```
% Housekeeping
clc
clear
g = 9.81; % [m/s^2]
syms f(h_n, h_0) f(tn, tn_1) f(ts, h_0)

% Trial_1 = table2array(readtable('0756 Data'));
% Trial_2 = table2array(readtable('0757 Data'));
% Trial_3 = table2array(readtable('0759 Data'));
% Trial_4 = table2array(readtable('0760 Data'));
% Trial_5 = table2array(readtable('0761 Data'));
%
% time = [Trial_1(:,1); Trial_2(:,1); Trial_3(:,1); Trial_4(:,1);
 Trial_5(:,1)];
% X = [Trial_1(:,2); Trial_2(:,2); Trial_3(:,2); Trial_4(:,2);
 Trial_5(:,2)];
% Y = [Trial_1(:,3); Trial_2(:,3); Trial_3(:,3); Trial_4(:,3);
 Trial_5(:,3)];

%%Data
load('Data_Height.mat');
load('Data_Time.mat');

time = [Data_Time(:,1); Data_Time(:,2); Data_Time(:,3);
 Data_Time(:,4); Data_Time(:,5)];
Y = [Data_Height(:,1); Data_Height(:,2); Data_Height(:,3);
 Data_Height(:,4); Data_Height(:,5)];
Y0 = [Data_Height(1,1); Data_Height(1,2); Data_Height(1,3);
 Data_Height(1,4); Data_Height(1,5)];
Ts = [sum(Data_Time(:,1)); sum(Data_Time(:,2)); sum(Data_Time(:,3));
 sum(Data_Time(:,4)); sum(Data_Time(:,5))];

% Find standard deviations:
std_T = std(time);
% std_X = std(X);
std_Y = std(Y);
std_Y0 = std(Y0);
std_Ts = std(Ts);

% error propagation for first derivative
% First find partials:
f(h_n, h_0) = (h_n/h_0)^(1/2);
p1 = diff(f(h_n, h_0),h_n);
p2 = diff(f(h_n, h_0),h_0);

% iterating through a vector of heights
% make a loop to get an error of partials
p1_val = zeros(35,1);
p2_val = zeros(35,1);

for i = 2:35
    p1_val(i) = 1/(2*Y(i-1)*(Y(i)/Y(i-1))^(1/2));
    p2_val(i) = -Y(i)/(2*Y(i-1)^2*(Y(i)/Y(i-1))^(1/2));
end


% end up with an array of errors and average them --> that is your
% sensitivity

% Use general method
error_1 = zeros(35,1);

for j = 1:35
    error_1(j) = sqrt((p1_val(j)*std_Y)^2 + (p2_val(j)*std_Y)^2);
end

% error propagation for second derivative
f(tn, tn_1) = (tn/(tn_1));
p3 = diff(f(tn, tn_1),tn);
p4 = diff(f(tn, tn_1),tn_1);

p3_val = zeros(25,1);
p4_val = zeros(25,1);

for i = 2:25
    p3_val(i) = 1/time(i-1);
    p4_val(i) = -time(i)/time(i-1)^2;
end

% General method
error_2 = zeros(472,1);

for j = 1:25
    error_2(j) = sqrt((p2_val(j)*std_Y)^2 + (p3_val(j)*std_Y)^2);
end

% error propagation for third derivative
f(ts, h_0) = (ts - sqrt((2*h_0)/g))/(ts + sqrt((2*h_0)/g));
p5 = diff(f(ts, h_0),ts);
p6 = diff(f(ts, h_0),h_0);

p5_val = zeros(25,1);
p6_val = zeros(25,1);

for i = 2:25
    p5_val(i) = 1/(time(i) + ((200*Y(i-1))/981)^(1/2)) - (time(i) -
 ((200*Y(i-1))/981)^(1/2))/(time(i) + ((200*Y(i-1))/981)^(1/2))^2;
    p6_val(i) = - 100/(981*((200*Y(i-1))/981)^(1/2)*(time(i)
 + ((200*Y(i-1))/981)^(1/2))) - (100*(time(i) -
 ((200*Y(i-1))/981)^(1/2)))/(981*((200*Y(i-1))/981)^(1/2)*(time(i) +
 ((200*Y(i-1))/981)^(1/2))^2);
end

% General method
error_3 = zeros(472,1);
```

```
for j = 1:25
    error_3(j) = sqrt((p5_val(j)*std_Y)^2 + (p6_val(j)*std_Y)^2);
end

% Average each array of errors
m1 = mean(error_1);
m2 = mean(error_2);
m3 = mean(error_3);

% Display error for each method
fprintf('Method 1 Error: %f\n',m1);
fprintf('Method 2 Error: %f\n',m2);
fprintf('Method 3 Error: %f\n',m3);
```

```
Method 1 Error: 1.089135
Method 2 Error: 0.060919
Method 3 Error: 0.019755
```

Published with MATLAB® R2019a