```
function [ymin,ymax,tstep,tauTime, isHeating] = M4Alg2_014_05(data)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% ENGR 132
% Program Description
%   Calculates min max timestep and tau variables and outputs them
% Version Changes: Modified some hard coded values to deal with all
 the data
%
% Function Call
%  [ymin,ymax,tstep,tauTime, isHeating] = M4Alg2_014_05(data)
%
% Input Arguments
%   1. data: contains the raw data for noisy/clean cooling/heating
%
% Output Arguments
% 1. ymin - the mimimum value of the data
% 2. ymax - the maximum value of the data
%   3. tstep - timeStep of the passed data
% 4. tauTime - the time constant of the data
%   5. isHeating - boolean variable thats true or false depeding on if
 the
%   data is heating or cooling
%
% Assigment Information
%   Assignment:       Final Project
%   Authors:             Peter Swales, pswales@purdue.edu
%       Colin Jamison, cjamison@purdue.edu
%   Team ID:            014-05
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Peter Swales: coded lines 1-40
% Colin Jamison coded lines 42-56
time = data(:,1); % column vector of time(sec)
temp = data(:,2); % column vector of temperature(C)
tempFirst100 = temp(1:100); %seperates the first hundred temp data
 points
avgPrior = mean(tempFirst100); %inititalizes the average before the
 value of n
avgDiff = 0; % initializes the average difference to enter the while
 loop
n = 101; %initializes data point number to the first value after the
 100 used to initialize the average

while abs(avgDiff) < 1 %tests to see if the loop has reached tstep
    avgPrior = mean(temp(1:n)); %average of all temp data before the
 nth data point
    ntest = n + 50; %calculates the max for the range of n values to
 test the average for
    avgAfter = mean(temp(n:ntest)); % average of temp values from the
 nth value to the "n+50"th value
    avgDiff = avgPrior - avgAfter; % difference between the averages:
 will be greater than one at tstep
```

```
    n = n + 5; % increments n by 5 to reduce redundant calculations
end
tstep = time(n); % the last n value will be the location of tstep
ymin = avgPrior; % ymin is the average of all data points before tstep

nNew = n; %initializes a new value of n to keep track of tsteps
 location
ymaxData = temp(length(temp) - 50:length(temp)); % the last 50
 temperatures of the data set
ymax = mean(ymaxData); % takes the average of the last 50 temps to
 find ymax
fac1 = .632; % initializes a factor for heating
fac2 = 5; %intitializes a factor for heating, the value 5 is used to
 ensure the loop collects all data points needed to calculate tau
isHeating = 1; %EDITED BY ALEX PIEPRZYCKI
if ymin > ymax %true if data set is for cooling instead of heating
    isHeating = 0; %EDITED BY ALEX PIEPZYCKI
    yminTemp = ymin; %these three lines flip the values of ymin and
 ymax
    ymin = ymax;     %^
    ymax = yminTemp; %^
    fac1 = 1 - fac1; %changes the factor(.632) for cooling
    fac2 = -5;       %changes the factor(5) for cooling
end

tauTemp = (ymax - ymin) * fac1 + ymin; %calculates the temperature at
 the time constant
tempNew = 1000; %intitalizes the variable temp to enter the loop
tauCount = 0;   %the number of data points used to calculate tau
tauTimeTot = 0; %the sum of all data points used to calculate tau

slope = abs((mean(temp(n+50:n+55)) - mean(temp(n:n+5)))/(mean(time(n
+50:n+55))-mean(time(n:n+5))));
slopeRange = sqrt(slope)-5;
%fprintf('Slope range: %.4f\n',slopeRange);
amountOfPoints = length(find( temp < (tauTemp + slopeRange) & temp >
 (tauTemp - slopeRange)));

i = 1;
endLoop = 0;
counter = 1;
tauTempHigh = tauTemp + slopeRange;    %in theory a bigger number
 here will give a better SSE but a slower run time (HAS TO BE GREATER
 THAN .4, THAT IS MINIMUM RANGE)
tauTempLow = tauTemp - slopeRange;     %same as tauTempHigh, but
 subtracted instead of added, this sets crude range to find values
while(~endLoop && i < length(temp))
    if(temp(i) >= tauTempLow && temp(i) <= tauTempHigh)
        timeIndex(counter) = i; %Array of indexs that fall in the
 correct range
        timeVector(counter) = time(i);  %Time vector is a vector of
 times in the tau range
        counter = counter + 1;
    end
    i = i+1;
end

i = timeVector(1);
lowestTauSSE = 50000;   %Set really high initially so it triggers the
 if statment right away and it gets reasigned
tauTime = 0;
exitCount = 0;  %if SSE fails to go down after 4 steps forward, loop
 terminates
while(i <= timeVector(length(timeVector)))
    SSE = M4Calibration_014_05(ymin, ymax, tstep, i - tstep, temp,
 time, isHeating);    %Checks SSE for the current step
    if(SSE < lowestTauSSE)
                  %Updates the tau and sse if current step has a lower
 sse
        tauTime = i - tstep;
        lowestTauSSE = SSE;
    else
        exitCount = exitCount + 1;
    end

    if(exitCount > 3)
        i = timeVector(length(timeVector)) + 1;
    end

    i = i + .0005;  %.0005 is the step of time intervals to check
end
```

```
ymin =

    0.5077


ymax =

  100.8127


tstep =

    1.3633


tau =

    0.2342


isHeating =

     0
```

Published with MATLAB® R2016a