**User-defined Functions in MATLAB**

MATLAB provides users with the ability to define their own functions. This capability is useful in generating solutions to specific problems that a user may encounter. In MATLAB, you have built-in functions such as \texttt{cos} for the cosine function and \texttt{log} for the natural logarithm. When you enter a command such as

\begin{verbatim}
>> log(10)
\end{verbatim}

MATLAB executes a function call to the \texttt{log} function, computes $ln(10) = 2.3026$, and responds with

\begin{verbatim}
ans =
2.3026
\end{verbatim}

**How to write your own function in MATLAB**

**Step 1. Define the problem.**
Here are some example problems that could be simulated in MATLAB with user-defined functions:
- Solve for the compressibility of a block of concrete
- Solve for the time required to heat a potato in a microwave oven
- Solve for the projected path of a hurricane

**Step 2. Define the input and output parameter(s) and the function name.**
Names must have no spaces or special characters such as * - + / $ ! @ ~ ( ) # % &. Names cannot start with a digit. Valid names are shown in Table 1.

<table>
<thead>
<tr>
<th>Example names for input parameters</th>
<th>Example names for output parameters</th>
<th>Example names for user-defined functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base</td>
<td>MilesPerGallon</td>
<td>GasMileage</td>
</tr>
<tr>
<td>Height</td>
<td>CircleArea</td>
<td>Calc_Hypotenuse</td>
</tr>
<tr>
<td>Gallons_of_gas</td>
<td>VolumeDistributionOf</td>
<td>ConcreteCompression</td>
</tr>
<tr>
<td></td>
<td>PotatoTemerature</td>
<td></td>
</tr>
<tr>
<td>Diameter</td>
<td>PercentError</td>
<td>PotatoHeating</td>
</tr>
<tr>
<td>SourceVoltage</td>
<td></td>
<td>HurricaneForecast</td>
</tr>
<tr>
<td>PotatoMass</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PotatoVolume</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Avoid non-descript names such as X, y, x1, y2, X2, Y1, input1, output2, ans. It is much more informative to use descriptive names, such as the ones provided above.

Avoid using reserved words for variable and function names. Reserved words are the names of built-in functions and functions contained in MATLAB toolboxes. If you use a reserved word for a user-defined function or variable name, then you will not be able to use the function you replaced. For example, if you create a variable called `cos`, then you will not be able to use MATLAB’s built-in cosine function.

To erase a variable or function name that is created in error, use the `clear` command. If you are not sure whether or not a name is a reserved word, use the `which` command to search for reserved words.

**Step 3. Write an algorithm to compute the output(s) given the input(s)**

**Step 4. Test the algorithm with a known solution**

**Step 5. Format the function header information into a standard format.**

An example of the header information for a user-defined function is provided in Figure 1.

```matlab
function h = daub(Nh)
    % function h = daub(Nh)
    %
    % Generate filter coefficients for the Daubechies orthogonal wavelets.
    %
    % h = filter coefficients of Daubechies orthonormal compactly supported wavelets
    % Nh = length of filter.
    %
    % Written by Kevin S. Amaratunga 08/25/95
    % Email: ksamarat@MIT.EDU
    % url: http://www.mit.edu:8001/people/ksamarat/kevin.html
```

Figure 1. Header information in a user-defined function

Notice that all the descriptive information is within comments. The `%` sign forces a line to be commented. This means that the information on the commented line will not be executed.

When one invokes the `help` command followed by the name of the user-defined function, the descriptive information in comments will be displayed. For example, if one were to enter the command
Step 5. Write a main program with a function call.
Step 5. Add user-interface, plotting routines, other supporting stuff to the main program.

**Detailed User-Defined Function Development**

**R-L Series**

**AC Circuit Analysis**

Goal: Write a MATLAB program to compute circuit parameters in an R-L series circuit driven by an AC voltage source.

**Step 1) Define user-provided inputs.**

We need to define various circuit parameters of the R-L circuit. These are resistance in \( \Omega \), inductance in henries, source frequency in Hz, and source voltage. Now that we have decided upon the input parameters, let’s think of variable names. Suppose we choose the following variable names:

- Resistance
- Inductance
- SourceFreq
- SourceVoltage

**Step 2) Define computed outputs.**

Suppose we would like to compute the electric current in the R-L series circuit, the phase angle in degrees between the source voltage and current and the power dissipated in Watts. Now we need to decided upon variable names for the output parameters. Suppose we choose the following output variable names:

- Current
- PhaseAngle
- PowerDissipated

**Step 3) Decide upon a function name.**

For the RL series problem, we will choose the name, `RLseries`.

**Step 4) Write the function to compute the outputs given the input parameters.**

The entire user-defined function, `RLseries`, is shown in Table 1. This file is saved with the name `RLseries.m`

**Step 5) Write a main program to execute the function for specific input values.**

The entire user-defined main program, `RLseriesMain`, is shown in Table 2. This file is saved with the name `RLseriesMain.m`. The two sets of circuit parameters modeled in this example are

- \( R = 10 \) ohms, \( L = 0.1 \) henries, \( f = 60 \) Hz, and \( V = 100 \) volts and
- \( R = 5 \) ohms, \( L = 0.1 \) henries, \( f = 60 \) Hz, and \( V = 100 \) volts

**Step 6) Execute the program by entering the name of the main program in the MATLAB command window.**

For the MATLAB main program, `RLseriesMain`, we would enter the command `>> rlseriesmain`. Notice that upper and lower case is not important in executing program names.
Entering this command provides the current in amps, phase angle in degrees and power in Watts dissipated for two sets of circuit parameters:


g > rlseriesmain
Current =
  2.5639
PhaseAngle =
   -75.1439
PowerDissipated =
    65.7366
Current =
  2.6296
PhaseAngle =
   -82.4450
PowerDissipated =
    34.5728

So we can summarize from this result that the circuit with

R = 10 ohms, L = 0.1 henries, f = 60 Hz, and V = 100 volts

produces a current of 2.5639 amps, in which the current lags the source voltage by 75.1439°, and the power dissipated in the circuit is 65.7366 Watts. The second set of results tells us that the circuit with

R = 5 ohms, L = 0.1 henries, f = 60 Hz, and V = 100 volts

produces a current of 2.6296 amps, in which the current lags the source voltage by 82.445°, and the power dissipated in the circuit is 34.5728 Watts.
function   [Current, PhaseAngle, PowerDissipated]  = RLseries(Resistance, Inductance, SourceFreq, SourceVoltage)

%   [Current, PhaseAngle, PowerDissipated]  = RLseries(Resistance, Inductance, SourceFreq, SourceVoltage)
%    This function computes the current, phase angle and real power in an
%    RL series circuit which is driven by an AC source.
%
% Inputs:
%       Resistance  =  resistance in ohms
%       Inductance  =  inductance in henries
%       SourceFreq  =  source frequency in Hz
%       SourceVoltage = source voltage in volts
% Outputs:
%       Current     =  current in amps
%       PhaseAngle  =  phase angle in degrees between the current and the source voltage
%       PowerDissipated  =  power in Watts dissipated in the circuit
%
% Name, date
% Western Carolina University

% radian frequency
w = 2 * pi * SourceFreq;

% reactance
Xl = w * Inductance;

% impedance
Z = Resistance + j * Xl;

% Current
Current = abs(SourceVoltage / Z);

% phase angle in degrees
PhaseAngle = angle(SourceVoltage / Z) * 180 / pi;

% Power dissipated
PowerDissipated = Current^2 * Resistance;
% Main program for RL series circuit analysis
% Simulate a series RL circuit with:
%   R = 10 ohms, L = .1 henries, f = 60 Hz, and V = 100 volts
[Current, PhaseAngle, PowerDissipated] = RLseries(10, .1, 60, 100)

% Simulate a series RL circuit with:
%   R = 5 ohms, L = .1 henries, f = 60 Hz, and V = 100 volts
[Current, PhaseAngle, PowerDissipated] = RLseries(5, .1, 60, 100)