Boise freezing pipes

In this assignment we will look at historical data for air temperature in Boise, and will try to find out how deep we need to bury water pipes so that they do not freeze in winter.

Task 1

In the file `boise_temp_data.mat` you have a table with data containing average monthly temperature in Boise between 1940 and 2022. The first column contains the year, and the subsequent 12 columns correspond to months. You can access data in the array by typing `boise_data(i,j)` where `i` is the index for the year (spanning 83 years from 1940 to 2022) and `j` is the index for a month (from 1 to 12). If you want to extract and entire row of data, say the 10th row, you can write `boise_data(10,:)`, and if you want to extract the entire column you can do it by typing `boise_data(:,10)` (if the 10th column is what you wanted).

For each year, compute maximum, minimum and mean temperature and plot all three as a function of time (year). Make sure your plot contain axis labels, etc, and the label for time is in fact the year (i.e. 1940, etc).

```
 load boite_temp_data.mat

T = boise_temp(:,2:end)
```

```
T = 83×12
   33.0000   38.9000   45.6000   50.1000   61.3000   69.6000   74.9000   74.2000 ...
   33.2000   39.5000   45.4000   50.1000   58.5000   62.9000   75.5000   70.9000
   20.8000   31.5000   40.3000   51.8000   53.0000   60.8000   75.7000   72.9000
   28.2000   35.9000   39.9000   54.0000   54.0000   61.6000   73.3000   70.9000
   25.0000   34.1000   38.3000   48.9000   58.6000   60.8000   72.3000   70.7000
   32.1000   37.6000   40.0000   46.0000   56.6000   61.1000   74.9000   73.4000
   29.0000   33.8000   43.8000   51.9000   57.7000   65.4000   74.9000   73.3000
   23.3000   38.1000   44.9000   49.4000   62.2000   61.8000   75.1000   72.2000
   33.0000   32.1000   37.9000   48.1000   56.3000   66.5000   70.3000   70.4000
   10.3000   30.7000   43.2000   53.4000   61.5000   65.7000   74.4000   73.6000
      ⋮
```

```
time = boise_temp(:,1)
```

```
time = 83×1
        1940
        1941
        1942
        1943
        1944
        1945
        1946
        1947
        1948
        1949
      ⋮
```

```
n = 83; % number of years
y = zeros(n,1);
```

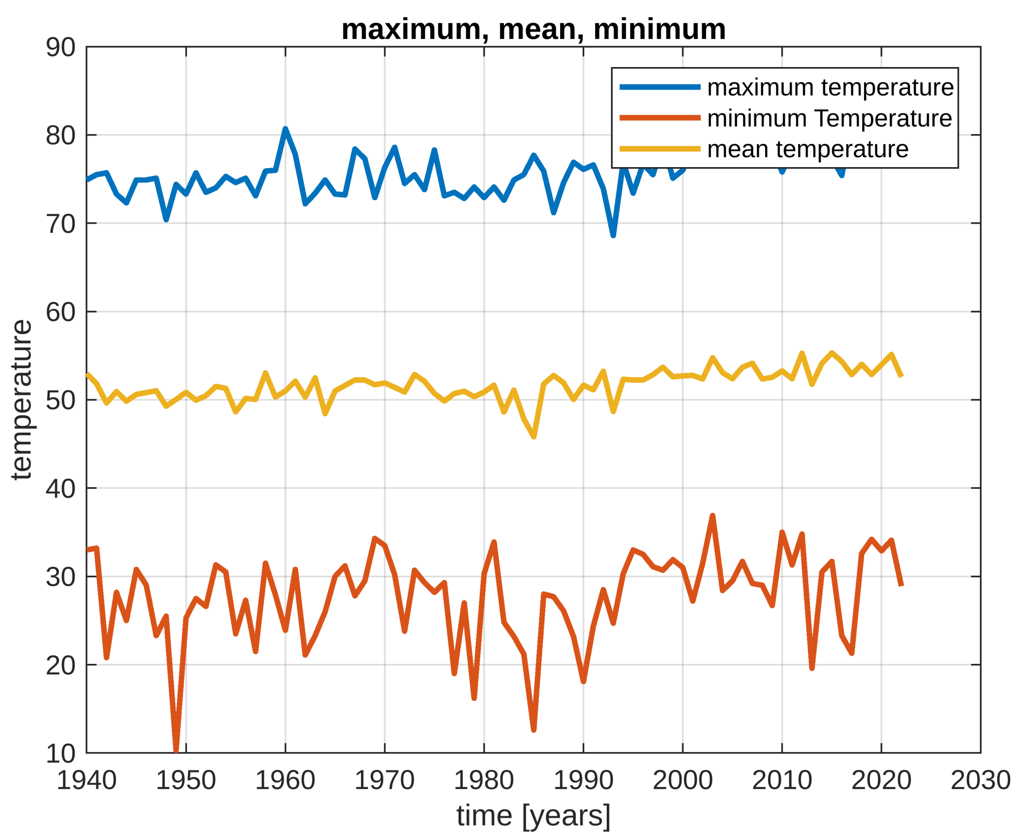
```
x = zeros(n,1);
z = zeros(n,1);

for i=1:n

    y(i) = min(T(i,:));
    x(i) = max(T(i,:));
    z(i) = mean(T(i,:));

end


plot(time,x, LineWidth=2)
hold on;
plot(time,y,LineWidth=2)
hold on;
plot(time,z,LineWidth=2)
title('maximum, mean, minimum')
legend('maximum temperature','minimum Temperature', 'mean temperature')
xlabel('time [years]')
ylabel('temperature')
grid on;
```




```
%boise_temp(:,1) = []
%Data = boise_temp
```

```
%where i is the index for the year, and j is the index for a month.(i,j)
%years = boise_data(:,1)
%yr1940 = boise_data(1,:)
```

boise_data(i,j)

What can you conclude from this plot? Do those statistics rise, fall or remain the same over time? Remember we deal with average montly temperatures, not daily temperatures.

Answer: All of the plots slowly rise over the years but rise and fall throught the months, this is expected however recent low temperatures in the last 20 years no longer reach the recorded low values observed in the first half. The overall trend shows how temperature is increasing, with winters becoming drastically much more mild.

Task 2

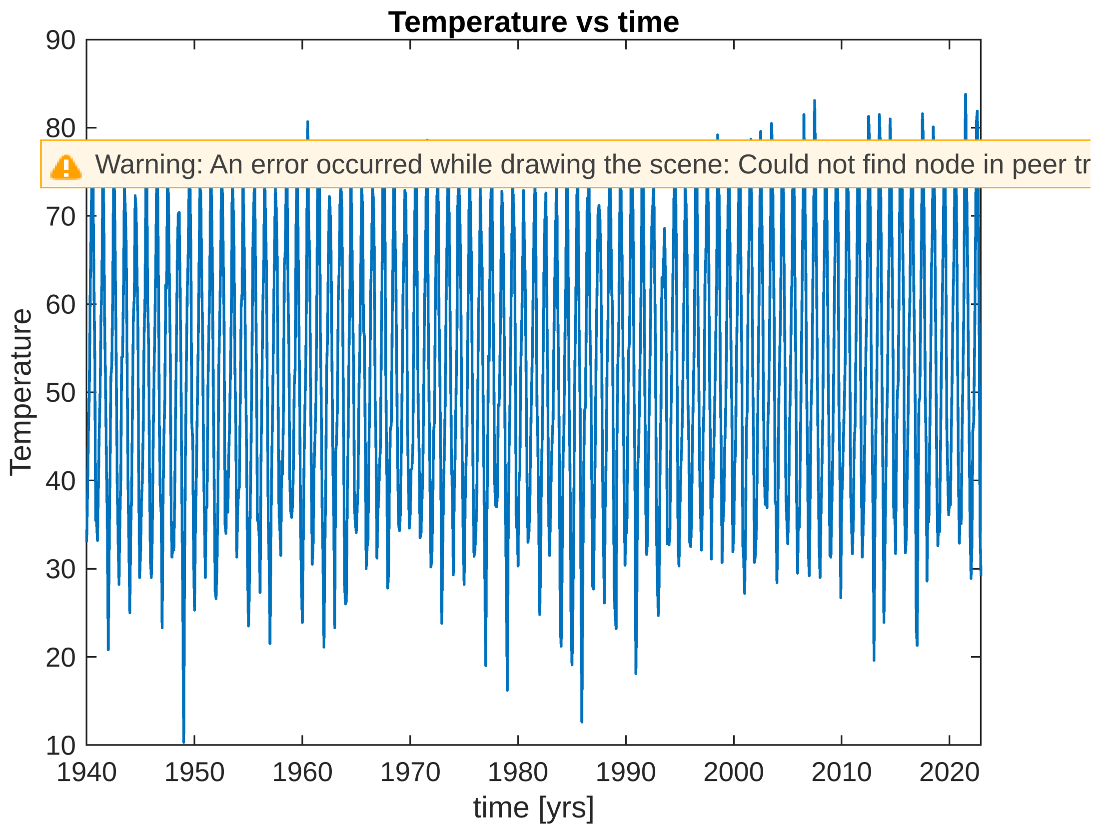
We would like to convert this two-dimensional array into a one-dimensional vector with chronological data, starting at January 1940 going all the way up to December 2022. Come up with a way to reshape the array, you can use Matlab built-in functions or simple algorithm with loops. Once ready, plot your data as a function of time.

```
close all

K = T(:);
OneDArray = reshape(T',[1 size(T,1)*size(T,2)])';% one demenstional array
time = (datetime(1940,1,1)):calmonths(1):(datetime(2022,12,31))
```

```
time = 1×996 datetime
01-Jan-1940 01-Feb-1940 01-Mar-1940 01-Apr-1940 01-May-1940 01-Jun-1940 01-Jul-1···
```

```
plot(time,OneDArray, LineWidth=1)
xlabel('time [yrs]')
ylabel('Temperature')
title('Temperature vs time')
```



It is ok to use months beginning from January 1940 (i.e. January 1940 is 1, February 1940 is 2, etc). If you want to be extra fancy you can use function `datetime()` to create a time array, which you can then use to plot the data. `datetime(y,m,d)` takes in at least three arguments (year, month, day) as numbers and returns a date format. For example, `datetime(2023,2,1)` will return `01-Feb-2023`.

Is your plot consistent with the results of Task 1?

Answer:

Yes, the plot is consistent with task 1. This graph continues to show a positive trend simmilar to the previous grapph. This plot further enhances the changes we are observing, if we look at the top section of the graph we can see that temperature is gradually increasing, the [eaks represent the summer months witha consistent gradual increase in temerature each year. However if we look at the trophs, we can see how unstable the winters become, with very little consistency over the years.

Task 3

The following equation models the soil temperature $T(x,t)$ $[^oC]$ at depth x [m] after exposure to average air temperature T_s $[^oC]$ for a time period t [s].

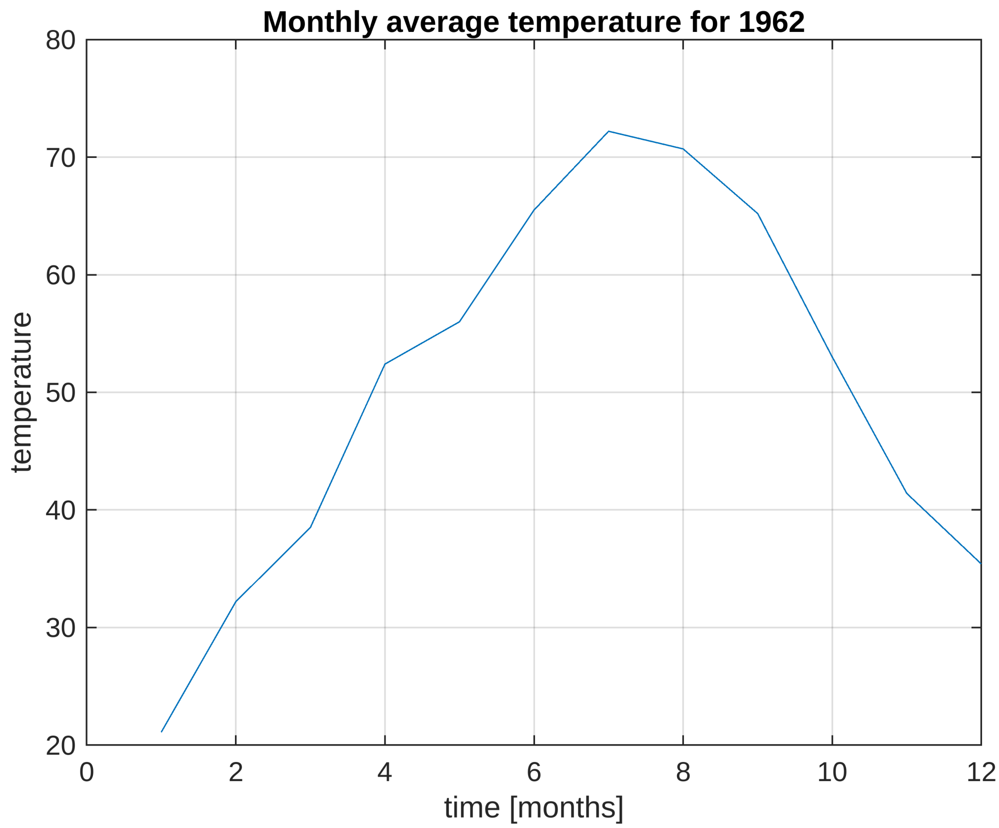
$$\frac{T(x,t) - T_s}{T_i - T_s} = \mathrm{erf}\left(\frac{x}{2\sqrt{\alpha t}}\right)$$

T_i $[^oC]$ is the soil temperature at the depth of 10 m, $\alpha = 0.138 \cdot 10^{-6}$ $[m^2/s]$ is an estimate of soil thermal conductivity (assuming uniform soil composition), and $\mathrm{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2}\ dt$ is the error function available in MATLAB as `erf()`.

Create a function which accepts the depth below the ground level and other required parameters and returns the value of temperature.

- Pick a year (you can take the year of your birth, some other significant date, or any other you like). Plot the average monthly temperatures in that year (make sure the plot label tells me which year it is).
- Let T_s be the minimum monthly temperature in that year, and T_i be the average air temperature in that year (remember to convert those temperatures to ^oC).
- Plot the soil temperature $T(x)$ for $x \in [0, 10]$ m, assuming that the ground was exposed to temperature T_s for 30 days (remember the equation assumes the time is expressed in seconds). If you want, you can invert the y axis so that the positive x actually points down, giving you a better intution of what actually happens.
- Use any technique we discussed in class (Bisection, Newton, Secant, fzero, etc) to find at which depth the temperature reaches $0\,^oC$ that month. We will call this value of x the frost penetration depth. Explain your reasons for choosing the method.
- Explain how you know that this result is correct. If you think it is incorrect, check your code again, maybe there is a bug, or you forgot to use appropriate units for temperature or time?
- Compare the result which you got with the frost penetration depth for Boise (recommended depth to bury water pipes, lay concrete foundations for fence posts, etc). How does your result compare?

```
B = boise_temp(23,2:end); % monthly temperature of 1962
plot(B)
xlabel('time [months]')
ylabel('temperature')
title('Monthly average temperature for 1962')
grid on;
```




```
Celcius = (B - 32)*(5/9); % farenhit to Celcius conversion
```

```
Ts = min(Celcius)
```

```
Ts = -6.0556
```

```
Ti = mean(Celcius)
```

```
Ti = 10.1667
```

```
a = 0.138e-6; %alpha
t = 30*60*60*24; % seconds in a month
m = 1000;

SoilTemp = Stemp(Ts,Ti,x,t,a)
```

```
SoilTemp = 83×1
   10.1667
   10.1667
   10.1667
   10.1667
   10.1667
   10.1667
   10.1667
   10.1667
   10.1667
   10.1667
```

⋮

```
n = 996;
x = linspace(0,10,n)
```

```
x = 1×996
         0    0.0101    0.0201    0.0302    0.0402    0.0503    0.0603    0.0704 ···
```

```
for i=1:n


    SoilTemp = Stemp(Ts,Ti,x(i),t,a);
     soilArray(i) = SoilTemp;
end

soilArray % temperature at depth
```

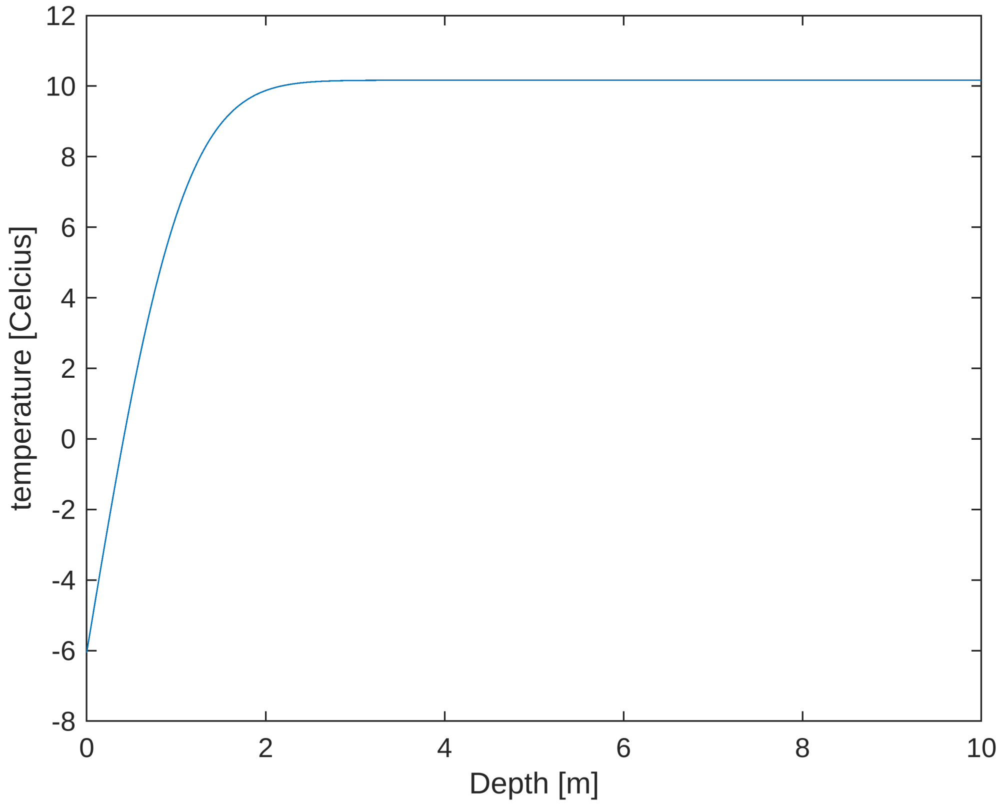
```
soilArray = 1×996
   -6.0556   -5.9018   -5.7480   -5.5943   -5.4406   -5.2870   -5.1335   -4.9802 ···
```

```

% T = @(x) Ts + (Ti-Ts)*erf(x./(2*sqrt(alpha*t)))
% x = linspace(0,10,m);
```

```
% p = T(x)
% plot(x,p)

plot(x,soilArray)
xlabel('Depth [m]')
ylabel('temperature [Celcius]')
```




```
epsilon = 1e-10;
% T1 = @Stemp
T1 = @(x) Stemp(Ts,Ti,x,t,a)
```

```
T1 = function_handle with value:
    @(x)Stemp(Ts,Ti,x,t,a)
```

```
x1 = .1
```

```
x1 = 0.1000
```

```
x0 = 0
```

```
x0 = 0
```

```
[x,k] = secant1(T1,x0,x1,epsilon)
```

```
x = 0.4114
k = 6
```

Answers:

I used the secant method to calculate, so that I dont have to compute the derivative and make things more complicated, and it also converges faster than if I were to use the bisection method.

the temperature reaches $0\,^{o}C$ that month at a depth of about 0.5 meters when we use the secant method. The secant method is more flexible compared to the newton method that is more limited in its aproximation. Ts represents the minimum temperature for 30 days, at -12 degrees celcius the ground shouls be frozen, however temprature increases with depth, this makes sense according to my understanding of the geothermal gradient. If I had to give consulting advice for the depth at which a pipe might need to be burried so that water running through the pipe doesnt freeze over the coldest temperatures, I would have to say tht the depth must be at least 0.5 meters bellow the surface.

Task 4

Let's do the historical analysis of the frost penetration depth in Boise, based on our model and the temperature data.

- Using the technique developed in Task 3, compute frost penetration depth for every month in our dataset. In some months there will be no frost, obviously, so you need to come up with a way to deal with this. For example, whenever the frost penetration depth is unphysical (i.e. smaller than 0, or maybe shows up as NaN - not a number), replace it with 0. You can also think of ways of anticipating whether there will be any frost line for a given month, and thus not wasting time to find frost penetration depth at all, and just assuming it is 0 for warm enough months. Please explain your approach. If you decide to use fzero, the function will write a warning message each time it does not find a zero - please find a way of dealing with this before submitting the assignment, otherwise I will have a lot of scrolling to do. You can either write your code in a smart way (i.e. you don't check for zero if there is no chance of having one), or you look up the documentation of `fzero`.
- For the T_i value take either the average air temperature across the entire time period in our data set, or choose the average temperature for the year you are currently analysing. Your choice should depend on the result of your analysis in Task 1. If the yearly average does not change much over time, it is ok to use constant value for T_i. If you conclude that there is a significant change in the average, use the yearly average instead.
- Plot the frost line depth over time. Find the maximum frost penetration depth, and the time at which it happened. Indicate this data point on the plot with a marker.
- On the same plot, mark the official Boise frost depth recommendation (you can use a horizontal dashed line, i.e. plot the same value for all times). Were there any historical occurences when our model predicted frost depth below the official recommendation? How many months like that happened so far (you may want to write an algorithm to count those points, if it is not obvious from the plot).

```
Celcius = (OneDArray - 32)*(5/9); % farenhit to Celcius conversion
Frozen_months= 1:length(Celcius);

%          Celcius(i) = frozen(i);

for i=1:length(Celcius)

    if (Celcius(i) < 0)

        Frozen_months(i) = Celcius(i);
```

```
    else

          Frozen_months(i) = 0;


    end


end

Frozen_months
```

```
Frozen_months = 1×996
         0         0         0         0         0         0         0         0···
```

```
T
```

```
T = 83×12
   33.0000   38.9000   45.6000   50.1000   61.3000   69.6000   74.9000   74.2000···
   33.2000   39.5000   45.4000   50.1000   58.5000   62.9000   75.5000   70.9000
   20.8000   31.5000   40.3000   51.8000   53.0000   60.8000   75.7000   72.9000
   28.2000   35.9000   39.9000   54.0000   54.0000   61.6000   73.3000   70.9000
   25.0000   34.1000   38.3000   48.9000   58.6000   60.8000   72.3000   70.7000
   32.1000   37.6000   40.0000   46.0000   56.6000   61.1000   74.9000   73.4000
   29.0000   33.8000   43.8000   51.9000   57.7000   65.4000   74.9000   73.3000
   23.3000   38.1000   44.9000   49.4000   62.2000   61.8000   75.1000   72.2000
   33.0000   32.1000   37.9000   48.1000   56.3000   66.5000   70.3000   70.4000
   10.3000   30.7000   43.2000   53.4000   61.5000   65.7000   74.4000   73.6000
      ⋮
```

```
n = length(T); % number of years
% y = zeros(n,1);
% x = zeros(n,1);

z = mean(Celcius) % Average air temperature across the time period in C
```

```
z = 10.9369
```

```
z1 = mean(OneDArray) % Average air temperature across the time period in F
```

```
z1 = 51.6863
```

```
[x,k] = secant1(T1,x0,x1,epsilon)
```

```
x = 0.4114
k = 6
```

```
Celcius_T = (T - 32)*(5/9) % farenhit to Celcius conversion
```

```
Celcius_T = 83×12
    0.5556    3.8333    7.5556   10.0556   16.2778   20.8889   23.8333   23.4444···
    0.6667    4.1667    7.4444   10.0556   14.7222   17.1667   24.1667   21.6111
   -6.2222   -0.2778    4.6111   11.0000   11.6667   16.0000   24.2778   22.7222
   -2.1111    2.1667    4.3889   12.2222   12.2222   16.4444   22.9444   21.6111
```

```
   -3.8889    1.1667    3.5000    9.3889   14.7778   16.0000   22.3889   21.5000
    0.0556    3.1111    4.4444    7.7778   13.6667   16.1667   23.8333   23.0000
   -1.6667    1.0000    6.5556   11.0556   14.2778   18.5556   23.8333   22.9444
   -4.8333    3.3889    7.1667    9.6667   16.7778   16.5556   23.9444   22.3333
    0.5556    0.0556    3.2778    8.9444   13.5000   19.1667   21.2778   21.3333
  -12.0556   -0.7222    6.2222   11.8889   16.3889   18.7222   23.5556   23.1111
     ⋮
```

```
T_i = (mean(OneDArray) - 32)*(5/9);


distance = (0:.1:10)
```

```
distance = 1×101
         0    0.1000    0.2000    0.3000    0.4000    0.5000    0.6000    0.7000 ⋯
```

```

n = length(Celcius);
depth =  zeros(length(T), 1);
for i=1:n

    % SoilTemp  =  Stemp(Celcius_T(i),T_i,x(i),t,a);

    depth(i) = secant(Stemp,x0,x1,epsilon,Frozen_months(i),T_i,t,a);
    if depth(i) < 0
        depth(i) = 0
    end
end
```

```
Not enough input arguments.

Error in Stemp (line 4)
SoilTemp = Ts + (Ti-Ts)*erf(x/(2*sqrt(a*t)));
```

Answers: