TEACHING PHYSIOLOGICAL ENGINEERING USING A MATLAB-BASED WEB ENVIRONMENT

G. Schneckenreither[1], G. Zauner[2], F. Judex[1]
[1]Vienna University of Technology, Austria, [2]'die Drahtwarenhandlung' – Simulation Services, Austria

Corresponding author: G. Schneckenreither
Vienna University of Technology, Institute for Analysis and Scientific Computing
1040 Wien, Wiedner Hauptstraße 8-10, Austria, `gschneck@osiris.tuwien.ac.at`

Abstract. This contribution deals with the application of a web-based e-learning system in classes and during autonomous learning. Based on the discussion of an example in the field of physiology we present and define the aims and options of our system in education. The example is a compartment model, which simulates glucose regulation in the human body. Certain trade-offs between the hormones glucagon respectively insulin and glucose can be described through a system of ordinary differential equations (ODEs). High concentration of glucose leads to production of insulin in the pancreas, which serves as a down-regulator for the glucose concentration (blood sugar level). A low level of glucose on the other side initiates production of the up-regulator glucagon. Implementation of the ODE approach involves invocations of ODE-solvers as well as non-continuous and threshold-dependant elements in the right hand side of the differential equations. Furthermore a block-model approach can be implemented with the SIMULINK environment. Accordingly this is an ideal example for demonstrating dynamic behaviour of a model, simulation (of physiological dysfunctions) and the application of differential equations and a corresponding block model approach in modelling and simulation.

1 Introduction

Since our e-learning system is based on a growing number of diverse applications and programs there is lots of room for deployment. Some of our applications are primarily intended to serve as interactive demonstrations during lectures, like for example interpolation problems. Directly through the graphical feedback of the remote program, the lecturer can highlight the differences between different algorithms. There also exist collections of examples [1], which are organised in a way, so that they cover the mathematics of a whole course. Accordingly the system can be used by students in parallel to the lectures as a pool for training and experimenting.

Other examples involve rather basic geometric problems but use vectors, transformation into polar coordinates or "vectorisation of loops" (MATLAB specific) on the programming level and therefore can also serve as illustrations for programming techniques during lectures. Again these remote applications are always accessible to students, so that they can be used in homework, tasks or autonomously.

More enhanced programs are settled in the area of modelling and simulation. These examples involve often a textual description or documentation. Some of them are based on the ARGESIM Comparisons, which are available on the internet [2]. Main focus of these examples lies in the process of modelling, the implementation of models (p.e. ODEs, block models), or simulation, interpretation and demonstration of results. Students form simulation classes, who are assigned to a certain task or exercise, can view the behaviour and code of the implemented examples and gather information useful for the process of creating such an application.

For demonstrating dynamic behaviour of models or programs during presentations (or lectures) it is often necessary to run one or even more programming environments on a laptop in order to show graphical output via a projector. Since connection to the internet is more and more often available and almost any computer runs an ordinary web-browser, it is easier and more convenient to access programs of different type through only one application and execute them on a remote and stable runtime environment.

The programs on our e-learning platform were mostly written by members of the modelling and simulation department partially in cooperation with the German faculty of the University of Technology Odessa. However, some examples were implemented in the scope of an internship or as exercises in special courses.

At the time the programming language is MATLAB in nearly all cases. There also exist calls to MODELICA instructions, which are based on the Java language. With very little restrictions it is also possible to execute SIMULINK models, which are invoked from MATLAB programs. For the near future we plan to reorganise the runtime environment to a more flexible structure, which will make it much more easy to incorporate different programming languages or platforms. The original interface to the MATLAB engine was provided by the "MATLAB Server"-application by MathWorks.

The following sections will present an example in the area of physiology. We discuss the structure of this model, methods of implementation and the process and purpose of interaction with the resulting programs.

2 Glucose Regulation - A Compartment Model

The model [3, 4, 5] has a clearly visible compartment structure and can be described by a system of differential equations or a block model with transfer functions. It returns the concentration of the substances glucose, glucagon and insulin over time and uses piecewise constant factors and some threshold dependant influences to model reactions of the pancreas to changes in the glucose level or to infusions of substances.

2.1 Glucose Regulation in General

Glucose (sugar with the chemical formula $C_6H_{12}O_6$) – among other features – serves as energy carrier to many biochemical systems and is crucial for enzymatic reactions in the human body. Therefore concentration of this carbohydrate is an important and carefully regulated physiological factor. The regular blood sugar level is about 90 mg of glucose per 100 ml of blood. Especially two hormones are involved in the regulation of glucose, namely glucagon and insulin (Figure 1).

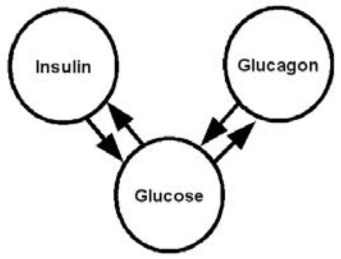


Figure 1: This strongly simplified block diagram shows the compartment structure of the glucose-regulation model.

A high level of glucose concentration leads to insulin production in the pancreas. Insulin favours the inclusion of glucose in the liver and muscles and accordingly serves as down-regulator for the glucose concentration.

Insulin [6] is synthesised by pancreatic β-cells and enhances conversion of glucose to glycogen and storage in liver and muscles, entry of glucose into cells and accordingly its breakdown to water and carbon dioxide during aerobic metabolism (glycolysis), as well as conversion to fatty acids.

A low level of glucose on the other side triggers glucagon production in the pancreas (α-cells). The up-regulator glucagon frees glucose, which has been converted and enclosed in the liver and leads to synthesis of glucose from amino or fatty acids.

A very high concentration of glucose (hyperglycaemia) leads to direct expulsion through the kidney.

Symptoms and ailments caused by dysfunction of the glucose regulation mechanism are among others [6]:

- in the case of too high glucose concentration: thirst, frequent urination, abdominal pain, cramps, tiredness, concentration disorder and headache
- in the case of low blood-sugar: decrease of brain activity, spasms, unexpected surges of adrenaline and acute perspiration.

The according model can be used for simulating regulation of the glucose level during infusions, dysfunctions of the pancreas such as Diabetes I, Diabetes II and Hyperinsulinism or countermeasures to these diseases.

2.2 Formulation as Differential Equations

The classical formulation of this model is by a system of linear ordinary differential equations including time dependent but piecewise constant parameters and threshold-functions. The system can be directly obtained from the textual model description.

$$\begin{aligned}
\dot{\Phi}_1(t) &= a_1 \cdot \big(b_{1,1}\cdot\Phi_1 + b_{1,2}\cdot\Phi_2 + b_{1,3}\cdot\Phi_3 + c_1(t) + d_1(\Phi_1)\big)\\
\dot{\Phi}_2(t) &= a_2 \cdot \big(\qquad\quad b_{2,2}\cdot\Phi_2 \qquad\quad + c_2(t) + d_2(\Phi_1)\big)\\
\dot{\Phi}_3(t) &= a_3 \cdot \big(\qquad\qquad\qquad\quad b_{3,3}\cdot\Phi_3 + c_3(t) + d_3(\Phi_1)\big)
\end{aligned} \tag{1}$$

It is clearly visible that glucose concentration directly depends on the concentration of the up- and down-regulators glucagon and insulin. Accordingly parameters $b_{1,2}$ and $b_{1,3}$ describe the functionality of these regulative substances. Further all concentrations decrease by time caused by natural decomposition (parameters $b_{i,i}$).

The piecewise constant inflow functions and the threshold dependant regulatory functions are defined by

$$c_i(t) = c_{i,0} + c_{i,1} \cdot \mathbb{I}_{[t_{i,1}, t_{i,2}]}(t) \tag{2}$$

$$d_i(\Phi_1) = \begin{cases} 0 & \Phi_1 \le T_i \\ m_i \cdot (\Phi_1 - n_i) & \Phi_1 > T_i, \end{cases} \tag{3}$$

where $\mathbb{I}$ denotes the indicator or characteristic function.

The functions $c_i(t)$ allow to define infusions of glucose, insulin and glucagon into the blood circuit. The functions d_i model the onset of production and expulsion of substances dependant on the glucose level. Accordingly these functions present the actual regulation mechanisms.

A detailed description of variables and parameters is given in Table 1.

$\Phi_1(t)$	...	glucose concentration
$\Phi_2(t)$	...	glucagon concentration
$\Phi_3(t)$	...	insulin concentration
a_i	...	compartment size factor
$b_{i,j}$	...	feedbacks and trade-offs between substances
$c_i(t)$	...	inflow of substances (production and infusions)
$d_i(\Phi_1(t))$	...	balancing functions (depend on glucose concentration)
$c_{i,0}$	...	constant production of a substance
$c_{i,1}$	...	additional infusion
$t_{i,j}$	...	time values for start and stop of the infusion
T_i	...	threshold value for glucose concentration
m_i	...	rate of production resp. evacuation of substances
n_i	...	target value for substance concentration

Table 1: Variable- and parameter-names.

2.3 Block Model Form

Due to the compartment structure of this model, it is possible to build an equivalent SIMULINK block model. The block diagram is shown in Figure 2. Students can download the source code and run and modify the model on a local desktop environment.

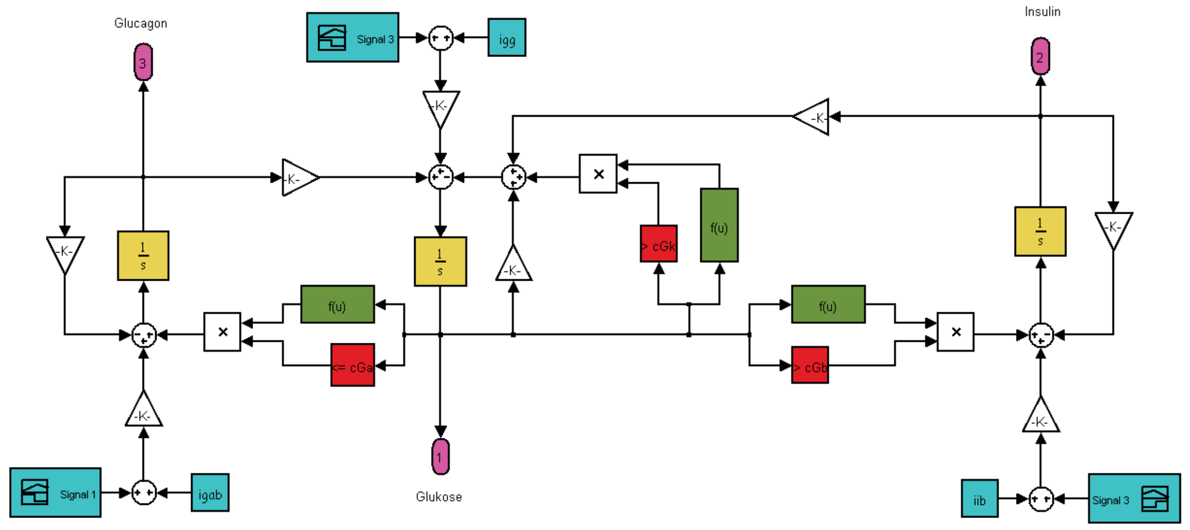


Figure 2: This is a screenshot from the SIMULINK environment, showing a detailed block diagram of the model implemented in SIMULINK. Clearly visible are the production and infusion blocks in turquoise, the transfer functions in yellow and algebraic threshold functions in green with zero-crossing detections in red.

Decomposition of all three concentrations can be found in the transfer function loops (compare $b_{i,i}$ in the ODE formulation). The effect of glucagon and insulin concentration on the glucose level is modelled by the top-most horizontal connections (corresponds to $b_{1,2}$ and $b_{1,3}$). Infusions are modelled by piecewise constant input signals (turquoise, corresponds to $c_i(t)$). And finally the regulation mechanisms (d_i) require a “threshold-crossing” detection, which delivers a logical value, and the description of the onset when a threshold is reached.

3 Remote Applications in Education

This section focuses on possible ways of application of our remote programs in education presented on basis of the glucose-regulation model. We define two main areas of application, which are presentations during lectures (section 3.1) and individual interaction (section 3.2). Even though the programs on our remote runtime environment are intended to serve a specific task in one of these areas, most of them can be used in both scenarios.

3.1 Application During Lectures

- The model allows to discuss the process of building an ODE-model (modelling) by using a detailed model description (textual or spoken). While explaining the trade-offs between glucose and insulin respectively glucose and glucagon or discussing the thresholds for production of insulin and glucagon, the lecturer can highlight the corresponding expressions in the ODE-system. This establishes a direct connection between reactions and behaviours in real physiological systems and mathematical expressions. For example the insulin concentration affects the glucose level in a linear manner $\dot{\Phi}_1(t) = \cdots + b_{1,3} \cdot \Phi_3 + \cdots$.
- The glucose-regulation model is an ideal example for control engineering in physiology. It involves concentrations of substances, their creation and evacuation, threshold dependant reactions and modelling of infusions, which represent external regulation interventions in case of a dysfunction of the physiological system.
- The lecturer can also present the conveniences of simulating physiological processes. Assuming the model is valid, countermeasures such as infusions and in particular the amount of the infusion can be tested. Furthermore the model allows to adapt parameters for individual patients, which could allow better treatment. But also general benefits and the use of modelling and simulation in various areas can be discussed.
- Due to the fact that the glucose-regulation model involves a formulation by ODEs it also requires solvers and a correct implementation suitable for invoking the solver-routines. This is interesting especially on the programming level since students must learn the MATLAB language. In particular the non-continuous parts of the right hand side require that the equation is formulated as a separate function after the main function, which contains the calls to the ODE-solver. More simple equations on the contrary would allow a definition as in-line-functions.
- Alternatively the model can also be implemented as a block model using for example the SIMULINK environment. Again the connections between real-life physiological reactions or processes and the description by blocks are very interesting and representative for modelling in control engineering. Furthermore the lecturer can show that both approaches lead to the same qualitative and quantitative model behaviour, which can be deepened by performing Laplace transformations in the ODE-system.

3.2 Individual Interaction

- The remote applications normally use 2 to 6 input parameters, which the user enters in the web-interface. Experimenting with those values lets the students get an understanding for the dynamics and reactions of the system. Often several scenarios are implemented as different applications. So the students can compare results of simulations under different predefined circumstances.
- Students can view the source-code and run models, which have previously been discussed in lectures. Accordingly the examples serve as repetition and for better understanding apart from the lecture.
- In the scope of special courses, students are often advised to build or extend a model and simulate specific scenarios. The remote applications and their implementation can serve as a pool for tasks and exercises.
- Since on our server lots of models and examples implemented with various techniques are available, students can investigate the modelling and programming techniques of fully functional examples. This may be useful for building and implementing a model even apart from lectures, which work with the remote runtime-environment.

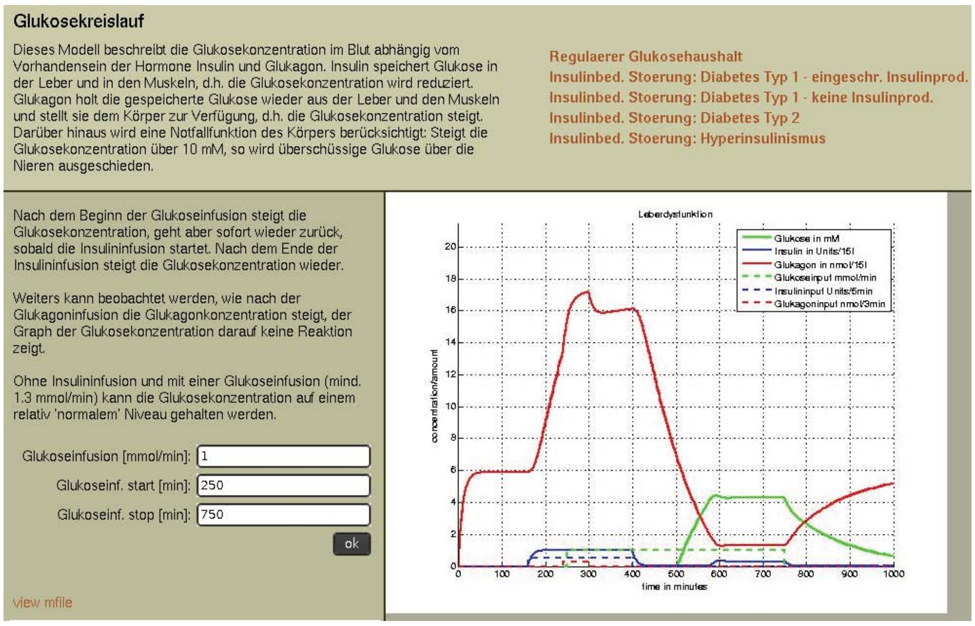
4 An Example of Application

To deepen the insight into the application of our remote system, we first focus on direct interaction through the web-interface (section 4.1) and then present a more global and long-term step-by-step process of interaction, which involves the application of the glucose-regulation model in lectures, during autonomous repetition and in tasks (section 4.2).

4.1 Interaction with the System

This section will give a detailed view on interaction with the e-learning system through the web-interface based on a specific example, which simulates a dysfunction of the glucose regulation mechanism called “liver-dysfunction”. The up-regulator glucagon is not able to increase the glucose level by freeing glucose stored in the liver and muscles. Therefore infusion is the only way to increase the glucose concentration. The user can control the infusion of glucose while an infusion of insulin is predefined in order to generate a reaction of the system, which is characterised by a decrease of the glucose level (Section 2.1). In Figure 3 it is visible that glucagon concentration (red) does not affect the level of glucose (green) while the infusion of insulin (blue) prevents an increase of the glucose level.

When the user invokes the example, predefined values are set in the input fields. By clicking on the execution button ("OK") the remote program calculates the development of concentrations over time and generates the output image, which then is shown in the right frame of the web-interface. By changing the values in the input fields and executing the application again, the user can analyse the reaction of the system under different conditions. If for example the glucose infusion level is set to a higher value, the insulin concentration/infusion cannot prevent the increase of the glucose level. In order to prevent futile input values, system crashes or extremely long calculation times the input values are limited and the html-interface shows a warning message beneath the input fields when a value is not allowed.


Glukosekreislauf

Dieses Modell beschreibt die Glukosekonzentration im Blut abhängig vom Vorhandensein der Hormone Insulin und Glukagon. Insulin speichert Glukose in der Leber und in den Muskeln, d.h. die Glukosekonzentration wird reduziert. Glukagon holt die gespeicherte Glukose wieder aus der Leber und den Muskeln und stellt sie dem Körper zur Verfügung, d.h. die Glukosekonzentration steigt. Darüber hinaus wird eine Notfallfunktion des Körpers berücksichtigt: Steigt die Glukosekonzentration über 10 mM, so wird überschüssige Glukose über die Nieren ausgeschieden.

Regulaerer Glukosehaushalt
Insulinbed. Stoerung: Diabetes Typ 1 - eingeschr. Insulinprod.
Insulinbed. Stoerung: Diabetes Typ 1 - keine Insulinprod.
Insulinbed. Stoerung: Diabetes Typ 2
Insulinbed. Stoerung: Hyperinsulinismus

Nach dem Beginn der Glukoseinfusion steigt die Glukosekonzentration, geht aber sofort wieder zurück, sobald die Insulininfusion startet. Nach dem Ende der Insulininfusion steigt die Glukosekonzentration wieder.

Weiters kann beobachtet werden, wie nach der Glukagoninfusion die Glukagonkonzentration steigt, der Graph der Glukosekonzentration darauf keine Reaktion zeigt.

Ohne Insulininfusion und mit einer Glukoseinfusion (mind. 1.3 mmol/min) kann die Glukosekonzentration auf einem relativ 'normalem' Niveau gehalten werden.

Glukoseinfusion [mmol/min]: 1
Glukoseinf. start [min]: 250
Glukoseinf. stop [min]: 750

Figure 3: This Figure shows the content of a web-browser connected to the e-learning interface and in particular to the glucose regulation chapter. In the top frame there is a short introduction (German) to the topic. Aside are links (orange) to eight remote applications (only five are in the picture). The lower frames show the contents of the selected example, which in this case simulates a malfunction of the liver where the glucose level can only be increased by infusion. The lower left frame exhibits another short description (shortened in this picture), three input fields, a button for execution and a link to the source code of the MATLAB program ("view mfile"). The first input field controls the level of glucose infusion and the other fields describe the start- and stop-time of the infusion. On the right side the graphical output of the model is shown. The concentration of glucose is plotted in green, glucagon in red and insulin in blue. Infusions are marked by slashed lines.

We assume that the user has now fully understood the model behaviour respectively the simulated disease and is interested in the implementation of the program. By clicking on the "view mfile" link on the bottom of the left frame, a new Tab opens with the source code of the model (figure 4). The code can be copied to a local desktop environment and be used to extend the program. A student may for example get the task to find (by trial) the lowest level of glucagon effectiveness (parameter $b_{1,2}$) that can provide a normal glucose level when the insulin concentration can take a certain range of values.

4.2 An Example Process of Interaction

The following discussion shall give an overview on a relatively large exercise like a small student-project. All tasks require interaction with the remote system and can take place autonomously as individual exercises usually do.

1. Students are motivated in a first step to read through a textual model description, which states the structure and the main features of the glucose regulation model (Section 2.1).

   Furthermore parameter names and linear dependencies are defined and discussed. Simultaneously a differential equations system (ODEs) is built with these parameters. By this students can retrace the process of modelling and setting up an equation system from a textual description.
2. Students have access to the server and can run the glucose-regulation model via the graphically enhanced web-interface with a few parameters to change but all values in a normal healthy range. This experimenting stage serves as a first introduction to the dynamics of the model.

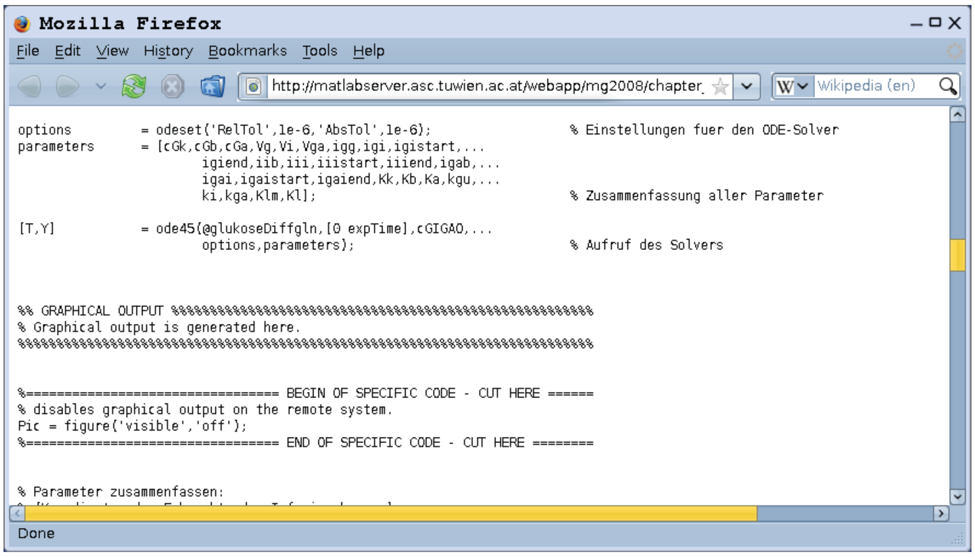
3. In a further task students have to analyse the reaction of the system to certain dysfunctions of the pancreas or the liver such as diabetes I, diabetes II, hyperinsulinism, etc., which are available as separate instances of the model on the server. Furthermore infusion parameters can be modified to simulate countermeasures.

   Here students can understand the models responses to certain inputs and get a deeper insight into the structure and behaviour of this model-example without having to study the source code of the model implementation.

4. Because the ODE-approach allows an equivalent formulation as block-model, students can compare the output of both approaches and find out that a model may allow several equivalent formulations.

5. In a next task students have to work with the source-code of the programme (ODE-implementation as well as SIMULINK block-model). This happens within an exercise, which for example demands to modify parts of the model, such as parameters and functions.

6. An additional task for the students is to add further parameters to the system and run simulations with the extended model. This forces students to get involved more deeply with the MATLAB/SIMULINK language/environment and makes a slow small-step introduction to the programming language through learning by doing.


Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://matlabserver.asc.tuwien.ac.at/webapp/mg2008/chapter
Wikipedia (en)
Done

```
options         = odeset('RelTol',1e-6,'AbsTol',1e-6);                 % Einstellungen fuer den ODE-Solver
parameters      = [cGk,cGb,cGa,Vg,Vi,Vga,igg,igi,igistart,...
                        igiend,iib,iii,iiistart,iiiend,igab,...
                        igai,igaistart,igaiend,Kk,Kb,Ka,kgu,...
                        ki,kga,Klm,Kl];                                % Zusammenfassung aller Parameter

[T,Y]           = ode45(@glukoseDiffgln,[0 expTime],cGIGA0,...
                        options,parameters);                           % Aufruf des Solvers


%% GRAPHICAL OUTPUT %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Graphical output is generated here.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


%================================= BEGIN OF SPECIFIC CODE - CUT HERE ======
% disables graphical output on the remote system.
Pic = figure('visible','off');
%================================= END OF SPECIFIC CODE - CUT HERE ========


% Parameter zusammenfassen:
```

Figure 4: This figure shows the source code of an instance of the glucose-regulation model loaded directly from the server. Since MATLAB is a script-programming language no compilation of code is necessary. The remote runtime environment executes exactly the same code as shown in the web-browser.

5 References

[1] Zauner G., Popper N., Breitenecker F.: *A PHP/MATLAB based e-learning system for education in engineering mathematics and in modeling and simulation*. In: Proceedings of the 6th EUROSIM Congress on Modelling and Simulation, Vol. 2, Ljubljana, 2007.

[2] Internet-reference: The ARGESIM web-server, `www.argesim.org`.

[3] Ernst A., Judex F., Höbarth J.: *Modelling the Human Blood Glucose Regulation – a MATLAB GUI for Educational Purposes*. Simulation News Europe SNE, 16 (2006) 1, 23–24.

[4] Fazekasch M.: *Projektpraktikum aus Techn. Mathematik – Simulation: Glukoseregulation durch die Nieren und den Pankreas*. Documentation of a Work-Study Program, Vienna University of Technology, 2008

[5] Höbarth J.: *System Analysis in Terms of Observability, Controllability and Stability for a Physiological System*. Diploma Thesis, Vienna University of Technology, 2005.

[6] Internet-reference: Wikipedia, `en.wikipedia.org`.