A PHP/MATLAB BASED E-LEARNING SYSTEM FOR EDUCATION IN ENGEINEERING MATHEMATICS AND IN MODELING AND SIMULATION

Günther Zauner[1, 2], Nikolas Popper[1], Felix Breitenecker[2]

[1]"die Drahtwarenhandlung" Simulation Services, Neustiftgasse 57-59, 1070 Wien
[2]Vienna University of Technology, Institute for Analysis and Scientific Computing, 1040 Vienna, Wiedner Hauptstraße 8-10 , Austria

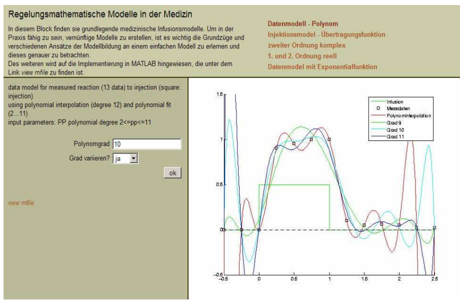
Guenther.zauner@drahtwarenhandlung.at

Abstract

The goal of this work is to present an e-learning tool based on the MATLAB Webserver technology. We offer an adaptive PHP framework which can be used for interactive learning in lessons and in project practices, as well as for web presentations of computer algebra solutions made in MATLAB (e.g. nonlinear fit problems for medical data). All mathematical/numerical solutions of the tasks are done in MATLAB. Another basic of the concept is not only to show the students the solution of a problem via internet for several different functions or parameters, but also to offer them the source code. The students can download the code and test other features and learn programming of mathematical solutions with a computer numeric/algebra package. An explanation of the detailed structure of the PHP – framework is given.

The main focus of this paper lies on model attempts for physiological systems. An example of a simple infusion model is used for interactive learning and system testing.

In several parts the way from a poor data interpolation to a data model with exponential functions up to a solution with transfer functions and parameter optimization is described. The outlook concerns the expandability of the defined framework, and will also focus on the restrictions of the system and how to deal with them.


Regelungsmathematische Modelle in der Medizin

In diesem Block finden sie grundlegende medizinische Infusionsmodelle. Um in der Praxis fähig zu sein, vernünftige Modelle zu erstellen, ist es wichtig die Grundzüge und verschiedenen Ansätze der Modellbildung an einem einfachen Modell zu erlernen und dieses genauer zu betrachten.
Des weiteren wird auf die Implementierung in MATLAB hingewiesen, die unter dem Link *view mfile* zu finden ist.

data model for measured reaction (13 data) to injection (square: injection)
using polynomial interpolation (degree 12) and polynomial fit (2...11)
input parameters: PP polynomial degree 2<=pp<=11

Polynomgrad: 10
Grad variieren?: ja

Keywords: e-learning, MATLAB, open code fragment, interactive learning.

Presenting Author's biography

Günther Zauner. He has earned a degree in mathematics with specialization in "mathematical computer science". With an interdisciplinary background based on higher technical school he has experience in the application and the development of numerical methods and different modeling approaches.

Current work focuses on simulation and modeling techniques, in this context especially model structure dynamics. Another main field of interest is the development of e-learning systems based on simulation environments.

1 General

At Vienna University of Technology a very similar problem like in other technical schools/universities in presenting lectures dealing with modeling and simulation basics, as well as summing up the necessary mathematical theory occurs: How to present the theory of modeling and simulation of special tasks based on examples, so that the students can follow it easily and thereby learn the most important basics?

As known from theory "learning by doing" is one of the best options. That is why the department for analysis and scientific computing designed the following structure for education. One of the main goals is to present dynamical models with a praxis interrelationship, which will be explained during the lessons, but should be also available via a web interface for advanced learning at home. Another principle is based on mathematics and computer numeric. For example it is much more comprehensible for students that the associative and distributive laws working with floating point numbers are hurt, when the conclusion is confirmed by interactive examples. In this case it is also important to show the programmed code in an easy readable language or with pseudo code.

This leads to the next benefit of the defined structure: the algorithmic part of the examples is all done in MATLAB and MATLAB/Simulink. This computer algebra/numeric package including the *symbolic math toolbox* is also used in the lessons "Einführung in das Programmieren für technische Mathematiker" and "Computermathematik". Therefore the main part of the visitors of the lessons, where the Webserver applications are included, are familiar with reading MATLAB code.

2 Background of MATLAB Webserver

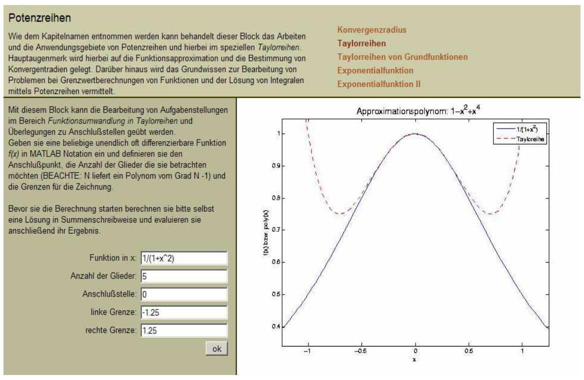
In general, as the examples are all realized in the MATLAB Release 2006a, the MATLAB Webserver application [1] needs an additional Webserver to act as server in client/server web architecture. In our case we use an Apache Webserver [2]. The interface used for input or input/output representation is defined by standard HTML (Hypertext Markup Language) [3] frames, which interact with MATLAB via a CGI – script.

In our case we decided to use PHP and interconnect the files directly with the Webserver. This has the following benefits:

- The system becomes more stable, because we have only two layers left instead of three.
- After defining the structure once, the whole system acts in modular concepts, which means that we can adapt examples and add new ones, without any code writing in PHP or HTML.

To get a general reusable system we have to define a global concept, capable to support the most important features for education in mathematics, modeling and simulation. Thus, we will get some restrictions in graphical representation and/or textual representation, but on the other hand a well defined structure supports easy model implementation and illustrating special content of teaching, which is not that easy explained in the common way at a blackboard.

The main part of the work is to create the basic structure. We chose the following frame definition, as shown in Fig. 1.


Potenzreihen

Wie dem Kapitelnamen entnommen werden kann behandelt dieser Block das Arbeiten und die Anwendungsgebiete von Potenzreihen und hierbei im speziellen *Taylorreihen*. Hauptaugenmerk wird hierbei auf die Funktionsapproximation und die Bestimmung von Konvergentradien gelegt. Darüber hinaus wird das Grundwissen zur Bearbeitung von Problemen bei Grenzwertberechnungen von Funktionen und der Lösung von Integralen mittels Potenzreihen vermittelt.

Konvergenzradius
Taylorreihen
Taylorreihen von Grundfunktionen
Exponentialfunktion
Exponentialfunktion II

Mit diesem Block kann die Bearbeitung von Aufgabenstellungen im Bereich *Funktionsumwandlung in Taylorreihen* und Überlegungen zu Anschlußstellen geübt werden.
Geben sie eine beliebige unendlich oft differenzierbare Funktion *f(x)* in MATLAB Notation ein und definieren sie den Anschlußpunkt, die Anzahl der Glieder die sie betrachten möchten (BEACHTE: N liefert ein Polynom vom Grad N -1) und die Grenzen für die Zeichnung.

Bevor sie die Berechnung starten berechnen sie bitte selbst eine Lösung in Summenschreibweise und evaluieren sie anschließend ihr Ergebnis.

Funktion in x:	1/(1+x^2)
Anzahl der Glieder:	5
Anschlußstelle:	0
linke Grenze:	-1.25
rechte Grenze:	1.25

Fig. 1 the general framework of the user interface including the first example

The upper left part of our input/output interface is the title of the chapter and a short explanation of the context, this part is dealing with.

In the same frame, beginning in the middle until the right side, there is the list of the examples corresponding to the chapter/block. This part has place for three columns with up to five rows providing the links to the specific examples. The lower part of the website, which is split into two frames, has an empty right side at the beginning of an example. On the left part a short description of the actual model is given. Under this part the definition and the settings of the selectable variables and parameters are placed.

3 Detailed structure

As already explained a general network is important to make an easy useable structure and to allow the fast growth of the system after a one-time detailed structural definition and implementation.

This is done in PHP with some special features. For fast transformation of the poor MATLAB m-file code into an MATLAB Webserver application we have to define some restrictions for the right side of the lower frame. This frame represents the output part of the system. As MATLAB allows different types of outputs and our system has restrictions in place and abilities for data representing (e.g. no rotation of 3d – graphics possible), we have to define strict rules for output creation and representation.

3.1 The three different output structures

In many cases of data transformation and in modeling and simulation, the user or developer is only interested in a textual form of the output. As known MATLAB can easily organize textual output to the command window (e.g. commands disp or print) or to a file. For our structure we define the synonym *retstr* as standard return value. This can be set to a string beginning with 'TEXT' and then filled with the return text, in which standard HTML commands, like <br> or <i>, can be implemented. The originated text is then transformed to standard HTML text in the output frame after executing the m-file.

The second sort of output is the classical plotting window. This is in our structure a so called 'IMAGE' – structure and therefore the *retstr* return value is set to this string. The implementation of such a structure needs a few extra code lines. An implementation of a graphical output can be done for example like this:

```
Pic = figure('visible','off');
…        % in this part the plot is defined in the same way
         % as this is done in standard MATLAB notation

drawnow;
wsprintjpeg(Pic,
instruct.mlimgfilename);
retstr = 'IMAGE';
```

The third output class is the so called 'ERROR' – class. To define the return value we first have to define what we consider to be an error. The first part where this class occurs is, when a MATLAB internal error arises. This can be for example because of wrong dimensions of input vectors or singularity of matrices. In these cases the original error message from MATLAB should be displayed via the web interface. The second sort of error message is the developer defined case. Such error messages are more or less equivalent to the work/usage of the TEXT construct and are in many cases used for programmer defined breaks before a MATLAB internal bug can occur.

3.2 Types of input variables

As everybody who is handling IO – interfaces for user applications knows, it is very hard to define a structure in which the user has a broad field of testing potentials, but concurrent catching all parameter settings and structures which are not allowed because this takes a lot of programming effort. Therefore we define special data types in PHP and perform the basic checks for the range directly in the definition part of the class.

The following code fragment shows all types of variables. The detailed explanation can be found below.

```
<?php
$pageVars['ml_mfile']='example_ergo3';
$form = new Form();
   $form->addField(new
ComboField('var1', 'paramter to
optimize:, 1,array(1 =>'TI',2 =>'KI',3
=>'TB',4 =>'KB')));
   $form->addField(new
FloatField('var2', 'starting value:',
0.1, 0.0001, 60));
   $form->addField(new
IntegerField('var3', 'Number of steps:',
5, 1, 30));
   $form->addField(new
TextField('var4,'Answer:', 'Optim.'));
$pageVars['form'] = &$form;
?>
```

The second code line defines the coupled MATLAB file, which runs within the defined structure. The following lines define examples for the four different input types: *ComboField*, *FloatField*, *IntegerField* and *TextField*.

The *ComboField* is a classical combo box (see also Fig. 2) as defined in several GUIs (General User Interface). In the example above the user can switch between four cases, which are than in MATLAB represented as numbers one to four.

The next part is the definition of a *FloatField*. The MATLAB name of the corresponding variable stands first and is the string var2. The next part separated by comma is the name the user will see in the interface. The third part is the default setting followed by the minimum value and the maximum, which can not be reached. The testing, if the defined value is valid, is executed by PHP, and in case it is not the input name is highlighted in red color and furthermore an error message occurs.

The definition of an *IntegerField* is done in a similar way as the *FloatField*. It is implemented because in many cases it does not make sense to define everything as floating point number. Moreover it improves the data filtering for the m-file.

The fourth example is a standard *TextField*. Compared to the *IntegerField* construction in this structure the last two parts of definition are missing. That is because a string does neither have a minimum value nor maximum value.

If we save this definition in the so called *initialize* text document, we get a system like depicted in Fig. 2.

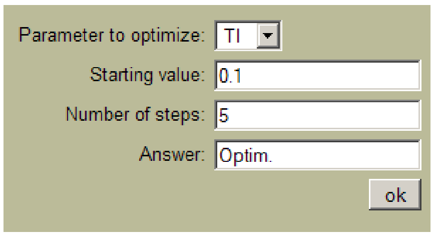

Parameter to optimize: TI
Starting value: 0.1
Number of steps: 5
Answer: Optim.

Fig. 2 graphical output corresponding to the code fragment on the left side of the page.

After explanation of the different data types the definition of the structure for the other components has to be done.

3.3 Input structure

We distinguish two different levels of our structure. This is done to ensure the reusability of the system and allow other staff, after a very short instruction period, to define new chapters and Webserver examples after defining the appropriate MATLAB code.

The main layer is the so called chapter level. The folder for one layer includes subfolders with the examples and four text files (There are several other files which are not important for the developer of new examples. These files include PHP code.). The first one, *description*, contains the text which is then represented in the top frame at the left side. This text can be defined in a standard editor using basic HTML commands.

The second, *headline*, is self-explanatory.

The third text file includes a few PHP commands in which the user can define a number for sorting the chapters. This part is optional and is only implemented for advanced system definition and extension for chapters in a higher hierarchical order.

The last file in the folder, *navigationLabel*, defines the chapter number. This block is also optional if we focus on only one chapter as a web application.

Now we have already defined the structure of an example chapter. One thing missing until now is the structure of the examples included in such a package. But as we will see the structure is quite similar and that is why the user does not have to learn many things before starting the implementation of examples.

Again we have a text file called *description*, where the short explanation of the file is created in textual form or with a few HTML additions. The *initialize* part is explained in detail in section 2.2.

The *navigationLabel* text file includes the name of the example. This name shows up as an entry in the link list in the upper frame of our example. Summing up this description we see, that after the user has written the *mfile*, only the input variables have to be set in PHP, all the rest is only writing text in an editor.

4 Application for transfer functions

After the definition of the whole framework – which is the main part of the work – we can go a step forward and show the structure and its benefits/restrictions within an application in the field of modeling and simulation in education.

In many cases it is easier for the students to understand system behavior of a class of problems using an example. Therfore a simple infusion model is chosen to explain the way from data measures up to a dynamic model structure.

4.1 Assignment of tasks

Backgrounds for the considerations in our model are physiological and metabolic processes

- which are observed over a time interval and for which measurements are available,
- which are influenced by factors from outside(e.g. medicine) and are reacting in a special way,

and

- systems where the attitude can be focused on relatively isolated from the surrounding, this means that the reaction to an excitation is not depending on other physiological components.

In the center of interest is the time dependent coherence between input and model reaction. The model has to be able to

- describe the time dependent characteristics in a mathematical comprehensible form,
- solve the problem with good correlation between input and output,
- reflect the physiological and biological coherence qualitatively well,
- to define the individual quantitative reaction only by parameter finding,
- fit the measurements as good as possible,

and

- predict the process reaction also with other conditions (e.g. changed excitation).

4.2 The model definition

The basic experimental structure of the model [4] is given by the following (virtual) assumptions:

An infusion with 500 ml of a substance over one hour is leading to an increase of the concentration of a well defined substance in the blood. Measurements c_i showing the increment of the substance in the blood are available at all time points t_i (every 15 minutes). Also important for the modeling point of view is what happens before the simulation starts. This is pointed out with two extra measurements half an hour and 15 minutes before the infusion starts. Fig. 3 shows the basic system.

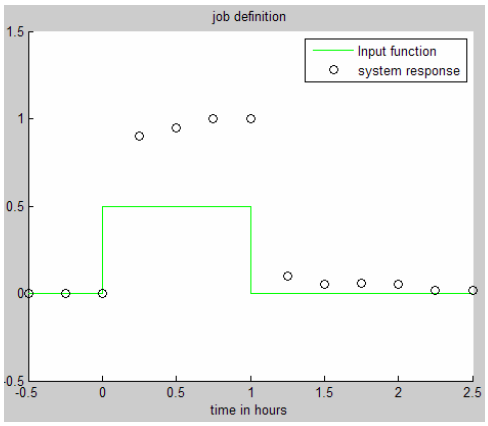


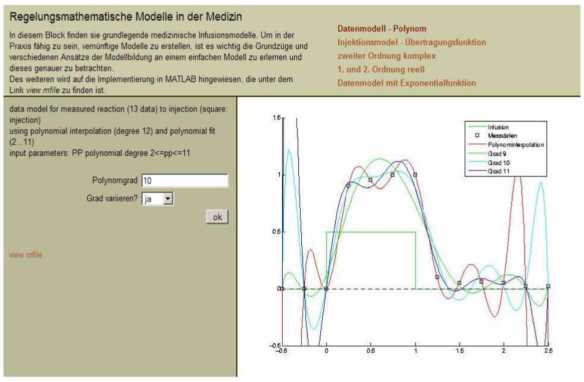
Fig. 3 model assumption

4.3 Data approach

The general question in the first iteration is, if there exist a mathematical model (formula) which can interpret the increase of concentration by a function which is at the measurement time close to the measured values. In the simplest case we are searching for an interpolation function *f(t)* with

$$f(t_i) = c_i \forall i \in \{1,\dots,n\} \tag{1}$$

One of the easiest ways to handle this is to make a polynomial interpolation. This is a classical data approach and therefore it is implemented as the first example in our MATLAB Webserver system. The modeling environment looks like the screenshot in Fig. 4. The students have the possibility to test the model with different compensation polynoms and with the interpolation polynom of order twelve.


Regelungsmathematische Modelle in der Medizin

In diesem Block finden sie grundlegende medizinische Infusionsmodelle. Um in der Praxis fähig zu sein, vernünftige Modelle zu erstellen, ist es wichtig die Grundzüge und verschiedenen Ansätze der Modellbildung an einem einfachen Modell zu erlernen und dieses genauer zu betrachten.
Des weiteren wird auf die Implementierung in MATLAB hingewiesen, die unter dem Link *view mfile* zu finden ist.

Datenmodell - Polynom
Injektionsmodel - Übertragungsfunktion
zweiter Ordnung komplex
1. und 2. Ordnung reell
Datenmodel mit Exponentialfunktion

data model for measured reaction (13 data) to injection (square: injection)
using polynomial interpolation (degree 12) and polynomial fit (2...11)
input parameters: PP polynomial degree 2<=pp<=11

Polynomgrad: 10
Grad variieren?: ja

Fig. 4 MATLAB Webserver example for a polynomial interpolation of the task defined in chapter 3.2

The students see the problems that occur when someone adds an extra measurement and that the interpolation is good in the data points, but outside it tends towards infinity. Also the problematic of the impossibility to handle other input functions and make a feasible model with the polynomial function can be shown. From modeling point of view this model is not appropriate because our system description acts with physiological impossible (negative) values.

The main problem of the function *f(t)* is, that it does not depend on the input function. It tries to handle the output, whereby the measurements are the only used knowledge. The infusion is never taken into account.

To sum it up we can see that all of this interpolation methods (polynom interpolation, splines,…) just build a data model – the model does exclusively represent the measurements c_i.

4.4 Exponential function approach

The next step towards an universally valid model is established on basic considerations:

From physiological point of view we know, that the function should be relatively smooth over time. Furthermore we see coherence between the input function defined as a rectangular by the infusion input.

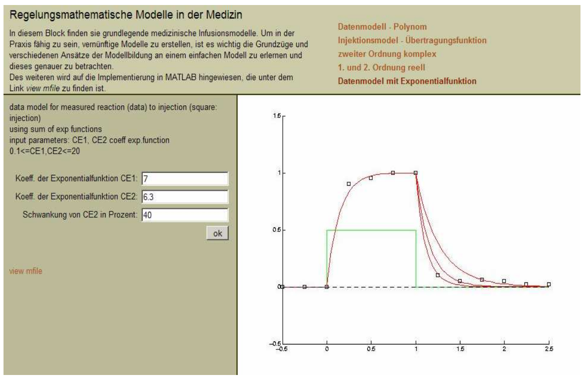
The coherence between infusion and the system reaction is described by exponential functions. The first time before the infusion starts to flow into the blood we assume the zero function as only valid solution. When the infusion starts, the output function tries to come up to a fixed level. In general this is explained by formular (2).

$$f(t) = a - b * e^{-c(t-t_o)} \; a,b,c \in \Re \tag{2}$$

In the end of the infusion time the input function jumps to zero. Afterwards the measured data seems to follow this function. This can be modelled again with a negative exponential function.

Summing up these results, we get a function which describes the output behaviour of the physiological system with three general exponential functions, whereby the first part, the zero function, is trivial.

The implementation of this model assumption is shown in Fig. 5. In this model the extra possibility of parameter variation for the third part can be done.


Regelungsmathematische Modelle in der Medizin

In diesem Block finden sie grundlegende medizinische Infusionsmodelle. Um in der Praxis fähig zu sein, vernünftige Modelle zu erstellen, ist es wichtig die Grundzüge und verschiedenen Ansätze der Modellbildung an einem einfachen Modell zu erlernen und dieses genauer zu betrachten.
Des weiteren wird auf die Implementierung in MATLAB hingewiesen, die unter dem Link *view mfile* zu finden ist.

Datenmodell - Polynom
Injektionsmodel - Übertragungsfunktion
zweiter Ordnung komplex
1. und 2. Ordnung reell
Datenmodel mit Exponentialfunktion

data model for measured reaction (data) to injection (square: injection)
using sum of exp functions
input parameters: CE1, CE2 coeff exp.function
0.1<=CE1,CE2<=20

Koeff. der Exponentialfunktion CE1: 7
Koeff. der Exponentialfunktion CE2: 6.3
Schwankung von CE2 in Prozent: 40

Fig. 5 MATLAB Webserver implementation for the exponential model assumption

As can be seen, the model output fits the measurements c_i very good. But what has to be taken into account is, that this model is no interpolation with exponential functions because the exponential functions used in the model act on different intervals.

A big disadvantage of the structure is that a change in the activation function (infusion) changes the whole reaction function. Nevertheless this model is not any more a simple data model because the input influences the output structure (defines when to switch between the different exponential functions).

4.5 Transfer function approach

As we want to use our MATLAB Webserver in education to teach the students to handle simple physiological examples, the solutions we give until now are not good enough. That is the reason why we make the next step towards a general system description.

Control theory leads to an appropriate model description between the input u(t) and the output x(t) in a structural-graphical, as well as in a mathematical-formulary way. The general structure is depicted in Fig. 6.

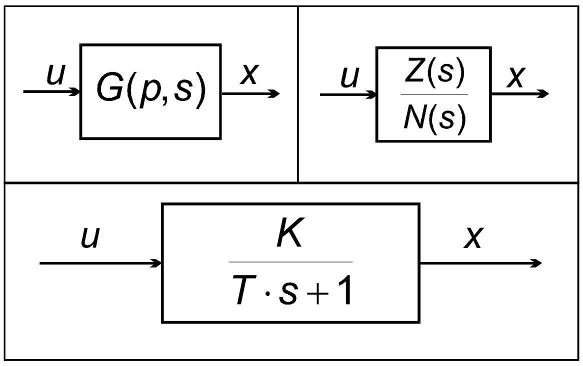
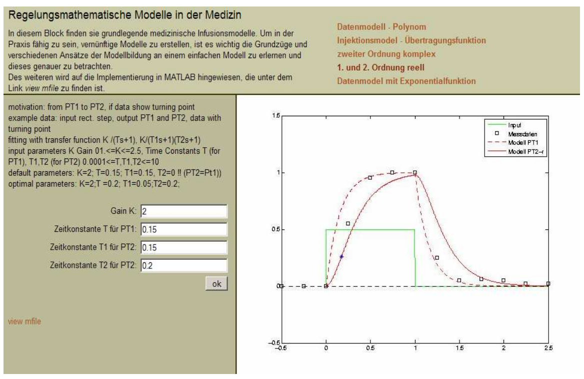


Fig. 6 structural-graphical representation of the relation; input *u(t)*/output *x(t)*, general model (top) and first order function as for the infusion model(down)

To get a feeling for the work of a transfer function and how to handle a system in an adequate way with this control structure we implemented several examples on the MATLAB Webserver. The solutions for the first order activation function and a parameter combination for a second order system with real zero points of the denominator are shown in Fig. 7.



Regelungsmathematische Modelle in der Medizin

In diesem Block finden sie grundlegende medizinische Infusionsmodelle. Um in der Praxis fähig zu sein, vernünftige Modelle zu erstellen, ist es wichtig die Grundzüge und verschiedenen Ansätze der Modellbildung an einem einfachen Modell zu erlernen und dieses genauer zu betrachten.
Des weiteren wird auf die Implementierung in MATLAB hingewiesen, die unter dem Link *view mfile* zu finden ist.

Datenmodell - Polynom
Injektionsmodel - Übertragungsfunktion
zweiter Ordnung komplex
1. und 2. Ordnung reell
Datenmodel mit Exponentialfunktion

motivation: from PT1 to PT2, if data show turning point
example data: input rect. step, output PT1 and PT2, data with turning point
fitting with transfer function K /(Ts+1), K/(T1s+1)(T2s+1)
input parameters K Gain 01.<=K<=2.5, Time Constants T (for PT1), T1,T2 (for PT2) 0.0001<=T,T1,T2<=10
default parameters: K=2; T=0.15; T1=0.15, T2=0 !! (PT2=Pt1))
optimal parameters: K=2;T =0.2; T1=0.05;T2=0.2;

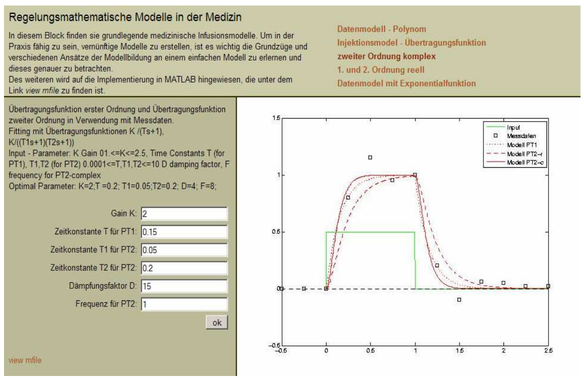
Gain K: 2
Zeitkonstante T für PT1: 0.15
Zeitkonstante T1 für PT2: 0.15
Zeitkonstante T2 für PT2: 0.2

Fig. 7 first order and second order with real zero points

The PT1 – element is plotted as dashed red line, in the representation of the second order transfer function the turning point of the function is visualized as a blue star.

The students can test optimization algorithms and for all the examples they have the button *view mfile* in the lower left corner of the example definition and parameter frame. This leads to another benefit of our system. The people working with this interface learn about modelling and simulation and, furthermore, by using this simple example they can advance their knowledge about programming in the computer numeric/algebra package MATLAB, which is in a wide area more or less the standard.

The next class of functions to be focussed on are the complex second order transfer functions. In this context the characteristic parameters are damping and frequency. A composition of all three systems (first order, second order real and second order complex) for our infusion model is shown in figure 8. The output functions are all coloured red, whereby the dashed line shows the PT2 – real model and the dotted one represents the first order transfer function.



Regelungsmathematische Modelle in der Medizin

In diesem Block finden sie grundlegende medizinische Infusionsmodelle. Um in der Praxis fähig zu sein, vernünftige Modelle zu erstellen, ist es wichtig die Grundzüge und verschiedenen Ansätze der Modellbildung an einem einfachen Modell zu erlernen und dieses genauer zu betrachten.
Des weiteren wird auf die Implementierung in MATLAB hingewiesen, die unter dem Link *view mfile* zu finden ist.

Datenmodell - Polynom
Injektionsmodel - Übertragungsfunktion
zweiter Ordnung komplex
1. und 2. Ordnung reell
Datenmodel mit Exponentialfunktion

Übertragungsfunktion erster Ordnung und Übertragungsfunktion zweiter Ordnung in Verwendung mit Messdaten.
Fitting mit Übertragungsfunktionen K /(Ts+1), K/((T1s+1)(T2s+1))
Input - Parameter: K Gain 01.<=K<=2.5, Time Constants T (for PT1), T1,T2 (for PT2) 0.0001<=T,T1,T2<=10 D damping factor, F frequency for PT2-complex
Optimal Parameter: K=2;T =0.2; T1=0.05;T2=0.2; D=4; F=8;

Gain K: 2
Zeitkonstante T für PT1: 0.15
Zeitkonstante T1 für PT2: 0.05
Zeitkonstante T2 für PT2: 0.2
Dämpfungsfaktor D: 15
Frequenz für PT2: 1

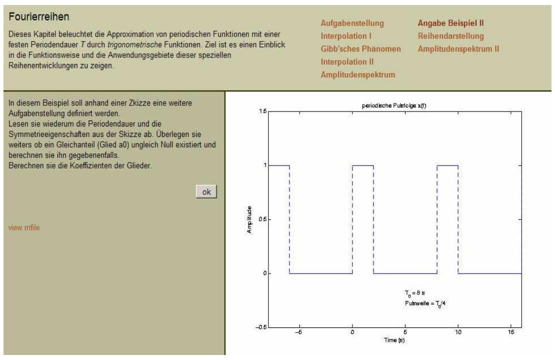
Fig. 8 three different transfer function approaches for the system from chapter 3.2

For all of these transfer function models we see, that we are now (far) away from the poor data model. Our system can now handle the dynamic features of this model and the solution which is generated with this method is capable to react in an adequate way to input changes.

5 Fourier series application

Another often used application in techniques is the approximation of a periodic function by a sum of trigonometric functions. The theory behind is called Fourier series [5] and is a part of teaching in all technical branches of study at Vienna University of Technology. For this reason additional blocks with examples are implemented on the MATLAB Webserver.

One class of examples starts with a picture of a predefined example function. A sample is shown in Fig. 9.


Fourierreihen

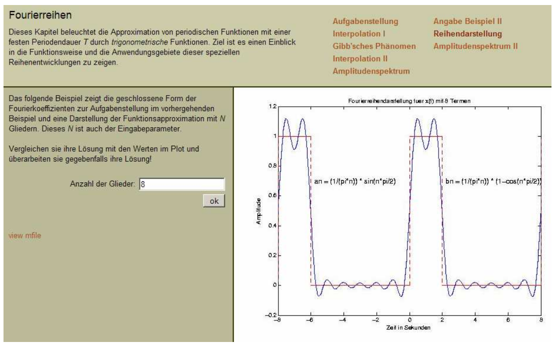
Dieses Kapitel beleuchtet die Approximation von periodischen Funktionen mit einer festen Periodendauer *T* durch *trigonometrische* Funktionen. Ziel ist es einen Einblick in die Funktionsweise und die Anwendungsgebiete dieser speziellen Reihenentwicklungen zu zeigen.

In diesem Beispiel soll anhand einer Zkizze eine weitere Aufgabenstellung definiert werden.
Lesen sie wiederum die Periodendauer und die Symmetrieeigenschaften aus der Skizze ab. Überlegen sie weiters ob ein Gleichanteil (Glied a0) ungleich Null existiert und berechnen sie ihn gegebenenfalls.
Berechnen sie die Koeffizienten der Glieder.

Fig. 9 setting of a standard task on our interface.

The students first have to answer questions and then go on with the solution of the system. The focus hereby does not only lie on the mathematical solution of the task into an indefinite sum, but also to explain effects like Gibbs phenomena and to communicate the feeling how many elements are necessary to get an effective solution for further calculations.

The approximation of the input function with a defined number of elements can be calculated. The solution for the function defined in Fig. 9 is shown in Fig. 10. Also the formulas for the coefficients of the *sinus* and *cosinus* function are returned.


Fourierreihen

Dieses Kapitel beleuchtet die Approximation von periodischen Funktionen mit einer festen Periodendauer *T* durch *trigonometrische* Funktionen. Ziel ist es einen Einblick in die Funktionsweise und die Anwendungsgebiete dieser speziellen Reihenentwicklungen zu zeigen.

Das folgende Beispiel zeigt die geschlossene Form der Fourierkoeffizienten zur Aufgabenstellung im vorhergehenden Beispiel und eine Darstellung der Funktionsapproximation mit *N* Gliedern. Dieses *N* ist auch der Eingabeparameter.

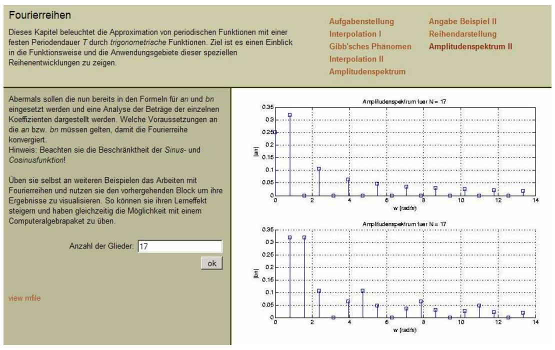
Vergleichen sie ihre Lösung mit den Werten im Plot und überarbeiten sie gegebenfalls ihre Lösung!

Anzahl der Glieder: 8

Fig. 10 solution with eight elements to the example defined like in figure 9.

The last questions discussed in the block are the frequency and absolute value spectra. This representation explains again the behavior of the absolute value for higher number of summing index.

For the discussed example the solution of the block is shown in Fig. 11.


Fourierreihen

Dieses Kapitel beleuchtet die Approximation von periodischen Funktionen mit einer festen Periodendauer *T* durch *trigonometrische* Funktionen. Ziel ist es einen Einblick in die Funktionsweise und die Anwendungsgebiete dieser speziellen Reihenentwicklungen zu zeigen.

Abermals sollen die nun bereits in den Formeln für *an* und *bn* eingesetzt werden und eine Analyse der Beträge der einzelnen Koeffizienten dargestellt werden. Welche Voraussetzungen an die *an* bzw. *bn* müssen gelten, damit die Fourierreihe konvergiert.
Hinweis: Beachten sie die Beschränktheit der *Sinus*- und *Cosinusfunktion*!

Üben sie selbst an weiteren Beispielen das Arbeiten mit Fourierreihen und nutzen sie den vorhergehenden Block um ihre Ergebnisse zu visualisieren. So können sie ihren Lerneffekt steigern und haben gleichzeitig die Möglichkeit mit einem Computeralgebrapaket zu üben.

Anzahl der Glieder: 17

Fig. 11 the upper plot shows the amplitudespectra of the a_n, which means the coefficients of the *cos* part. The lower plot depicts the coeff.of the *sin* elemts.

Within these two plots the students can see on of the necessary conditions for convergence of the sum of trigonometric functions: the coefficients have to be a null sequence. The coefficients of a convergent majorant series can be chosen to depict the knowledge graphical. Therefore the *symbolic math toolbox* is used again.

6 Outlook

As pointed out in sections 4 and 5 the MATLAB Webserver solution is compatible for use in modeling applications. Two main goals are of interest:

- Easy handling parameter/function-variation for given examples, and thus, supporting learning by doing

and

- the possibility to have a look at the source code in every step and make further work with it in the own MATLAB workspace.

The shown system is a tool which is not only used to present tasks in modeling and simulation, but also to impart the students in special fields of mathematics, where the graphical view combined with the formal solution helps a lot to understand the theory (e.g. Fourier Series, higher dimensional extreme value analysis, Taylor Polynom, Interpolation, …).

One of the most interesting next steps will be the implementation of a state flow simulation environment based on the MATLAB Webserver.

7 References

[1] http://www.mathworks.com/

[2] http://httpd.apache.org/

[3] M. Lubkowitz. Webseiten programmieren und gestalten – HTML, CSS, JavaScript, PHP, Perl, MySQL, SVG. Galileo Press, Bonn 2003, ISBN-10: 3898423131.

[4] F. Breitenecker, H .Ecker and I. Bausch-Gall. Simulation mit ACSL : eine Einführung in die Modellbildung, numerischen Methoden und Simulation . Braunschweig : Vieweg, 1993. - XI, 399 S

[5] H. Amann, J. Escher. Analysis II. Basel Bosten Berlin, Birkhäuser, 1999