ABOUT THE INTEGRATION OF SIMULINK INTO THE MATLAB-BASED SIMULATION AND EXPERIMENT SERVER MMT

A. Körner[(a)], I. Hafner[(b)] , M. Bicher[(c)], S. Winkler[(d)], U. Fitsch[(e)]

Vienna University of Technology
Institute for Analysis and Scientific Computing
Wiedner Hauptstraße 8-10
1040 Vienna

[(a)] andreas.koerner@tuwien.ac.at, [(b)] irene.hafner@tuwien.ac.at [(c)] martin.bicher@tuwien.ac.at
[(d)] stefanie.winkler@tuwien.ac.at [(e)] ursula.fitsch@tuwien.ac.at

ABSTRACT

At the Vienna University of Technology, Institute for Analysis and Scientific Computing, the Research Group for Mathematical Modelling uses the MMT-Server for teaching mathematical modelling.
MMT stands for Mathematics, Modelling and Tools. It is based on and a PHP interface, which is responsible for input and output of parameters and results. Most of the currently existing examples on the MMT systems are simulated with MATLAB.
As MATLAB is used for interactive experiments on this server, it's nearby to integrate Simulink as a second simulator in order to have the possibility to compare different modelling approaches. Simulink differs from MATLAB due to its graphical description of models by signal flow graphs.
Simulink works with linear and non-linear time-continuous and time-discrete systems. Therefore, specifications of Laplace- and z-transformations from linear system theory as well as solvers for ordinary differential equations of non-linear systems are available.

Keywords: Simulink, MATLAB, E-learning, Simulation, Mathematical Modelling.

1. BUILD-UP OF THE MMT-SERVER

The existing user interface of the current MMT-Server is illustrated in Figure 1. It is working with examples implemented in MATLAB.

On the left side, there is the Experiment Layer, which organises the structure of the experiments. A higher level denotes a subject out of mathematical modelling and a lower level represents experiments in different scenarios.
In the centre of the user interface, which is the main window, a precise briefing of the actual experiment is given.

The Experiment Description window below shows which parameter should be changed and gives an impression of the expected results.
The right side consists of Files and Source Code and is the important one regarding the experiment's implementation in MATLAB, because the links to the m-files are available there. Users can download these files to learn more about how to program in MATLAB and they can also tune and change the code to experiment with other scenarios.

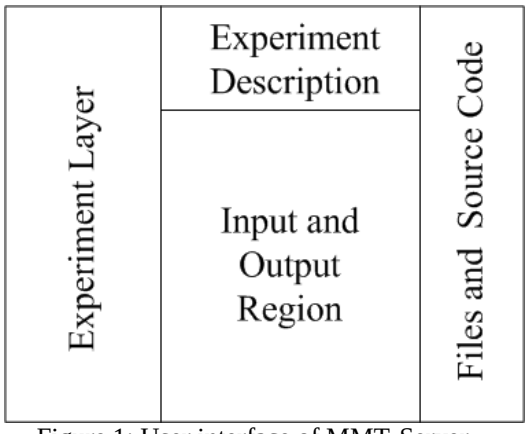

Figure 1: User interface of MMT-Server

The input and output region offers the possibility to enter model and/or simulation parameter. These parameters are available as a string in the MATLAB code, stored in the `instruct.varN` variable. N denotes the number of the variable which is assigned from the PHP-based input field in the input region. The instruct variable is transformed into a MATLAB variable

```
K = str2double(instruct.var1).
```

After this conversion, the MATLAB specific code follows.
At the end of the MATLAB code, a section for graphical output is offered. Ordinary plots are handled by the system and are displayed in the output region.

The following example illustrates the visual nature of an example or experiment on the MMT-server.
Let's observe a mathematical pendulum, described by the ordinary differential equation

$$m \cdot l \cdot \varphi''(t) = -m \cdot g \cdot \sin(\varphi(t)) - l \cdot d \cdot \varphi'(t).$$

l denotes the length of the pendulum, d the damping coefficient and φ is considered as $\varphi(0) \in \left(-\frac{\pi}{2}, \frac{\pi}{2}\right)$ and the beginning velocity $\dot{\varphi}(0)$ can be chosen. The corresponding look of the example on the MMT-server is shown in Figure 2.

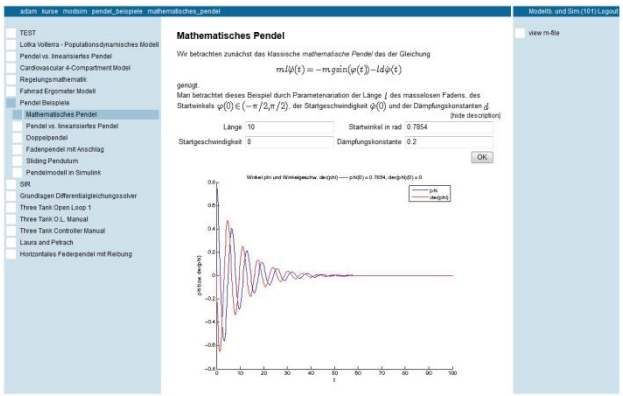

Figure 2: Front-end of MMT-server example of the mathematical pendulum

This structure is important for the integration of Simulink, because Simulink examples on the MMT-server work in the same way focused on the front-end.

2. SIMULINK AT THE MMT-SERVER

To allow comparisons between two or more different simulators, it is necessary to implement more than only MATLAB on the MMT-server. The fact that Simulink is MATLAB-based allows including Simulink on the MMT-server with minimal effort.

2.1. Mathematical description of Simulink Models

A signal flow graph is defined by an oriented graph $G_w = (V, E, w)$, where V is the set of nodes, E is the set of edges and w is a mapping

$$w: E \to \mathbb{R}_0^+,$$

which relates each edge with a real positive number, the so-called weight of this edge. Furthermore μ defines a mapping

$$\mu: V \to L^2(\mathbb{R}_0^+),$$

which maps each node to a signal of the signal space $L^2(\mathbb{R}_0^+)$. The mapping w operates along the edge $e_j = \langle v_j, v_l \rangle \in E$ according to

$$x_l(t) = w_{e_j}(x_j(t)).$$

The triple $(G_w, L^2(\mathbb{R}_0^+), \mu)$ is called a (generalised) signal flow graph.

In case of more than one connected edge to one node v_l, that means for $r \in \mathbb{N}$ with the edges

$$e_j = \langle v_j, v_l \rangle, e_{j+1} = \langle v_{j+1}, v_l \rangle, \dots, e_{j+r} = \langle v_{j+r}, v_l \rangle,$$

the mapping w operates according to

$$x_l(t) = \sum_{s=0}^{r} w_{e_{j+s}}(x_{j+s}(t)).$$

In this definition, $L^2(\mathbb{R}_0^+)$ is used. This is the vector space of quadratic integrable functions.
Because of the fact, that a Simulink model will be implemented on a computer, a closer look to the signals out of $L^2(\mathbb{R}_0^+)$ is appropriate.

Let's assume $I = [a, b] \subset \mathbb{R}_0^+$ and denote

$$\mathcal{Z} = \{t_1, t_2, \dots, t_k\}, \quad k \in \mathbb{N},$$

a decomposition of the interval I with $t_1 = a$ and $t_k = b$. fulfils the condition $t_{i+1} - t_i = \Delta t$ for all $i = 1, \dots, k$, is called equidistant. Through

$$x(t_i) = x_i$$

a mapping $\delta_a: L^2(I) \to \mathbb{R}$ is defined, the sampling function. By this sampling δ_a, a set of values

$$X = \{x_1 = x(t_i), \dots, x_k = x(t_k)\}$$

is defined. In computers only this set of sampled values are available.

This definition allows a mathematical characterisation of Simulink models as signal flow graphs. Simulink offers a graphical interface to build these signal flow graphs in the model editor.

2.2. Some basic signal flow graphs

To emphasise the mathematical characterisation of Simulink models, some basic examples are presented in this subsection.
The multiplication of one signal is represented by the signal flow graph in Figure 3. There, the traditional and an alternative description are depicted.

Figure 3: Signal flow representation of a multiplication in classical and alternative form

In Figure 3, the alternative form of the signal flow representation reminds of the block structure of Simulink. The corresponding mapping is given by

$$x_l(t) = a \cdot x_j(t).$$

For the addition of two signals in a signal flow graph, in case of more than one edge, the mapping w can be adapted.

The weighting is set to be the identity and in case of two signals, the corresponding mapping is given by

$$x_l(t) = x_j(t) + x_{j+1}(t).$$

An even more complex operation is the integration of a signal. In this setting, the construction of the weighting w is visible. It is necessary, that the weighting is a general mapping between the signals without structural limitations. This makes it possible to describe a multiplication as well as an integration. The integration is defined by

$$x_l(t) = \int_0^t x_j(t)\,\mathrm{d}t.$$

After introducing the integration operator into the signal flow concept, a higher operation – the convolution – can be analysed. This operation is very important for linear time invariant systems. Thereby, the description in the time domain is done with the convolution. The convolution is defined on the set of integrable functions on $\Omega \subseteq \mathbb{R}_0^+$, denoted by

$$L^1(\Omega) = \left\{ f: \Omega \to \mathbb{R}: \int_\Omega |f(x)|\mathrm{d}x < \infty \right\}.$$

For $f, g \in L^1(\Omega)$, the convolution is defined by

$$(f * g)(t) = \int_\Omega f(\tau) g(t-\tau)\,\mathrm{d}\tau.$$

At this point, a problem with the definition of a signal flow graph occurs. Let's assume a signal x_j is related to the signal x_l via convolution by a function $h \in L^1(\Omega)$ through

$$x_l(t) = (x_j * h)(t).$$

This relation is not representable as the mapping w in the definition of a signal flow graph. To handle this problem, a well known transformation for linear time invariant systems is used, the Laplace-transform. It is defined for functions $f: [0,\infty) \to \mathbb{R}$, which are from exponential order, meaning that the function f fulfils the condition

$$|f(t)| \leq M \cdot e^{s_0 t} \qquad \forall t > T,$$

with constants $s_0, M, T > 0$. By the use of

$$\mathcal{L}(f)(s) = \int_0^\infty f(t) \cdot e^{-st}\,\mathrm{d}t,$$

a mapping from the set of functions from exponential order to the set of complex functions is defined and called the Laplace-transform. By the use of this Laplace-transform, the description of the convolution in a signal flow graph is realisable. The Laplace-transform offers the Theorem of convolution (see [5]), which transforms the convolution to a multiplication in the s-plane. If capital letters denote the corresponding Laplace-transform, the relation in the s-plane is described by

$$X_l(s) = X_j(s) \cdot H(s).$$

With this extension of the specification of a signal flow graph, the convolution can also be interpreted in this context.
Amongst others, these are basic operations which can be used to construct complex systems in Simulink.

2.3. Implementation of Simulink Models

This graphical representation built in the model editor corresponds to a MATLAB file which stores all used blocks and components and all connections between them. Even constants and possible variables, controlled out of MATLAB, as well as parameters, are stored in this MATLAB-file.
It is possible to run this file in MATLAB without a graphical representation in the Simulink model explorer. The model editor is a more comfortable tool to construct the models in Simulink, but simulation base is the MATLAB executable code in the mdl-file.

Simulink Models are executed via MATLAB on the MMT-Server. The model-file is build up like the schematic sketch in Figure 4.

```
Model {
  Name "simple"

  BlockParameterDefaults {

  [...]

  }

  Block {

  [...]

  }

  [...]

}
```

Figure 4: Cut-out of a (schematic) description of the MATLAB code of a Simulink model

In the MATLAB description of a Simulink model, all names, parameter values, connections, (default) adjustment of all blocks, etc. are stored. This file is executed by a MATLAB file which includes all values and parameters that are required for the simulation at the MMT-server.
The front-end of the MMT-server is equal to the one for MATLAB examples, only the simulator in the background is different. For visualisation, the graphical Simulink model is also available in the file-section, see Figure 1. If users download this file in a normal web browser, only the MATLAB code, as specified in Figure 2, can be observed. The signal flow graph representation of the Simulink model can be observed via local MATLAB/Simulink version. If the file with MATLAB code inside is opened by Simulink, the signal flow graph of the Simulink model is depicted.

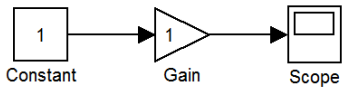

Figure 5: A simple Simulink model

To get an impression, the very simple Simulink model depicted in Figure 5 needs 730 lines MATLAB code to be created. This shows that the model explorer in Simulink is necessary for building more complex systems in Simulink.

3. SIMULINK EXPERIMENTS

Simulink offers a wide range of linear and non-linear time-continuous and time-discrete blocks. Important linear time-continuous block diagrams are the addition and multiplication block, integrator, transfer-function and state space block. These blocks are also available for linear and time discrete system, so-called sampling systems.
Simulink experiments have a wide range of tools with these precast systems. The class of linear systems enables varied methods to describe linear time invariant systems given by the system theory. On the one hand, these systems are characterised via linear ordinary differential equation of n-th order with constant coefficients. These can be transformed into n ordinary differential equations of first order. The differential equation system, combined with an equation, describing the output of a system, formulates the equations of the state space description of a linear time invariant system. An alternative form is a transfer function. Using the Laplace-transform, the state space description is translated in the complex s-plane, where the differential equations appear as algebraic equations. These equations allow formulating an input-output-relation for the linear systems. So a complex system, before specified by a differential equation, is represented in the complex s- plane via rational function.
This is one of the important features of Simulink: different model approaches are not only available for linear systems.
For the numerical simulation, Simulink provides numerical solvers for ordinary differential equations. In this way, linear as well as non-linear systems can be handled.

The following subsections show different examples implemented in Simulink on the MMT-server.

3.1. Binary Data Transmission

The following example shows, that Simulink provides many specialised blocks for a large variety of problems. In this particular case, a binary data transmission is observed. Binary data $\{0,1\}$ is mapped to (abstract) discrete symbols $\{s_0 = -1, s_1 = 1\}$.
After this mapping to discrete symbols, the transmission channel adds Gaussian noise. After the transmission, the binary data is reconstructed by a decision maker. The whole system is represented in Figure 6.

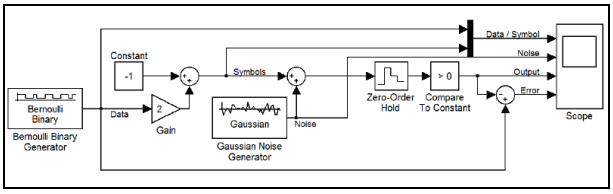

Figure 6: Simulink model of the binary data transmission

3.2. Matter transport of a substance in the human kidney

This example shows the application of a compartment model and its implementation in Simulink. The transport of a substance in the human kidney can be modelled via compartments, shown in Figure 7.

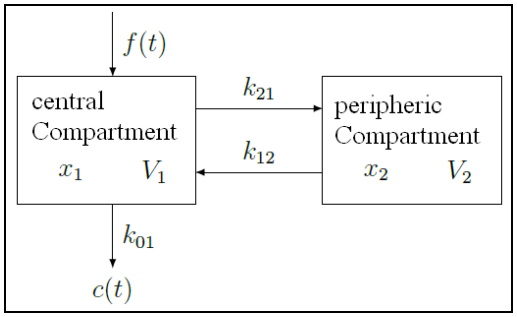

Figure 7: Model of the transport of a substance in the human kidney

The transport process is described by a system of differential equations and a the corresponding injection f. The differential equation system is given by

$$\frac{dx_1}{dt} = f(t) - (k_{01} + k_{21})x_2 + k_{12}x_2$$

$$\frac{dx_2}{dt} = k_{21}x_1 - k_{12}x_2$$

and the injection is modeled by

$$f(t) = \frac{D}{\tau} \quad \text{for} \quad 0 \leq t \leq \tau,$$

where D denotes the full amount of the substance.

In the experiment, the influences of the parameters k_{01}, k_{21} and k_{12} can be observed as well as the volume V_1 of the central compartment, the amount of substance D and the injection time .

Mainly, the model is described by a linear differential equation system with constant coefficients, which is managed differently in MATLAB than in Simulink. For these linear systems, MATLAB offers a number of functions and methods. The model implemented in Simulink is shown in Figure 8.

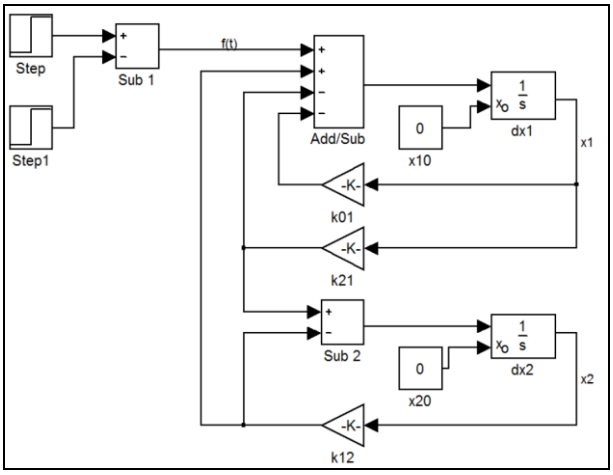

Figure 8: Simulink implementation of the transport model of a substance in the human kidney

This example is a good one to show the differences between the code-oriented implementation in MATLAB and the block-oriented implementation in Simulink.

3.3. Basic Pendulum Models

In section 1, about the built-up of the MMT-server, an example about the mathematical pendulum is used to explain an experiment at the server. This example is suitable to explain the usage of MATLAB and Simulink at the MMT-server.

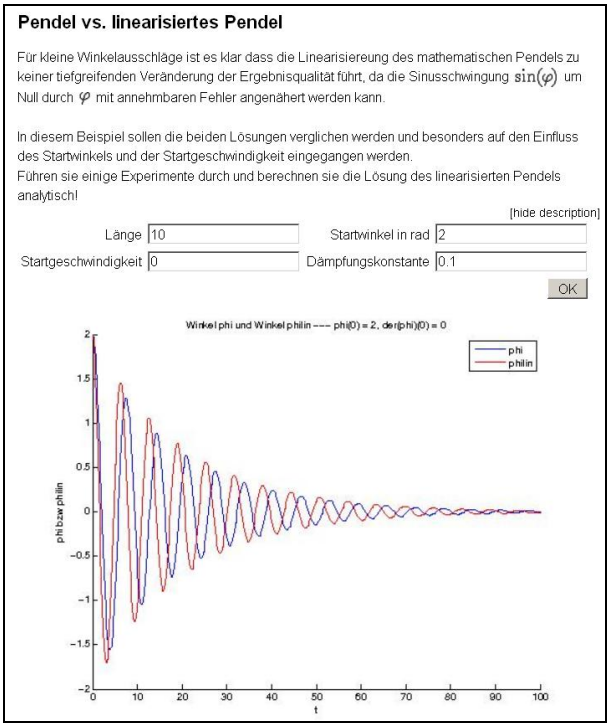

Figure 9: Pendulum experiment to compare linear and non-linear differential equation

The mathematical pendulum is described by a non-linear differential equation, so the second step is to linearise this equation and compare its solution to the non-linearised one. The linearised pendulum equation is received, when the sine function is replaced by the first order Taylor-expansion. The result is

$$m \cdot l \cdot \ddot{\varphi}(t) = -m \cdot g \cdot \varphi(t) - l \cdot d \cdot \dot{\varphi}(t).$$

The experiment at the MMT-server compares the solution of different parameters and different initial values. The result is, that for major starting angles the linear and non-linear solution diverges.

In Figure 9, the example of linear versus non-linear pendulum for a chosen set of values is shown.

Following to this basic example, some experiments with a double pendulum, a constrained pendulum and a sliding pendulum are available.
The last example of this introduction to the topic of modelling and simulation is the pendulum model implemented in Simulink.
In this model, Simulink and some MATLAB functions are compared. The pendulum is a rod pendulum and the example is implemented via Simulink model, shown in Figure 10.

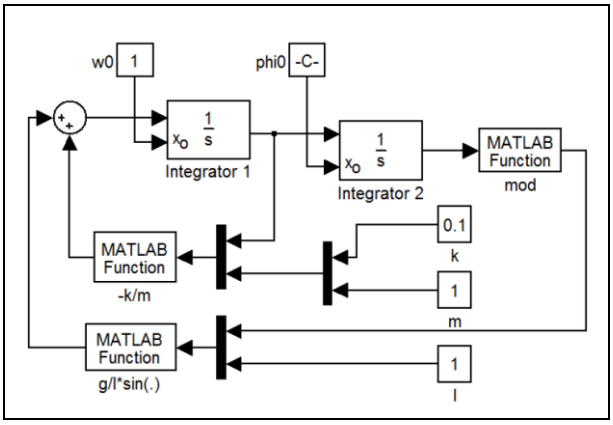

Figure 10: Simulink implementation of a rod pendulum

In this example, a good illustration of the combination of MATLAB functions with Simulink systems is possible. Simulink expands and complements the possibilities to build models at the MMT-server.

3.4. Pendulum with Friction

On the basis of the previous pendulum examples, even a more challenging example in the context of pendulums is available on the MMT-server. The pendulum moves in the horizontal plane and has friction respected. To visualise this specification, Figure 11 shows the pendulum.

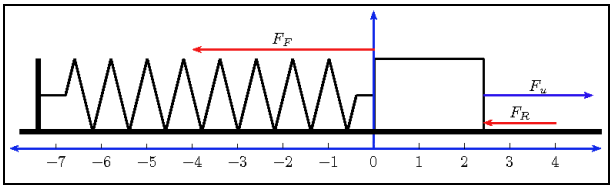

Figure 11: Horizontal pendulum with friction

The goal is to control the position of the mass by a PI-controller, while the process is given by the horizontal pendulum with friction.

This example is exclusively designed in Simulink, both the controller and the process. The controller is a well known structure, simply representable in Simulink. For the process model in Simulink, the pendulum process and the friction are built in two parts. The pendulum part is implemented and as an add-on, the friction process is done in an own model. The friction process depends on the velocity v and the force F, and the Simulink model is shown in Figure 12.

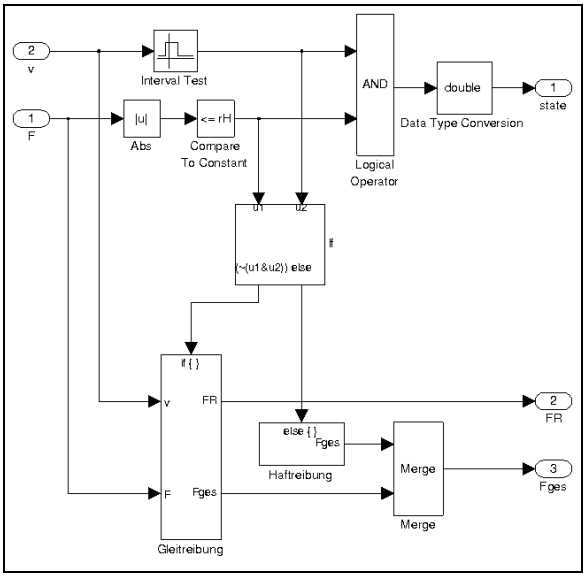

Figure 12: Friction process of the advanced pendulum model

In this case, the model is very large and many subsystems are involved. In that sense, the MMT-server can be used to show how to construct more complex systems in an economic way. Also examples can be offered to analyse the behaviour of several subsystems on their own on the one hand, and working together and influence each other on the other hand.

4. SUMMARY AND OUTLOOK

A long period of development work, which started in 2009, completed a multifaceted experiment server in 2012. The server started containing basic examples implemented in MATLAB to present typical simple tasks of mathematical modelling and simulation.
The initial MMT environment developed to become a multifunctional MATLAB simulation system.
Simulink models are represented as signal flow graphs. These are composed of small prepared blocks, which are offered by Simulink.
In the education of mathematical modelling and simulation, the server is used to teach different model approaches and to illustrate different simulation tools through MATLAB and Simulink.

Future plans include expanding the MMT-server with Simscape examples, as well as other available toolboxes of MATLAB/Simulink.
The MMT-server expands and evolutes with every designed example, because each new example creates new ideas to visualise or implement the results in a novel way.

ACKNOWLEDGEMENT

The MMT server, which is introduced and used in this work, is administered in cooperation with the dwh Simulation services.

REFERENCES

Körner, A., Zauner, G., Schneckenreither, G., 2009. Ein e-learning System für MMT – Mathematik, Modellbildung und Tools, Systemerweiterung und Einbindung von graphischer Modellbildung. *20th Symposium Simulation Techniques*, pages 87–94, Cottbus (Germany)

Winkler, S., Körner, A., Hafner, I., 2010. MMT – A web-based e-learning System for Mathematics, Modeling and Simulation using MATLAB. *7th EUROSIM Congress on Modelling and Simulation*, paper 231, Prague (Czech Republic)

Körner, A., Hafner, I., Bicher, M., Winkler, S., Breitenecker, F., 2011. MMT - A Web Environment for Education in Mathematical Modelling and Simulation. *ASIM 21. Symposium Simulationstechnik*, Winterthur (Switzerland), ISBN: 978-3-905745-44-3

Hafner, I., Bicher, M., Winkler, S., Fitsch, U., Körner, A., 2012. MMT _ A System Offering E-Learning of Modelling and Simulation in a Different Way. *MATHMOD 2012 - 7th Vienna Conference on Mathematical Modelling*, Vienna (Austria)

Körner, A., 2012, Analyse und Simulation dynamischer Systeme in Simulink mit Web-Interface, diploma thesis, Vienna University of Technology

AUTHORS BIOGRAPHY

Andreas Körner, born on 22.03.1982, is currently phd student at Vienna University of Technology. He is a member oft the group Mathematical Modelling and Simulation at the Institute for Analysis and Scientific Computing.