

```
% % Mfile name
%   mtl_aae_sim_errorprop.m

% Version:
%   Matlab R2007a

% Revised:
%   August 28, 2008

% Authors:
%   Luke Snyder, Dr. Autar Kaw
%   University of South Florida
%   kaw@eng.usf.edu
%   Website: http://numericalmethods.eng.usf.edu

% Purpose
%   To show error propogation in a multivariable function.

% Keywords
%   Young's Modulus
%   Strain

% Clearing all data, variable names, and files from any other source and
% clearing the command window after each successive run of the program.
clc
clear all

% Inputs:
%   Enter inputs to calculate strain based on the formula
%   Strain = F/(h^2E)

%   This is the only place in the program where the user makes the changes
%   based on their wishes.
% Enter Force (in N):
Force = 72;

% Enter relative measurement error in percentage for force:
RME_Force = 2.5;

% Enter length or width of cross section (in m):
h = 4e-3;

% Enter relative measurement error in percentage for h:
RME_h = 2.5;

% Enter Young's Modulus (in Pa):
E = 70e9;

% Enter relative measurement error in percentage for Modulus:
RME_Young = 2.5;

% *****

disp(sprintf('Error Propagation in a Simple Function Evaluation'))
```

```

disp(sprintf('\nUniversity of South Florida'))
disp(sprintf('United States of America'))
disp(sprintf('kaw@eng.usf.edu'))
disp(sprintf('Website: http://numericalmethods.eng.usf.edu'))
disp(sprintf('\nNOTE: This worksheet illustrates the use of Matlab to use a simple'))
disp(sprintf('equation to show how error propagates in a multivariable function.'))

disp(sprintf('\n*****Introduction*****'))

disp(sprintf('\nThe following worksheet illustrates how error propagates in a '))
disp(sprintf('simple function evaluation. This is due largely to the fact that '))
disp(sprintf('relative error calculations are based on round-off errors or/and '))
disp(sprintf('measurement errors that exist inherently in simple devices used to '))
disp(sprintf('take measurements. The user will enter force, length or width of '))
disp(sprintf('the cross section, and Youngs Modulus data in the Input section'))
disp(sprintf('of the program, as well as the corresponding relative'))
disp(sprintf('measurement error for each input. From here, Matlab will '))
disp(sprintf('be used to demonstrate how this error propagates.'))

disp(sprintf('\nThe formula used for illustration is Strain = F/(h^2E)'))
% Calculating the actual measurement error for each input.
Meas_Error_Force = Force * (RME_Force/100);
Meas_Error_h = h * (RME_h/100);
Meas_Error_Young = E * (RME_Young/100);

disp(sprintf('\n\n*****Input Data*****'))
disp(sprintf('\nForce applied, Force = %g N',Force));
disp(sprintf('\nRelative Measurement Error for force, RME_Force = %g %%',RME_Force));
disp(sprintf('\nCross Sectional length or width, h = %g',h));
disp(sprintf('\nRelative Measurement Error for cross section dimension, RME_h = %g %%',
RME_h));
disp(sprintf('\nYoungs Modulus, E = %g',E));
disp(sprintf('\nRelative Measurement Error for Youngs Modulus, RME_Young = %g %%',
RME_Young));

disp(sprintf('\n\n*****Procedure*****'))
disp(sprintf('\nForce applied, Force = %g N',Force));
disp(sprintf('\nRelative Measurement Error for force, RME_Force = %g %%',RME_Force));
disp(sprintf('\nMeasurement Error for Force = +/-%g N',Meas_Error_Force));

disp(sprintf('\nCross Sectional length or width, h = %g',h));
disp(sprintf('\nRelative Measurement Error for cross section dimension, RME_h = %g %%',
RME_h));
disp(sprintf('\nMeasurement Error for h = +/-%g m',Meas_Error_h));

disp(sprintf('\nYoungs Modulus, E = %g',E));
disp(sprintf('\nRelative Measurement Error for Youngs Modulus, RME_Young = %g %%',
RME_Young));
disp(sprintf('\nMeasurement Error for Youngs Modulus = +/-%g Pa',Meas_Error_Young));

% First we must calculate the strain in the simulation given by the
% equation  $e = F/h^2 * E$ .
strain = Force/(h^2 * E);

```

```
disp(sprintf('\n\nThe nominal strain calculated at the nominal value for all inputs'))
disp(sprintf('is, strain = %g',strain));

% Using the formula for maximum possible error to derive the maximum
% possible error in the measured strain. The "syms" command is used
% because we need to differentiate the formula for strain with respect to
% each variable.

% Note that different variable names are used because the previous variable
% names represented actual numbers meaning that if the partial differential
% is taken, in all cases it will be zero. The subs command is then used to
% input the values given by the user into the partial differential equation.

syms F h1 E1

strain1 = F/((h1^2)*E1);
% Partial with respect to Force:
partial_wrt_Force = diff(strain1,F);

% Calculating the error for Force:
a = subs(partial_wrt_Force,E1,E);
partial_wrt_Force = abs(subs(a,h1,h));

% Partial with respect to length or width dimension (h):
partial_wrt_h = diff(strain1,h1);

% Calculating the error for h:
a = subs(partial_wrt_h,h1,h);
b = subs(a,E1,E);
partial_wrt_h = abs(subs(b,F,Force));

% Partial with respect to Young's Modulus:
partial_wrt_Young = diff(strain1,E1);

% Calculating the error for Young's Modulus:
a =subs(partial_wrt_Young,E1,E);
b = subs(a,h1,h);
partial_wrt_Young = abs(subs(b,F,Force));

% Calculating the range of value for each measured quantity based in the
% RME percentages given.
Range_Force = (RME_Force/100) * Force;

Range_h = (RME_h/100) * h;

Range_Young = (RME_Young/100) * E;

% Calculating the Maximum possible error by adding the absolute value of
% the partial differential equations multiplied by their respective range
% of variation in the measured value.

% Using the subs command to input the values given by the user into the
% partial differential equation.
% Partial with respect to Force:
```

```

a = subs(partial_wrt_Force,E1,E);
partial_wrt_Force = subs(a,h1,h);

% Calculating the individual error contributions due to Force, Area, and
% the length or width dimension respectively.
% With Respect to Force:
Error_Force = partial_wrt_Force * Range_Force;

% With Respect to h:
Error_h = partial_wrt_h * Range_h;

% With Respect to Young's Modulus:
Error_Young = partial_wrt_Young * Range_Young;

% Calculating total error by adding all the individual contributions given
% by the user defined inputs.
Tot_Error = (Error_Force + Error_h + Error_Young);
disp(sprintf('\nThe maximum total Error calculated is, Tot_Error = %g',Tot_Error));
disp(sprintf('\nThe contribution to total error due to force is, Error_Force = %g',
Error_Force));

% Calculating the percent contribution of force error to the total error.
Rel_Force_Error = (Error_Force/Tot_Error) * 100;
disp(sprintf('Percentage of total error, Rel_Force_Error = %g %%',Rel_Force_Error))

disp(sprintf('\nThe contribution to total error due to h is, Error_h = %g',Error_h))

% Calculating the percent contribution of h error to total error.
Rel_h_Error = (Error_h/Tot_Error) * 100;
disp(sprintf('Percentage of total error, Rel_h_Error = %g %%',Rel_h_Error));

disp(sprintf('\nThe contribution to total error due to Youngs Modulus is, '));
disp(sprintf('Error_Young = %g',Error_Young));

% Calculating the percent contribution of Young's Modulus error to total
% error.
Rel_Young_Error = (Error_Young/Tot_Error) * 100;
disp(sprintf('Percentage of total error, Rel_Young_Error = %g %%',Rel_Young_Error));

% Calculating the effective range that the axial strain could be within
% using the total error calculation.
Low_strain = (strain - Tot_Error);

High_strain = (strain + Tot_Error);

disp(sprintf('\n-----'));
disp(sprintf('\nThe nominal strain calculated at the nominal value for all inputs'))
disp(sprintf('is, strain = %g',strain));

disp(sprintf('Minimum value of strain = %g',Low_strain));
disp(sprintf('Maximum value of strain = %g',High_strain));

% Calculating the percent relative measurement error for strain.
RME_strain = (Tot_Error/strain) * 100;

```

```
disp(sprintf('The maximum measurement error for strain is RME_strain = %g %%',RME_strain))
```

```
disp(sprintf('\n\n*****Conclusion*****'))
disp(sprintf('If a calculation is made with numbers that are not exact, then the'))
disp(sprintf('calculation itself will have an error. Since the final results of an'))
disp(sprintf('experiment are not usually directly measured but are some '))
disp(sprintf('function of one or more of the measured quantities, it is important '))
disp(sprintf('to understand and utilize propagation of error concepts to better '))
disp(sprintf('interpret and represent experimental results.'))
```

```
disp(sprintf('\n\n*****References*****'))
disp('See: <a href = "http://numericalmethods.eng.usf.␣
edu/mws/gen/01aae/mws_gen_aae_txt_propagationoferrors.pdf">Propagation of Errors</a>')
```

```
disp(sprintf('\n\nLegal Notice: The copyright for this application is owned'))
disp(sprintf('by the author(s). Neither MathWorks nor the author(s)'))
disp(sprintf('are responsible for any errors contained within and are '))
disp(sprintf('not liable for any damages resulting from the use of this'))
disp(sprintf('material. This application is intended for non-commercial,'))
disp(sprintf('non-profit use only. Contact the author for permission if'))
disp(sprintf('you wish to use this application in for-profit activities.'))
```