```
clc
clear
load('Parameter.mat')
load('Temp_cal.mat')
folder = uigetdir();
files = dir([folder,'\*.mat']);
[~,idx] = sort([files.datenum]); % Erstellt Indizes f?r die Sortierung
 der Daten nach der 2. Zeile
files = files(idx,:); % Sortiert die Daten anhand der Indizies und
 schreibt sie in das 'Sorted' array

for i=1: length(files) % load files and write data into Values array
    load([folder,'\',files(i).name]) % loads the file
    raw(i,:) = data(1,:);
    data_corr(i,:) = data(1,:)* sqrt((243.88-temp_cal)/(243.88-Temp));
    voltages(i,1) = mean(data(1,:)); % writes the average of all
 mesured Voltages into the forst column
    voltages_corr(i,1) = mean(data(1,:)* sqrt((243.88-temp_cal)/
(243.88-Temp)));
end

pos = linspace(0,184,24);
geschw = speed(param, voltages);
figure(1)
%plot(pos, geschw,'-rx')
%hold on
geschw_corr = speed(param, voltages_corr);
plot(pos, geschw_corr,'-bx')
%title('Nachlaufdelle')
legend({'geschw.','geschw.
 corr.'},'Location','southwest','FontSize',12)
xlabel('Position [mm]','FontSize',18)
ylabel('Geschwindigkeit [m/s]','FontSize',18)
%hold off

%_____________________________________________________________________
%
%Fourier Transform: muss noch f?r alle eingelesen datenpunkte
 angepasst
L = length(data); % signal length
Fs = 1000; % sample frequency
T = 1/Fs; % sample time
t = (0:L-1)*T; %time vector

%fft
NFFT = 2^nextpow2(L); % Next power of 2 from length of y
f = Fs/2*linspace(0,1,NFFT/2+1);

for i=1:length(files)
    Y(i,:) = fft(data_corr(i,:)-mean(data_corr(i,:)),NFFT)/L;
end
for i=1: size(Y,1)
```

```
    [~,I]= max(abs(Y(i,1:NFFT/2+1)));
    max_freq(i) = f(I);
end

% Plot single-sided amplitude spectrum.
figure(2)
plot(f,2*abs(Y(12,1:NFFT/2+1)))
%title('Amplituden Spektrum der Daten')
xlabel('Frequenz [Hz]','FontSize',18)
ylabel('Magnitude','FontSize',18)
pos = linspace(0,184,24);
figure(3)
plot(pos,max_freq,'-x')
%title('Dominante Frequenz')
xlabel('Position [mm]','FontSize',18)
ylabel('Frequenz [Hz]','FontSize',18)

char_length = 0.0534;

figure(4)
plot_idx = 3000;
contourf(f(1:plot_idx).*char_length/16,pos,2*abs(Y(:,1:plot_idx)))
%title('Contour Plot Frequenzen')
xlabel('Strouhal Zahl','FontSize',18)
ylabel('Position [mm]','FontSize',18)

char_length = 0.0534;
for i=1: length(max_freq)
    strouhal(i) = (max_freq(i) * char_length) / Speed;
end
figure(5)
plot(pos,strouhal,'-x')
%title('Strouhal Nummern')
xlabel('Position [mm]','FontSize',18)
ylabel('Strouhal Zahl','FontSize',18)

figure(6)
Cd = wake(geschw_corr);

for i=1:size(raw,1)
    standard_dev(i) = std(raw(i,:));
end
figure(8)
plot(pos, standard_dev,'-x')
%title('Standardabweichung')
xlabel('Position [mm]','FontSize',18)
ylabel('Standardabweichung','FontSize',18)
Speed_diff = max(geschw_corr)-min(geschw_corr);

Error using dir
Invalid path. The path must not contain a null character.

Error in Auslesen_cta (line 6)
files = dir([folder,'\*.mat']);
```

Published with MATLAB® R2019b

```
clc
clear

folder = uigetdir(); % ?ffnet Auswahlfenster f?r Ordner mit
 Kalibrierungsdaten
files = dir([folder, '\*.mat']); % Struct mit file-namen
Values = zeros(180,3); % Speicherzuweisung
ydata = zeros(36,1); % Speicherzuweisung
xdata = zeros(36,1); % Speicherzuweisung

for i=1: length(files) % L?dt files und schreibt sie in ein array
    load([folder,'\', files(i).name]) % L?dt files
    Values(i,1) = mean(data(1,:)); % Schreibt den Mittelwert aller
 gemessenen Spannungen einer Geschwindigkeit in die erste Spalte
    Values(i,2) = Speed; % Schreibt die gemessene Geschwindigkeit in
 die zweite Spalte
    Values(i,3) = Temp; % Schreibt die Temperatur w?hrend der
 Kalibrierung in die dritte Spalte
end

[~,idx] = sort(Values(:,2)); % Erstellt Indizes f?r die Sortierung der
 Daten nach der 2. Zeile
Sorted = Values(idx,:); % Sortiert die Daten anhand der Indizies und
 schreibt sie in das 'Sorted' array

plot(Sorted(:,2), Sorted(:,1),'ro') % Plottet das Sorted array als
 rote Kreise
axis([0 45 1.7 2.6]) % Achsenbema?ung
hold on
grid on


for i=1 : 36 % Bildet den Mittelwert aller Daten mit der gleichen
 Geschwindigkeit und schreibt sie in das ydata array
    if i == 1
        ydata(i) = mean(Sorted(1:5,1));
    else
        ydata(i) = mean(Sorted((i-1)*5+1:i*5,1));
    end
end

for i=1 : 36 % Nimmt jeden Geschwindigkeitswert einmal und schreibt
 ihn in das xdata array
    xdata(i) = Sorted(i*5,2);
end

plot(xdata, ydata, 'b.') % Plottet xdata und ydata als blaue Punkte

param0 = [0, 0, 0.45]; % Array mit vermuteten Anfangsparametern
fun = @(param, x) sqrt(param(1) + param(2) * x.^param(3)); % Funktion
```

```
param = lsqcurvefit(fun, param0, xdata, ydata); % Parameter der
 Funktion an die x und y Daten anpassen. Paramter in param array
 speichern

Volt = voltage(param, xdata); % Berechnen der Spannungen anhand der
 neuen Funktion mit den berechneten Parametern
figure(1)
plot(xdata, Volt,'r')
xlabel('Geschwindigkeit (m/s)','FontSize',18)
ylabel('Ausgangsspannung (V)','FontSize',18)
hold off

speed_calc = speed(param, ydata); % Berechnen der Geschwindigkeiten
 aus den gemessenen Spannungen mithilfe der invertierten Funktion
figure(2)
plot(ydata, speed_calc, '-bo')
ylabel('Geschwindigkeit (m/s)','FontSize',18)
xlabel('Ausgangsspannung (V)','FontSize',18)

deviation = speed_calc - xdata;
standard_deviation = std(deviation);

save('Parameter.mat', 'param')
temp_cal = Values(1,3);
save('Temp_cal.mat', 'temp_cal')

%function v = voltage(param, x) % Funktion V(u)
    %v = sqrt(param(1) + param(2)*x.^ param(3));
%end

%function u = speed(param,x) % Funktion u(V) Inverse von V(u)
    %u = ((x.^(2) + (-1)* param(1))/param(2)).^(1/param(3));
%end
```

Error using dir
Invalid path. The path must not contain a null character.

Error in Kalibrierung (line 5)
files = dir([folder, '.mat']); % Struct mit file-namen*

Published with MATLAB® R2019b

```
function [Cd] = wake(geschw_corr)
data_speed = geschw_corr;

% pos=(1:24);
pos = linspace(0, 0.184, 24);
pos = pos';
[~,id] = min(data_speed);
middle = pos(id);
beta = [1000,-1000,min(data_speed),max(data_speed),-middle];
options = optimoptions('lsqcurvefit','MaxFunctionEvaluations',2000);
formula1 = @(beta,x) (1+beta(1).*(x+beta(5)).^2) .* exp(beta(2).*(x
+beta(5)).^2) * (beta(3) - beta(4)) + beta(4);
beta1 = lsqcurvefit(formula1,beta,pos,data_speed,[0, -Inf, -Inf, -Inf,
 -Inf,],[Inf, Inf, Inf, Inf, Inf],options);
disp(beta1)

%plot(linspace(0,184,24), geschw_corr,'b*')
%hold on


delle = formula1(beta1, pos);
plot(linspace(0,184,24),delle)
%legend({'geschw.','function
 fit'},'Location','Southwest','FontSize',12)
xlabel('Position [mm]','FontSize',18)
ylabel('Geschwindigkeit [m/s]','FontSize',18)
hold on

ref_length = 0.0534;
fun = @(x) (formula1(beta1, x)/max(data_speed)).*(1-(formula1(beta1,
 x)/max(data_speed)));
Cd = 2 * integral(fun,0,0.184)/ref_length;
end
```

```
Not enough input arguments.

Error in wake (line 2)
data_speed = geschw_corr;
```

Published with MATLAB® R2019b