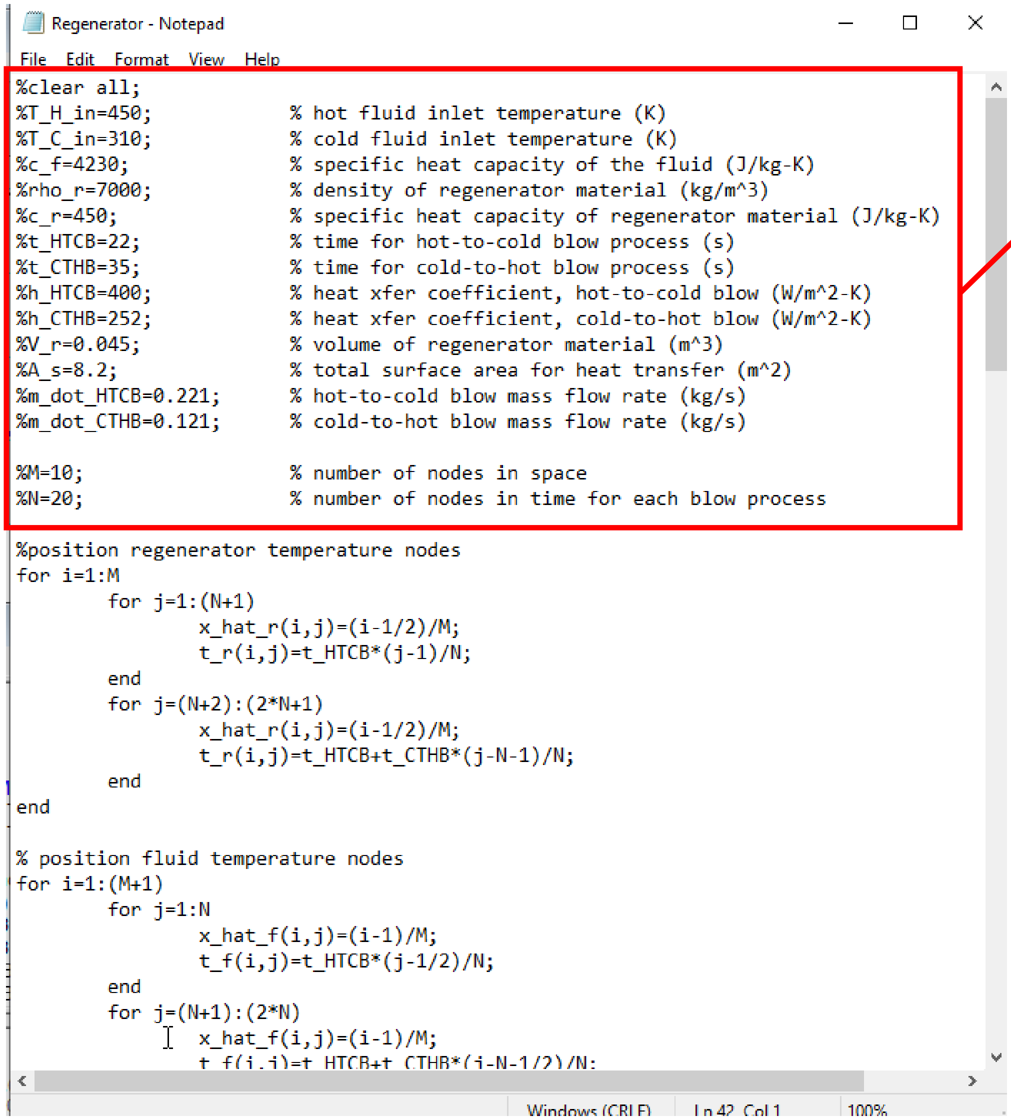
## Closing MATLAB

The **MATLAB.Quit** Macro command closes the instance of MATLAB that was opened previously with the MATLAB.Open command. The open process is slow, so don’t close MATLAB until you are done using it. Again, repeated use of the MATLAB.Open command does **not** open multiple instances of MATLAB.

## Example using MATLAB with EES

The book Heat Transfer by Nellis and Klein (2009) develops a numerical model of a regenerator in MATLAB in Section 8.10.5.  Here let's illustrate the use of EES with MATLAB by setting up the regenerator operating conditions and geometry in EES and using EES' internal property database and the Heat Transfer library to determine the required inputs for the MATLAB model, assuming helium is flowing through a packed bed of spheres.  These inputs are provided to MATLAB where the model is run and then finally the results (in this case the temperature of the regenerator as a function of time and space) are returned to EES and used to generate a plot.

The script developed in MATLAB is saved to a folder that lies in MATLAB's path as Regenerator.m and the original inputs that were hard-coded in the script are removed.



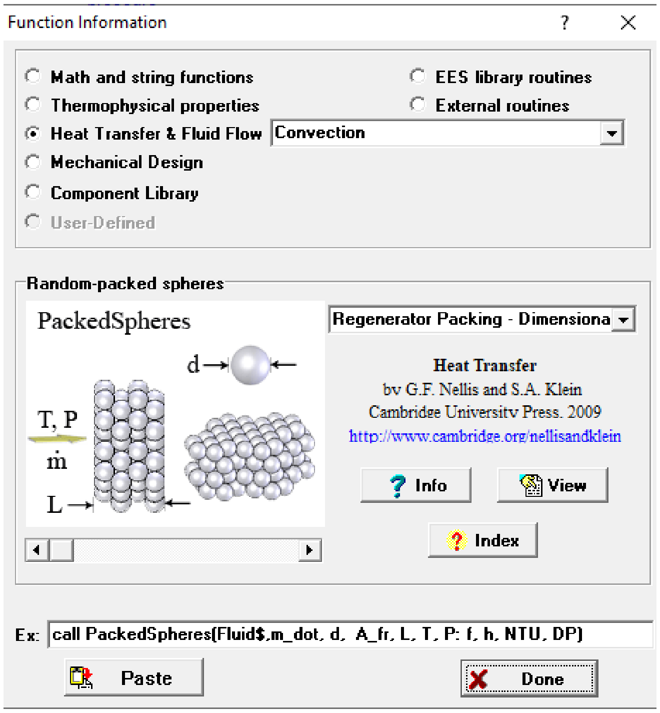
```
%clear all;
%T_H_in=450;            % hot fluid inlet temperature (K)
%T_C_in=310;            % cold fluid inlet temperature (K)
%c_f=4230;              % specific heat capacity of the fluid (J/kg-K)
%rho_r=7000;            % density of regenerator material (kg/m^3)
%c_r=450;               % specific heat capacity of regenerator material (J/kg-K)
%t_HTCB=22;             % time for hot-to-cold blow process (s)
%t_CTHB=35;             % time for cold-to-hot blow process (s)
%h_HTCB=400;            % heat xfer coefficient, hot-to-cold blow (W/m^2-K)
%h_CTHB=252;            % heat xfer coefficient, cold-to-hot blow (W/m^2-K)
%V_r=0.045;             % volume of regenerator material (m^3)
%A_s=8.2;               % total surface area for heat transfer (m^2)
%m_dot_HTCB=0.221;      % hot-to-cold blow mass flow rate (kg/s)
%m_dot_CTHB=0.121;      % cold-to-hot blow mass flow rate (kg/s)

%M=10;                  % number of nodes in space
%N=20;                  % number of nodes in time for each blow process

%position regenerator temperature nodes
for i=1:M
        for j=1:(N+1)
                x_hat_r(i,j)=(i-1/2)/M;
                t_r(i,j)=t_HTCB*(j-1)/N;
        end
        for j=(N+2):(2*N+1)
                x_hat_r(i,j)=(i-1/2)/M;
                t_r(i,j)=t_HTCB+t_CTHB*(j-N-1)/N;
        end
end

% position fluid temperature nodes
for i=1:(M+1)
        for j=1:N
                x_hat_f(i,j)=(i-1)/M;
                t_f(i,j)=t_HTCB*(j-1/2)/N;
        end
        for j=(N+1):(2*N)
                x_hat_f(i,j)=(i-1)/M;
                t_f(i,j)=t_HTCB+t_CTHB*(j-N-1/2)/N;
```

These inputs are all commented out of the MATLAB pro-

# Example using MATLAB with EES (continued)

The EES program takes as inputs the fluid, inlet temperatures, and pressures as well as the geometry and packing material.

```
$UnitSystem SI Mass J K Pa

"Operating conditions"
F$='Helium'                           "fluid"
T_H_in=20 [K]                         "hot inlet temperature"
T_C_in=5 [K]                          "cold inlet temperature"
P=500 [kPa]*Convert(kPa,Pa)   "pressure"
m_dot=0.5 [g/s]*Convert(g/s,kg/s) "mass flow rate"
time=5 [s]                             "blow time"

"Geometry"
D=5 [cm]*Convert(cm,m)           "diameter"
L=6 [cm]*Convert(cm,m)            "length"
d_s=0.5 [mm]*Convert(mm,m)    "sphere size"
phi=0.34 [-]                           "porosity"
R$='Lead'                              "packing material"
```



Function Information

○ Math and string functions
○ Thermophysical properties
● Heat Transfer & Fluid Flow — Convection
○ Mechanical Design
○ Component Library
○ User-Defined
○ EES library routines
○ External routines

Random-packed spheres

Regenerator Packing - Dimensiona

Heat Transfer
bv G.F. Nellis and S.A. Klein
Cambridge Universitv Press. 2009
http://www.cambridge.org/nellisandklein

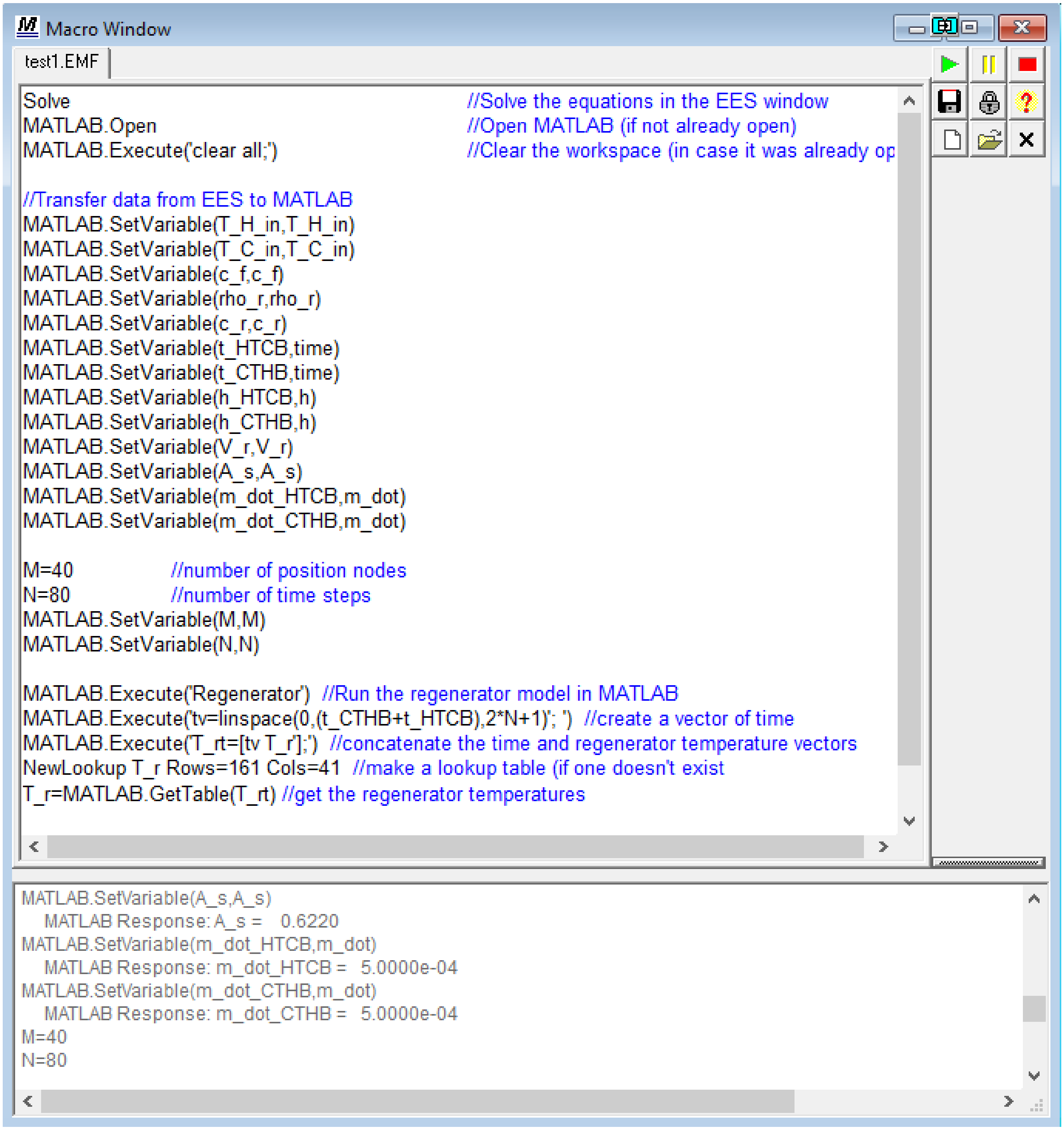
Ex: call PackedSpheres(Fluid$,m_dot, d, A_fr, L, T, P: f, h, NTU, DP)

The properties required are computed using EES' internal property database and the heat transfer co-efficient is computed using the PackedSpheres procedure.

```
T_bar=(T_H_in+T_C_in)/2                       "average temperature"
c_f=cP(F$,T=T_bar,P=P)                        "fluid specific heat capacity"

rho_r=Density(R$,T=T_bar)                     "density of regenerator packing"
c_r=cP(R$,T=T_bar)                            "specific heat capacity of regenerator packing"

A_fr=pi*D^2/4                                 "frontal area"
call PackedSpheres(F$,m_dot, d_s,  A_fr, L, T_bar, P: , h, , ) "get heat transfer coefficient"
V_r=A_fr*L*(1-phi)                            "volume of packing"
A_s=4*(1-phi)*A_fr*L/d_s                      "surface area"
```

## Example using MATLAB with EES (continued)

A Macro is created that opens MATLAB, transfers the required input variables to MATLAB, runs the program and then sends some of the resulting data back to EES where it is stored in a Lookup Table.

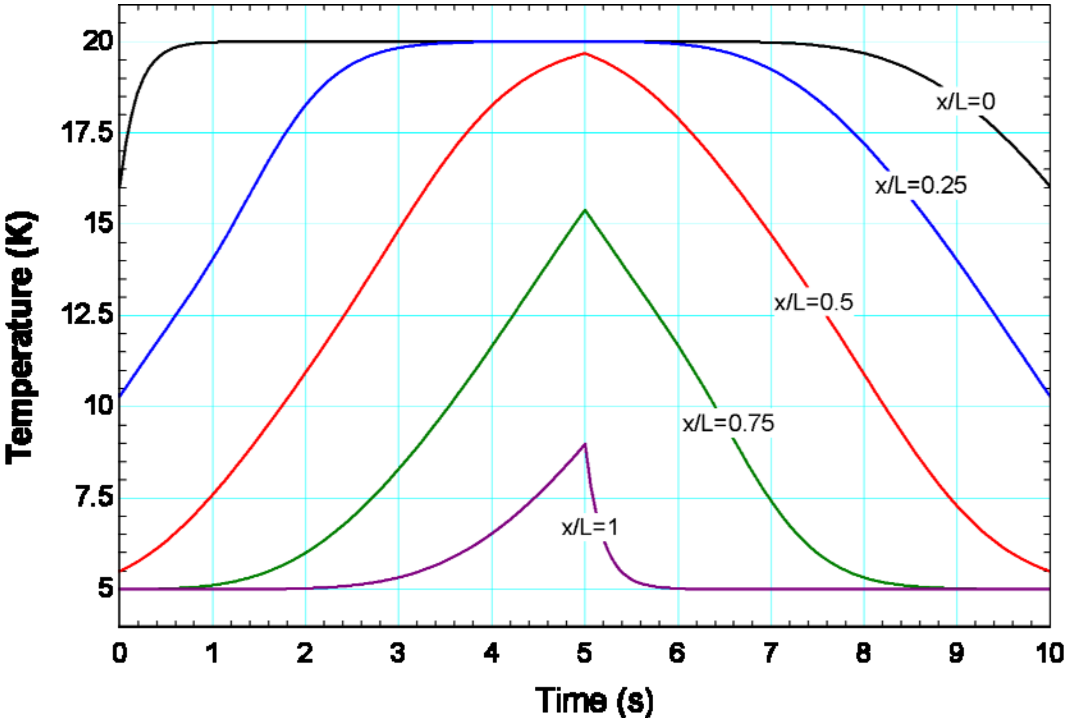


```
Solve                                   //Solve the equations in the EES window
MATLAB.Open                             //Open MATLAB (if not already open)
MATLAB.Execute('clear all;')            //Clear the workspace (in case it was already op

//Transfer data from EES to MATLAB
MATLAB.SetVariable(T_H_in,T_H_in)
MATLAB.SetVariable(T_C_in,T_C_in)
MATLAB.SetVariable(c_f,c_f)
MATLAB.SetVariable(rho_r,rho_r)
MATLAB.SetVariable(c_r,c_r)
MATLAB.SetVariable(t_HTCB,time)
MATLAB.SetVariable(t_CTHB,time)
MATLAB.SetVariable(h_HTCB,h)
MATLAB.SetVariable(h_CTHB,h)
MATLAB.SetVariable(V_r,V_r)
MATLAB.SetVariable(A_s,A_s)
MATLAB.SetVariable(m_dot_HTCB,m_dot)
MATLAB.SetVariable(m_dot_CTHB,m_dot)

M=40            //number of position nodes
N=80            //number of time steps
MATLAB.SetVariable(M,M)
MATLAB.SetVariable(N,N)

MATLAB.Execute('Regenerator')  //Run the regenerator model in MATLAB
MATLAB.Execute('tv=linspace(0,(t_CTHB+t_HTCB),2*N+1)'; ')  //create a vector of time
MATLAB.Execute('T_rt=[tv T_r];')  //concatenate the time and regenerator temperature vectors
NewLookup T_r Rows=161 Cols=41  //make a lookup table (if one doesn't exist
T_r=MATLAB.GetTable(T_rt) //get the regenerator temperatures
```

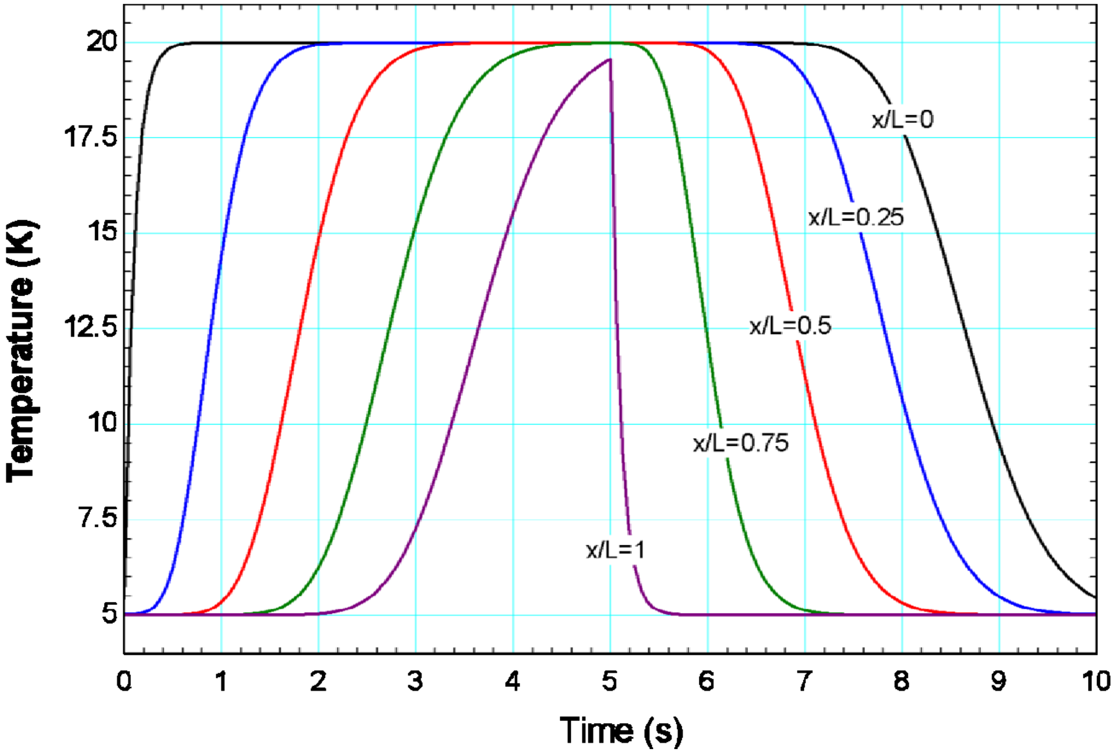
```
MATLAB.SetVariable(A_s,A_s)
   MATLAB Response: A_s =    0.6220
MATLAB.SetVariable(m_dot_HTCB,m_dot)
   MATLAB Response: m_dot_HTCB =   5.0000e-04
MATLAB.SetVariable(m_dot_CTHB,m_dot)
   MATLAB Response: m_dot_CTHB =   5.0000e-04
M=40
N=80
```

## Example using MATLAB with EES (continued)

The data in the Lookup Table can be used to create a plot (shown below).



If the Automatic Update feature is selected in the plot then it is possible to change an input in EES and simply re-run the Macro to see how the temperature distribution changes. For example, if the mass flow rate is doubled, the new temperature distribution is shown below.



## Recent Changes to EES

- Additional macro commands have been added to facilitate interactions between EES and MATLAB
- The LaTexPDF macro command automatically prepares a .pdf or .TeX file
- Thermodynamic and transport property for Acetylene, R417A, R515A, and R545A have been added
- The NormalBoilingPt function can be used for brines as well as other fluids
- Condenser5_CL, Condenser6_CL, Evaporator5_CL and Evaporator6_CL have been added to the component library. These models are for design and simulation of condensers and evaporators that use a liquid rather than air.
- Unit checking can optionally be applied while entering equations in the Lookup Table
- The ModifyPlot macro command can now be used to change the size and location of the plot as well as the gridline color and whether the plot toolbar is visible.
- Clicking on a tab in the Macro Window brings up a popup menu that allows you to save, close, or clear the log file for that macro.
- Improved correlations have been implemented for Hydrogen and Parahydrogen.
- The PsychProps procedure has been added which returns all psychrometric properties in one call (like the RealThermalProps procedure)
- The Delete Parametric Tables, Delete Lookup Tables, and Delete Plot Windows dialogs have been revised to use check box lists allowing easy manipulation of multiple tables and plots.
- The color of the constant property lines on a property plot can be specified using the Property Plot command.

F-Chart Software

PO Box 44042
Madison, WI, 53744

Phone/FAX: 608-274-4262
Internet: http://fchart.com
E-mail: info@fchart.com

## Instant Update & Technical Service

EES uses a different model for updating than most other programs. Each time that there is a change in the EES program, either to correct a problem or to add a new feature, the version number is incremented by 0.001 and the latest version of EES is placed on our website. Although the program has become very robust and stable, there have been many new versions of EES released since the last EESy Solutions was distributed.

Any user who has a current subscription to our Instant Update Service can download the latest version. All new licenses of EES are provided with one year of Instant Update Service. The fee to continue Instant Update Service after the first year is 20% of the current cost of the program per year if renewed within 12 months after expiration of the Service. Contact F-Chart Software if you wish to resubscribe to the Instant Update Service.