Handbook for NISI Code

Christian Dürnhofer
Fabian Hufgard

12/10/2020

Universität Stuttgart
Institut für Raumfahrtsysteme
Pfaffenwaldring 29
70569 Stuttgart
Germany
Tel: +49 711 685 62387
Fax: +49 711 685 63596

HEFDiG
High Enthalpy Flow Diagnostics Group

Report Documentation Page

Project:	
Document:	
Doc. No.:	
Issue:	1.0

Prepared		Date
(Main Author)	C. Dürnhofer, F. Hufgard	12/10/2020
Checked		Date
(Tech. Resp.)	Stefan Löhle	
Released		Date
(Doc. Management)	-	

Document change record

Issue	Date	DCN No./Change Description	Pages Affected
1.0	12/10/2020	Initiation of document	All

Contents

1. Setting up the NISI Code

This handbook covers the application of the NISI code written by the HEFDiG group at IRS. This includes the necessary setup (folder structure, data format, variables) for basic execution of the program.
In this handbook a color scheme represents different inputs. Green colored text represents a variable in Matlab. The variable value is written in red. Folders are highlighted in blue. Lastly, files are marked in orange. It is advised to set up pfade.mat (see subsection 1.1) and then following the regression check and example (see section 2) for insight in the proper setup.

1.1. Folder Structure

The NISI code requires a certain folder structure on the used device. At first, the two folders for the data and the code must be defined. Therefore, the code will request the file pfade.mat. This file contains two paths, one to the Data folder and the other one to the nisi folder. The pfade.mat file must be filled with two variables as in the following example:

- pathDataFolder 'C:\Users\"username"\Data\'
- pathNisiFolder 'C:\Users\"username"\Matlab\nisi\'

Figure 1 depicts the basic folder tree scheme[1]. Note that the nisi folder and all its subfolders must be added to the active path of Matlab.
It is essential that all data files for evaluation are saved in the Data folder. The names of Dataset, CalFolder and MeasFolder can be chosen arbitrarily, but must be defined later on (see subsection 1.2 & subsection 1.3). The foldernames within the Data branch will serve as placeholders for this handbook. Dataset is a folder containing calibration and measurement subfolders. Within every Dataset folder, there must be at least one calibration folder.

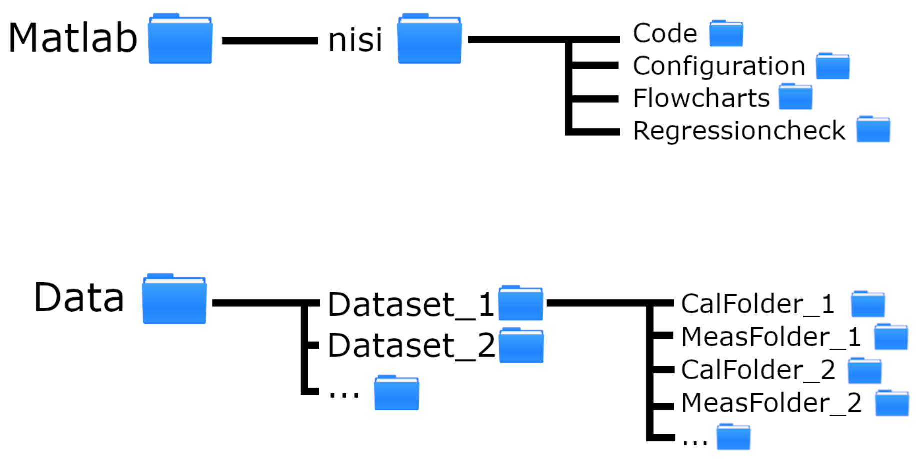


Fig. 1: Folder tree for NISI code

[1]Folder image by https://icon-library.net/icon/folder-icon-svg-4.html

1.2. NISI Input

This subsection describes all data files, which are mandatory as NISI input. This covers the file naming and the variables within those files. Within every CalFolder and MeasFolder a certain data structure and naming convention is necessary. The input data must have the folder structure from Figure 1 in its name. A calibration folder needs exactly one data file with the following name convention:

Dataset_CalFolder_C_NISI.mat

This file contains all calibration data. Note that the underscores are essential part of the name. The calibration file contains three variables:

- Calibration_Temperature(M,N,:) / Calibration_Pressure(M,N,:)
- Calibration_Heat_Flux(M,N,:)
- Time(1,:)

By definition, the Calibration_Temperature and Calibration_Heat_Flux must be 3-dimensional. Figure 2 shows the temperature data matrix format for N sensors and M surfaces. The row specifies the surface and the column the sensor. Superscripts stand for the sensor identifier, subscripts for the surface identifier. Finally, for every surface/sensor pair the data is stored inside the matrix pages (indicated by the :). For 1D NISI this simplifies to a (1,1,:) matrix. The heat flux data matrix format is slightly different. Figure 3 depicts the heat flux data matrix format. The heat flux must be the same for every sensor, since there can only be one heat flux per surface.

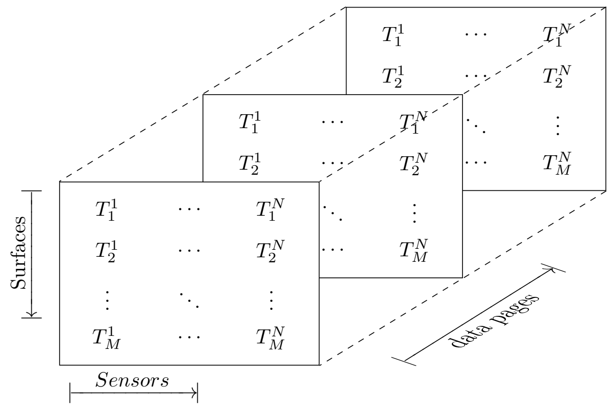


Fig. 2: Temperature data matrix format

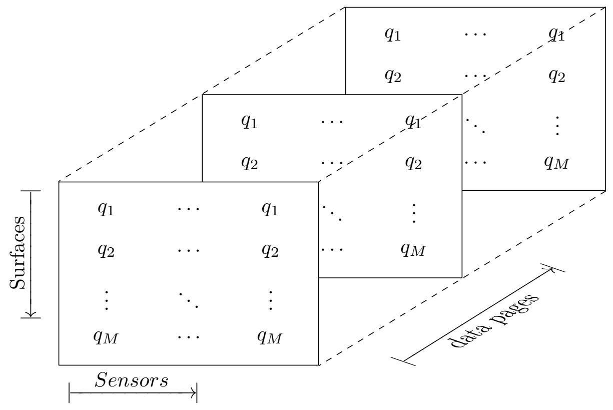


Fig. 3: Heat flux data matrix format

The Time must be an array of the same data size.

After the first NISI calibration run a list of found parameters, the impulse response and the solution will be saved as seperate files to the CalFolder folder. A measurement run can only be performed after a calibration run.

The name convention for a measurement file is as follows:

Dataset_MeasFolder_M_NISI.mat

The measurement file also contains three variables with naming:

- Msrmnt_Temperature(N,:) /Msrmnt_Pressure(N,:)
- Input_Heat_Flux(M,:)
- Time(1,:)

The variable Msrmnt_Temperature can be a data array for 1D NISI or a matrix for 3D NISI. The data structure is simpler compared to the calibration case, since the measurement can only account for the sensors. The Input_Heat_Flux(:,1) variable is optional.

1.3. Config - File

The conig-file contains all variables required by the NISI code. Every config-file must be saved to the Configuration folder within the nisi folder structure. Within this folder is a template called 00_Default_Config.m. The name of every config-file must correspond to the Dataset folder it belongs to.

The function name of a config file must always be of the form of Dataset_Config(). The file Dataset_Config.m must be saved as this compound name to the Configuration folder. Now the Dataset is defined and the following path definition covers the calibration and measurement folders. The variable
nameCalFolder has as value 'CalFolder', which is the calibration folder name as string. The final path definition variable nameMeasFolder has the value 'MeasFolder' as string.

1.3.1. Program Control

The program control panel specifies the requested calculations. Table 1 shows the available options. All settings use logical values. Setting all of the options in Table 1 to 0 or false will only execute the base program with an trial inversion of the calibration data.

Frequency_Analysis activates a fourier transformation on the data and plots them.

The option analyzeIR is a tool to perform a swift system identification and get a grasp on the resulting impulse response for a calibration run. When this option is active, the inversion step at the end of a NISI run is omitted, which reduces the calculation time significantly.

flagNL, currentTemp, pastTemps, chemieIdentifikation and chemieSolver are not implemented in this version of the code.

Tab. 1: Program Control

Name	Explanation	Settings
Frequency_Analysis	Acitivates Frequency Domain Visualisation	0, 1
analyzeIR	Only perform system identification and impulse response calculation	0, 1
flagNL	Calculation of Non-linear System Impulse Response	0, 1
currentTemp	Calculation with Impulse Response of current Temperature	0, 1
pastTemps	Calculation with Impulse Responses of past Time steps	0, 1
chemieIdentifikation	Activates chemical terms in identification	0, 1
chemieSolver	Activates Chemical Solver	0, 1

1.3.2. Program Configuration

The program configuration initializes the core inputs for the NISI algorithm. Those core inputs are their possible settings.

The Pre_Calc_Calibration and the Pre_Calc_Measurement enable filtering of the data before the NISI system identification. For the first evaluation of experiment data, it is mandatory to set both of those to 1. If this is set to 0, the code will evaluate the already filtered ..._C_NISI_filtered.mat.

The Sensor array defines the sensor/surface pairing for the calculation. The default setting is [0;0] and it evaluates all possible pairings. This is only relevant for NISI-3D applications.

The Penetration_Time describes the time delay between an incoming heat flux to a signal at the measurement position. For temperature measurement the following estimation for a semi-infinite slab can be used [1]:

$$t_p = \left(\frac{x}{3.6\sqrt{\frac{k}{\rho c_p}}} \right)^2 \tag{1}$$

Contradictory to the analytical solution of the heat equation, it takes a finite time for a temperature rise to occur at a certain location. This is modelled by adding a time delay to the calculated impulse response.

The variable Future_Time_Window defines a time window used for the inversion process. This stabilizes the solution of the IHCP. The inversion algorithms need the Future_Time_Window to calculate the reconstructed heat flux [2].
Timestep_Reduction_C and Timestep_Reduction_M reduce the number of data points for the evaluation. The code will skip the number of data points defined in those variables.

Temperature_Zero_C and Temperature_Zero_M set the time in s before the experiment. This will set the baseline for the evaluation. Note that a constant temperature level is required during this time span.

Tab. 2: Program Configuration

Name	**Explanation**	**Settings**
Pre_Calc_Calibration	Perform input data calculations Filters and reduction etc	0, 1
Pre_Calc_Measurement	Perform input data calculations Filters and reduction etc	0, 1
Sensor	Define the sensor/surface pairing for calculation	[0;0] = all [n;m] = Sensor n/ Surface m
Penetration_Time	Thermal penetration time in s	single-precision >0
Future_Time_Window	Window for Future Time Steps in s	single-precision >0
Timestep_Reduction_C	Only uses every n-th timestep for calibration	integer
Timestep_Reduction_M	Only uses every n-th timestep for measurement	integer
Temperature_Zero_C	Time in s for baseline calibration temperature	single-precision >0
Temperature_Zero_M	Time in s for baseline measurement temperature	single-precision >0
Auto_Param_Finder	Activation of the automatic parameter finder	0, 1
Max_Parameter_n_o	Amount of Parameters checked for viability (numerator)	integer
Max_Parameter_d_o	Amount of Parameters checked for viability (denominator)	integer
Parameter_Config_n	Configuration of the used parameters of the transfer function (numerator)	1 - 4
Parameter_Config_d	Configuration of the used parameters of the transfer function (denominator)	1 - 4
Manual_n_o	Heat flux time derivative orders and constant	2-row matrix
Manual_d_o	Tempearturetime derivative orders and constant	2-row matrix

At the end of the program configuration the user can manually define heat flux- and temperature terms Manual_n_o and Manual_d_o respectively. Those are the flux- and temperature time derivatives and their respective constant used for the non-integer system identification (NISI)[3]mehr quellen. The NISI method is derived by laplace transformation of the heat equation. This results in a system dependent transfer function. Rewriting the transfer function to the time domain leads to the basic model

$$\sum_{n=M_0}^{M} \alpha_n D^{n/2} T(t) = \sum_{n=L_0}^{L} \beta_n D^{n/2} \Phi(t) \tag{2}$$

where $D^{n/2}$ are the non-integer time derivatives, T is the temperature, Φ is the heat flux, α_n the scaling factors for the temperature terms, β_n the scaling factor for the heat flux terms. Equation 2 restricts the order to integers and half-integers. Equation 3 depicts an example heat flux- and Equation 4 a temperature term setting. The first row defines the order of the derivatives, the second row the constants. All entries in the second row that are set to 0 will be calculated when running the NISI code. By default all entries in the second row are set to 0. One constant can be set to a certain value, e.g. 1, in order to prevent the code to calculate such small numbers that Matlab reaches the bottom part of its numbers regime. The time derivatives used range from -0.5 to 2 and only one constant is defined with the value 1.

Heat Flux Terms:

$$Manual\_n\_o = \begin{bmatrix} -0.5 & 0 & 0.5 & 1 & 1.5 & 2 \\ 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{3}$$

Temperature Terms:

$$Manual\_d\_o = \begin{bmatrix} -0.5 & 0 & 0.5 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{4}$$

The amount of terms used for temperature and flux can be of different size. These terms are the core of the NISI approach and will influence the result significantly. To get further insight to the parameter selection please refer to [3]alle quellen.

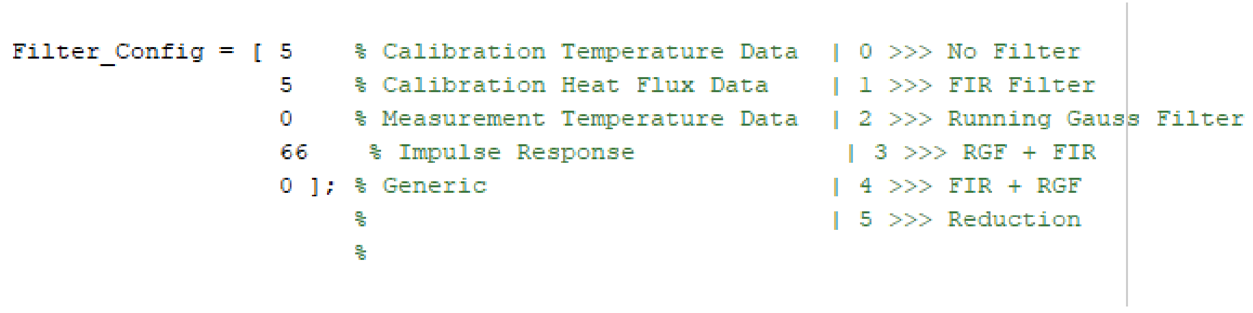
1.3.3. Filter Configuration

In this part of the Config-file the used filters are defined. The vector Filter_Config contains the chosen options. The available filter options are summarized in Table 3. Note that for all settings, except option 0, timestep reduction will be performed.

Tab. 3: List of Filter Options

Setting	Filter Option
0	No Filter
1	Finite-Impulse-Response Filter (FIR)
2	Running Gauss Filter (RGF)
3	First: RGF , Second: FIR
4	First: FIR , Second: RGF
5	Only Timestep Reduction
6	Interpolate (impulse response only [Nelder-mead simplex fit])
66	Interpolate (impulse response only [squared fit])
7	Sharp Edge Filter (for NISI pulses)

The Filter_Config is a column vector with 5 entries. Each entry defines the filters for a specific set of data. Figure 4 shows the vector in the config file and the row defines the setting for the data.



```
Filter_Config = [ 5    % Calibration Temperature Data  | 0 >>> No Filter
                  5    % Calibration Heat Flux Data    | 1 >>> FIR Filter
                  0    % Measurement Temperature Data  | 2 >>> Running Gauss Filter
                  66    % Impulse Response              | 3 >>> RGF + FIR
                  0 ]; % Generic                       | 4 >>> FIR + RGF
                       %                               | 5 >>> Reduction
                       %
```

Fig. 4: Filter config data arrangement

FIR
If a FIR filter is chosen, the filter conditions must be defined. Within the config file, two settings must be made. The vector FIR_Config defines the number of points taken for filtering. The vector FIR_Runs

sets the number of performed filtering runs. It is the number of times, the filter will be applied to the selected input data.

RGF
The vector Cut_Off_Frequency sets the frequency for the cut off applied to the specified input.

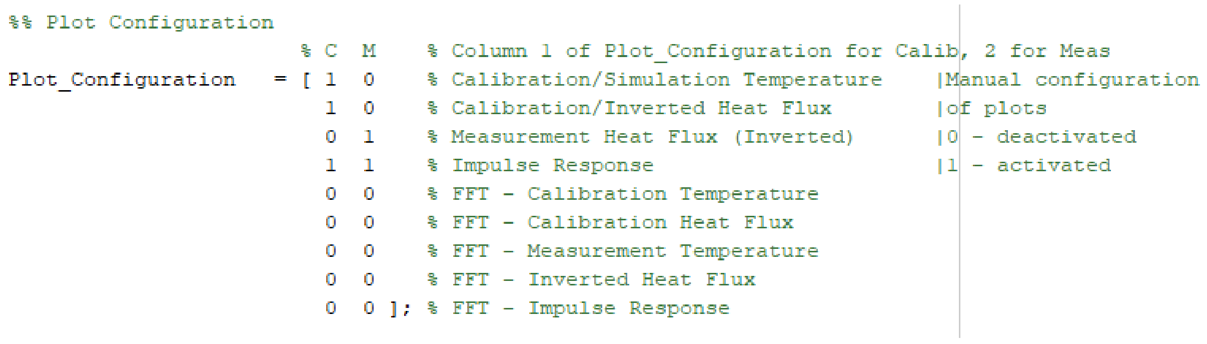
Interpolate
The interpolate functions for the impulse response can be used to eliminate singularities.

1.3.4. Plot Configuration

This part of the config-file defines the generated output plots. Figure 5 shows the default setting and all availablemplot options. The default setting covers the basic plots of interest for calibration and measurement evaluation.

For the calibration those default plots feature the calibration temperature and the simulated temperature after system identification, the generated impulse and the calibration reconstruction. This temperature plot is a visual indicator of a successful system identification process, since the calibration- and simulation temperature should overlap.
For a measruement run input data and the used impulse response are plotted by default.

By activating Frequency_Analysis (see Table 1) the frequency domain plots become available. Those must be activated in the Plot_Configuration seperately.



```
%% Plot Configuration
                         % C  M    % Column 1 of Plot_Configuration for Calib, 2 for Meas
Plot_Configuration   = [ 1  0    % Calibration/Simulation Temperature    |Manual configuration
                         1  0    % Calibration/Inverted Heat Flux        |of plots
                         0  1    % Measurement Heat Flux (Inverted)      |0 - deactivated
                         1  1    % Impulse Response                      |1 - activated
                         0  0    % FFT - Calibration Temperature
                         0  0    % FFT - Calibration Heat Flux
                         0  0    % FFT - Measurement Temperature
                         0  0    % FFT - Inverted Heat Flux
                         0  0 ]; % FFT - Impulse Response
```

Fig. 5: Plot configuration

1.3.5. Solver Configuration

The solver configuration sets the used inversion algorithm. Four algorithms are employed. All of the algorithms use an impulse response for calculation of the heat flux. The options are the sequential function estimation by Beck [2] and Van-Cittert [4].

Tab. 4: List of solvers

Setting	Solver
0	Phased Van Cittert
1	Sequential Function Estimation (constant, only 1 future time step)
3	Sequential Function Estimation (constant, all future time steps)
4	Sequential Function Estimation (linear, all future time steps)

1.4. Code Execution

After setting up the proper folder structure, creating the input file and defining the conig file the code can be executed. The Matlab function for a calibration run is:

```
NISI('Dataset','C')
```

The Dataset input will lead the code to the corresponding config file and the 'C' defines a calibration run. The code will then start automatically and a multi waitbar will pop up. After a successful execution the plot module will generate plots as defined in the config file. To perform a measurement run use the following Matlab function:

```
NISI('Dataset','M')
```

2. Regression Check

The regression check included features a 1D and a 3D calculation. Both consist of a calibration with a followed measurement run. The regression check is automatized in the Matlab script Regressioncheck.m. It can be found in the Regressioncheck subfolder in the nisi folder. By executing the script, a figure window for each individual run will appear. The created plots should look like the Figure A.1 to Figure A.4. After checking the resulting figure after each run, continue the regression check by pressing any key in the command window.

After the regression check the script will pop up a dialogue window. This window asks, if the user wants to clean up all by regression testing generated data on the local directory. ”Yes” will delete all generated folders, ”No” will leave them in the Datafolder directory. Use the option ”No”, to get detailed insight into the generated folder structure on the PC.

3. Example: Regression Check 1

The following example uses the regression check 1 to explain the NISI setup. Keep the generated data from Regressioncheck.m for comparison. Use the following code in the command window to execute the NISI code for a calibration regression check 1 once again:

NISI('regr1','C')

After the successful execution access the data folder defined in pfade.mat. The folder structure for the regression check 1 should look like this:

...\Data\regr1\Calibration

...\Data\regr1\Measurement

In accordance to Figure 1, the data folder is simply called Data. The dataset name is regr1. The subfolders Calibration and Measurement are CalFolder and MeasFolder respectively.
First check the data inside Calibration. Since the code was already executed for this setup, all input and output files exist. The mandatory input file here is: (check subsection 1.2)

regr1_Calibration_C_NISI.mat

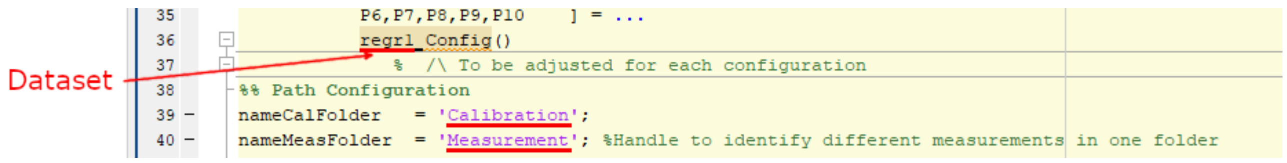
containing the variables:

Calibration_Temperature(1,1,1051)

Calibration_Heat_Flux(1,1,1051)

Time(1,1051)

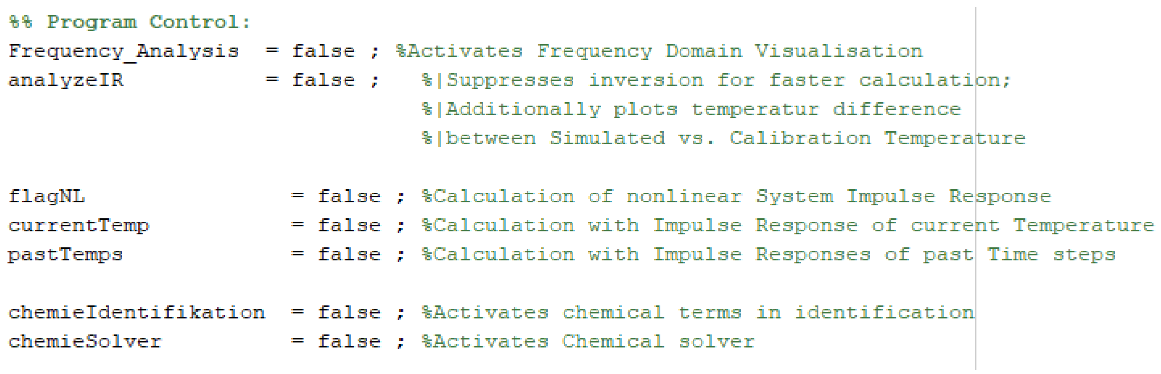
All other files are output of the code.
regr1_Calibration_C_Config_Copy.m is a copy of the used config file for this calculation and the following explanations will refer to this file. The original file (the code requires this file, not the copy!) is stored at the required ...\nisi\Configuration directory. At the start, the data path configuration must be given. Figure 6 shows the setup for nameDataset, nameCalFolder and nameMeasFolder.



```
            P6,P7,P8,P9,P10     ] = ...
            regr1_Config()
              %  /\ To be adjusted for each configuration
%% Path Configuration
nameCalFolder    = 'Calibration';
nameMeasFolder   = 'Measurement'; %Handle to identify different measurements in one folder
```

Fig. 6: Data path setup for regr1 (Regressioncheck 1)

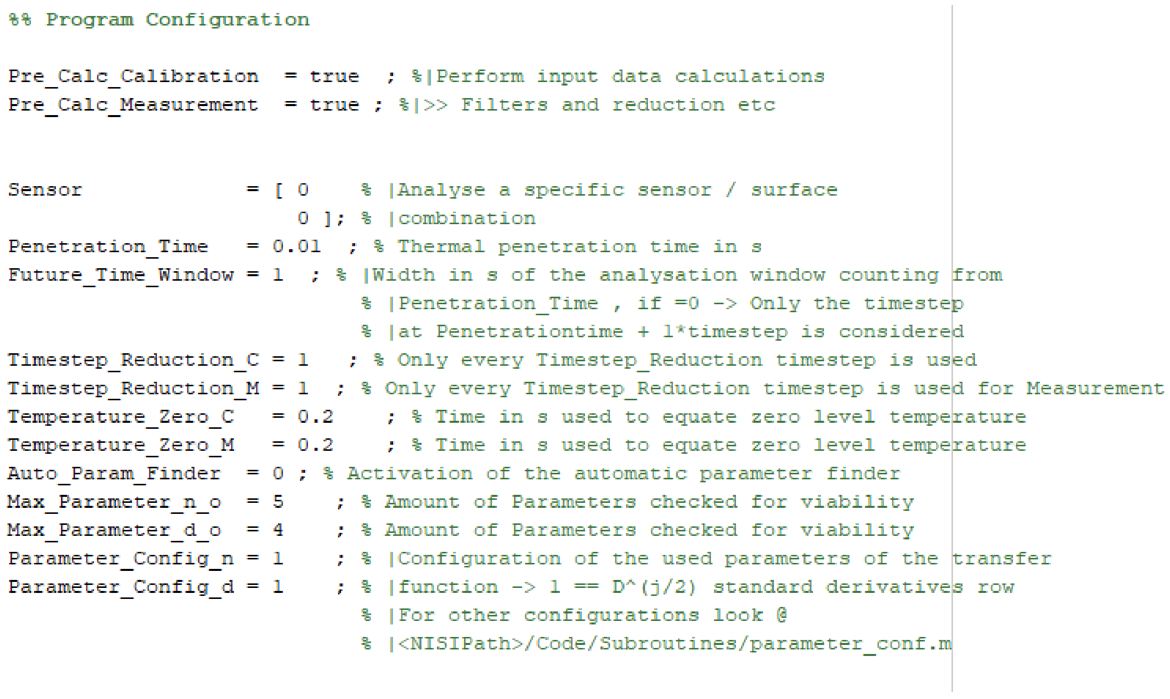
Figure 7 shows the variables defined in the program control section. For the regr1 test case, all variables are switched off. Figure 8 shows the program configuration variables. Since Pre_Calc_Calibration is set to true, the code will use the unfiltered data and perform filtering on them. The Sensor is set to account all surface/sensor pairs, which is one surface to one sensor in this setup. The Penetration_Time and Future_Time_Window are 0.01 s and 1 s respectively. Timestep_Reduction_C is 1, which means every timestep is considered in the calculation. The baseline is defined by Temper-



```
%% Program Control:
Frequency_Analysis  = false ; %Activates Frequency Domain Visualisation
analyzeIR           = false ;   %|Suppresses inversion for faster calculation;
                                %|Additionally plots temperatur difference
                                %|between Simulated vs. Calibration Temperature

flagNL                = false ; %Calculation of nonlinear System Impulse Response
currentTemp           = false ; %Calculation with Impulse Response of current Temperature
pastTemps             = false ; %Calculation with Impulse Responses of past Time steps

chemieIdentifikation  = false ; %Activates chemical terms in identification
chemieSolver          = false ; %Activates Chemical solver
```

Fig. 7: Program control setup for regr1

ature_Zero_C. In this example it is set to 0.2 s. This means, the first 0.2 s of the temperature data are averaged and will be subtracted from all data points. Lastly, the Auto_Param_Finder is switched off to 0. This means the parameters for the NISI approach have to be defined manually in the following part of the config file. The last four variables are not considered in this calculation, since Auto_Param_Finder is switched off.

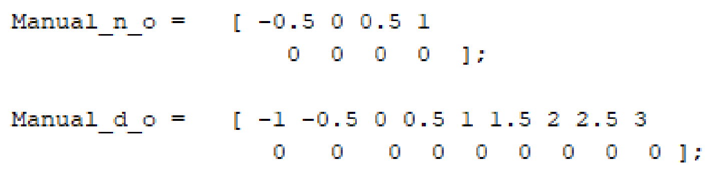


```
%% Program Configuration

Pre_Calc_Calibration  = true  ; %|Perform input data calculations
Pre_Calc_Measurement  = true ; %|>> Filters and reduction etc


Sensor              = [ 0    % |Analyse a specific sensor / surface
                        0 ]; % |combination
Penetration_Time    = 0.01  ; % Thermal penetration time in s
Future_Time_Window = 1  ; % |Width in s of the analysation window counting from
                             % |Penetration_Time , if =0 -> Only the timestep
                             % |at Penetrationtime + 1*timestep is considered
Timestep_Reduction_C = 1   ; % Only every Timestep_Reduction timestep is used
Timestep_Reduction_M = 1  ; % Only every Timestep_Reduction timestep is used for Measurement
Temperature_Zero_C   = 0.2    ; % Time in s used to equate zero level temperature
Temperature_Zero_M   = 0.2    ; % Time in s used to equate zero level temperature
Auto_Param_Finder  = 0 ; % Activation of the automatic parameter finder
Max_Parameter_n_o  = 5    ; % Amount of Parameters checked for viability
Max_Parameter_d_o  = 4    ; % Amount of Parameters checked for viability
Parameter_Config_n = 1    ; % |Configuration of the used parameters of the transfer
Parameter_Config_d = 1    ; % |function -> 1 == D^(j/2) standard derivatives row
                             % |For other configurations look @
                             % |<NISIPath>/Code/Subroutines/parameter_conf.m
```

Fig. 8: Program configuration setup for regr1

Figure 9 shows the parameter/derivative selection used for the calculation. The arrays can be of variable length, only the number of parameters and derivatives must match. Check Equation 2 to get further insight into the correct parameter/derivative setup.

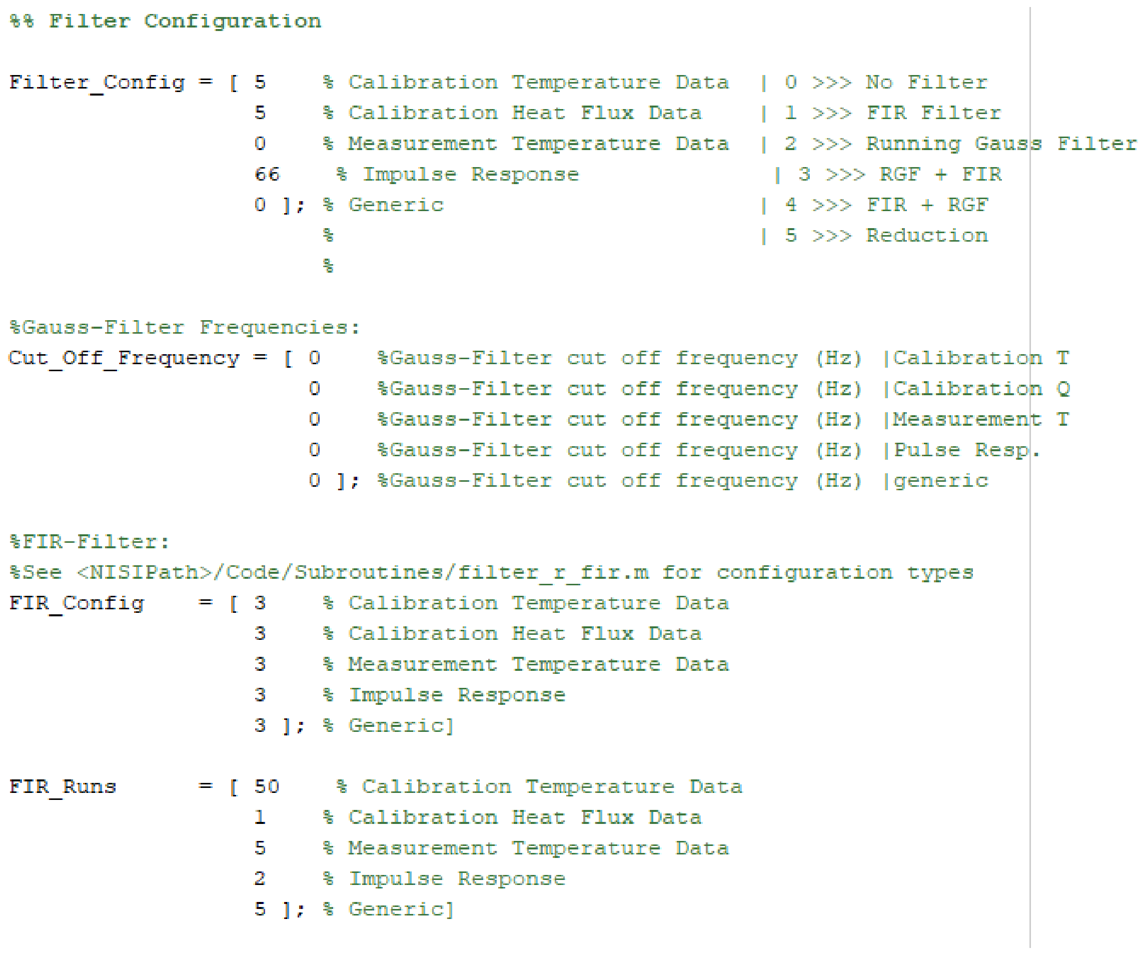
Figure 10 depicts the filter options used for the regression check 1. The Filter_Config defines the employed filters. The calibration temperature and heat flux data are only timestep reduced. No other



```
Manual_n_o =   [ -0.5 0 0.5 1
                   0  0  0  0  ];

Manual_d_o =   [ -1 -0.5 0 0.5 1 1.5 2 2.5 3
                  0   0   0  0  0  0  0  0  0 ];
```

Fig. 9: Parameter/derivative setup for regr1

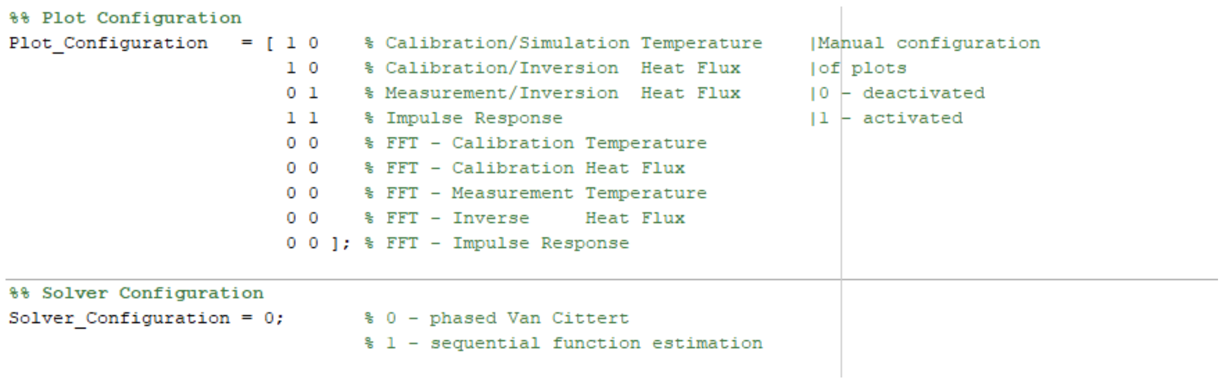
filters are applied to them. Since no measurement data are available in a calibration run, the filter can be switched off. The impulse response is handled differently than temperature and heat flux data. Instead of filtering, an interpolation module using a quadratic fit at the first time steps is employed. All vales of Cut_Off_Frequency are switched to 0, since no RGF is used. Although values for FIR_Config and FIR_Runs are present, they will not be taken into account. If they were active, the FIR filter always use 3 data points for the number of runs defined.



```
%% Filter Configuration

Filter_Config = [ 5    % Calibration Temperature Data   | 0 >>> No Filter
                  5    % Calibration Heat Flux Data     | 1 >>> FIR Filter
                  0    % Measurement Temperature Data   | 2 >>> Running Gauss Filter
                  66    % Impulse Response               | 3 >>> RGF + FIR
                  0 ]; % Generic                        | 4 >>> FIR + RGF
                       %                                | 5 >>> Reduction
                       %

%Gauss-Filter Frequencies:
Cut_Off_Frequency = [ 0    %Gauss-Filter cut off frequency (Hz) |Calibration T
                      0    %Gauss-Filter cut off frequency (Hz) |Calibration Q
                      0    %Gauss-Filter cut off frequency (Hz) |Measurement T
                      0    %Gauss-Filter cut off frequency (Hz) |Pulse Resp.
                      0 ]; %Gauss-Filter cut off frequency (Hz) |generic

%FIR-Filter:
%See <NISIPath>/Code/Subroutines/filter_r_fir.m for configuration types
FIR_Config     = [ 3    % Calibration Temperature Data
                   3    % Calibration Heat Flux Data
                   3    % Measurement Temperature Data
                   3    % Impulse Response
                   3 ]; % Generic]

FIR_Runs       = [ 50    % Calibration Temperature Data
                   1    % Calibration Heat Flux Data
                   5    % Measurement Temperature Data
                   2    % Impulse Response
                   5 ]; % Generic]
```

Fig. 10: Filter setup for regr1 (Regressioncheck 1)

Figure 11 is the last section of the example config file and shows the plot and solver setup. The default plot configuration is activated here.
The used solver is the phased Van Cittert.



```
%% Plot Configuration
Plot_Configuration   = [ 1 0    % Calibration/Simulation Temperature    |Manual configuration
                         1 0    % Calibration/Inversion  Heat Flux      |of plots
                         0 1    % Measurement/Inversion  Heat Flux      |0 - deactivated
                         1 1    % Impulse Response                      |1 - activated
                         0 0    % FFT - Calibration Temperature
                         0 0    % FFT - Calibration Heat Flux
                         0 0    % FFT - Measurement Temperature
                         0 0    % FFT - Inverse     Heat Flux
                         0 0 ]; % FFT - Impulse Response

%% Solver Configuration
Solver_Configuration = 0;       % 0 - phased Van Cittert
                                % 1 - sequential function estimation
```

Fig. 11: Plot and solver setup for regr1 (Regressioncheck 1)

4. Quick Start

- Define pfade.mat (First time only)

  -pathDataFolder C: \. . . \Data \

  -pathNisiFolder C: \. . . \nisi \

- Create dataset folder
- Create calibration folder within dataset folder
- Save Dataset_CalFolder_C_NISI.mat at calibration folder (filename must match dataset and calibration folder name)

  -Calibration_Temperature(1,1,:) or Calibration_Pressure(1,1,:)

  -Calibration_Heat_Flux(1,1,:)

  -Time(1,:)

- Create measurement folder within dataset folder (filename must match dataset and measurement folder name)

  -Msrmnt_Temperature(1,:) or Msrmnt_Pressure(1,:)

  -Input_Heat_Flux(1,:) (if available)

  -Time(1,:)

- Costumize Config file

  -Open template 00_Default_Config.m

  -Replace 00_Default of the file and the function header with the dataset name

  -Add name of calibration folder at variable nameCalFolder

  -Add name of measurement folder at variable nameMeasFolder

  -Adjust Flux- and Temperature Terms as necessary

- Execute 'NISI('Dataset','C') or 'NISI('Dataset','M') at Matlab terminal for calibration and measurement respectively

5. Bibliography

[1] Kaviany, M., *Principles of Heat Transfer*, John Wiley and Sons, 2001.

[2] Beck, J., Blackwell, B., and St. Clair, C. R., *Inverse Heat Conduction: Ill-posed Problems*, Wiley, 1985.

[3] Loehle, S. and Frankel, J., "Physical Insight into NISI Parameters," *50th Aerospace Science Meeting and Exhibit*, AIAA, 2012.

[4] van Cittert, P. H., "Zum Einfluss der Spaltbreite auf die Intensitätsverteilung in Spektrallinien," *Zeitschrift für Physik*, Vol. 65, No. 547, 1931.

A. Appendix

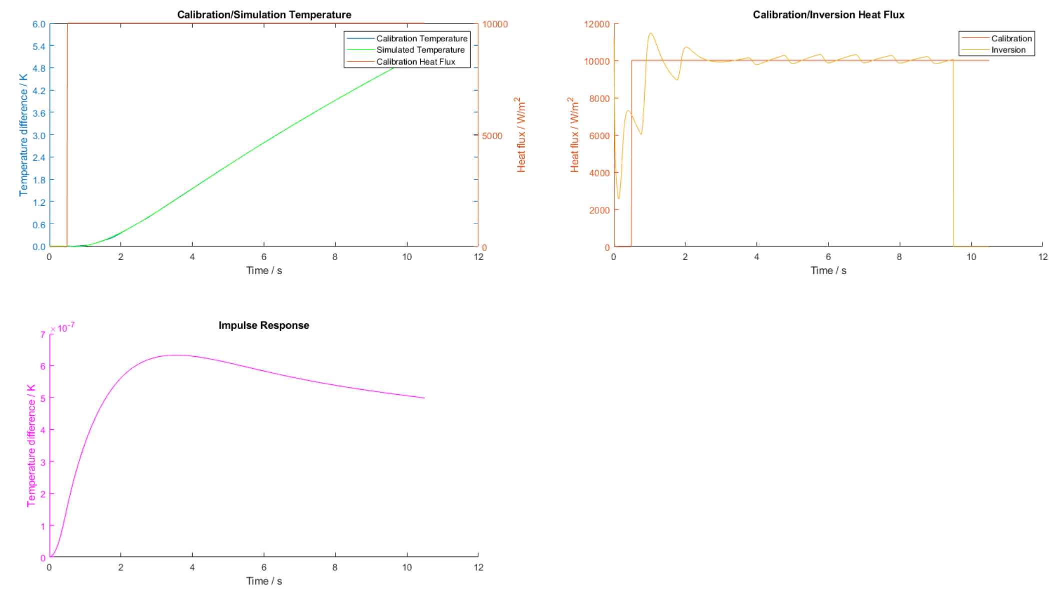


Fig. A.1: 1D Calibration Setup regression test

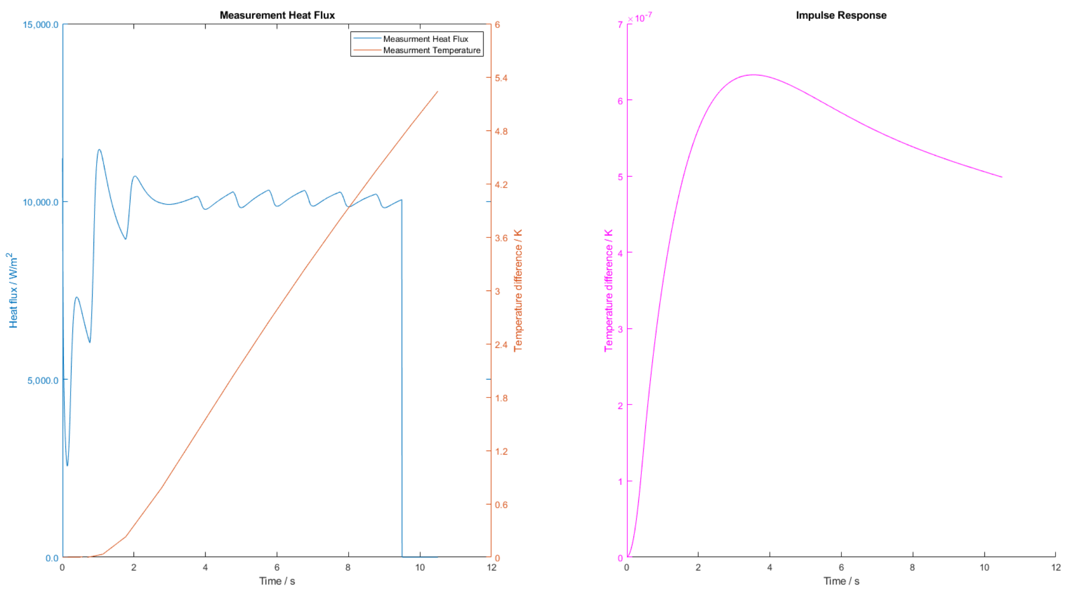


Fig. A.2: 1D Measurement Setup regression test

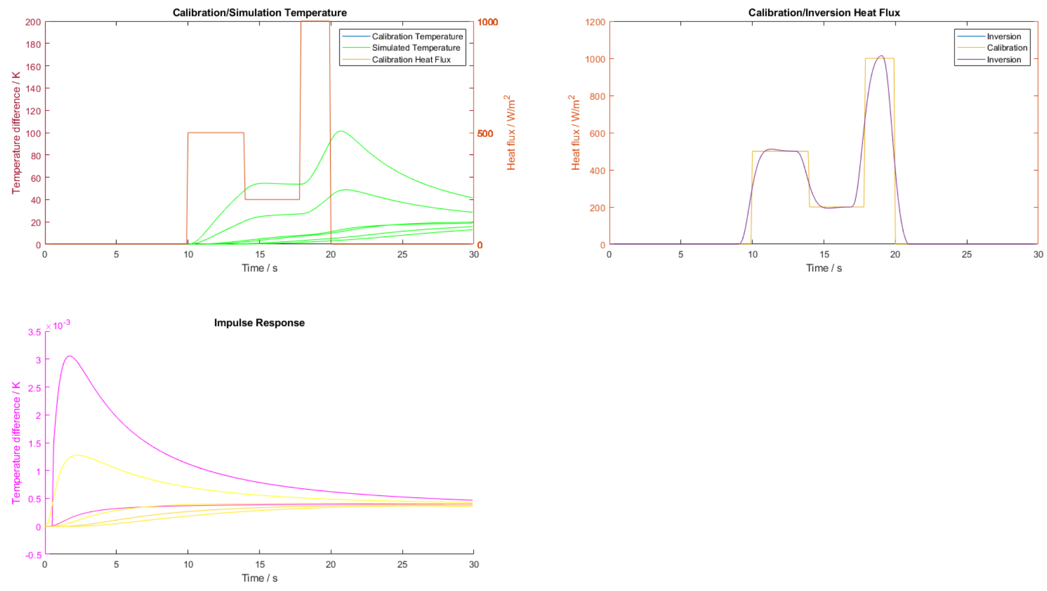


Fig. A.3: 3D Calibration Setup regression test

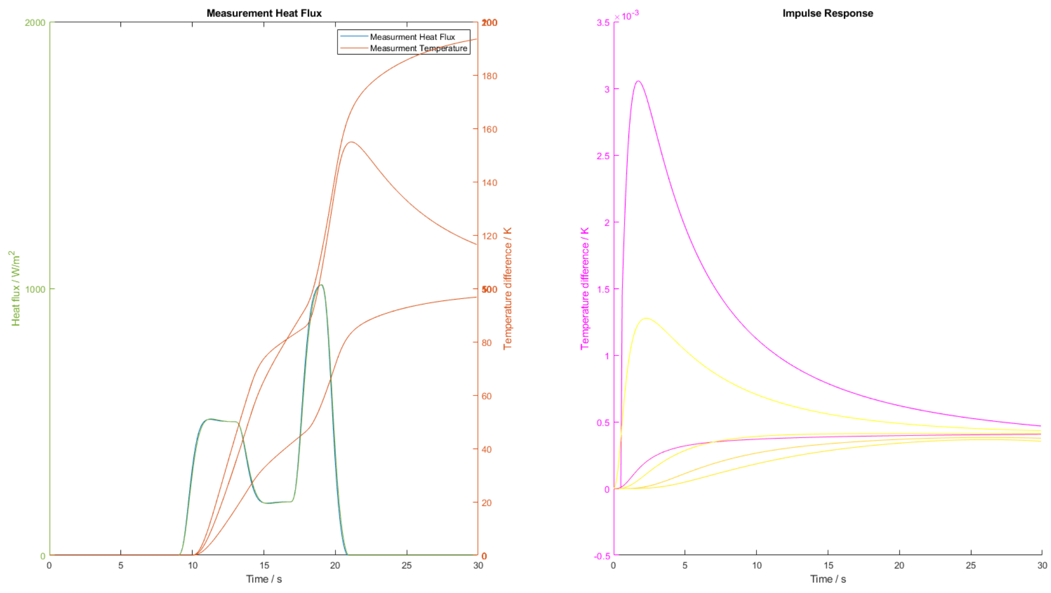


Fig. A.4: 3D Measurement Setup regression test