Aalborg Universitet

AALBORG UNIVERSITY

Simple Data Pre-processing of the Laser Flash Analysis Results from the LFA 447 Apparatus

Johra, Hicham

Publication date:
2019

Document Version
Publisher's PDF, also known as Version of record

Link to publication from Aalborg University

Citation for published version (APA):
Johra, H. (2019). *Simple Data Pre-processing of the Laser Flash Analysis Results from the LFA 447 Apparatus*. Department of Civil Engineering, Aalborg University. DCE Lecture notes No. 72

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

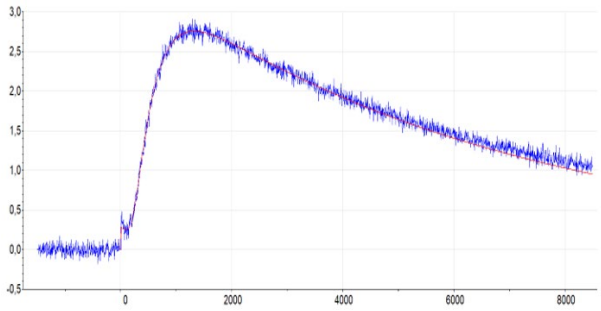
- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal -

Take down policy
If you believe that this document breaches copyright please contact us at vbn@aub.aau.dk providing details, and we will remove access to the work immediately and investigate your claim.

Downloaded from vbn.aau.dk on: July 02, 2025

Simple Data Pre-processing of the Laser Flash Analysis Results from the LFA 447 Apparatus

Hicham Johra



DEPARTMENT OF CIVIL ENGINEERING
AALBORG UNIVERSITY

ISSN 1901-7286
DCE Lecture Notes No. 72

Aalborg University
Department of Civil Engineering
Architectural Engineering

DCE Lecture Notes No. 72

Simple Data Pre-processing of the Laser Flash Analysis Results from the LFA 447 Apparatus

by

Hicham Johra

October 2019

© Aalborg University

Scientific Publications at the Department of Civil Engineering

Technical Reports are published for timely dissemination of research results and scientific work carried out at the Department of Civil Engineering (DCE) at Aalborg University. This medium allows publication of more detailed explanations and results than typically allowed in scientific journals.

Technical Memoranda are produced to enable the preliminary dissemination of scientific work by the personnel of the DCE where such release is deemed to be appropriate. Documents of this kind may be incomplete or temporary versions of papers—or part of continuing work. This should be kept in mind when references are given to publications of this kind.

Contract Reports are produced to report scientific work carried out under contract. Publications of this kind contain confidential matter and are reserved for the sponsors and the DCE. Therefore, Contract Reports are generally not available for public circulation.

Lecture Notes contain material produced by the lecturers at the DCE for educational purposes. This may be scientific notes, lecture books, example problems or manuals for laboratory work, or computer programs developed at the DCE.

Theses are monograms or collections of papers published to report the scientific work carried out at the DCE to obtain a degree as either PhD or Doctor of Technology. The thesis is publicly available after the defence of the degree.

Latest News is published to enable rapid communication of information about scientific work carried out at the DCE. This includes the status of research projects, developments in the laboratories, information about collaborative work and recent research results.

Published 2019 by
Aalborg University
Department of Civil Engineering
Thomas Manns Vej 23
DK-9220 Aalborg Ø, Denmark

Printed in Aalborg at Aalborg University

ISSN 1901-7286
DCE Lecture Notes No. 72

Contents

1. Foreword

The aim of this technical report is to explain how to use simple scripts (VBA Excel Macro and MATLAB) to pre-process Laser Flash Analysis raw data results from the LFA 447 Apparatus (Netzsch Gerätebau GmbH [1]) at the Building Material Characterization Laboratory of Aalborg University - Department of Civil Engineering [2]. These scripts perform cleaning and ordering of the measurement points and mean average/standard deviation calculation for arbitrary decimation range of a given parameter.

2. Introduction

The raw data measurements of the Laser Flash Analysis Apparatus LFA 447 are processed and analyzed with the Netzsch Proteus® LFA Analysis software [1]. The processed measurement result points of a given test sample can be exported as a text file report. It is recommended to export the processed measurement result points of a given test sample as a text file with “tabulation” column separator and “point” decimal symbol (see *Figure 1* and *Figure 2*).

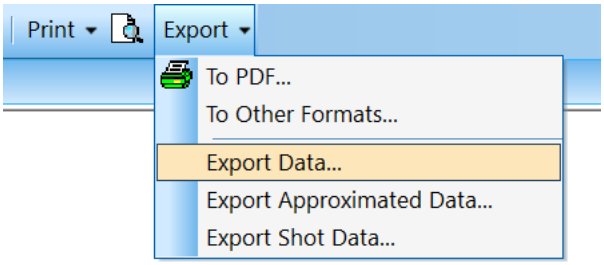


Figure 1: *Export all measurement points from the Netzsch Proteus® LFA Analysis software.*

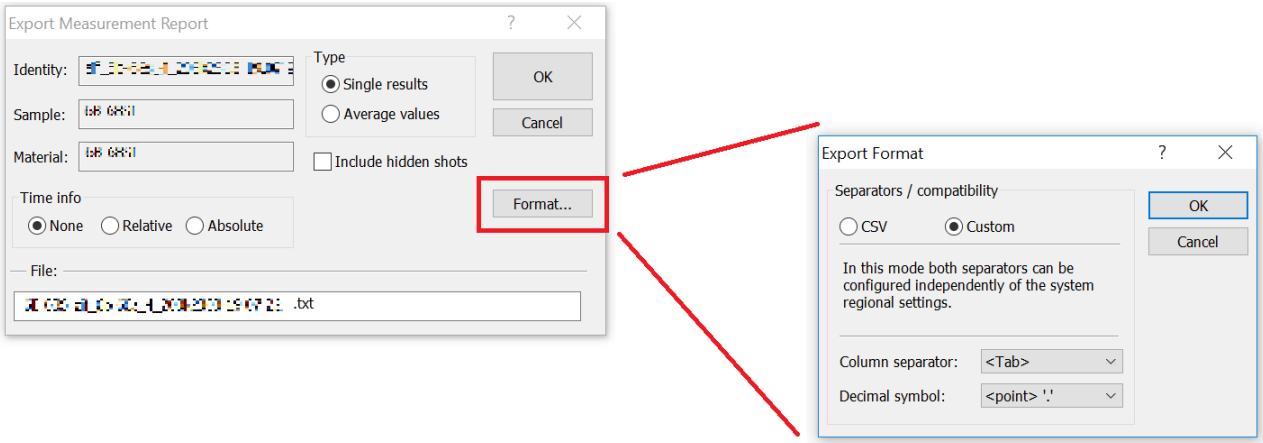


Figure 2: *Export the measurement data as a text file report with “tabulation” as column separator and “point” as decimal symbol.*

The exported measurement report text file is composed of a header describing the sample characteristics, and then followed by the measurement points organized by temperature range (see *Figure 3*). However, this data arrangement is not convenient to extract rapidly the temperature, thermal diffusivity and specific heat capacity of all measurements and have then placed consecutively in columns.

```
##Thermal_diffusivity

##General_information
#Database       lfa_results.lfa.mdb
#Instrument     #LFA_447
#Identity       XXXXXXXXXXX XX.XX.XX
#Date   20XX-XX-XX
#Material       XXXXXX
#Ref_temperature /°C    22.0
#Ref_density /(g/cm^3)  2.430
#Sample XXXXXXXX
#Type   #Single_layer
#Thickness_RT/mm        1.5110
#Diameter/mm    5.550
#Sensor InSb
#Operator       XXXXXX
#Remark_mment   XXX
#Cp_table       XXXXXXXX #01
#Expansion_table        dL_const
#CalcCode       R+p/1/0-0-0

##Results
#Shot_number    #Temperature/°C #Model  #Diffusivity/(mm^2/s)   #Pulse_type
1       47.3    Rad. + pc.      0.402   3
2       46.9    Rad. + pc.      0.392   3
3       47.0    Rad. + pc.      0.401   3
4       47.1    Rad. + pc.      0.399   3
5       46.8    Rad. + pc.      0.400   3
6       46.9    Rad. + pc.      0.399   3
7       47.1    Rad. + pc.      0.404   3
8       47.1    Rad. + pc.      0.399   3
9       46.9    Rad. + pc.      0.405   3
10      47.0    Rad. + pc.      0.400   3
#Mean   47.0            0.400
#Std_Dev        0.1             0.004
11      48.2    Rad. + pc.      0.402   3
12      48.0    Rad. + pc.      0.399   3
13      47.9    Rad. + pc.      0.399   3
14      48.0    Rad. + pc.      0.402   3
15      48.2    Rad. + pc.      0.401   3
16      47.9    Rad. + pc.      0.397   3
17      47.9    Rad. + pc.      0.402   3
18      48.0    Rad. + pc.      0.405   3
19      48.2    Rad. + pc.      0.398   3
20      47.8    Rad. + pc.      0.407   3
#Mean   48.0            0.401
#Std_Dev        0.1             0.003
21      49.0    Rad. + pc.      0.380   3
22      49.5    Rad. + pc.      0.397   3
23      48.9    Rad. + pc.      0.400   3
24      48.8    Rad. + pc.      0.400   3
25      49.1    Rad. + pc.      0.396   3
26      49.1    Rad. + pc.      0.400   3
27      48.9    Rad. + pc.      0.395   3
28      48.9    Rad. + pc.      0.404   3
29      49.0    Rad. + pc.      0.397   3
30      49.2    Rad. + pc.      0.402   3
#Mean   49.0            0.397
#Std_Dev        0.2             0.007
```

Figure 3: *LFA 447 measurement data report text file.*

The content of the report text file can directly be copy/pasted into Excel, but one can see in *Figure 4* that there are intermediate lines and columns which are unnecessary and should be deleted. However, deleting many lines manually on Excel is very tedious and time-consuming. That is why a VBA Macro or MATLAB Script performing this task automatically is a great time-saver.

3	##General_information				
4	#Database	lfa_results.lfa.mdb			
5	#Instrument	#LFA_447			
6	#Identity	XXXXXXXXXXXX XX.XX.XX			
7	#Date	20XX-XX-XX			
8	#Material	XXXXXX			
9	#Ref_temperature /°C	22			
10	#Ref_density /(g/cm^3)	2.430			
11	#Sample	XXXXXXXX			
12	#Type	#Single_layer			
13	#Thickness_RT/mm	15.110			
14	#Diameter/mm	5.550			
15	#Sensor	InSb			
16	#Operator	XXXXXX			
17	#Remark_mment	XXX			
18	#Cp_table	XXXXXXXX #01			
19	#Expansion_table	dL_const			
20	#CalcCode	R+p/l/0-0-0			
21					
22	##Results				
23	#Shot_number	#Temperature/°C	#Model	#Diffusivity/(mm^2/s)	#Pulse_type
24	1	47.3	Rad. + pc.	0.402	3
25	2	46.9	Rad. + pc.	0.392	3
26	3	47	Rad. + pc.	0.401	3
27	4	47.1	Rad. + pc.	0.399	3
28	5	46.8	Rad. + pc.	0.4	3
29	6	46.9	Rad. + pc.	0.399	3
30	7	47.1	Rad. + pc.	0.404	3
31	8	47.1	Rad. + pc.	0.399	3
32	9	46.9	Rad. + pc.	0.405	3
33	10	47	Rad. + pc.	0.4	3
34	#Mean	47		0.4	
35	#Std_Dev	0.1		0.004	
36	11	48.2	Rad. + pc.	0.402	3
37	12	48	Rad. + pc.	0.399	3
38	13	47.9	Rad. + pc.	0.399	3
39	14	48	Rad. + pc.	0.402	3
40	15	48.2	Rad. + pc.	0.401	3
41	16	47.9	Rad. + pc.	0.397	3
42	17	47.9	Rad. + pc.	0.402	3
43	18	48	Rad. + pc.	0.405	3
44	19	48.2	Rad. + pc.	0.398	3
45	20	47.8	Rad. + pc.	0.407	3
46	#Mean	48		0.401	
47	#Std_Dev	0.1		0.003	

Figure 4: *Example of a measurement report text file copy/pasted into Excel with unnecessary intermediate lines and columns which should be deleted (framed in red).*

In addition, the measurement points are not necessarily set in order according to temperature and the number of points per temperature range can have arbitrary value. It is therefore tedious to group scattered measurement points by temperature and calculate mean average and standard deviation for specific range of temperatures.

The different scripts presented in this report clean, order, decimate and calculate the mean average and standard deviation of the LFA 447 measurement data from the report text files.

3. VBA Excel Macro for pre-processing (cleaning) LFA 447 measurement report

The Microsoft Excel Macro-Enable Worksheet named “LFA_447_data_result_pre-processing” contains an Excel VBA Macro which removes automatically the unnecessary columns containing the strings “#Model” and “#Pulse_type”, and removes the unnecessary lines containing the strings “#Mean” and “#Std_Dev”. All other data is left as it is.

To “clean” the LFA 447 result report from the aforementioned columns and lines, open the result report text file with the “Notepad” software (see *Figure 5*). Select all the text of the report text file (Ctrl + A) and copy it (Ctrl + C).

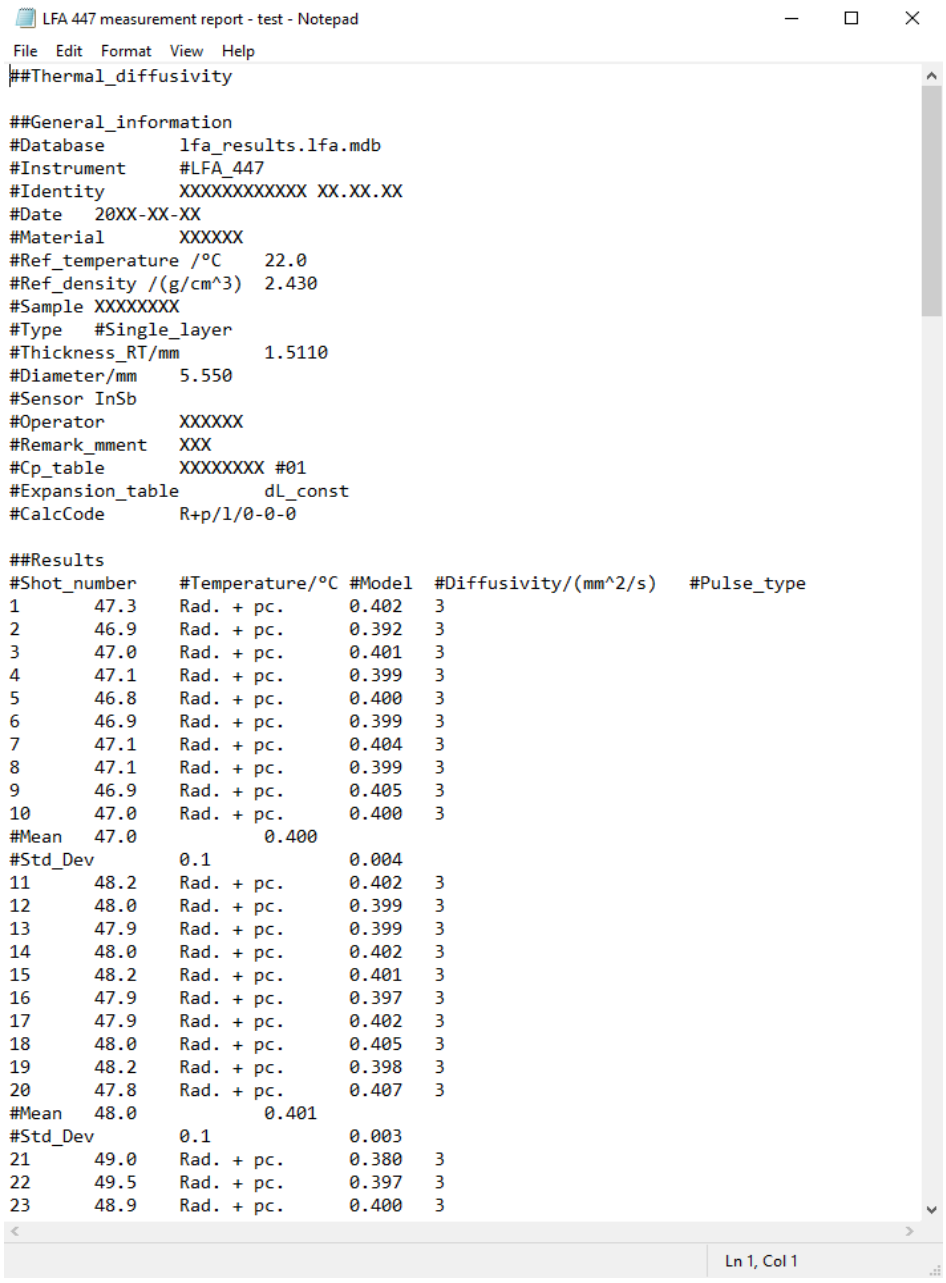
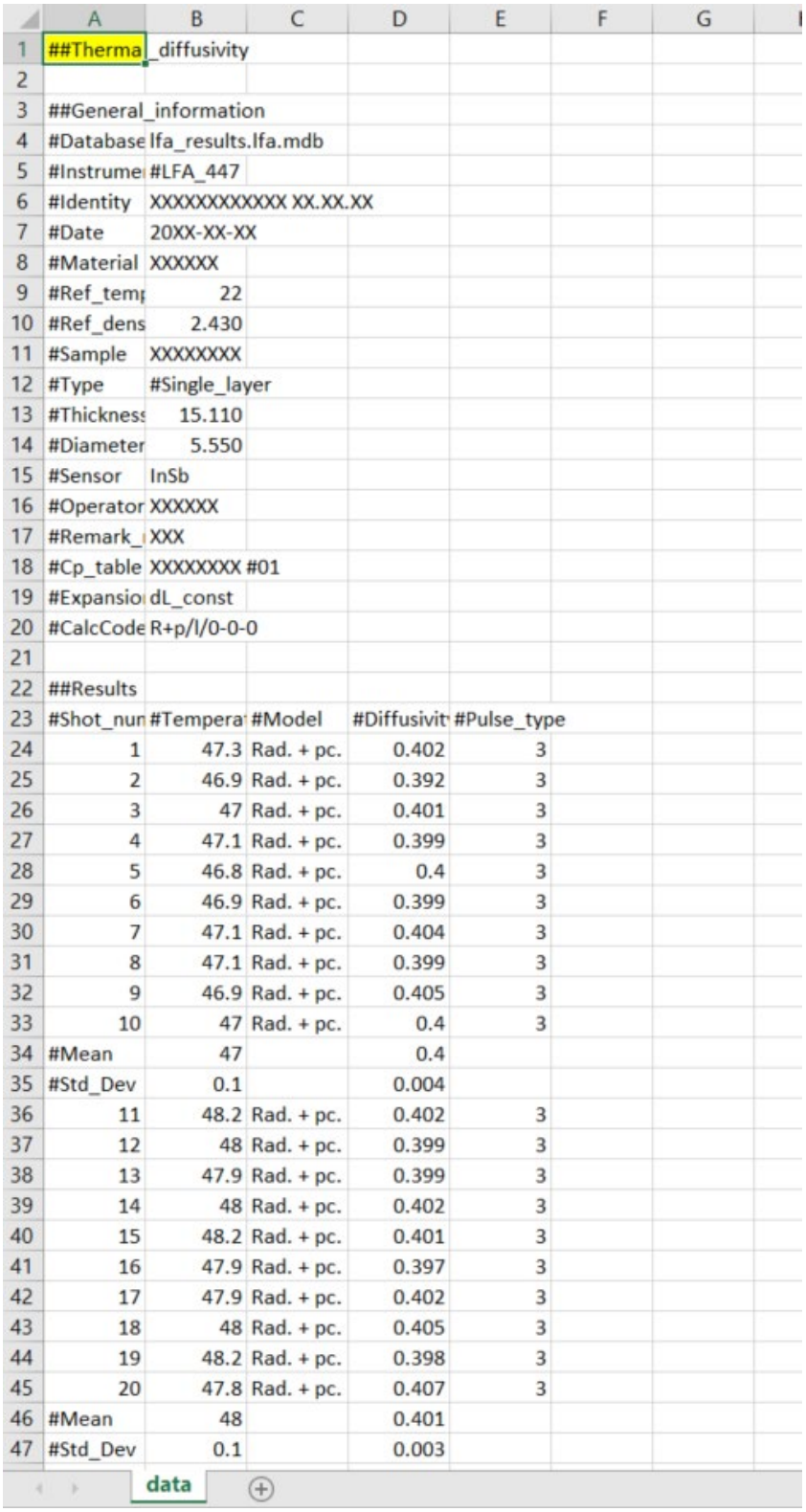


Figure 5: *LFA 447 measurement results report text file.*

Open the Excel file “LFA_447_data_result_pre-processing” and paste (Ctrl + V) the entire content of the report text file into the yellow cell “A1” of the only sheet (named “data”) of the Excel file (see *Figure 6*).



	A	B	C	D	E	F	G
1	##Therma	_diffusivity					
2							
3	##General_information						
4	#Database	lfa_results.lfa.mdb					
5	#Instrume	#LFA_447					
6	#Identity	XXXXXXXXXXX XX.XX.XX					
7	#Date	20XX-XX-XX					
8	#Material	XXXXXX					
9	#Ref_temp	22					
10	#Ref_dens	2.430					
11	#Sample	XXXXXXXX					
12	#Type	#Single_layer					
13	#Thickness	15.110					
14	#Diameter	5.550					
15	#Sensor	InSb					
16	#Operator	XXXXXX					
17	#Remark_	XXX					
18	#Cp_table	XXXXXXXX #01					
19	#Expansio	dL_const					
20	#CalcCode	R+p/l/0-0-0					
21							
22	##Results						
23	#Shot_nun	#Tempera	#Model	#Diffusivit	#Pulse_type		
24	1	47.3	Rad. + pc.	0.402	3		
25	2	46.9	Rad. + pc.	0.392	3		
26	3	47	Rad. + pc.	0.401	3		
27	4	47.1	Rad. + pc.	0.399	3		
28	5	46.8	Rad. + pc.	0.4	3		
29	6	46.9	Rad. + pc.	0.399	3		
30	7	47.1	Rad. + pc.	0.404	3		
31	8	47.1	Rad. + pc.	0.399	3		
32	9	46.9	Rad. + pc.	0.405	3		
33	10	47	Rad. + pc.	0.4	3		
34	#Mean	47		0.4			
35	#Std_Dev	0.1		0.004			
36	11	48.2	Rad. + pc.	0.402	3		
37	12	48	Rad. + pc.	0.399	3		
38	13	47.9	Rad. + pc.	0.399	3		
39	14	48	Rad. + pc.	0.402	3		
40	15	48.2	Rad. + pc.	0.401	3		
41	16	47.9	Rad. + pc.	0.397	3		
42	17	47.9	Rad. + pc.	0.402	3		
43	18	48	Rad. + pc.	0.405	3		
44	19	48.2	Rad. + pc.	0.398	3		
45	20	47.8	Rad. + pc.	0.407	3		
46	#Mean	48		0.401			
47	#Std_Dev	0.1		0.003			

data

Figure 6: *Paste all report text content into the yellow cell “A1” of the only sheet (named “data”) of the Excel file “LFA_447_data_result_pre-processing”.*

Press “Ctrl + m” to execute the Excel macro and clean the data (see *Figure 7*). One can see that all data points are now continuous and contiguous. The process will also work if other results are in the report text file such as thermal conductivity, specific heat capacity or contact resistance.

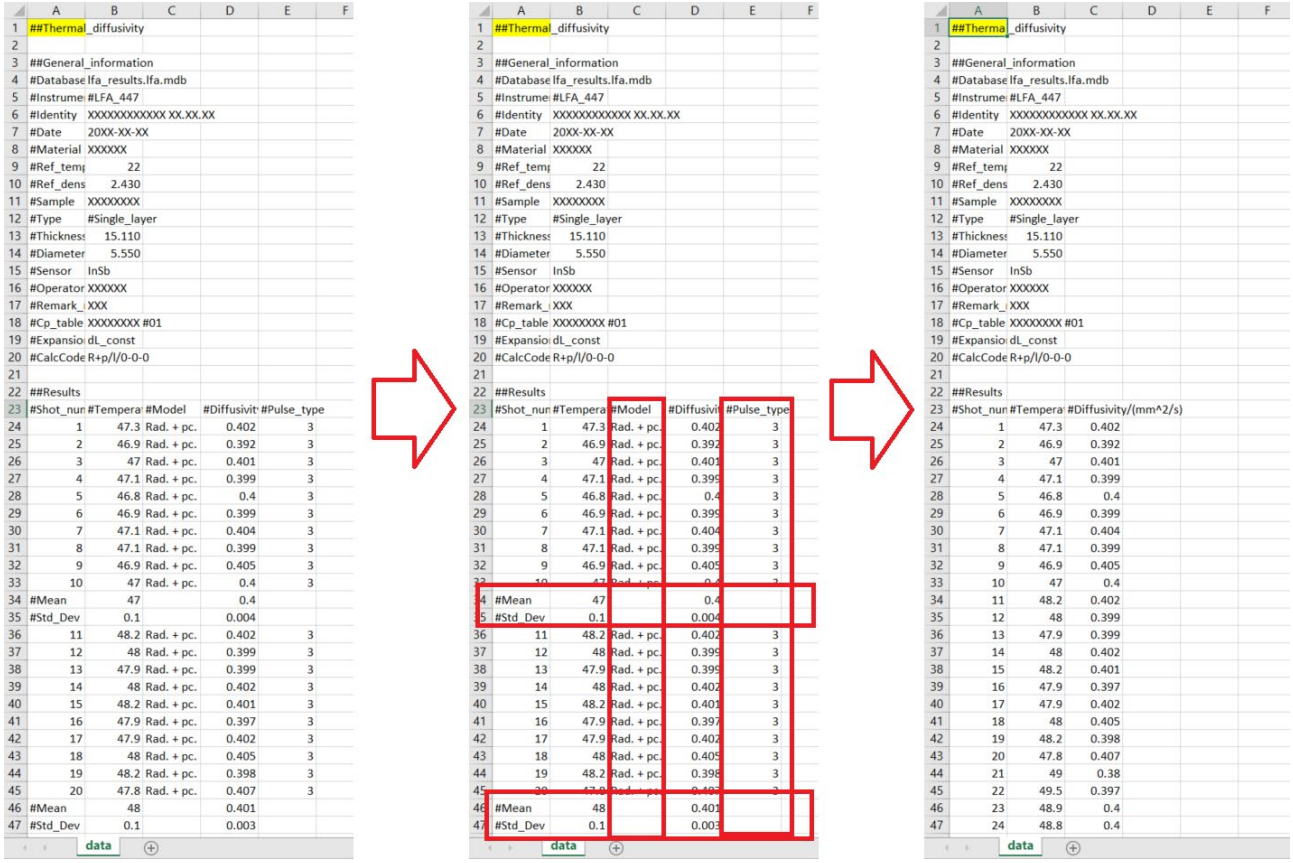


Figure 7: *Report text file content before data cleaning (left); columns and lines which will be deleted from the sheet (center); report text file content after data cleaning (right).*

4. MATLAB script for pre-processing (cleaning) LFA 447 measurement report

The MATLAB script named “LFA_447_data_result_pre_processing.m” takes a LFA 447 measurement report text file as input and creates and new text file with the report’s data without the unnecessary columns containing the strings “#Model” and “#Pulse_type”, and without the unnecessary lines containing the strings “#Mean” and “#Std_Dev”. The report information before the data header is not kept. All other data is left as it is. The output text file is placed in the same folder as the input measurement report text file. The name of the output text file is the same as the one of the input file and preceded by “clean_data_”.

To generate the “clean” LFA 447 result report, open and run the MATLAB script “LFA_447_data_result_pre_processing.m”. Select the input measurement report text file (see *Figure 8*).

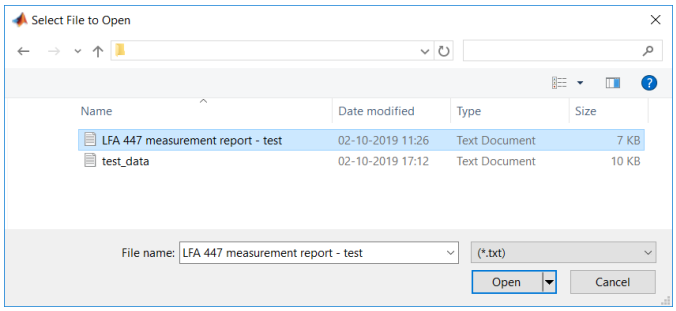


Figure 8: *Select the input measurement report text file when running the MATLAB script.*

If no error message has been displayed in the MATLAB command window, the operation has been completed successfully and a new text file has been generated in the same folder as the input measurement report text file without the unnecessary lines and columns (see *Figure 9*).

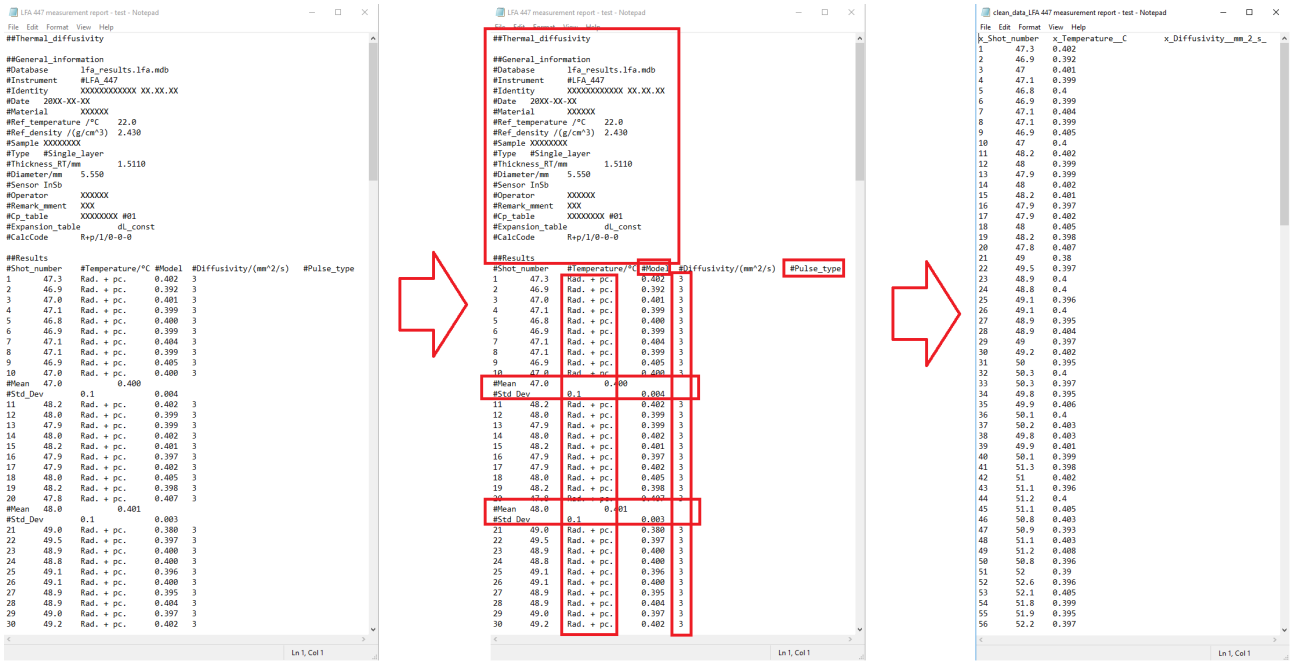


Figure 9: *Input report text file content before data cleaning (left); columns and lines which are not kept from the input file (center); output text file after data pre-processing by the MATLAB script (right).*

5. MATLAB script for averaging, decimating regularly, and calculating standard deviation of scattered data points

The MATLAB script named “average_decimate_std_dev_scattered_data.m” takes a tabulation delimited, point decimal delimiter data text file with 2 columns of scattered data points as input. From this file, it sets in order, averages and decimates (resampling) the second data column as a function of the first data column of the text file with a specific decimation/resampling step size. The standard deviation of the first data column is calculated for each decimation/resampling step. The decimation/resampling result is saved in a new output text file with the same name as the input file and preceded by “avrg_decimated_std_dev_”. The output text file is placed in the same folder as the input text file. A detailed example of the script processing is presented hereafter.

To decimate / resample a data file with 2 data columns, open and run the MATLAB script “average_decimate_std_dev_scattered_data.m”. Select the input text file (see *Figure 10*).

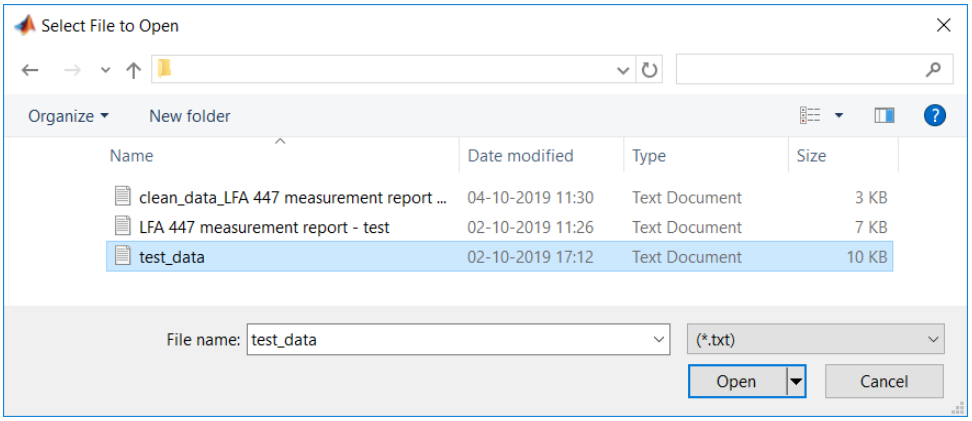


Figure 10: *Select the input text file when running the MATLAB script.*

Choose decimation / resampling step size (see *Figure 11*).

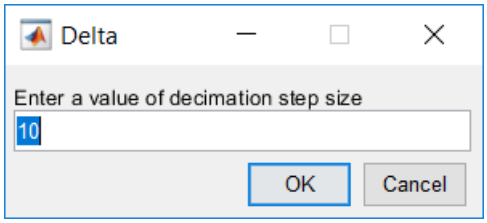


Figure 11: *Indicate the decimation / resampling step size.*

If no error message has been displayed in the MATLAB command window, the operation has been completed successfully and a new text file has been generated in the same folder as the input file. One can see in *Figure 12 (on the left)* that the input file has scattered data in the first column (the data points are not ordered in ascending or descending order). The input file has values in the first column ranging from 9.6 to 60.5. In the current example, the decimation / resampling step size is 10, meaning that the data in the second column will be averaged (and standard deviation calculated) for corresponding first column's values of 0 ±5, 10 ±5, 20 ±5, 30 ±5, 40 ±5, 50 ±5, 60 ±5 and 70 ±5. Consequently, one can see in the output file (*Figure 12 on the right*) that the data points of column 2 have been averaged (and standard deviation calculated) for column 1 average values of 0, 10, 20, 30, 40, 50, 60 and 70. In the case of column 1 average values of 0 and 70, there is no column 2 data corresponding to column 1 data points 0 ±5 or 70 ±5. The corresponding average data for column 2 is thus set to "NaN" and the standard deviation is left as an empty cell.

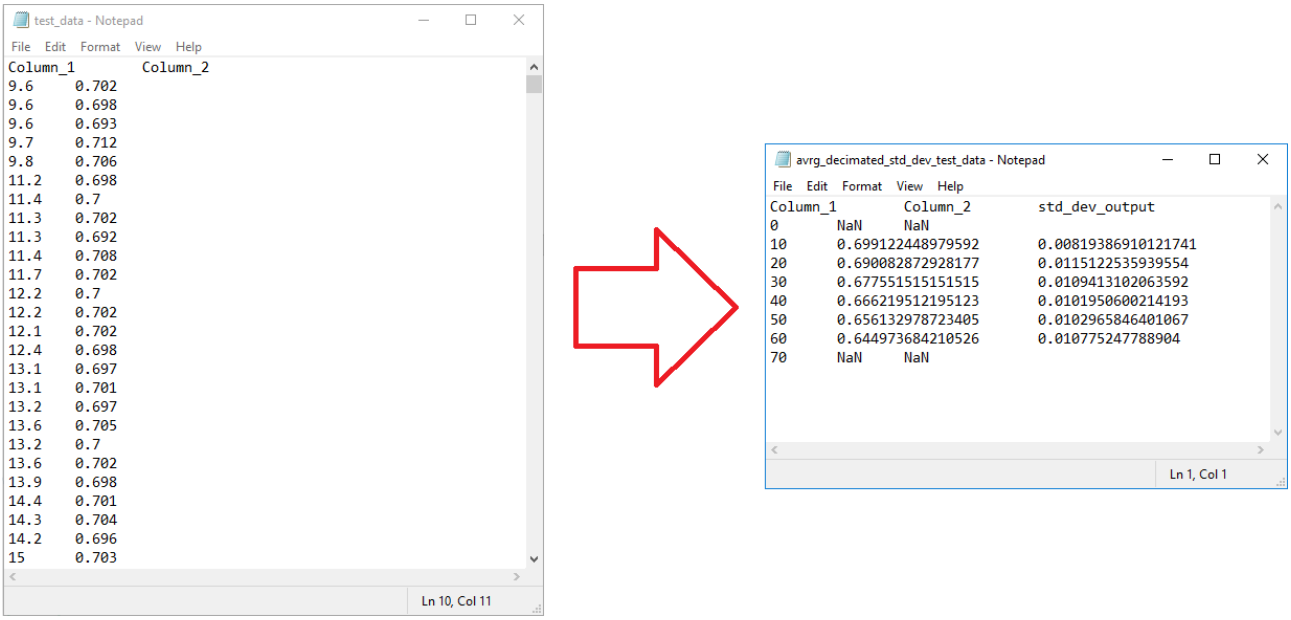


Figure 12: *Input file with scattered data in 2 columns (left); output file with first column set in ascending order and decimated according to decimation / resampling step size, second column with average of the data according to the first data column, and third column with the standard deviation of the second column for each step (right).*

6. Appendices

6.1. VBA Excel Macro code for pre-processing (cleaning) LFA measurement report

```
Sub clean_data_LFA_1()
'
' clean_data_LFA_1 Macro
' Remove the entire columns containing "#Model" and "#Pulse_type"
' Remove all the lines containing "#Mean" and "Std_Dev"

Dim FindString As String
Dim Rng As Range


' Find the column containing "#Model" and delete it entirely
With Sheets("data").UsedRange
    FindString = "#Model"
    Set Rng = .Find(What:=FindString, _
    After:=.Cells(.Cells.Count), _
    LookIn:=xlValues, _
    LookAt:=xlWhole, _
    SearchOrder:=xlByRows, _
    SearchDirection:=xlNext, _
    MatchCase:=False)

    ' If there is no "#Model" found, an error message is displayed and
the script is terminated
    If Rng Is Nothing Then
        MsgBox "Nothing found"
        Exit Sub
    End If
    Columns(Rng.Column).EntireColumn.Delete
End With


' Find the column containing "#Pulse_type" and delete it entirely
With Sheets("data").UsedRange
    FindString = "#Pulse_type"
    Set Rng = .Find(What:=FindString, _
    After:=.Cells(.Cells.Count), _
    LookIn:=xlValues, _
    LookAt:=xlWhole, _
    SearchOrder:=xlByRows, _
    SearchDirection:=xlNext, _
    MatchCase:=False)
```

```
    ' If there is no "#Pulse_type" found, an error message is displayed
and the script is terminated
    If Rng Is Nothing Then
        MsgBox "Error"
        Exit Sub
    End If
    Columns(Rng.Column).EntireColumn.Delete
End With


' Find all the lines containing "#Mean" and delete them
' Find the first occurrence of "#Mean"
FindString = "#Mean"

With Sheets("data").UsedRange
    Set Rng = .Find(What:=FindString, _
    After:=.Cells(.Cells.Count), _
    LookIn:=xlValues, _
    LookAt:=xlWhole, _
    SearchOrder:=xlByRows, _
    SearchDirection:=xlNext, _
    MatchCase:=False)
End With

' Loop to delete the entire line containing "#Mean" until there is no
"#Mean" which can be found
Do While Not Rng Is Nothing
Rows(Rng.Row).EntireRow.Delete

With Sheets("data").UsedRange
    Set Rng = .Find(What:=FindString, _
    After:=.Cells(.Cells.Count), _
    LookIn:=xlValues, _
    LookAt:=xlWhole, _
    SearchOrder:=xlByRows, _
    SearchDirection:=xlNext, _
    MatchCase:=False)
End With
Loop

' Find all the lines containing "#Std_Dev" and delete them
' Find the first occurrence of "#Std_Dev"
FindString = "#Std_Dev"

With Sheets("data").UsedRange
    Set Rng = .Find(What:=FindString, _
```

```
    After:=.Cells(.Cells.Count), _
    LookIn:=xlValues, _
    LookAt:=xlWhole, _
    SearchOrder:=xlByRows, _
    SearchDirection:=xlNext, _
    MatchCase:=False)
End With

' Loop to delete the entire line containing "#Std_Dev" until there is no
"#Std_Dev" which can be found
Do While Not Rng Is Nothing
Rows(Rng.Row).EntireRow.Delete

With Sheets("data").UsedRange
    Set Rng = .Find(What:=FindString, _
    After:=.Cells(.Cells.Count), _
    LookIn:=xlValues, _
    LookAt:=xlWhole, _
    SearchOrder:=xlByRows, _
    SearchDirection:=xlNext, _
    MatchCase:=False)
End With
Loop

Application.Goto Range("A1")

End Sub
```

6.2. MATLAB code for pre-processing (cleaning) LFA measurement report

```
%% Removes unnecessary columns and lines from a LFA 447 measurement
report text file and save it in a new text file
% Prompt user for location of input file
% Remove the unnecessary columns containing "#Model" and "#Pulse_type"
% Remove the unnecessary lines containing "#Mean" and "#Std_Dev"
% Remove all text before the tabulated data (before the data header)
% A new result file is generated and saved in the same folder as the
input file
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%

% Prompt user for input file
[file,path,indx] = uigetfile('*.txt');
if isequal(file,0)
   disp('Error: the data pre-processing has been terminated.')
   return;
end

% get file name
filename = [path file];
[fileID,errmsg] = fopen(filename);
if isempty(errmsg)
   disp('Input file has been loaded correctly.')
else
   disp('Error: the data pre-processing has been terminated.')
   return;
end

% Import data
cdata = readtable(filename);
fclose(fileID);

% Get list of all variable from the table
list_var_names = cdata.Properties.VariableNames;

% Find index of variable "Shot_number"
index_shot = find(contains(list_var_names,'Shot_number'));
if isempty(index_shot)
    disp('The report text file is not valid. This script will be
terminated immediately.');
    return
end

% Find index of variable "Model"
index_model = find(contains(list_var_names,'Model'));
if isempty(index_model)
    disp('The report text file is not valid. This script will be
terminated immediately.');
    return
end
```

```
% Find index of variable "Pulse_type"
index_pulse_type = find(contains(list_var_names,'Pulse_type'));
if isempty(index_pulse_type)
    disp('The report text file is not valid. This script will be
terminated immediately.');
    return
else
    disp('The report text file is valid.');
end

% Delete columns "model" and "Pulse_type"
cdata =
removevars(cdata,{char(string(cdata.Properties.VariableNames(index_model)
)), char(string(cdata.Properties.VariableNames(index_pulse_type)))});

% Get back the new Shot_number variable number where the #Mean and
#Std_Dev string should be found
index_shot = find(contains(list_var_names,'Shot_number'));

% Get the real name of the shot var
shot_var_name = char(cdata.Properties.VariableNames(index_shot));

% Create new table without the #Mean and #Std_Dev lines
clean_data = cdata((cdata.(shot_var_name) ~= "#Mean" &
cdata.(shot_var_name) ~= "#Std_Dev"),:); % cdata.(shot_var_name) with the
"()" because "shot_var_name" is not a variable but an expression which
give a char variable name

% Create a new text file with the clean data table in same folder as
input file
new_filename = [path 'clean_data_' file];
writetable(clean_data,new_filename,'Delimiter','tab');

% Create final message and display it with name of the new created file
msg = strcat('A clean result data text file named "clean_data_', file, '"
has been created in the same folder as the input file.');
disp(msg);
```

6.3. MATLAB code for averaging, decimating regularly, and calculating standard deviation of scattered data points

```
%% Average and decimate regularly scattered data with fixed defined step
% Prompt user for location of input file
% Prompt user for step size for decimation of the averaged data
% Group Y scattered data (only 1 Y column in a 2 column data set with X
as first and Y as second column) in chunck and average it and decimate it
according to the step size of decimation of X
% Minimum and maximum are defined according to extremas in the X data and
according to the decimation step size
% Standard deviation is calculated for each step adn stored in additional
column
% A new result file is generated and saved in the same folder as the
input file
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%

% Prompt user for input file
[file,path,indx] = uigetfile('*.txt');
if isequal(file,0)
   disp('Error: the data treatment has been terminated.')
   return;
end

% get file name
filename = [path file];
[fileID,errmsg] = fopen(filename);
if isempty(errmsg)
   disp('Input file has been loaded correctly.')
else
   disp('Error: the data treatment has been terminated.')
   return;
end

% Import data
cdata = readtable(filename);
fclose(fileID);

% Check that there is at least 2 columns (because only the 2 first
columns
% are used for data treatment
if length(cdata.Properties.VariableNames) < 2
    disp('Error: input file data is wrong. The data treatment has been
terminated.')
   return;
end

% Check that data is longer than 1 row
if length(cdata{:,1}) > 1 && length(cdata{:,2}) > 1
   disp('Input data is valid.')
else
```

```
   disp('Error: wrong data. The data treatment has been terminated.')
   return;
end

% Prompt user for decimation step size
prompt = {'Enter a value of decimation step size'};
dlgtitle = 'Delta';
definput = {'1'};
opts.Interpreter = 'tex';
answer = inputdlg(prompt,dlgtitle,[1 40],definput,opts);
delta = str2double(answer); % convert answer into numerical

% Get first point in X data and last point in X data
first = min(cdata{:,1});
last = max(cdata{:,1});
distance = last - first;

% Check that the step size is not zero and that it is not larger than the
% distance between the first and last point
if delta > 0 && delta < distance
   disp('Step size for decimation is valid.')
else
   disp('Error: wrong step size for decimation. The data treatment has
been terminated.')
   return
end

% round to the closest first data point or round lower according to
% decimation step size
starting_point = delta*floor(first/delta);
% do the same for the last data point
ending_point = delta*ceil(last/delta);
% Because of numerical truncation, there might be one step too many at
the
% beginning and / or at the end. But this step will be empty or NaN

X_output = (starting_point:delta:ending_point)';
Y_output = NaN(length(X_output),1);
std_dev_output = NaN(length(X_output),1);

for i = 1:length(X_output)
% for each step in X_output, get the average of the Y data values
% at distance delta/2 from the current X_output.
   selection_array = (cdata{:,1} >= X_output(i)-delta/2).*(cdata{:,1} <=
X_output(i)+delta/2); % have to fulfill 2 conditions with ".*"
   sub_table=(cdata(logical(selection_array),2));
   Y_output(i) = mean(sub_table{:,1});
   std_dev_output(i) = std(sub_table{:,1});
end

table_output = table(X_output,Y_output,std_dev_output); % Arrange all
data in a table
```

```
table_output.Properties.VariableNames([1 2]) =
cdata.Properties.VariableNames([1 2]); % Rename table header with header
of original input file

% Write table data into text file in same folder as input file
new_filename = [path 'avrg_decimated_std_dev_' file];
writetable(table_output,new_filename,'Delimiter','tab');

disp('Result file has been created in the same file as the input file.');
```

References

[1] Netzsch Gerätebau GmbH. Operating Instructions Nano-Flash-Apparatus LFA 447, 2001.

[2] Building Material Characterization Laboratory of Aalborg University, Department of Civil Engineering, Aalborg, Denmark.
https://buildingmaterials.civil.aau.dk