classdef Rho < matlab.apps.AppBase

% Properties that correspond to app components

properties (Access = public)

RhoUIFigure matlab.ui.Figure

GridLayout matlab.ui.container.GridLayout

LeftPanel matlab.ui.container.Panel

ThermocoupleDropDown matlab.ui.control.DropDown

ThermocoupleDropDownLabel matlab.ui.control.Label

PlotDataButton matlab.ui.control.Button

LorenzNumberWK2EditField matlab.ui.control.NumericEditField

LorenzNumberWK2EditFieldLabel matlab.ui.control.Label

DropDown matlab.ui.control.DropDown

OutliersDegreesofFreedomEditField matlab.ui.control.NumericEditField

OutliersDegreesofFreedomEditFieldLabel matlab.ui.control.Label

NoiseLabel matlab.ui.control.Label

DiscsLabel matlab.ui.control.Label

ToEditField matlab.ui.control.NumericEditField

ToEditFieldLabel matlab.ui.control.Label

FromEditField matlab.ui.control.NumericEditField

FromEditFieldLabel matlab.ui.control.Label

WdiscDiametermEditField matlab.ui.control.NumericEditField

DiametermEditFieldLabel matlab.ui.control.Label

WdiscLengthmEditField matlab.ui.control.NumericEditField

LengthmEditFieldLabel matlab.ui.control.Label

DiameterEditField matlab.ui.control.NumericEditField

DiametermEditField\_2Label matlab.ui.control.Label

diametererror matlab.ui.control.NumericEditField

Label\_2 matlab.ui.control.Label

lengtherror matlab.ui.control.NumericEditField

Label matlab.ui.control.Label

CurrentAEditField matlab.ui.control.NumericEditField

CurrentAEditFieldLabel matlab.ui.control.Label

LengthEditField matlab.ui.control.NumericEditField

LengthmLabel matlab.ui.control.Label

Step2optionalLabel\_2 matlab.ui.control.Label

LoadDataButton matlab.ui.control.Button

filename matlab.ui.control.EditField

ConstantsLabel matlab.ui.control.Label

SampleDimensionsLabel matlab.ui.control.Label

RemoveButton matlab.ui.control.Button

DataSelectionLabel matlab.ui.control.Label

RightPanel matlab.ui.container.Panel

TabGroup matlab.ui.container.TabGroup

DataSelectionTab matlab.ui.container.Tab

V\_NegativeLabel matlab.ui.control.Label

V\_PositiveLabel matlab.ui.control.Label

T\_AfterLabel matlab.ui.control.Label

T\_BeforeLabel matlab.ui.control.Label

NotSelectedLabel matlab.ui.control.Label

figure2 matlab.ui.control.UIAxes

figure1 matlab.ui.control.UIAxes

ResistivityTab matlab.ui.container.Tab

figure3 matlab.ui.control.UIAxes

OutputTab matlab.ui.container.Tab

Table matlab.ui.control.Table

InformationTab matlab.ui.container.Tab

ForfeedbackpleasecontactMeryemBerradaatmberradauwocaLabel matlab.ui.control.Label

TextArea matlab.ui.control.TextArea

FiguresTextArea matlab.ui.control.TextArea

FiguresTextAreaLabel matlab.ui.control.Label

InstructionsTextArea matlab.ui.control.TextArea

InstructionsTextAreaLabel matlab.ui.control.Label

end

% Properties that correspond to apps with auto-reflow

properties (Access = private)

onePanelWidth = 576;

end

properties (Access = public)

%thresholds

n=2; %2 relates to n2

n1=-1e-4; %-1e-4 relates to amplitude of negative voltage values

n3; %20 controls outliers, max difference between T and V values CHANGE THIS \*\*\*\*\*\*\*\*\*\*

n4=1.3e-4; %1.3e-4 differentiates TB, TA from difference between T and V values

n5=3; %3 minimum number of elements to consider

n6=10; %10 number of elements for T, dont change

%constants

I=[]; % constant current [A]

Lnum=[]; % Lorenz number [W\*Ohm\*K^-2]

L=[]; % sample length [m]

D=[]; % sample diameter [m]

errL=[]; % error from microscope [m]

errD=[];

Lp=[]; % disc length [m] (4t)

Dp=[]; % disc diameter [m] (50t)

%This identifies all negative values

data;

negative\_index;

negative\_index\_number;

aa;

%selecting sections based on finding the sections with negative numbers

k = 1;

neg\_diff = []; %create an empty matrix

interval\_neg2=[]; %create an empty matrix

%Calling variables

Vpositive;

Vnegative;

Tbefore;

Tafter;

Vpos;

Vneg;

Tbef;

Taft;

emf;

Td;

Tc;

T;

Vt;

pW;

Va;

Vfinal;

p;

K;

errVP;

errVN;

errTa;

errTb;

errT;

errV;

errp;

errK;

VP\_dif\_max;

VN\_dif\_max;

Ta\_dif\_max;

Tb\_dif\_max;

dif\_VN;

dif\_VP;

temp\_data\_pos;

temp\_data\_neg;

value;

brushedData;

data1;

BD;

BDa;

BDb;

t;

xmin;

xmax;

TFe;

pFe;

end

% Callbacks that handle component events

methods (Access = private)

% Button pushed function: LoadDataButton

function LoadDataButtonPushed(app, event)

[fname, fpath] = uigetfile('.xlsx', '.csv', '.xls');

if ~ischar(fname); return; end %user cancel

file=fullfile(fpath,fname);

app.filename.Value=file;

app.data = importdata(app.filename.Value);

app.data = struct2cell(app.data);

app.data = cell2mat (app.data);

app.data = app.data.';

% plot of raw data

app.figure1;

clf(app.figure1);

cla(app.figure1)

plot(app.figure1,app.data,'.b');

hold(app.figure1,'on');

end

% Value changed function: filename

function filenameValueChanged(app, event)

app.Value = app.filename.Value;

app.filename.Value=file;

end

% Value changed function: LengthEditField

function LengthEditFieldValueChanged(app, event)

app.L = app.LengthEditField.Value;

end

% Value changed function: lengtherror

function lengtherrorValueChanged(app, event)

app.errL = app.lengtherror.Value;

end

% Value changed function: DiameterEditField

function DiameterEditFieldValueChanged(app, event)

app.D = app.DiameterEditField.Value;

end

% Value changed function: diametererror

function diametererrorValueChanged(app, event)

app.errD = app.diametererror.Value;

end

% Value changed function: WdiscLengthmEditField

function WdiscLengthmEditFieldValueChanged(app, event)

app.Lp= app.WdiscLengthmEditField.Value;

end

% Value changed function: WdiscDiametermEditField

function WdiscDiametermEditFieldValueChanged(app, event)

app.Dp = app.WdiscDiametermEditField.Value;

end

% Value changed function: OutliersDegreesofFreedomEditField

function OutliersDegreesofFreedomEditFieldValueChanged(app, event)

app.n3 = app.OutliersDegreesofFreedomEditField.Value;

end

% Value changed function: CurrentAEditField

function CurrentAEditFieldValueChanged(app, event)

app.I = app.CurrentAEditField.Value;

end

% Value changed function: LorenzNumberWK2EditField

function LorenzNumberWK2EditFieldValueChanged(app, event)

app.Lnum = app.LorenzNumberWK2EditField.Value;

end

% Button pushed function: PlotDataButton

function PlotDataButtonPushed(app, event)

clear final

%Get Values

app.L = app.LengthEditField.Value;

app.errL = app.lengtherror.Value;

app.D = app.DiameterEditField.Value;

app.errD = app.diametererror.Value;

app.Lp= app.WdiscLengthmEditField.Value;

app.Dp = app.WdiscDiametermEditField.Value;

app.n3 = app.OutliersDegreesofFreedomEditField.Value;

app.I = app.CurrentAEditField.Value;

app.Lnum = app.LorenzNumberWK2EditField.Value;

app.k=1;

%\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Identify voltage/temperature\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

%This identifies all negative values

a = app.data;

b = [app.data(2:end) 0];

c = [app.data(3:end) 0 0];

d = [app.data(4:end) 0 0 0];

ab = a.\*b;

ac = a.\*c;

ad = a.\*d;

app.negative\_index = ab<0 & ac<0 & ad<0; % finds if ab, ac and ad all contain negative numbers at same locations

app.negative\_index\_number = (find(app.negative\_index)); %finds the values that are negatives

app.aa = a(app.negative\_index\_number);

%loop to identify when the first voltage is positive or negative

for i=1:(length(app.negative\_index\_number)-1)

interval\_neg = (app.negative\_index\_number(i)+1:app.negative\_index\_number(i+1));

if sum(app.data(interval\_neg)<0)>app.n5 && mode(app.data(interval\_neg))<app.n1 %mode returns the most frequent value

app.temp\_data\_neg = app.data(interval\_neg); %defines a vector of negative values

app.temp\_data\_neg = app.temp\_data\_neg(app.temp\_data\_neg<0);

interval\_neg = interval\_neg(app.temp\_data\_neg<0); % inditifies all sections of negative values

%plotfigure 1 by highlighting negative values in red

app.figure1;

plot(app.figure1,interval\_neg,app.temp\_data\_neg,'.r'); % Plot x vs y on the uiaxes object.

hold(app.figure1,'on');

temp\_data\_before = app.data(interval\_neg(1)-app.n6:interval\_neg(1)-1); %selects n6 values before first negative values

temp\_data\_after = app.data(interval\_neg(end)+1:interval\_neg(end)+app.n6); %selects n6 values after first negative values

if app.k==44

1;

end

% in the case where the sequence is TB VP VN TA

%If the difference between Tb-negative values with Ta-negative

%values is less than n4, then interval of negative values is

%defined from first to last element of negative values

%whereas if Tb-negative values is smaller than Ta-negative

%values, then the positive values are first part of negative values

n2 = max(ceil(app.n\*length(interval\_neg)),app.n6); %ceil rounds each entry of bracket, max finds the highest number

if abs(abs(abs(median(temp\_data\_before))-abs(median(app.temp\_data\_neg)))- abs(abs(median(temp\_data\_after))-abs(median(app.temp\_data\_neg))))<app.n4

app.interval\_neg2 = [app.interval\_neg2 interval\_neg(1) interval\_neg(end)];

elseif (abs(abs(median(temp\_data\_before))-abs(median(app.temp\_data\_neg))) < abs(abs(median(temp\_data\_after))-abs(median(app.temp\_data\_neg))))

interval\_pos = (interval\_neg(1)-n2:interval\_neg(1)-1); %first part of negative values

app.temp\_data\_pos = app.data(interval\_pos); %defines positive values as y axis values

V\_ind = (min([interval\_pos , interval\_neg]):max([interval\_pos , interval\_neg])); %creates one vector of all voltage values

VP\_ind = V\_ind(app.data(V\_ind)>=0); %defines positive voltage

isempty(VP\_ind); %find empty cells, should be 0

VP = app.data(VP\_ind); %VP is y values of the VP indices

VN\_ind = V\_ind(app.data(V\_ind)<0); %defines negative voltage

VN = app.data(VN\_ind); %VN is y values of the VN indices

app.dif\_VN = diff(VN); %calculates difference between adjacent elements in VN

T\_a\_ind = (VN\_ind(end)+1:VN\_ind(end)+app.n6); %defines indexes for Ta as n6 values after VN

T\_a = app.data(T\_a\_ind); %T\_a is y values of Ta indices

%repeat the previous for each data point

[~,ind1] = max(abs(diff(VP(1:end-app.n5))));

if ~isempty(ind1)

ind = VP\_ind(1)+ind1;

VP\_ind = (ind:VP\_ind(end));

VP = app.data(VP\_ind);

T\_b\_ind = ind-app.n6:ind-1;

T\_b = app.data(T\_b\_ind);

V\_ind = (VP\_ind(1):VN\_ind(end)); %creates a vector of all VP and VN data points

final.VP{app.k} = [VP;VP\_ind]; %creates vector of all VP data points

final.VN{app.k} = [VN;VN\_ind]; %creates vector of all VN data points

final.Tb{app.k} = [T\_b;T\_b\_ind]; %creates vector of all Tb data points

final.Ta{app.k} = [T\_a;T\_a\_ind]; %creates vector of all Ta data points

app.k = app.k+1;

%updates figure 1 to identify all data points with this sentence

plot(app.figure1,V\_ind,app.data(V\_ind),'\*k',T\_b\_ind,T\_b,'.r',T\_a\_ind,T\_a,'.g');

end

% in the case where the sequence is TB VN VP TA

%define positive values as second part of negative values

else

interval\_pos = (interval\_neg(end)+1:interval\_neg(end)+n2); %second part of negative values

app.temp\_data\_pos = app.data(interval\_pos); %defines positive values as y axis values

V\_ind = (min([interval\_pos , interval\_neg]):max([interval\_pos , interval\_neg]));%creates one vector of all voltage values

VP\_ind = V\_ind(app.data(V\_ind)>=0);%defines positive voltage

isempty(VP\_ind); %find empty cells, should be 0

VP = app.data(VP\_ind); %VP is y values of the VP indices

VN\_ind = V\_ind(app.data(V\_ind)<0); %defines negative voltage

VN = app.data(VN\_ind); %VN is y values of the VN indices

app.dif\_VN = diff(VN); %calculates difference between adjacent elements in VN

if ~isempty(VN\_ind)

T\_b\_ind = (VN\_ind(1)-app.n6:VN\_ind(1)-1); %defines Tb as n6 values before VN

T\_b = app.data(T\_b\_ind); %defines y values of Tb

%repeat the previous for each data point

[~,ind1] = max(abs(diff(VP(app.n5:end))));

if ~isempty(ind1)

ind = VP\_ind(1)+ind1+(app.n5-1);

VP\_ind = (VP\_ind(1):ind-1);

VP = app.data(VP\_ind);

T\_a\_ind = ind:ind+(app.n6-1);

T\_a = app.data(T\_a\_ind);

V\_ind = (VN\_ind(1):VP\_ind(end));

final.VP{app.k} = [VP;VP\_ind];

final.VN{app.k} = [VN;VN\_ind];

final.Tb{app.k} = [T\_b;T\_b\_ind];

final.Ta{app.k} = [T\_a;T\_a\_ind];

app.k = app.k+1;

%updates figure 1 to identify all data points with this sentence

plot(app.figure1,V\_ind,app.data(V\_ind),'\*k',T\_b\_ind,T\_b,'.r',T\_a\_ind,T\_a,'.g');

end

end

end

end

end

%loop to identify Ta and Tb depending on the previous scenarios

for jj=1:length(app.interval\_neg2)-1

interval\_neg = app.interval\_neg2(jj):app.interval\_neg2(jj+1);

n2 = max(ceil(app.n\*length(interval\_neg)),app.n6);

if length(interval\_neg)>1 && mode(app.data(interval\_neg))<app.n1

app.temp\_data\_neg = app.data(interval\_neg); %defines a vector of negative values

app.temp\_data\_neg = app.temp\_data\_neg(app.temp\_data\_neg<0);

interval\_neg = interval\_neg(app.temp\_data\_neg<0); %identifies all sections of negative values

%updates figure 1 by highlighting negative values in red

plot(app.figure1,interval\_neg,app.temp\_data\_neg,'.r')

hold(app.figure1,'on');

temp\_data\_before = app.data(interval\_neg(1)-n2:interval\_neg(1)-1); %selects first part of negative values

temp\_data\_after = app.data(interval\_neg(end)+1:interval\_neg(end)+n2); %selects second part of negative values

% in the case where the sequence is TB VP VN TA

%if Tb>Ta then positive interval is first part of negative values

if max(abs(diff(temp\_data\_before)))>max(abs(diff(temp\_data\_after)))

interval\_pos = (interval\_neg(1)-n2:interval\_neg(1)-1); %first part of negative values

app.temp\_data\_pos = app.data(interval\_pos); %defines y axis values of positive values

V\_ind = (min([interval\_pos , interval\_neg]):max([interval\_pos , interval\_neg])); %creates one vector of all negative values

VP\_ind = V\_ind(app.data(V\_ind)>=0); %defines positive voltage

isempty(VP\_ind); %find empty cells, should be 0

VP = app.data(VP\_ind); %VP is y values of the VP indices

VN\_ind = V\_ind(app.data(V\_ind)<0); %defines negative voltages

VN = app.data(VN\_ind); %VN is y values of the VN indices

app.dif\_VN = diff(VN); %calculate differences between adjacent elements in VN

T\_a\_ind = (VN\_ind(end)+1:VN\_ind(end)+app.n6); %defines Ta as n6 values after VN

T\_a = app.data(T\_a\_ind); %defines y values of Ta

%repeat the previous for each data point

[~,ind1] = max(abs(diff(VP(1:end-app.n5))));

if ~isempty(ind1)

ind = VP\_ind(1)+ind1;

VP\_ind = (ind:VP\_ind(end));

VP = app.data(VP\_ind);

T\_b\_ind = ind-app.n6:ind-1;

T\_b = app.data(T\_b\_ind);

V\_ind = (VP\_ind(1):VN\_ind(end));

final.VP{app.k} = [VP;VP\_ind];

final.VN{app.k} = [VN;VN\_ind];

final.Tb{app.k} = [T\_b;T\_b\_ind];

final.Ta{app.k} = [T\_a;T\_a\_ind];

app.k = app.k+1;

%updates figure 1 to identify all data points with this sentence

plot(app.figure1,V\_ind,app.data(V\_ind),'\*k',T\_b\_ind,T\_b,'.r',T\_a\_ind,T\_a,'.g');

end

% in the case where the sequence is TB VN VP TA

else

interval\_pos = (interval\_neg(end)+1:interval\_neg(end)+n2); %second part of negative values

app.temp\_data\_pos = app.data(interval\_pos); %defines positive values as y axis values

V\_ind = (min([interval\_pos , interval\_neg]):max([interval\_pos , interval\_neg])); %creates one vector of all negative values

VP\_ind = V\_ind(app.data(V\_ind)>=0); %defines positive voltage

isempty(VP\_ind); %find empty cells, should be 0

VP = app.data(VP\_ind); %VP is y values of the VP indices

VN\_ind = V\_ind(app.data(V\_ind)<0); %defines negative voltage

VN = app.data(VN\_ind); %VN is y values of the VN indices

app.dif\_VN = diff(VN); %calculates differences between adjacent elements in VN

if ~isempty(VN\_ind)

T\_b\_ind = (VN\_ind(1)-app.n6:VN\_ind(1)-1); %defines Tb as n6 values before VN

T\_b = app.data(T\_b\_ind); %defines y values of Tb

%repeat the previous for each data point

[~,ind1] = max(abs(diff(VP(app.n5:end))));

if ~isempty(ind1)

ind = VP\_ind(1)+ind1+(app.n5-1);

VP\_ind = (VP\_ind(1):ind-1);

VP = app.data(VP\_ind);

T\_a\_ind = ind:ind+(app.n6-1);

T\_a = app.data(T\_a\_ind);

V\_ind = (VN\_ind(1):VP\_ind(end));

final.VP{app.k} = [VP;VP\_ind];

final.VN{app.k} = [VN;VN\_ind];

final.Tb{app.k} = [T\_b;T\_b\_ind];

final.Ta{app.k} = [T\_a;T\_a\_ind];

app.k = app.k+1;

%updates figure 1 to identify all data points with this sentence

plot(app.figure1,V\_ind,app.data(V\_ind),'\*k',T\_b\_ind,T\_b,'.r',T\_a\_ind,T\_a,'.g');

end

end

end

end

end

%plot of final selections, without outliers

app.figure2;

clf(app.figure2);

cla(app.figure2)

plot(app.figure2,app.data,'.b');

app.VP\_dif\_max = []; %create an empty matrix

app.VN\_dif\_max = []; %create an empty matrix

app.Ta\_dif\_max = []; %create an empty matrix

app.Tb\_dif\_max = []; %create an empty matrix

%loop for creating final variables without outliers

for kk = 1:length(final.VP)

if kk==1

Ta\_ind = final.Ta{kk}(2,:); %defines TA

VP\_b\_ind = final.VP{kk+1}(2,:); %scenario TB VP VN TA

VN\_b\_ind = final.VN{kk+1}(2,:); %scenario TB VN VP TA

for i=1:length(Ta\_ind)

item = Ta\_ind(i);

%if VP or VN indices overlap with TA, then indice is -1

if any(VP\_b\_ind==item) || any(VN\_b\_ind==item)

Ta\_ind(i) = -1;

end

end

%all indices not equal to -1 are labelled as TA

Ta = [final.Ta{kk}(1,Ta\_ind~=-1);Ta\_ind(Ta\_ind~=-1)];

final.Ta{kk} = Ta;

elseif kk==length(final.VP) %for the rest of VP elements

Tb\_ind = final.Tb{kk}(2,:); %defines TB

VP\_a\_ind = final.VP{kk-1}(2,:); %scenario TB VP VN TA

VN\_a\_ind = final.VN{kk-1}(2,:); %scenario TB VN VP TA

for i=1:length(Ta\_ind)

item = Tb\_ind(i);

%if VP or VN indices overlap with TB, then indice is -1

if any(VP\_a\_ind==item) || any(VN\_a\_ind==item)

Tb\_ind(i) = -1;

end

end

%all indices not equal to -1 are labelled as TB

Tb = [final.Tb{kk}(1,Tb\_ind~=-1);Tb\_ind(Tb\_ind~=-1)];

final.Tb{kk} = Tb;

else

Ta\_ind = final.Ta{kk}(2,:); %opposite scenario as previous loop for TA

VP\_a\_ind = final.VP{kk-1}(2,:);

VN\_a\_ind = final.VN{kk-1}(2,:);

Tb\_ind = final.Tb{kk}(2,:); %opposite scenario as previous loop for TB

VP\_b\_ind = final.VP{kk+1}(2,:);

VN\_b\_ind = final.VN{kk+1}(2,:);

for i=1:length(Ta\_ind) %same as previous loop but for opposite scenario

item = Ta\_ind(i);

if any(VP\_b\_ind==item) || any(VP\_a\_ind==item) || any(VN\_b\_ind==item) || any(VN\_a\_ind==item)

Ta\_ind(i) = -1;

end

item = Tb\_ind(i); %same as previous loop but for opposite scenario

if any(VP\_b\_ind==item) || any(VP\_a\_ind==item) || any(VN\_b\_ind==item) || any(VN\_a\_ind==item)

Tb\_ind(i) = -1;

end

end

%rewrites all TA and TB by avoiding overlapping indices

Ta = [final.Ta{kk}(1,Ta\_ind~=-1);Ta\_ind(Ta\_ind~=-1)];

final.Ta{kk} = Ta;

Tb = [final.Tb{kk}(1,Tb\_ind~=-1);Tb\_ind(Tb\_ind~=-1)];

final.Tb{kk} = Tb;

end

% identify y-axis values as first row of matrix final

VP = final.VP{kk}(1,:);

VN = final.VN{kk}(1,:);

Ta = final.Ta{kk}(1,:);

Tb = final.Tb{kk}(1,:);

% identify x-axis values as second row of matrix final

VP\_ind = final.VP{kk}(2,:);

VN\_ind = final.VN{kk}(2,:);

Ta\_ind = final.Ta{kk}(2,:);

Tb\_ind = final.Tb{kk}(2,:);

% rewrite VP without including values that have a difference larger than n3

VP\_dif = sort(abs(diff(VP)));

if length(VP\_dif)>app.n5

VP\_dif = VP\_dif(end-app.n5);

VP\_dif1 = [abs(diff(VP)) max(abs(diff(VP)))];

VP\_dif2 = [max(abs(diff(VP))) abs(diff(VP))];

app.n3 = app.OutliersDegreesofFreedomEditField.Value;

VP\_del\_ind = VP\_dif1>app.n3.\*VP\_dif & VP\_dif2>app.n3.\*VP\_dif;

VP = VP(~VP\_del\_ind);

VP\_ind = VP\_ind(~VP\_del\_ind);

end

% rewrite VN without including values that have a difference larger than n3

VN\_dif = sort(abs(diff(VN)));

if length(VN\_dif)>app.n5

VN\_dif = VN\_dif(end-app.n5);

VN\_dif1 = [abs(diff(VN)) max(abs(diff(VN)))];

VN\_dif2 = [max(abs(diff(VN))) abs(diff(VN))];

VN\_del\_ind = VN\_dif1>app.n3\*VN\_dif & VN\_dif2>app.n3\*VN\_dif;

VN = VN(~VN\_del\_ind);

VN\_ind = VN\_ind(~VN\_del\_ind);

end

% rewrite TA without including values that have a difference larger than n3

Ta\_dif = sort(abs(diff(Ta)));

if length(Ta\_dif)>app.n5

Ta\_dif = Ta\_dif(end-app.n5);

Ta\_dif1 = [abs(diff(Ta)) max(abs(diff(Ta)))];

Ta\_dif2 = [max(abs(diff(Ta))) abs(diff(Ta))];

Ta\_del\_ind = Ta\_dif1>app.n3\*Ta\_dif & Ta\_dif2>app.n3\*Ta\_dif;

Ta = Ta(~Ta\_del\_ind);

Ta\_ind = Ta\_ind(~Ta\_del\_ind);

end

% rewrite TB without including values that have a difference larger than n3

Tb\_dif = sort(abs(diff(Tb)));

if length(Tb\_dif)>app.n5

Tb\_dif = Tb\_dif(end-app.n5);

Tb\_dif1 = [abs(diff(Tb)) max(abs(diff(Tb)))];

Tb\_dif2 = [max(abs(diff(Tb))) abs(diff(Tb))];

Tb\_del\_ind = Tb\_dif1>app.n3\*Tb\_dif & Tb\_dif2>app.n3\*Tb\_dif;

Tb = Tb(~Tb\_del\_ind);

Tb\_ind = Tb\_ind(~Tb\_del\_ind);

end

% rewrite final matrix

final.VP{kk} = [VP;VP\_ind];

final.VN{kk} = [VN;VN\_ind];

final.Ta{kk} = [Ta;Ta\_ind];

final.Tb{kk} = [Tb;Tb\_ind];

% calculating mean values for each data point

final\_avg.VP(kk) = mean(VP);

final\_avg.VN(kk) = mean(VN);

final\_avg.Ta(kk) = mean(Ta);

final\_avg.Tb(kk) = mean(Tb);

% calculating standard deviation for each data point

final\_avg.errVP(kk) = std(VP);

final\_avg.errVN(kk) = std(VN);

final\_avg.errTa(kk) = std(Ta);

final\_avg.errTb(kk) = std(Tb);

% update figure 2 using final variables

hold(app.figure2,'on');

plot(app.figure2,final.VP{kk}(2,:),final.VP{kk}(1,:),'\*k',...

final.VN{kk}(2,:),final.VN{kk}(1,:),'\*m',...

final.Ta{kk}(2,:),final.Ta{kk}(1,:),'.r',...

final.Tb{kk}(2,:),final.Tb{kk}(1,:),'.g');

end

% \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_resistivity calculation\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

%creating the final variables that will be used in equations

app.Vpositive=final\_avg.VP;

app.Vnegative=final\_avg.VN;

app.Tbefore=final\_avg.Tb;

app.Tafter=final\_avg.Ta;

%replacing outliers by interpolation

app.Vpos=filloutliers(app.Vpositive,'linear','mean');

app.Vneg=filloutliers(app.Vnegative,'linear','mean');

app.Tbef=filloutliers(app.Tbefore,'linear','mean');

app.Taft=filloutliers(app.Tafter,'linear','mean');

% Convert emf to temperature

app.emf=1/2.\*(app.Taft+app.Tbef); % vector of emf value for each data point

app.Td=app.emf.\*1000+0.342;

switch app.ThermocoupleDropDown.Value

case 'Type-C' % this is in celcius

app.Tc=74.124732.\*app.Td-4.28082813.\*app.Td.^2+0.52113892.\*app.Td.^3 ...

-0.0457487201.\*app.Td.^4+0.00280578284.\*app.Td.^5 ...

-0.000113145137.\*app.Td.^6+0.00000285489684.\*app.Td.^7 ...

-0.0000000407643828.\*app.Td.^8+0.000000000251358071.\*app.Td.^9;

case 'Type-S at 1 GPa'

app.Tc=6\*10^(-22).\*app.Td.^6-2\*10^(-18).\*app.Td.^5+8\*10^(-16).\*app.Td.^4 ...

+4\*10^(-12).\*app.Td.^3-7\*10^(-09).\*app.Td.^2+8\*10^(-06).\*app.Td-0.0003;

case 'Type-S at 2 GPa'

app.Tc=3\*10^(-21).\*app.Td.^6-2\*10^(-17).\*app.Td.^5+4\*10^(-14).\*app.Td.^4 ...

-4\*10^(-11).\*app.Td.^3+2\*10^(-08).\*app.Td.^2+4\*10^(-06).\*app.Td-0.0001;

case 'Type-S at 3 GPa'

app.Tc=3\*10^(-21).\*app.Td.^6-2\*10^(-17).\*app.Td.^5+4\*10^(-14).\*app.Td.^4 ...

-4\*10^(-11).\*app.Td.^3+2\*10^(-08).\*app.Td.^2+6\*10^(-06).\*app.Td-0.0002;

case 'Type-S at 4 GPa'

app.Tc=2\*10^(-21).\*app.Td.^6-1\*10^(-17).\*app.Td.^5+2\*10^(-14).\*app.Td.^4 ...

-2\*10^(-11).\*app.Td.^3+8\*10^(-09).\*app.Td.^2+1\*10^(-05).\*app.Td-0.0004;

case 'Type-S at 5 GPa'

app.Tc=2\*10^(-21).\*app.Td.^6-1\*10^(-17).\*app.Td.^5+3\*10^(-14).\*app.Td.^4 ...

- 4\*10^(-11).\*app.Td.^3+1\*10^(-08).\*app.Td.^2+1\*10^(-05).\*app.Td-0.0005;

otherwise %type-S at 1 atm (up to 1760 C)

app.Tc=1\*10^(-12).\*app.Td.^6-6\*10^(-10).\*app.Td.^5+2\*10^(-07).\*app.Td.^4- ...

2\*10^(-05).\*app.Td.^3+0.0015.\*app.Td.^2+0.038.\*app.Td-0.2997;

end

app.T=app.Tc+273.15; % gives T in kelvin

% drop down value changed

% W data from Littleton et al. (2019), Pt data from Gomi and Yoshino

% (2019), Re data from Littleton et al. (2019)

switch app.DropDown.Value

case 'W at 2 GPa'

app.pW=3\*10^(-6).\*app.T.^2+0.0094.\*app.T-0.7407;

case 'W at 3 GPa'

app.pW=3\*10^(-6).\*app.T.^2+0.0098.\*app.T-0.3533;

case 'W at 4 GPa'

app.pW=3\*10^(-6).\*app.T.^2+0.0085.\*app.T-0.3197;

case 'W at 5 GPa'

app.pW=2\*10^(-6).\*app.T.^2+0.0086.\*app.T-0.4985;

case 'Pt at 1 atm'

app.pW=0.0325.\*app.T+3.5675;

case 'Pt at 10 GPa'

app.pW=0.0269.\*app.T+3.6386;

case 'Pt at 20 GPa'

app.pW=0.023.\*app.T+3.628;

case 'Re at 2 GPa'

app.pW=-1\*10^(-5).\*app.T.^2+0.0753.\*app.T+0.2602;

case 'Re at 3 GPa'

app.pW=-1\*10^(-5).\*app.T.^2+0.0661.\*app.T+4.0377;

case 'Re at 4 GPa'

app.pW=-1\*10^(-5).\*app.T.^2+0.9432.\*app.T+0.9432;

case 'Re at 5 GPa'

app.pW=-1\*10^(-5).\*app.T.^2+0.0614.\*app.T+3.3132;

otherwise

app.pW=0.\*app.T;

end

app.Vt=1/2.\*abs(app.Vpos-app.Vneg); % total voltage for each data point

app.Va=app.pW\*10^-8.\*app.Lp.\*app.I\*4/(pi.\*app.Dp.^2); % V contribution from 1 plug

app.Vfinal=app.Vt-2\*app.Va; % V from sample

app.p=10^8\*(pi.\*app.D.^2.\*app.Vfinal)./(4.\*app.L.\*app.I); % resistivity [microOhm\*cm]

app.K=10^8\*app.Lnum.\*app.T./app.p; % thermal conductivity [W\*K^-1\*m^-1]

%error bar calculations

app.errVP=final\_avg.errVP;

app.errVN=final\_avg.errVN;

app.errTa=final\_avg.errTa;

app.errTb=final\_avg.errTb;

% Error Propagation, assumes no error on Lorenz number or current

app.errT=1/2.\*(app.errTa+app.errTb); % error on T

app.errV=1/2\*sqrt(app.errVP.^2+app.errVN.^2); % error on V

app.errp=app.p.\*sqrt((2\*app.errD./app.D).^2+(app.errV./app.Vfinal).^2+(app.errL./app.L).^2); % error p in cm

app.errK=app.errp./app.p.\*app.K; % error on k

% rho of pure Fe at 1 atm (Van Zytveld, 1980)

app.TFe=[153.59147; 202.72715; 258.41425; 310.82563; 363.23702; 441.8541; 494.26549; 537.39569; 599.63421; 661.87273; 707.7327; 763.4198; 799.45262; 832.20974; 874.79399; 910.82682; 943.58394; 966.51392; 1005.82246; 1028.75244; 1045.131; 1048.40671; 1068.06098; 1084.43954; 1120.47237; 1153.77544; 1199.6354; 1258.59821; 1301.18246; 1340.491; 1399.45381; 1468.24376; 1546.86084; 1638.58077; 1710.64642; 1746.67925; 1783.25803; 1783.25803; 1789.80946; 1796.36088; 1832.39371; 1881.52938; 1930.66506; 1979.80073; 2045.31497; 2110.8292; 2166.5163; 2238.58196; 2310.64761; 2402.91349; 2488.082; 2596.18048];

app.pFe=[3.17505; 5.62855; 7.8367; 11.02625; 15.19721; 19.6544; 22.5986; 26.52421; 31.67656; 37.31961; 42.96267; 48.60572; 54.24878; 58.17438; 63.81743; 69.70584; 76.33029; 81.48265; 88.88404; 93.30035; 98.4527; 100.17015; 101.3969; 102.869; 105.56786; 107.28531; 110.22951; 113.41906; 115.38186; 116.36326; 118.32607; 120.28887; 122.49702; 124.21447; 124.70517; 125.68657; 126.66797; 129.85752; 132.55638; 134.76453; 135.25523; 135.74593; 136.23663; 137.21803; 138.19943; 139.18083; 140.65293; 141.63433; 143.10643; 144.57853; 146.05064; 147.52274];

%figure of resistivity data

app.figure3;

plot(app.figure3,app.TFe,app.pFe,'-',app.T,app.p,'\*')

legend (app.figure3,'Fe at 1 atm','This study','location', 'southeast')

%generate table

app.t = [app.T' app.p' app.errp' app.K' app.errK']; %create matrix of table data

app.Table.Data = app.t; % Add data to the Table UI Component

end

% Cell edit callback: Table

function TableCellEdit(app, event)

app.Table=app.table(app.T,app.p,app.errp,app.K,app.errK,'VariableNames',{'app.T' 'app.p' 'app.errp' 'app.K' 'app.errK'});

end

% Button pushed function: RemoveButton

function RemoveButtonPushed(app, event)

%remove brushed values from plot

clear final

%get range of values

%with brush function

%app.BD = evalin('base','brushedData'); %get brushedData variable from base workspace

%without brush function

app.xmin = app.FromEditField.Value;

app.xmax = app.ToEditField.Value;

app.BDa=(app.xmin:app.xmax);

app.BD = app.data(:,app.BDa);

app.data1 = ismember(app.data,app.BD); %identify section of data that matches brushedData

app.data(app.data1) = NaN; %remove that section

% plot of raw data

app.figure1;

cla(app.figure1)

plot(app.figure1,app.data,'.b');

hold(app.figure1,'on');

end

% Value changed function: FromEditField

function FromEditFieldValueChanged(app, event)

app.xmin = app.FromEditField.Value;

end

% Value changed function: ToEditField

function ToEditFieldValueChanged(app, event)

app.xmax = app.ToEditField.Value;

end

% Changes arrangement of the app based on UIFigure width

function updateAppLayout(app, event)

currentFigureWidth = app.RhoUIFigure.Position(3);

if(currentFigureWidth <= app.onePanelWidth)

% Change to a 2x1 grid

app.GridLayout.RowHeight = {400, 400};

app.GridLayout.ColumnWidth = {'1x'};

app.RightPanel.Layout.Row = 2;

app.RightPanel.Layout.Column = 1;

else

% Change to a 1x2 grid

app.GridLayout.RowHeight = {'1x'};

app.GridLayout.ColumnWidth = {282, '1x'};

app.RightPanel.Layout.Row = 1;

app.RightPanel.Layout.Column = 2;

end

end

end

% Component initialization

methods (Access = private)

% Create UIFigure and components

function createComponents(app)

% Create RhoUIFigure and hide until all components are created

app.RhoUIFigure = uifigure('Visible', 'off');

app.RhoUIFigure.AutoResizeChildren = 'off';

app.RhoUIFigure.Position = [100 100 703 400];

app.RhoUIFigure.Name = 'Rho';

app.RhoUIFigure.Icon = 'icon1.jpg';

app.RhoUIFigure.SizeChangedFcn = createCallbackFcn(app, @updateAppLayout, true);

app.RhoUIFigure.HandleVisibility = 'on';

% Create GridLayout

app.GridLayout = uigridlayout(app.RhoUIFigure);

app.GridLayout.ColumnWidth = {282, '1x'};

app.GridLayout.RowHeight = {'1x'};

app.GridLayout.ColumnSpacing = 0;

app.GridLayout.RowSpacing = 0;

app.GridLayout.Padding = [0 0 0 0];

app.GridLayout.Scrollable = 'on';

% Create LeftPanel

app.LeftPanel = uipanel(app.GridLayout);

app.LeftPanel.Layout.Row = 1;

app.LeftPanel.Layout.Column = 1;

app.LeftPanel.Scrollable = 'on';

% Create DataSelectionLabel

app.DataSelectionLabel = uilabel(app.LeftPanel);

app.DataSelectionLabel.HorizontalAlignment = 'center';

app.DataSelectionLabel.FontSize = 15;

app.DataSelectionLabel.FontWeight = 'bold';

app.DataSelectionLabel.Position = [7 371 267 22];

app.DataSelectionLabel.Text = 'Data Selection';

% Create RemoveButton

app.RemoveButton = uibutton(app.LeftPanel, 'push');

app.RemoveButton.ButtonPushedFcn = createCallbackFcn(app, @RemoveButtonPushed, true);

app.RemoveButton.FontWeight = 'bold';

app.RemoveButton.FontColor = [0 0.4471 0.7412];

app.RemoveButton.Position = [197 291 69 22];

app.RemoveButton.Text = 'Remove';

% Create SampleDimensionsLabel

app.SampleDimensionsLabel = uilabel(app.LeftPanel);

app.SampleDimensionsLabel.FontWeight = 'bold';

app.SampleDimensionsLabel.Position = [10 231 141 22];

app.SampleDimensionsLabel.Text = 'Sample Dimensions';

% Create ConstantsLabel

app.ConstantsLabel = uilabel(app.LeftPanel);

app.ConstantsLabel.FontWeight = 'bold';

app.ConstantsLabel.Position = [11 156 63 22];

app.ConstantsLabel.Text = 'Constants';

% Create filename

app.filename = uieditfield(app.LeftPanel, 'text');

app.filename.ValueChangedFcn = createCallbackFcn(app, @filenameValueChanged, true);

app.filename.Position = [137 351 89 22];

% Create LoadDataButton

app.LoadDataButton = uibutton(app.LeftPanel, 'push');

app.LoadDataButton.ButtonPushedFcn = createCallbackFcn(app, @LoadDataButtonPushed, true);

app.LoadDataButton.FontWeight = 'bold';

app.LoadDataButton.FontColor = [0 0.4471 0.7412];

app.LoadDataButton.Position = [65 351 68 22];

app.LoadDataButton.Text = 'Load Data';

% Create Step2optionalLabel\_2

app.Step2optionalLabel\_2 = uilabel(app.LeftPanel);

app.Step2optionalLabel\_2.WordWrap = 'on';

app.Step2optionalLabel\_2.FontName = 'Segoe UI';

app.Step2optionalLabel\_2.FontSize = 10;

app.Step2optionalLabel\_2.Position = [19 317 248 15];

app.Step2optionalLabel\_2.Text = 'Optional - Indicate a range of indexes to be removed.';

% Create LengthmLabel

app.LengthmLabel = uilabel(app.LeftPanel);

app.LengthmLabel.HorizontalAlignment = 'right';

app.LengthmLabel.Position = [17 208 62 22];

app.LengthmLabel.Text = 'Length [m]';

% Create LengthEditField

app.LengthEditField = uieditfield(app.LeftPanel, 'numeric');

app.LengthEditField.ValueChangedFcn = createCallbackFcn(app, @LengthEditFieldValueChanged, true);

app.LengthEditField.Position = [84 208 62 22];

app.LengthEditField.Value = 0.001068;

% Create CurrentAEditFieldLabel

app.CurrentAEditFieldLabel = uilabel(app.LeftPanel);

app.CurrentAEditFieldLabel.HorizontalAlignment = 'right';

app.CurrentAEditFieldLabel.Position = [9 135 63 22];

app.CurrentAEditFieldLabel.Text = 'Current [A]';

% Create CurrentAEditField

app.CurrentAEditField = uieditfield(app.LeftPanel, 'numeric');

app.CurrentAEditField.ValueChangedFcn = createCallbackFcn(app, @CurrentAEditFieldValueChanged, true);

app.CurrentAEditField.Position = [77 135 34 22];

app.CurrentAEditField.Value = 0.2;

% Create Label

app.Label = uilabel(app.LeftPanel);

app.Label.HorizontalAlignment = 'right';

app.Label.Position = [131 208 25 22];

app.Label.Text = '±';

% Create lengtherror

app.lengtherror = uieditfield(app.LeftPanel, 'numeric');

app.lengtherror.ValueChangedFcn = createCallbackFcn(app, @lengtherrorValueChanged, true);

app.lengtherror.Position = [161 208 62 22];

app.lengtherror.Value = 1.27e-05;

% Create Label\_2

app.Label\_2 = uilabel(app.LeftPanel);

app.Label\_2.HorizontalAlignment = 'right';

app.Label\_2.Position = [143 183 25 22];

app.Label\_2.Text = '±';

% Create diametererror

app.diametererror = uieditfield(app.LeftPanel, 'numeric');

app.diametererror.ValueChangedFcn = createCallbackFcn(app, @diametererrorValueChanged, true);

app.diametererror.Position = [173 183 62 22];

app.diametererror.Value = 1.27e-05;

% Create DiametermEditField\_2Label

app.DiametermEditField\_2Label = uilabel(app.LeftPanel);

app.DiametermEditField\_2Label.HorizontalAlignment = 'right';

app.DiametermEditField\_2Label.Position = [17 183 74 22];

app.DiametermEditField\_2Label.Text = 'Diameter [m]';

% Create DiameterEditField

app.DiameterEditField = uieditfield(app.LeftPanel, 'numeric');

app.DiameterEditField.ValueChangedFcn = createCallbackFcn(app, @DiameterEditFieldValueChanged, true);

app.DiameterEditField.Position = [96 183 62 22];

app.DiameterEditField.Value = 0.000508;

% Create LengthmEditFieldLabel

app.LengthmEditFieldLabel = uilabel(app.LeftPanel);

app.LengthmEditFieldLabel.HorizontalAlignment = 'right';

app.LengthmEditFieldLabel.Position = [18 31 59 22];

app.LengthmEditFieldLabel.Text = 'Length [m]';

% Create WdiscLengthmEditField

app.WdiscLengthmEditField = uieditfield(app.LeftPanel, 'numeric');

app.WdiscLengthmEditField.ValueChangedFcn = createCallbackFcn(app, @WdiscLengthmEditFieldValueChanged, true);

app.WdiscLengthmEditField.Position = [79 31 66 22];

app.WdiscLengthmEditField.Value = 1.017e-05;

% Create DiametermEditFieldLabel

app.DiametermEditFieldLabel = uilabel(app.LeftPanel);

app.DiametermEditFieldLabel.HorizontalAlignment = 'right';

app.DiametermEditFieldLabel.Position = [16 7 73 22];

app.DiametermEditFieldLabel.Text = 'Diameter [m]';

% Create WdiscDiametermEditField

app.WdiscDiametermEditField = uieditfield(app.LeftPanel, 'numeric');

app.WdiscDiametermEditField.ValueChangedFcn = createCallbackFcn(app, @WdiscDiametermEditFieldValueChanged, true);

app.WdiscDiametermEditField.Position = [93 7 52 22];

app.WdiscDiametermEditField.Value = 0.00127;

% Create FromEditFieldLabel

app.FromEditFieldLabel = uilabel(app.LeftPanel);

app.FromEditFieldLabel.HorizontalAlignment = 'right';

app.FromEditFieldLabel.Position = [16 291 33 22];

app.FromEditFieldLabel.Text = 'From';

% Create FromEditField

app.FromEditField = uieditfield(app.LeftPanel, 'numeric');

app.FromEditField.ValueChangedFcn = createCallbackFcn(app, @FromEditFieldValueChanged, true);

app.FromEditField.Position = [53 291 47 22];

% Create ToEditFieldLabel

app.ToEditFieldLabel = uilabel(app.LeftPanel);

app.ToEditFieldLabel.HorizontalAlignment = 'right';

app.ToEditFieldLabel.Position = [94 291 25 22];

app.ToEditFieldLabel.Text = 'To';

% Create ToEditField

app.ToEditField = uieditfield(app.LeftPanel, 'numeric');

app.ToEditField.ValueChangedFcn = createCallbackFcn(app, @ToEditFieldValueChanged, true);

app.ToEditField.Position = [126 291 44 22];

% Create DiscsLabel

app.DiscsLabel = uilabel(app.LeftPanel);

app.DiscsLabel.FontWeight = 'bold';

app.DiscsLabel.Position = [18 57 63 22];

app.DiscsLabel.Text = 'Discs';

% Create NoiseLabel

app.NoiseLabel = uilabel(app.LeftPanel);

app.NoiseLabel.FontWeight = 'bold';

app.NoiseLabel.Position = [10 325 41 22];

app.NoiseLabel.Text = 'Noise';

% Create OutliersDegreesofFreedomEditFieldLabel

app.OutliersDegreesofFreedomEditFieldLabel = uilabel(app.LeftPanel);

app.OutliersDegreesofFreedomEditFieldLabel.HorizontalAlignment = 'right';

app.OutliersDegreesofFreedomEditFieldLabel.Position = [16 260 159 22];

app.OutliersDegreesofFreedomEditFieldLabel.Text = 'Outliers Degrees of Freedom';

% Create OutliersDegreesofFreedomEditField

app.OutliersDegreesofFreedomEditField = uieditfield(app.LeftPanel, 'numeric');

app.OutliersDegreesofFreedomEditField.ValueChangedFcn = createCallbackFcn(app, @OutliersDegreesofFreedomEditFieldValueChanged, true);

app.OutliersDegreesofFreedomEditField.Position = [184 260 24 22];

app.OutliersDegreesofFreedomEditField.Value = 20;

% Create DropDown

app.DropDown = uidropdown(app.LeftPanel);

app.DropDown.Items = {'None', 'W at 2 GPa', 'W at 3 GPa', 'W at 4 GPa', 'W at 5 GPa', 'Pt at 1 atm ', 'Pt at 10 GPa', 'Pt at 20 GPa', 'Re at 2 GPa', 'Re at 3 GPa', 'Re at 4 GPa', 'Re at 5 GPa'};

app.DropDown.Position = [60 57 100 22];

app.DropDown.Value = 'None';

% Create LorenzNumberWK2EditFieldLabel

app.LorenzNumberWK2EditFieldLabel = uilabel(app.LeftPanel);

app.LorenzNumberWK2EditFieldLabel.HorizontalAlignment = 'right';

app.LorenzNumberWK2EditFieldLabel.Position = [10 114 141 22];

app.LorenzNumberWK2EditFieldLabel.Text = 'Lorenz Number [WΩK^-2]';

% Create LorenzNumberWK2EditField

app.LorenzNumberWK2EditField = uieditfield(app.LeftPanel, 'numeric');

app.LorenzNumberWK2EditField.ValueChangedFcn = createCallbackFcn(app, @LorenzNumberWK2EditFieldValueChanged, true);

app.LorenzNumberWK2EditField.Position = [156 114 60 22];

app.LorenzNumberWK2EditField.Value = 2.44e-08;

% Create PlotDataButton

app.PlotDataButton = uibutton(app.LeftPanel, 'push');

app.PlotDataButton.ButtonPushedFcn = createCallbackFcn(app, @PlotDataButtonPushed, true);

app.PlotDataButton.FontWeight = 'bold';

app.PlotDataButton.FontColor = [0 0.4471 0.7412];

app.PlotDataButton.Position = [174 6 99 22];

app.PlotDataButton.Text = 'Plot Data';

% Create ThermocoupleDropDownLabel

app.ThermocoupleDropDownLabel = uilabel(app.LeftPanel);

app.ThermocoupleDropDownLabel.HorizontalAlignment = 'right';

app.ThermocoupleDropDownLabel.FontWeight = 'bold';

app.ThermocoupleDropDownLabel.Position = [10 86 92 22];

app.ThermocoupleDropDownLabel.Text = 'Thermocouple ';

% Create ThermocoupleDropDown

app.ThermocoupleDropDown = uidropdown(app.LeftPanel);

app.ThermocoupleDropDown.Items = {'Type-C', 'Type-S at 1 atm', 'Type-S at 1 GPa', 'Type-S at 2 GPa', 'Type-S at 3 GPa', 'Type-S at 4 GPa', 'Type-S at 5 GPa'};

app.ThermocoupleDropDown.Position = [107 86 75 22];

app.ThermocoupleDropDown.Value = 'Type-C';

% Create RightPanel

app.RightPanel = uipanel(app.GridLayout);

app.RightPanel.Layout.Row = 1;

app.RightPanel.Layout.Column = 2;

app.RightPanel.Scrollable = 'on';

% Create TabGroup

app.TabGroup = uitabgroup(app.RightPanel);

app.TabGroup.Position = [7 6 408 387];

% Create DataSelectionTab

app.DataSelectionTab = uitab(app.TabGroup);

app.DataSelectionTab.Title = 'Data Selection';

% Create figure1

app.figure1 = uiaxes(app.DataSelectionTab);

title(app.figure1, 'Figure 1: Temporary Selection')

xlabel(app.figure1, 'Index')

ylabel(app.figure1, 'Data')

app.figure1.XGrid = 'on';

app.figure1.XMinorGrid = 'on';

app.figure1.YGrid = 'on';

app.figure1.YMinorGrid = 'on';

app.figure1.GridAlpha = 0.15;

app.figure1.Position = [8 204 391 150];

% Create figure2

app.figure2 = uiaxes(app.DataSelectionTab);

title(app.figure2, 'Figure 2: Final Selection')

xlabel(app.figure2, 'Index')

ylabel(app.figure2, 'Data')

app.figure2.XGrid = 'on';

app.figure2.XMinorGrid = 'on';

app.figure2.YGrid = 'on';

app.figure2.YMinorGrid = 'on';

app.figure2.Position = [1 44 407 154];

% Create NotSelectedLabel

app.NotSelectedLabel = uilabel(app.DataSelectionTab);

app.NotSelectedLabel.FontWeight = 'bold';

app.NotSelectedLabel.FontColor = [0 0 1];

app.NotSelectedLabel.Position = [1 8 78 22];

app.NotSelectedLabel.Text = 'Not Selected';

% Create T\_BeforeLabel

app.T\_BeforeLabel = uilabel(app.DataSelectionTab);

app.T\_BeforeLabel.FontWeight = 'bold';

app.T\_BeforeLabel.FontColor = [0.3922 0.8314 0.0745];

app.T\_BeforeLabel.Position = [90 8 58 22];

app.T\_BeforeLabel.Text = 'T\_Before';

% Create T\_AfterLabel

app.T\_AfterLabel = uilabel(app.DataSelectionTab);

app.T\_AfterLabel.FontWeight = 'bold';

app.T\_AfterLabel.FontColor = [1 0 0];

app.T\_AfterLabel.Position = [148 8 48 22];

app.T\_AfterLabel.Text = 'T\_After';

% Create V\_PositiveLabel

app.V\_PositiveLabel = uilabel(app.DataSelectionTab);

app.V\_PositiveLabel.FontWeight = 'bold';

app.V\_PositiveLabel.Position = [195 8 66 22];

app.V\_PositiveLabel.Text = 'V\_Positive';

% Create V\_NegativeLabel

app.V\_NegativeLabel = uilabel(app.DataSelectionTab);

app.V\_NegativeLabel.FontWeight = 'bold';

app.V\_NegativeLabel.FontColor = [1 0 1];

app.V\_NegativeLabel.Position = [260 8 70 22];

app.V\_NegativeLabel.Text = 'V\_Negative';

% Create ResistivityTab

app.ResistivityTab = uitab(app.TabGroup);

app.ResistivityTab.Title = 'Resistivity';

% Create figure3

app.figure3 = uiaxes(app.ResistivityTab);

title(app.figure3, 'Figure 3: Electrical Resistivity')

xlabel(app.figure3, 'Temperature [K]')

ylabel(app.figure3, 'ρ [μΩcm]')

app.figure3.XGrid = 'on';

app.figure3.XMinorGrid = 'on';

app.figure3.YGrid = 'on';

app.figure3.YMinorGrid = 'on';

app.figure3.Position = [0 5 399 349];

% Create OutputTab

app.OutputTab = uitab(app.TabGroup);

app.OutputTab.Title = 'Output';

% Create Table

app.Table = uitable(app.OutputTab);

app.Table.ColumnName = {'T [K]'; 'ρ [µΩcm]'; 'err ρ'; 'κ [W/(mK)]'; 'err κ'};

app.Table.RowName = {};

app.Table.ColumnEditable = true;

app.Table.CellEditCallback = createCallbackFcn(app, @TableCellEdit, true);

app.Table.Position = [8 8 391 349];

% Create InformationTab

app.InformationTab = uitab(app.TabGroup);

app.InformationTab.Title = 'Information';

% Create InstructionsTextAreaLabel

app.InstructionsTextAreaLabel = uilabel(app.InformationTab);

app.InstructionsTextAreaLabel.HorizontalAlignment = 'right';

app.InstructionsTextAreaLabel.Position = [8 325 67 22];

app.InstructionsTextAreaLabel.Text = 'Instructions';

% Create InstructionsTextArea

app.InstructionsTextArea = uitextarea(app.InformationTab);

app.InstructionsTextArea.Position = [90 236 309 113];

app.InstructionsTextArea.Value = {'Step 1: Load data from .xlsx, .csv, or .xls.'; ''; ''; 'Step 2: Update the input parameters to those of your experiment. '; ''; 'Noise (optional) - input the range of indexes to be removed from raw data. This step has to be done before clicking on Plot Data.'; ''; 'Outliers Degrees of Freedom - Condition to limit the deviations from the pattern of voltage and temperature.'; ''; 'Discs - The default scenario does not account for the voltage drop contribution of the discs (option ''None''). Otherwise, the fitted data is substracted from the total voltage drop measurements. '; ''; ''; 'Step 3: Plot Data'; ''};

% Create FiguresTextAreaLabel

app.FiguresTextAreaLabel = uilabel(app.InformationTab);

app.FiguresTextAreaLabel.HorizontalAlignment = 'right';

app.FiguresTextAreaLabel.Position = [8 181 46 22];

app.FiguresTextAreaLabel.Text = 'Figures';

% Create FiguresTextArea

app.FiguresTextArea = uitextarea(app.InformationTab);

app.FiguresTextArea.Position = [74 98 325 107];

app.FiguresTextArea.Value = {'After loading the data, Figure 1 will display the raw data as seen by the multimeter.'; ''; 'When plotting the data, Figure 1 is updated to show the temperature (blue) and voltage (black) selections. This is a temporary selection.'; ''; 'The final selection (temperature before, temperature after, voltage positif and voltage negative) is displayed in Figure 2. '};

% Create TextArea

app.TextArea = uitextarea(app.InformationTab);

app.TextArea.FontSize = 9;

app.TextArea.BackgroundColor = [0.9412 0.9412 0.9412];

app.TextArea.Position = [8 25 391 56];

app.TextArea.Value = {'References: '; ''; 'Chu, T. K. & Chi, T. C. (1981) Properties of Selected Ferrous Alloying Elements, Vol. III-1., McGraw-Hill.'; ''; 'Gomi, H., & Yoshino, T. (2019). Resistivity, Seebeck coefficient, and thermal conductivity of platinum at high pressure and temperature. Physical Review B, 100(21), 214302. '; ''; 'Joshua A. H. Littleton, Richard A. Secco, Wenjun Yong, and Meryem Berrada (2019). "Electrical resistivity and thermal conductivity of W and Re up to 5 GPa and 2300 K", Journal of Applied Physics 125, 135901.'};

% Create ForfeedbackpleasecontactMeryemBerradaatmberradauwocaLabel

app.ForfeedbackpleasecontactMeryemBerradaatmberradauwocaLabel = uilabel(app.InformationTab);

app.ForfeedbackpleasecontactMeryemBerradaatmberradauwocaLabel.WordWrap = 'on';

app.ForfeedbackpleasecontactMeryemBerradaatmberradauwocaLabel.FontSize = 9;

app.ForfeedbackpleasecontactMeryemBerradaatmberradauwocaLabel.Position = [8 0 327 22];

app.ForfeedbackpleasecontactMeryemBerradaatmberradauwocaLabel.Text = 'For feedback, please contact Meryem Berrada at mberrada@uwo.ca';

% Show the figure after all components are created

app.RhoUIFigure.Visible = 'on';

end

end

% App creation and deletion

methods (Access = public)

% Construct app

function app = Rho\_exported

% Create UIFigure and components

createComponents(app)

% Register the app with App Designer

registerApp(app, app.RhoUIFigure)

if nargout == 0

clear app

end

end

% Code that executes before app deletion

function delete(app)

% Delete UIFigure when app is deleted

delete(app.RhoUIFigure)

end

end

end