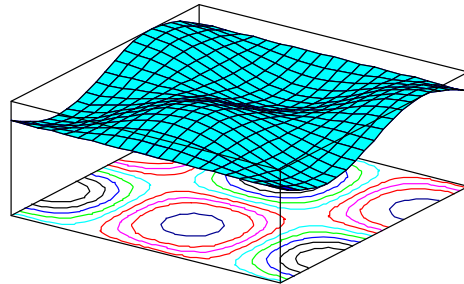


## FEATURES

- Implement your own ideas
- Write flexible, powerful scripts
- Exploit MatLab / Scilab
- Hide math in a container
- Create tools usable for everyone
- Extend dB-Lab functionality
- Share tools with coworkers



The MAT module is a programmable tool for realizing any kind of mathematical processing such as simulations, statistics or graphical display of data. In addition to a basic tool box coming with the KLIPPEL R&D System a user may write his own powerful applications using MatLab or SciLab. Processing is encapsulated in a Module, so coworkers not familiar with the high-level language may operate the MAT module using the common interface of dB-Lab. Thus the MAT module is an ideal basis for creating new tools which may be shared within the working team.

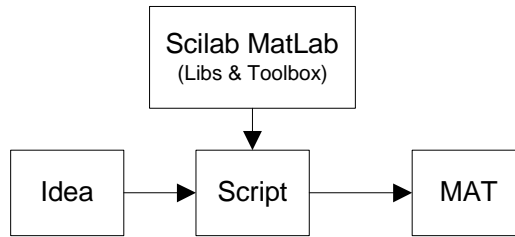
Article Number: 1001-100

## CONTENTS:

Programming mode .....	2
User mode .....	3
Input.....	4
Results.....	4
Examples.....	5

# Programming mode

**How it works**



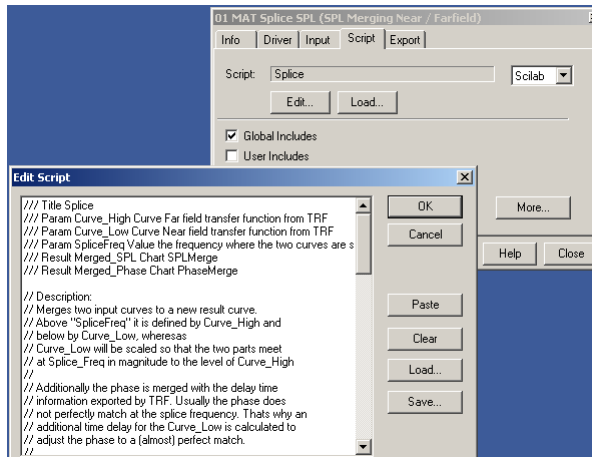
In the programming mode the developer creates his own MAT application. He implements his ideas in a script using a high level math language (MatLab or Scilab). Each script is divided into parts for

1. defining the input parameters
2. implementing the actual problem in a script editor
3. defining the results and their representation

The implemented user interface (step 1 and 3) provides a smooth integration into the Analyzer. This way anyone using the implemented solution does not need to know anything about implementation details.

**Script editor**

An internal editor for writing and modifying the script is provided. It is also possible to use an external editor and load the final script into the MAT module.

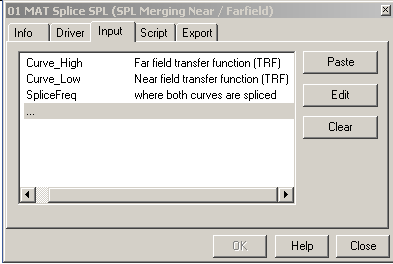


**Language**

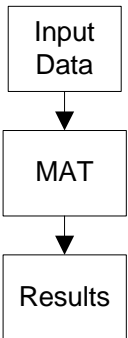
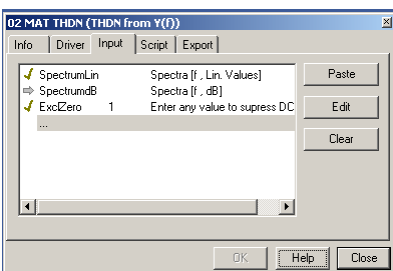
The Scripts can be written using MatLab or SciLab. MatLab is a commercially available product, SciLab is a free software package and has a similar syntax and functionality as MatLab. The user may select a language according to availability and personal preference.

**Libraries**

Predefined libraries and tool boxes coming with MatLab or SciLab can be used. It is also possible to create new libraries for often used functions and to share them within the team e.g. over the network.

<p><b>Creating User-interface</b></p>	<p><i>Defining input:</i></p> <p>The input parameters are defined with keywords to control the <i>input</i> property page. A comment can be added to inform the user, what data should be imported. Data import is done via the clipboard or an internal editor, allowing to import curves, setups or any external data. Scalar numbers can be entered directly.</p> <p><i>Defining output:</i></p> <p>Results (such as tables and curves) are defined and assigned to result windows by keywords, and are displayed automatically after the script finishes. Window and Curves (labels, colors, range etc.) can be configured freely in the script. Tables can be created using plain text. HTML syntax gives even more freedom for individual formatting of the output.</p>	
<p><b>Finishing</b></p>	<p>Helper functions make it simple to validate input parameters and display warnings and error messages to the user.</p> <p>The programmer should give a short description in the property page <i>INFO</i> about the usage of the application.</p> <p>The last step is to save the script as an operation template, so it can be used by any other user similar to an original module of the R&amp;D System. An appropriate name should be given to identify the operation intuitively.</p>	

## User mode

<p><b>How it works</b></p>	<p>The user has to perform the following steps:</p> <ul style="list-style-type: none"> <li>• Enter the input data</li> <li>• Start the operation</li> </ul> <p>MatLab or Scilab will process the script in the background. After that, the results are displayed in the respective result windows as specified by the programmer. The script may also contain further user interaction, such as basic input dialogs.</p> <p>The user does not need to know any details of the specific implementation. Using the comments on the <i>Info</i> and <i>Input</i> property page he is able to perform the operation.</p>	
<p><b>Input</b></p>	<p>The user selects the <i>Input</i> property page and enters or import the data required. Short comments for each parameter help identify the source and meaning of the input data. Each data set entered is marked with a green tick. Inputs which are not satisfied are marked by a gray arrow indication.</p>	
<p><b>Running the operation</b></p>	<p>Dialogues and messages in separate windows may appear and request user interaction depending on the application</p>	
<p><b>Output</b></p>	<p>The results are provided in up to 5 graphical result windows as well as one HTML result window. The settings defined in the script can be customized by the user, the same way as for all other Modules.</p>	

<b>Integration into R&amp;D system</b>	<p>All MAT applications are handled in the same way as any other operation (such as LSI, LPM). This allows a smooth integration into the project management within dB-Lab. Operations can be grouped in objects to keep measurements and MAT applications together.</p> <p>Furthermore the report system allows a flexible representation of all results embedded in a user defined HTML report document. Templates for each application can be created either by programmer or user.</p>
--	---

## Input

Input Curves	Data in a matrix form of any order
Input Variables	Single values
Script	High level math code containing the algorithm
Language	SciLab or MatLab compatible scripts
Libraries	Local and global locations for storing often used routines and functions own or toolbox

## Results

### Result Windows

Input Curves	Graphical representation of input data (first two columns only)
Input Variables	List of all input variables
Result Curve 1..5	Graphical representation of one or more curves. Standard formatting functionality from dB-Lab for modification and user defined settings are provided.
Result variables	Numbers and parameters can be output in tables or as text. The output can be formatted as plain text format or HTML.
Log	Output log file from Scilab or MatLab

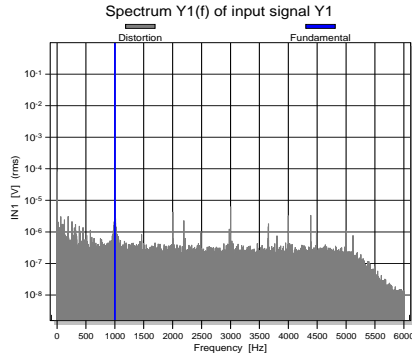
### Other

Export	All results are prepared in a list to be copied to the clipboard.
--------	---

# Examples

Several simple examples may demonstrate the capabilities of the MAT module.

## THDN of a spectrum



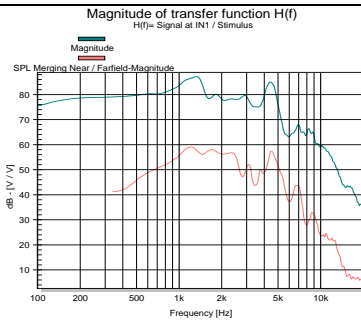
Name	Value	Description
THDN	89.568529 dB	Total Harmonic Distortion + Noise
f(signal)	1.00049 kHz	Signal Frequency
Lin. Signal	.998885	Amplitude of Signal
Signal in dB	-.0096902 dB	Amplitude of Signal in dB

Result table (HTML format)

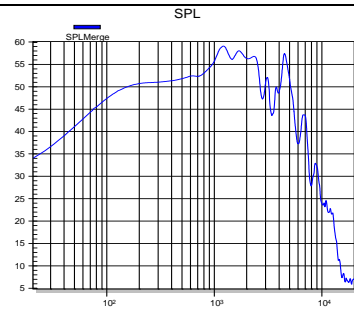
Input data (sinusoidal tone spectrum)

A spectrum in dB or linearly scaled is imported to the MAT module. The THDN as well as the frequency of the highest signal component is calculated. The result is displayed in an HTML output window.

## Splicing SPL (Near & Farfield)



Near and far-field response

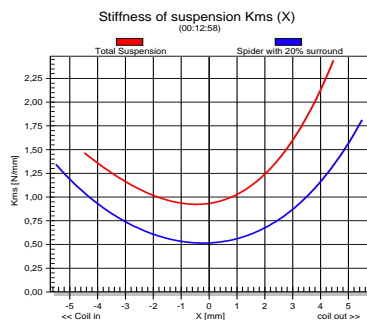


Spliced frequency response

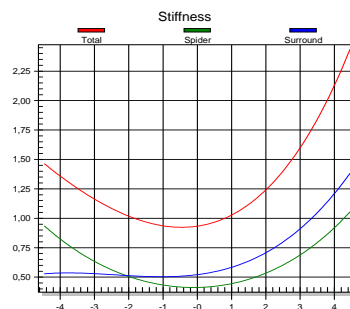
SPL transfer function measurement in smaller rooms often require merging of a near and far field measurement to get low frequency response and to suppress room influence. The script matches the magnitude and phase at the required splicing frequency, which is an input parameter.

Algorithm: Both transfer functions are imported as complex functions (3 columns: Frequency, Magnitude, Phase) from the TRF module. They are spliced that way, that the magnitude of the near field response matches the magnitude of the Far field response at the splicing frequency. Together with the complex transfer function the delay time is imported from TRF to splice the phase response exactly. This is an example for importing a matrix with more than 2 rows.

## Separating spider and surround



Stiffness measured by LSI



Stiffness of separated components using MAT

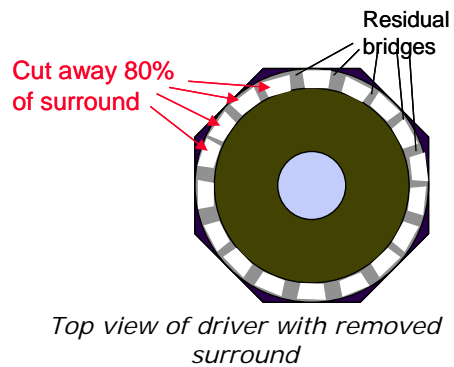
### 1. Measurement

The stiffness  $K_{ms}(x)$  of a loudspeaker driver is measured by the LSI module. To isolate the contribution from spider and surround, the surround is partially removed and the stiffness of the remaining part of suspension is determined by a second LSI measurement.

### 2. Postprocessing

After importing the results of both LSI measurements to the MAT module, the stiffness of each component (surround and spider) is calculated.

Algorithm: A weighted addition allows to subtract both stiffness curves from each other considering the error from the remaining part of the surround in the second measurement. This is an example for multiple curves in one output window with appropriate setting of the graphical properties.



Find explanations for symbols at <http://www.klippel.de/know-how/literature.html>

updated August 13, 2012



Klippel GmbH  
Mendelssohnallee 30  
01309 Dresden, Germany

[www.klippel.de](http://www.klippel.de)  
[info@klippel.de](mailto:info@klippel.de)

TEL: +49-351-251 35 35  
FAX: +49-351-251 34 31