Proceedings of the 17th Conference on Recent Advances in Robotics (FCRAR 2004)

Orlando, Florida, May 6-7 2004

The Making of a "Bio-Systems Modeling and Control" Course

Zvi S. Roth

Florida Atlantic University, Electrical Engineering Department, 777 Glades Road, Boca Raton, Florida 33431
(rothz@fau.edu)

Abstract

The paper discusses the philosophy, design and implementation of a graduate-level course entitled "Bio-Systems Modeling and Control", taught for the first time at FAU in Spring 2004. The course, unlike similar courses in some Biomedical Engineering programs, is aimed at engineering students who may have no prior Controls or Biology background.

The majority of biological and physiological control systems are nonlinear and the control is often accomplished parametrically. Therefore topics selection for the course and their order of delivery pose an educational challenge. The course strongly emphasizes dynamic simulations, initially in Matlab® and Simulink® environments, and later on, a case is made for supplementing it with Dymola® and Modelica® environments. There is a need to synchronize the simulation activities and tutorial examples with the evolving theory contents of the course, which has an impact on the topics order.

Via focusing, in more depth, on the various course topics, the paper outlines specific new concepts that were delivered. These are substantiated by selected homework project assignments.

The course started with the Population Dynamics models. These basic intuitive models formed a good start to convey what Modeling in general is. It also served as a starting point for introduction of the simulation software tools and some of their basic principles.

The next broad areas, albeit being loosely connected thematically, in this order, were Spread of Infectious Diseases and Enzyme Kinetics. With this background in place, students were finally ready to look at the first set of Linear Models, via study of Diffusion effects and Generalized RLC Electrical Models. At this juncture of the course a case for Dymola® , a software environment new to the students, was made.

The course went on to cover detailed physiological models, such as Heart and Blood Circulation, Oxygen Transport in the Blood and Gas Exchange in the Lungs, Cell Volume Control and the Hodgkin-Huxley Nerve Cell Action Potential models, and concluded with brief exposure to Compartmental Modeling and Drug Delivery Control.

This paper is primarily intended as a resource reference for instructors. Instructors who wish to receive a free CD of the course's full set of lectures material and all other course documents are invited to contact the author at the e-mail address shown above.

I. Planning Considerations

The basic introductory Controls theory is more or less the same in all undergraduate engineering control courses. The difference between, say, a Chemical Engineering and Aerospace Engineering versions of classical control, may lie only in the examples and discipline-related case studies.

A typical first control course focuses on linear control analysis and design. Instructors usually "love" to teach Controls, partly because everything there builds up so logically: starting with basic topics such as Transfer Functions, Poles and Zeros, Block Diagram modeling of Dynamic Systems, Feedback and Stability, and thereafter continuing with powerful analysis and control design tools such as Root Locus, Nyquist Stability Criterion and Bode Loop Shaping.

Actually, an educational price is paid for such structure of "easy-leading-to-more-difficult" order of topics. Most examples, that need to match such basic theory, tend to be highly simplified single-loop processes that only idealize reality. Most "real-life" case studies, as we all know, quickly run into multi-input multi-output coupled loops structures and severe nonlinearities, such as saturation, dead zone, backlash, and as such these "difficult" example are deemed to go beyond the scope of a first control course. The neglecting of coupling and nonlinear effects is invariably forgiven, because after all the main objective in a first control course is to introduce students, for the first time in their life, to "systems level" thinking.

In a typical Biomedical Engineering first control course, the examples should all come from the field of Physiology. Biomedical Engineering majors are all expected to have had prior exposure to a few basic Biology and Chemistry courses. Often, by the time that they take a controls course, they may even have been exposed to Physiology and Biochemistry courses. The "real-life syndrome" (those coupled multiple loops and nonlinearities) is particularly aggravated in physiological process examples. Things get even more complicated in such systems, as control is often applied, not in the form of additive signals, but parametrically. Yet again – the emphasis is on the control theory aspects, because this is what may really be new to Biomedical engineering students at that juncture of their education career. Example simplifications, via linearization, and other simplification methods, are willingly being tolerated.

The first task in planning for any new course is textbook selection. A good example to an introductory Physiological Control textbook to Biomedical Engineering students, with such scope and prerequisites, is Michael Khoo's text [1]. The book indeed focuses on Control Design techniques, including even basic System Identification methods, applicable to simplified physiological models. The book emphasizes Matlab® and Simulink® simulations, and provides a full collection of model files that can be easily downloaded from the web. Yet, I felt that this book may be unsuitable for students who must first master some of the Biology and Physiology concepts. Khoo's book may, on the other hand, be a perfect choice for a second (follow-up) course that might emphasizes Controls over Modeling.

In other engineering fields, students may not have a broad biology and chemistry background. Most engineering students nowadays have to take a single course in Chemistry, and often no courses at all in Biology. This paper discusses the development of a Bio-Systems Modeling and Control course, for students who have had no Biology background, and no prior Controls background. The only assumed prerequisite is a course in ordinary differential equations, not so much for the specific details that may or may not be covered in such a course, but rather to assure some minimum level of mathematics sofistication.

The textbook selected for the course was the one by Hoppensteadt and Peskin [2]. This wonderful little book covers what appears to be a random collection of topics in Physiology. It starts with Heart and Blood Circulation modeling. Then it goes on to other organs – Lungs, Kidneys and so on. The most amazing aspect of the book is its very realistic Matlab® simulators, available (again via downloading from the web) at the end of each chapter. The simulators allow the students to play "what-if" scenarios, introducing disease effects and so on. Even though that the authors of the book promise that the book is organized to go from "light-to-heavy" topics, starting the Bio-Systems Modeling and Control course with steady-state blood flow models and their electrical resistance-capacitance analogs, presented a formidable educational dilemma. I felt pretty strongly that the Biology, Physiology and Simulation Tools topics need to build up to a point where a discussion of specific organ models will come naturally. This paper covers in part some of the ideas of such preparatory topics.

This beginning-level graduate or senior-level undergraduate course, was designed for 15 weeks per semester and three weekly delivery hours. The course's grade was determined based on students' scores in weekly simulation projects (13 projects altogether) with no exams. Graded projects were allowed to be resubmitted once, corrected according to instructor's comments. The course was assisted by a Blackboard 6.0 web-instruction platform. The course's Blackboard site contained all PowerPoint lecture slides, all other handout material and resource documents (i.e. simulation software tutorials). It also contained an on-line grade-book, and a bulletin board on which instructor's announcements, homework hints and other useful information got posted.

I really wanted the course to be "a lot of biology" and only "a little bit of control".

II. List of Course Topics and their Order

The following is a list of course topics that were finally selected, and their order of delivery. Roughly speaking, following the course's textbook started with Heart and Blood Circulation models, only around the middle of the semester, after much other ground was covered.

1) Population Dynamics Part I (Logistic Models)
2) Population Dynamics Part II (Extinction and Predation)
3) Population Dynamics Part III (Delay Logistic models)
4) Population Dynamics Part IV (Predator-Prey models)
5) Population Dynamics Part V (Competition and Symbiosis models)
6) Linearization
7) Models for Spread of Infectious Diseases
8) Enzyme Kinetics Part I (What are enzymes?)
9) Enzyme Kinetics Part II (Chemical Reaction Rate)
10) Enzyme Kinetics Part III (Basic enzyme reactions)
11) Enzyme Kinetics Part IV (Cooperative Enzyme phenomena)
12) Enzyme Kinetics Part V (Activation and Inhibition)
13) Linear Biological Models 1 (Diffusion)
14) Linear Biological Models 2 (Generalized Electrical Models)
15) Heart and Blood Circulation I (Basic Circulation Concepts and Parameters)
16) Heart and Blood Circulation II (Heart Basics)
17) Heart and Blood Circulation III (Steady-State Flow-Pressure-Volume Relationships)
18) Heart and Blood Circulation IV (Baroreceptor Loop)
19) Heart and Blood Circulation V (Dynamic Models)
20) Heart and Blood Circulation VI (Hoppensteadt-Peskin Simulator
21) Heart and Blood Circulation VII (Control of systemic resistance)
22) Lungs Models I (Gas Exchange in a single Alveolus)

23) Lungs Models II (Gas Exchange in the Lungs – overall)
24) Lungs Models III (Hoppensteadt-Peskin Lung Simulator
25) Compartmental Modeling I (Preliminary concepts
26) Cellular Homeostasis I (Osmosis)
27) Cellular Homeostasis II (Cell Volume Control)
28) Compartmental Modeling II (Drug Delivery Modeling)
29) Cellular Homeostasis III (Ions Movement through Cell Membrane)
30) Cellular Homeostasis IV (Interaction of Cellular Electrical and Osmosis effects)
31) Course Cellular Homeostasis V (Hodgkin-Huxley Model)

III. Course Topics Discussion

Despite the nonlinearity of most population growth models, I strongly felt that the topic of Population Dynamics was a perfect opener for the course. The most immediate tasks that the instructor faces, at this phase of the course, are: a) teach the necessary simulation tools needed for the weekly projects, using examples that go along with the evolving course material, and b) convey the basic ideas of what "dynamic models" in general really are.

Population growth models are highly intuitive, and can be made progressively complex. One can start with linear (exponential) growth and decay models, and then progress to the Logistic model in which species' "carrying capacity" caps the growth. Students need to intuitively feel that more and more model terms can, on one hand, be added, to assure model plausibility, or be taken off, to achieve better simplicity. Model plausibility built based on a given data set is best judged by least squares curve fitting (see Appendix A). The logistic model can thereafter be augmented with either an "extinction" multiplicative term or "predation" additive terms of one kind or another.

Even at this early stage of the course, it is not too early to teach concepts such as "equilibrium point and its stability" and "phase plane plot", all are easy to plot and visualize for first-order systems. It is somewhat strange that the first control systems topics are such that are typically covered in a second control course, devoted to nonlinear dynamics. Yet, it is at this point in the course way too soon to even talk about the first linear model.

Adding time delay effects to single-species models illustrates to the students, for the first time, what a "limit cycle" is. It also serves a simulations-related purpose (related to selection of simulation parameters) that is elaborated further in the next section (and in Appendix C).

Multi-species theory starts with the classic Lotka-Volterra predator-prey model. Excellent references on this subject are [3,4]. The first leans in the direction of analytic solutions, but it is very comprehensive. The latter contains some outstanding suggestions for homework assignments.

Predator-prey models pave the way to Competition and Symbiosis models. All these models require more careful analysis of parametric dependence of the multiple equilibrium points, and their stability. Much of that can be accomplished numerically with built-in Matlab® features, however some hand calculations may be necessary to acquire "global understanding" of a given model. In Appendix D it is demonstrated that the running of a simulation model with a fixed set of parameters, or even with a family of sets of parameters may not be a substitute for some analytical analysis.

Murray's book [3] strongly advocates Model Normalization. Indeed many believe that this is one of the more crucial aspects of modeling. Normalization consists of time scaling and variables magnitude scaling, so that all problem variables (including time) become unit-less, and spread conveniently (for simulation purposes) within or near the interval [0,1]. Furthermore, normalization allows system analysis in which the number of relevant parameters is reduced to the very minimum. While it is good that students begin to recognize some common normalization tricks, it is to be understood that very often a normalization process can be very tricky and unobvious. It was decided, in this course, only to touch upon the subject, without ever delving too deeply into it.

The Competition and Symbiosis models topic is a good place in the course to cover the controls topics of "Linearization" and "Lyapunov's First Method". Students in all first control courses often have problems understanding what linearization really means, and what is it good for. Simulation assignments such as "How large is the range for which a linearized model is valid" often receive wrong treatment, sometimes due to "comparing apples to oranges" (see Appendix E)

The next course topic "Spread of Infectious Diseases", at its simplest form (the Susceptible-Infected-Recovered (SIR) models) [5], appears (on the surface) to introduce no fundamentally new theoretical or simulation principles. It appears to just give the student a chance to practice once again prior ideas – models for interaction between Infected and Susceptible are after all not conceptually different from interaction between predators and preys. Even the simulation assignment (Appendix G) appears to presents no new concepts (as it deals with a search for population vaccination strategy that will cause disease eradication in a specific prescribed time). The SIR model assignment does however allow the class to witness hands-on the merit of partial model normalization (i.e. a one in which no attempt is made to reduce the number of parameters to a minimum. Parameters are only "non-dimensionalized", to make simulations easier, in the sense of avoiding dealing with population numbers measured in millions). It is always fun to tie simulations to real data (such as that available regarding the Smallpox epidemics some years ago). There is however a new twist: For the first time, students

meet an equality constraint condition that ties the model variables to each other (i.e. in the SIR model the three population portion add up to the total population size). In a simulated model, attaching an integrator to each variable is a wrong approach. In the basic SIR model there are only two independent variable, and an algebraic constraint block (see Appendix G).

The transition from the Population Dynamics subject to that of Enzyme Kinetics is quiet natural, as species interaction and Chemistry's law of mass-action are mathematical synonyms. Here again, students meet algebraic constraint relationships. This topic is students' first exposure to Physiology. The new theoretical twist is the exposure to one of Biochemistry's most fundamental theoretical achievements – Michaelis-Menten theory [6-8]. Enzyme Kinetics is characterized by two process phases, occurring over drasticallly different time scales. There is a very fast enzyme-substrate complex formation, followed by a much slower product formation, during which some of the concentration maintain a state of "quasi-equilibrium" (that is, some of the variables begin to obey a nonlinear equality constraint, decaying together "hand-in-hand"). See Appendix H, for a basic Enzyme Kinetics assignment, that surprisingly proved to be very difficult for most students.

Enzyme Kinetics in its more advanced forms involves Cooperation and Inhibition (Competition and Allosteric) phenomena. These give students, for the first time, a glimpse at some important feedback control mechanisms that exist at the molecular level. I however had a hard time finding reasonably easy homework project assignments, to properly cover these issues. The assignments in [6] all looked way too advanced for my course. More about this issue will be discussed in the next section, in conjunction with capabilities of the Dymola® software.

It was almost the middle of the semester when we finally reached some mainstream linear modeling ideas. In some Biomedical Engineering Control Modeling courses, the topic of Electrical Analog models, is one of the first to be covered. Khoo's book [1] indeed starts with this topic. A reason for my decision to delay the delivery of Linear Biological Models for so long, had to do with my earlier plan to tie this topic to the Dymola® simulation software.

Diffusion, in its simplest form (that is, Fick's Law describing diffusion flow proportional to the concentration difference) often leads to linear models. It presents a natural transition from the topic of Enzyme Kinetics to tie mass-action law to Fick's law, in describing substance transfer problems occurring at the cell level. Discussion of Diffusion also serves to introduce students to the concept of "resistance to flow".

Representing mechanical, thermal, fluid and chemical processes by means of electrical analogs – generalized voltages and currents, and consequently – generalized resistances, capacitances (compliances) and inductances (inertances), is most appealing to electrical engineers, and very useful in general. It is intuitive (to electrical engineers) that, for instance, in a complex RC circuit, a dynamic model analysis starting point must be at the capacitors' voltages. These voltages are the circuit's "states" (variables that can be independently initialized). Transferring such a concept to other physical domains can help giving users a more systematic approach to the analysis problem. The dilemma of "should electrical analogs be used if simulation models are available to other physical systems" is discussed further in the next section, in conjunction with Dymola®, and in Appendix I.

At this point in the course, lectures finally began to adhere more closely to the required textbook [2]. We were able to cover only the first three chapter of the book (which is a little less than 50% of the book's material). The discussion of the various organs models was greatly assisted by some wonderful pictures from [9-10]. The model presented in chapter 1 of [2] for Heart and Blood Circulation is a simplified resistance-compliance model of the heart and groups of blood vessels. Students become exposed, for the first time, to steady-state models. A great deal of the homeostatic control mechanisms can be revealed simply by studying of sensitivity functions derived from the algebraic steady-state flow-pressure-volume relationships. The steady-state models pave the way to control ideas, such as the "Baro-receptor Loop" and "Resistance Control at the tissue level". Students also learn how the dynamic relationships are to be added, laying the ground for Hoppensteadt-Peskin Heart and Blood Circulation Simulator. The simulator and its educational relevance are discussed in the next section, and in Appendix J.

Chapter 2 of [2] deals with Gas Exchange in the Lungs, and here the students get to see a connection to the previous topic of Enzyme Kinetics, via the concepts of "partial pressure" and "oxygen-hemoglobin dissociation curve". Like any of the textbook's chapters, students are given with a simulator (a lung simulator, in chapter 2) that allows them to play "what-if" scenarios (for instance, study the effects of elevation changes on the oxygen concentrations in the blood).

Chapter 3 of the text introduces the students to the topic of Osmosis and to the topic of Ions Transfer through Cell Membrane, eventually leading to Cell Volume Control and to the celebrated Hodgkin-Huxley Nerve Cell Action Potential model. Appendix K deals with one of the students' favorite activity – Hoppensteadt-Peskin's Hodgkin-Huxley simulator.

The course ended with very brief control-oriented exposure to Compartmental Modeling (mainly along the lines of [11]). The need for compartments model identification, via controlled drug infusion or bolus injections is explored in Appendix L. All in all, an attempt was made to thematically tie all course topics to one another in a logical continuous stream.

III. Dynamic Simulation Instruction

One may opt to start a course like that with a sequence of Matlab® and Simulink® tutorials, prior to covering any Bio-Systems theory, as in [12]. After all we need to provide the students with all the tools that they may need for their weekly

assigned simulation projects. The problem with such an approach lies in the quality of examples that can be shown at such an early stage of the course. A better approach may be to interlace the evolving theory with simulation tools tutorials that progressively become more and more involved. The examples for the simulation tutorials can be ones that are directly relevant to the theory which is being covered at that time. Such a mix of theory and simulation tutorials was done in the first few weeks of the course. Thereafter (with the exception of the Dymola® software [13], that was introduced near the middle of the course) the course proceeded along mostly theoretical lines.

A lot of ground must be covered in the first few weeks of the semester. Most of the basic principles of Simulink® modeling can very adequately be done with single-species examples. Students must learn how to look for suitable parts in the various components libraries, how to lay out the block diagram, how to use integrator blocks as the central dynamic elements, and how to select basic output devices (that is "sinks", such as a Scope or a XY-Graph). At this early stage, it is not necessary, to delve too deeply into simulation parameters issues, such as selection of a numerical integration method. Basic knowledge of Simulink is attained even if student is only able to initiate a single-run. Any parametric analysis, of the early models, can adequately be done by manual varying of the parameters, by manually conducting multiple runs, and by manual data collection (via observation of scope outputs).

The students, a little bit later on, learn how to set up the simulation diagrams more efficiently. A "Fcn" block from the User-Defined Functions library, combined with a multiplexer block "Mux" from the Signal Routing library, allow the user to create complicated multivariate nonlinear functions in a single block, in which the nonlinear function is written algebraically as a text. At this point, students learn about the merit of keeping track of the different model parameters and initial conditions (so, for instance, a set of multiplier blocks working in conjunction with a constant blocks, may be superior to using a transfer function block, in which parameters can be changed only manually).

The next phase in enhancing students' Simulink® simulation skills starts when they learn how to conduct measurements in a more automated manner. For instance, if an analysis requires that a certain time duration be measured (say, a "settling time"), then there is a need to conduct such a measurement using an enabled sub-system block (see Appendix B).

One more important skill, is to teach students how to control the execution of Simulink® from a Matlab® m file (using the "sim" and related commands. Again, refer to Appendix B). There are numerous benefits to running Simulink® from Matlab®: (a) ability to plot family of phase-plane plots (in the cases of single or two species models), (b) ability to plot 3D trajectories (in cases of three species models), (c) ability to search the parameters space for specific parameters that cause the response to have certain specific values at certain specific times (as in Appendix G).

One of the reasons for studying the effects of pure time delay on single species models, this time around from a simulations skills instruction viewpoint, comes from exploiting the severe numerical sensitivity of systems with delay. A pure time delay block, even one that has a sufficiently large buffer size, can cause numerical stability and accuracy problems. The user must, in such cases, navigate carefully among the simulation parameters (particularly the "Relative tolerance" parameter). The key problem is to sense when is it that the simulation program may be "cheating" - producing erroneous simulation output. This is the point where benchmarking, utilizing models with known analytic solutions, or other known analytic solution features, becomes important (see Appendix C).

In conjunction with the study of multi-species models, students must learn the concept of "state". The number of integrators in the model equals the number of independent variables. (Appendix D contrasts a three-species model with no constraints to a model with constraint). Algebraic constraints can be handled by an algebraic constraint block, from the Math Operations library, a new feature of Simulink® Version 5.1.

The Enzyme Kinetics activities (Appendix H) exposes students, for the first time, to the potential use of stiff numerical integration methods. Most students, who took the course in Spring 2004, have missed this point.

Modelica® is a modeling environment in which sub-systems of potentially different physical nature are interconnected in a "electrical-circuit-like" manner [13]. The model is similar, in spirit, to that of an electronic SPICE model. In a SPICE model, every component is described by its nodes location, by its type and values. This method of modeling may have a big potential advantage over the "Simulink® paradigm" (in which everything evolves around the model integrators), in the sense that coupling effects among different loops do not need to be explicitly mathematically modeled. Dymola® provides the simulation engine to run the freeware models available from the Modelica® organization.

Such a case for Dymola/Modelica is featured in Appendix I, in which an electrical analog model is used to describe a simplified Respiratory Mechanics (an example taken from [1]).

The key new idea of the new dynamic simulation paradigm presented by Modelica/Dymola is library sharing. Modelica® is a large and continually expanded collection of dynamic models of all kinds – mechanical, fluid, thermal etc. In principle, the electrical analog of Appendix I is not really necessary, as one may opt to simulate the model, directly using hydraulic or pneumatic components. This however has not yet been exploited in the course, but it will be studied in future versions of the course.

The Modelica® Biochemistry library is still in development and as such not yet available to users in a "downloadable" form. It is however available in text form via [14]. Once such a tool becomes available, instructor would be able to assign realistic multi-step Enzyme Kinetics simulation assignments.

The last simulation skill that students learn in this course is the use of fully available models. The textbook by Hoppensteadt and Peskin [2] provides very detailed, and fully annotated, set of Matlab® simulators, for the various organs

or physiological sub-systems discussed in the book. Students are encouraged, via numerous suggested simulation exercises, to modify the basic software code in an attempt to explore effects of disease or other environmental influences. (See Appendices J and K). Some modifications, the easier ones, are nothing more than searching for different constant values. More complex code modifications may involve replacement of code lines. These simulators are a great finale for the course – students really have a good time playing with it.

IV. Conclusions and Future Work

There is a need for a "bridge course" in Biology for engineering students. A new text [15] aims at filling this void. Here is a quote from the book's preface: "… ([15]) focuses narrowly on what we perceive to be the essentials of New Biology, namely, genes and proteins, cells as basic units of life, cell division, and animal development… ([15]) introduces cells as robust complex networks of genes and proteins and adopts a systems view to discuss communication of cells with other cells and with the external environment". In this context, my course falls mostly on the "Old Biology" side…

As a lot of biological ground was covered in the course, it should be acknowledged however that much more has not been covered: Kidneys, Muscles, Hormones and Endocrine Systems, the Eye, the Ear, just to name a few, and most importantly all DNA and gene-related topics. So at this point it is time to ask a question that should have normally appeared at the paper's opening paragraphs "What are the course's goals?".

I shall start from a personal perspective. Teaching a course that does that much Biology and Physiology is a steep learning curve for an engineering professor with my background. I believe that I am not alone. Other faculty members, especially within the systems and control research community, may follow my steps and become, like me, more aware and more attuned to biological issues.

More and more engineering leaders believe that a new era of Biotechnology and other biological applications, may force all engineering disciplines to enhance students' biology background. It is important that more and more engineering-oriented biology courses be developed. At FAU several faculty took the initiative of doing just that. In the last year several new engineering courses were developed and offered in the areas of BioInformatics, Bioengineering, Biomechanics, Bioflow, Biometrics, and more are still forthcoming. My course fits this new trend and it should be followed up by more bio-modeling courses (some may aim at "New Biology") and more bio-control courses.

V. Acknowledgements

I should express deep gratitude to the many individuals who contributed to the making of the Bio-Systems Modeling and Control course. Special thanks to my department chair Dr. Salvatore Morgera who initiated the idea, encouraged me to develop the course and provided me with much needed release time for that. Dr. Michael Khoo (USC) kindly provided me with his PowerPoint® lecture slides, of which I gladly used the material for the Linear Biological Models. Much can still be used in a future follow-up controls course. My daughter Nitzan Roth (a medical student at USC) helped me with many useful brainstorming discussions, reviewed some of the lectures and offered several good suggestions for homework projects (such as the one in Appendix G about infectious diseases). Some of my students in the course worked hard on the solutions, and some of their effort is reflected in the appendices. Appendices A-B,D-F and H-J use material from Meta Leesirikul's solutions. Appendix G uses Daryl Sahai's solutions. Appendix K uses Jacob Mihalak's solution and Appendix L uses Michael Fields's solutions..

References

1. Michael C.K. Khoo, "Physiological control Systems – Analysis, Simulation and Estimation", IEEE Press 1999.
2. Frank C. Hoppensteadt and Charles S. Peskin, "Modeling and Simulation in Medicine and the Life Sciences". Second Edition, Springer 2001.
3. J.D. Murray, "Mathematical Biology", Springer-Verlag 1989.
4. Edward K. Yeargers, Ronald W. Shonkwiler and James V. Herod, "An Introduction to the Mathematics of Biology, With Computer Algebra Models", Birkhauser 1996.
5. Nicholas F. Britton, "Essential Mathematical Biology", Springer 2003.
6. James Keener and James Sneyd, "Mathematical Physiology", Springer 1998.
7. Arthur C. Guyton and John E. Hall, "Medical Physiology" Tenth Edition, W.B. Saunders Company 2000.
8. David Randall, Warren Burggren and Kathleen French, "Eckert Animal Physiology – Mechanisms and Adaptations" Fifth Edition, W.H. Feeman and Company 2001.
9. Valerie C. Scanlon and Tina Sanders, "Essentials of Anatomy and Physiology", Fourth Edition, F.A. Davis Company 2002.
10. Geoffrey M. Cooper, "The Cell – A Molecular Approach", ASM Press 1997.

11. Robert B. Northrop, “Endogenous and Exogenous Regulation and Control of Physiological Systems”, Chapman & Hall / CRC Press 2000.
12. James B. Dabney and Thomas L. Harman, “Mastering Simulink”, Prentice Hall 2004.
13. “Dymola – Dynamic Modeling Laboratory: User Guide” Version 5.1b, Dynasim AB 2003.
14. Emma Larsdotter Nilsson and Peter Fritzson, “BioChem – A Biological and Chemical Library for Modelica”, Proceedings of the 3rd International Modelica Conference, Linkoping, November 3-4, 2003.
15. Aydin Tozeren and Stephen W. Byers, “New Biology for Engineers and Computer Scientists”, Pearson / Prentice Hall 2004.

Appendix A: Modeling via simple curve-fitting

The following assignment, regarding the US Census data, was offered in [4]:

1790	3,929,214	1860	31,433,321	1930	122,775,046
1800	5,308,483	1870	39,818,449	1940	131,669,275
1810	7,239,881	1880	50,155,783	1950	151,325,798
1820	9,638,453	1890	62,947,714	1960	179,323,175
1830	12,866,020	1900	75,994,575	1970	203,545,805
1840	17,069,453	1910	91,972,266	1980	226,545,805
1850	23,191,876	1920	105,710,620	1990	248,709,873

Students were asked to determine the period of exponential growth (and its growth rate parameter), and to determine whether or not the data hints to any population logistic growth capping . The first part presented no problem to the students (who used Microsoft Excel® or Matlab® to do both the plotting and least-squares curve-fitting:
:

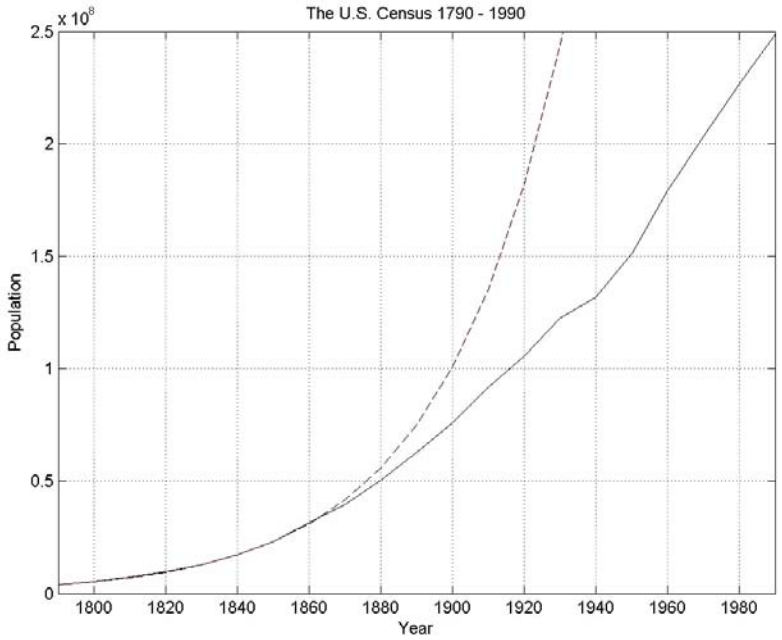


Fig A.1 Actual U.S population vs. best-fit exponential growth model
Fitting a logistic model was more of a challenge. Some of the students tried to fit the “best” linear per-capita growth rate curve to a set of very noisy data:

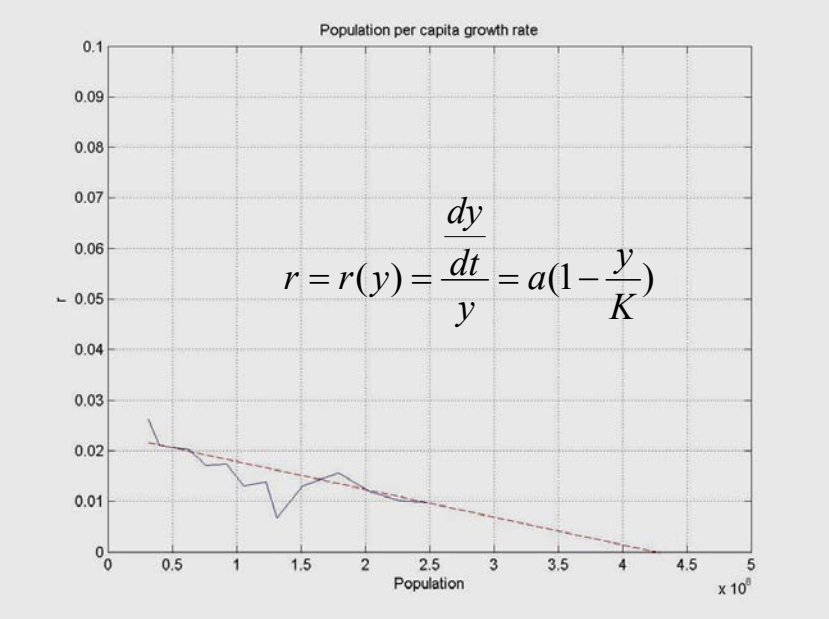


Fig A.2 Carrying capacity estimation via linear curve fitting to a finite-difference approximation of the per-capita growth rate, revealing a 420,000,000 estimated carrying capacity of the US population.

Appendix B: Automated Time Measurements Data Collection in Simulink®

In the following Simulink® model, of a normalized logistic model, a simple sub-system, is enabled only when the enabling input signal is positive. This, and the control from Matlab®, aid in the automated data collection for a 2% settling time as function of the system's initial condition.

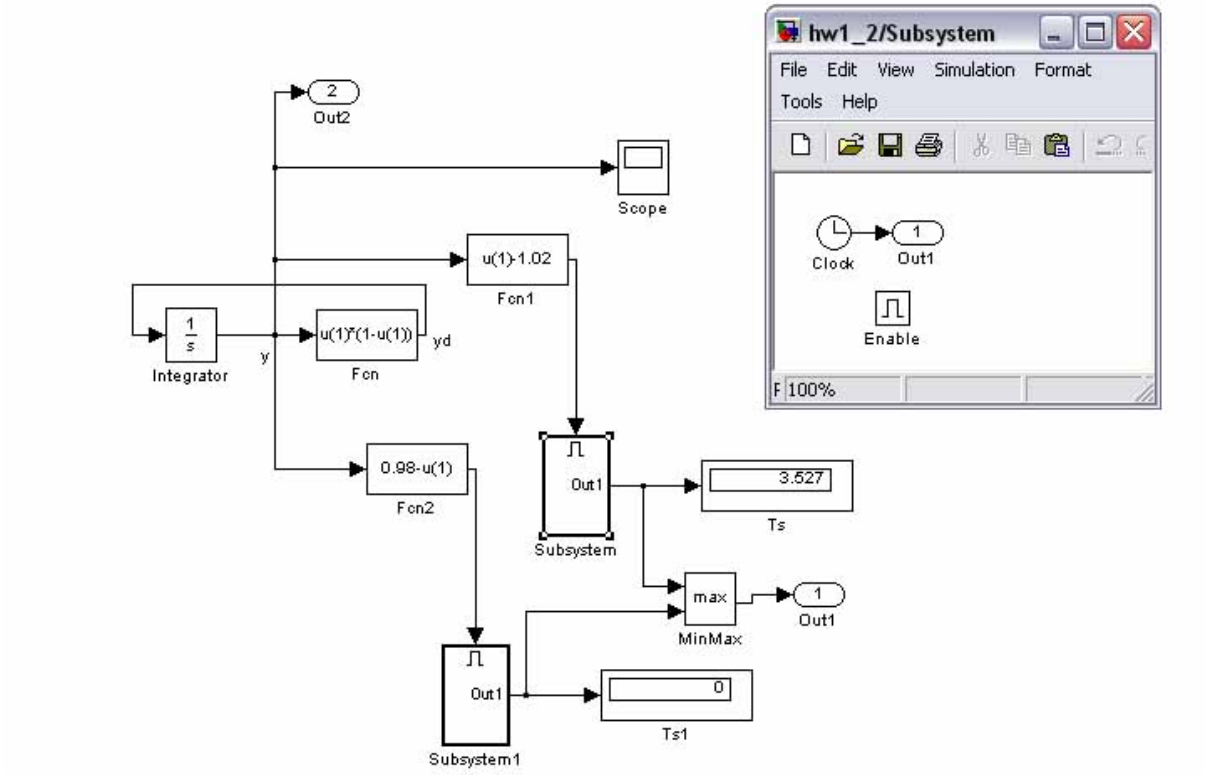


Fig B.1 Simulink® main and subsystem diagram for measuring a 2% settling time of a normalized logistic model.

A Matlab® control code allows the plotting of the settling time as function of the initial conditions:

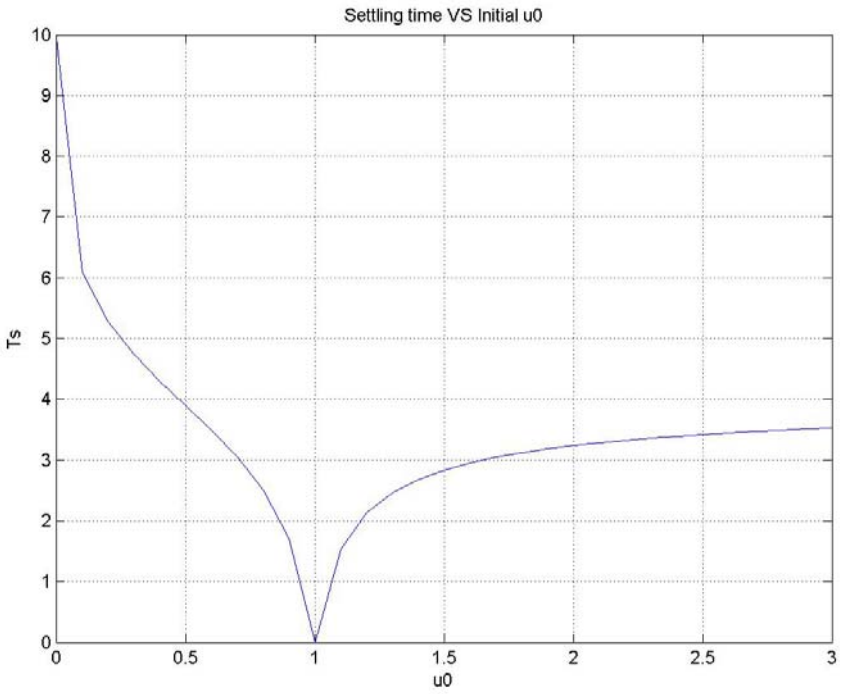


Fig B.2. Plot of settling time vs normalized initial condition

MATLAB Code:

```
clc;
clear all;
color=['b','g','r','c','m','y','k'];
i=1;TSindex=1;
for u0=0.0:0.1:3;                          % varying the u0
   opts=simset('InitialState',u0);
   sim('hw1_2',10,opts);                   % run simulink
   plot(tout,yout(:,2),color(i));          % plot u VS time
   grid on;
   hold on;
   xlim([0,4]);
   xlabel('time');
   ylabel('u');
   [m,n] = size(yout)
   TS(TSindex)=yout(m,1);
   i=i+1;
   TSindex=TSindex+1;
   if i>7
      i=1;
   end
end
figure;
u0=0.0:0.1:3;
plot(u0,TS);                               %PLOT ts VS initial u0
grid on;
xlabel('u0');
ylabel('Ts');
title('Settling time VS Initial u0');
```

Appendix C: Simulation Parameters Selection

The following normalized logistic model, with pure-time delay is analyzed in [3]:

$$\frac{dx(\tau)}{d\tau} = x(\tau)[1 - x(\tau - D)]$$

A useful "benchmark" result is that "a limit cycle develops if the normalized time delay D>π/2".
The Simulink model is shown below for D=1:

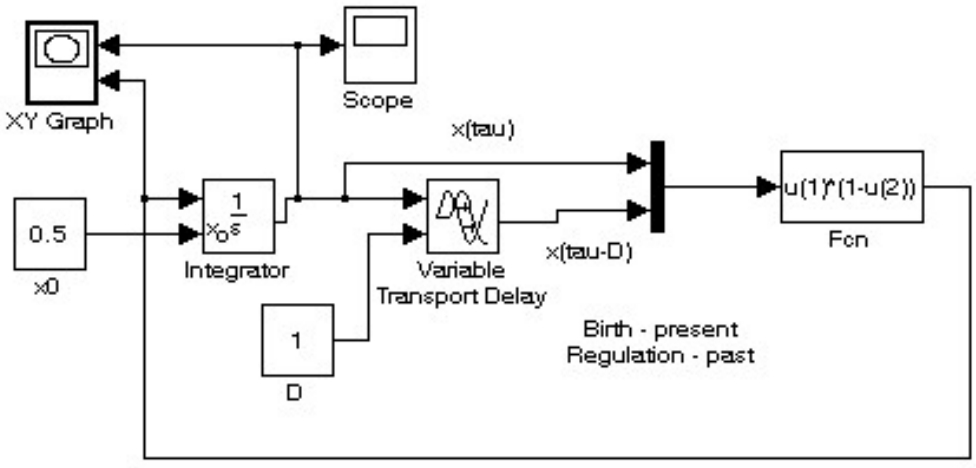


Figure C.1. A logistic model with delay.

A simulation for D=1.6, using default simulation parameters, produced the following transient plot:

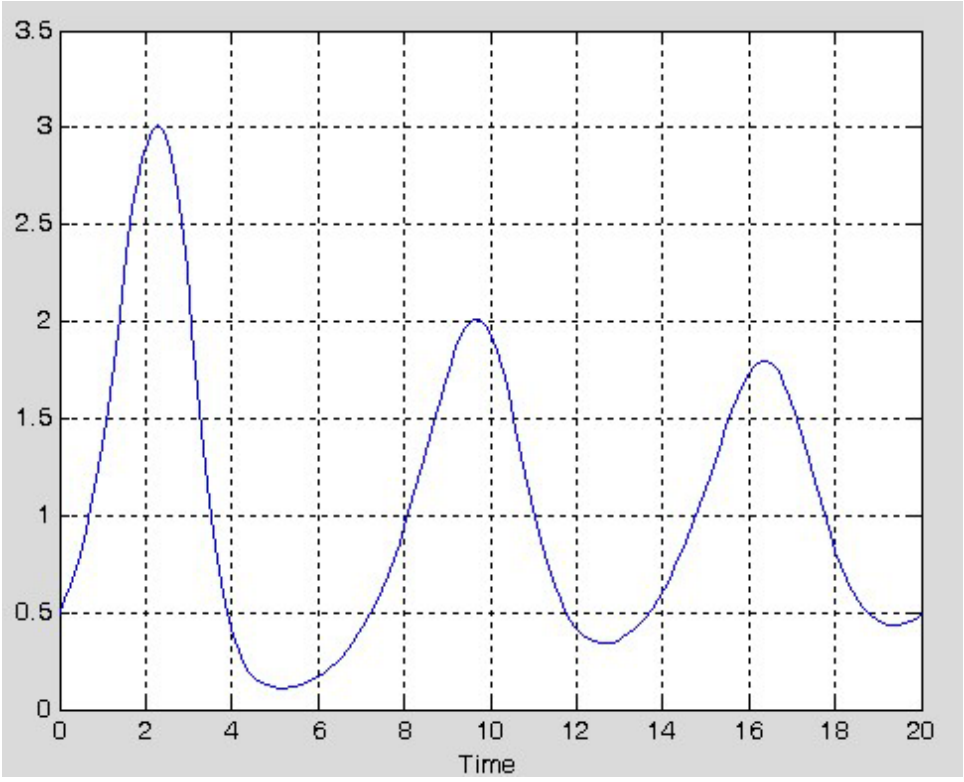


Figure C.2. population transient – need to increase simulation final time.

As the final time is increased, actual erroneous outputs arise showing a normalized population convergence to 1. There is a need to increase the "Refine Output" parameter, increase as needed the delay block's buffer size, and most importantly reduce the "Relative tolerance" parameter by orders of magnitude, before the correct periodic solution is revealed.

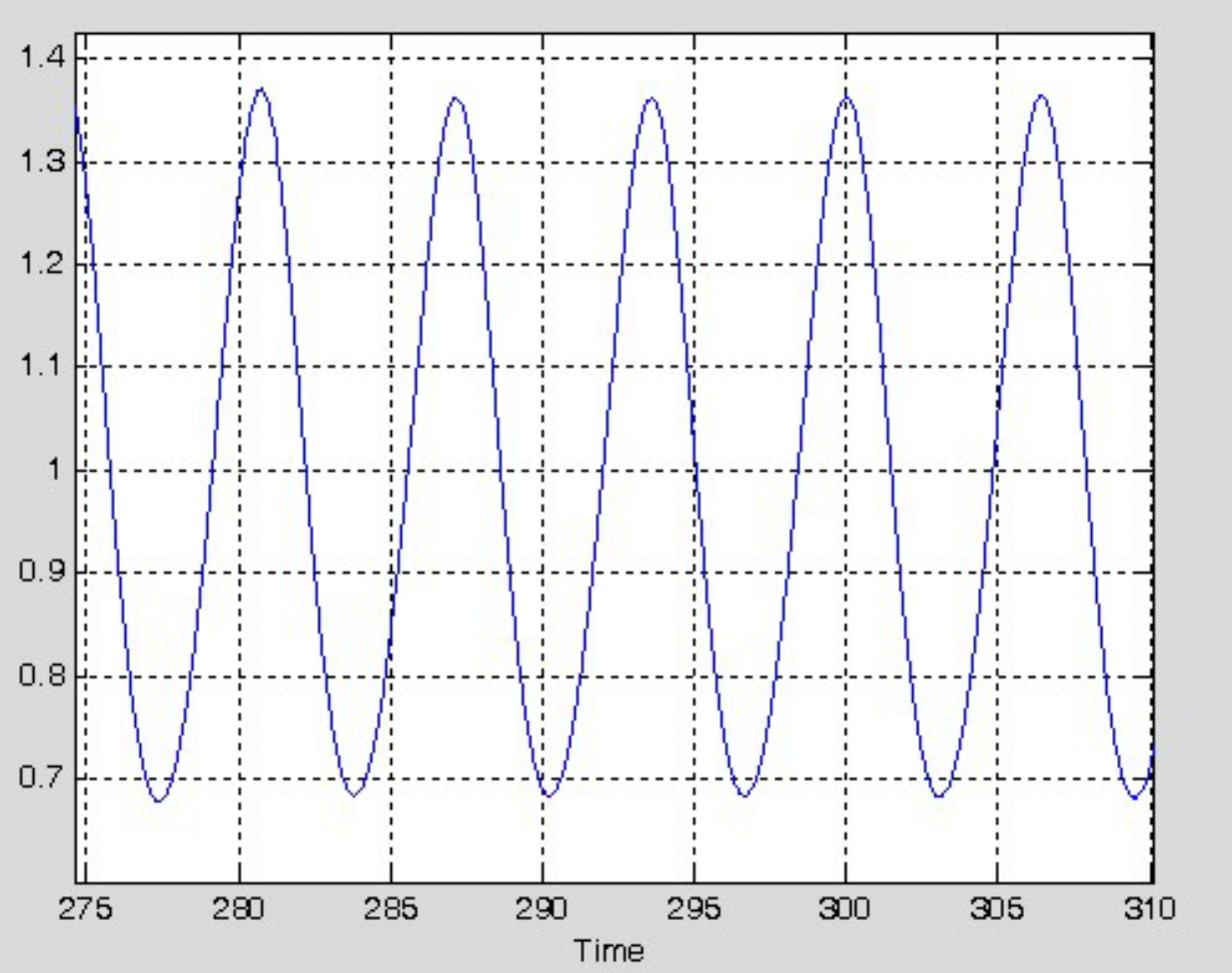


Figure C.3. Population periodic solution.

Appendix D: Overall understanding of a given model's performance

The following example is taken from [3]. A two-species model was given, and students were asked to determine what type of a model is it (Competition, Symbiosis, etc)

$$\frac{dN_1}{dt} = r_1 N_1 (1 - \frac{N_1}{K_1 + b_{12} N_2})$$

$$\frac{dN_2}{dt} = r_2 N_2 (1 - \frac{N_2}{K_2 + b_{21} N_1})$$

What most students did was to pick up some sample values for the coefficients, run a Simulink® simulation, observe the result and draw some conclusion. Some students used Matlab® built-in "trim" command (for finding the equilibrium points of a given sub-system) and "linmod" command (for linearizing a model about a specific equilibrium point). The solution below does the analysis via more comprehensive varying of the system's parameters:

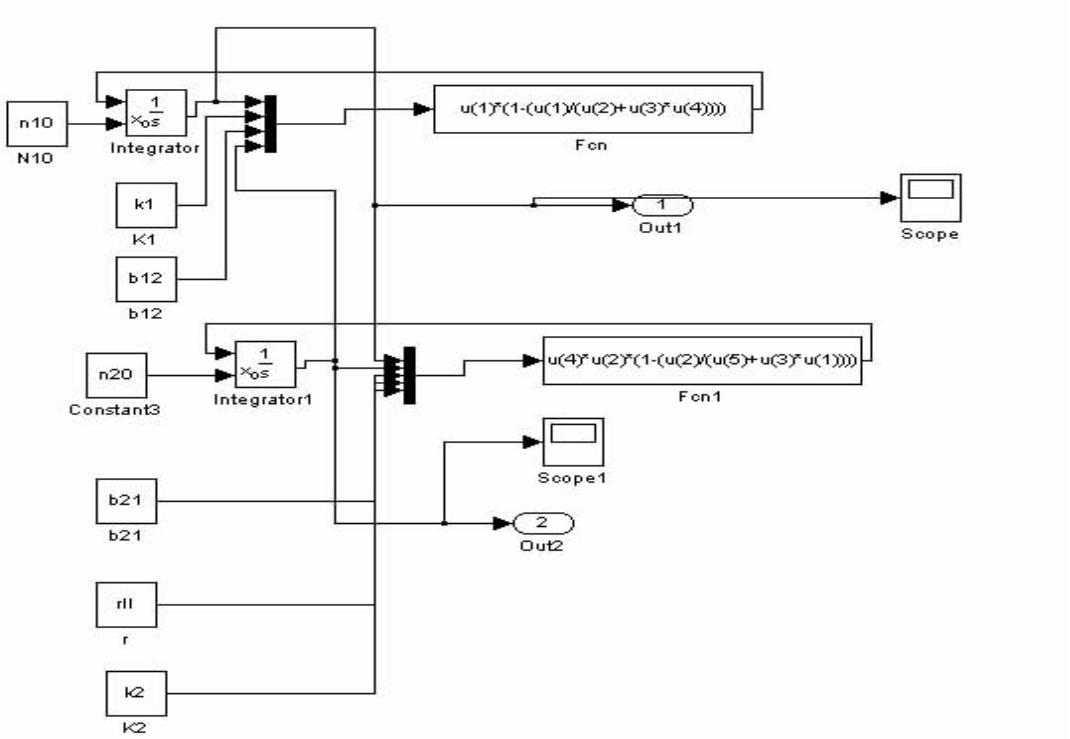


Fig D.1 Simulink Model of the "mystery model", allowing for parametric variation via Matlab®.

MATLAB program

```
clc;
clear all;
b12=0.1;
b21=0.5;
n10=1;
n20=1;
rll=3;
k1=5;
k2=5;
index=1;
fact=0;
for x1=0:0.2:1.5
   for x2=0:0.2:1.5
      temp=trim('hw4_2',[x1;x2]);
      temp=round(temp*10)/10;
      if index~=1
         for ind2=1:1:index-1
            if temp.'==xeq(ind2,:)
               fact=1;
            end
         end
      end;
      if fact==0
         xeq(index,:)=temp.';
         index=index+1;
      end;
      fact=0;
   end;
end;
xeq
```

```
for ind2=1:1:index-1
   a=linmodv5('hw4_2',xeq(ind2,:));
   eig(a)
end;
sim('hw4_2',30);
plot(tout,yout(:,1));
hold on;
plot(tout,yout(:,2),'r--');
figure;
plot(yout(:,1),yout(:,2));
```

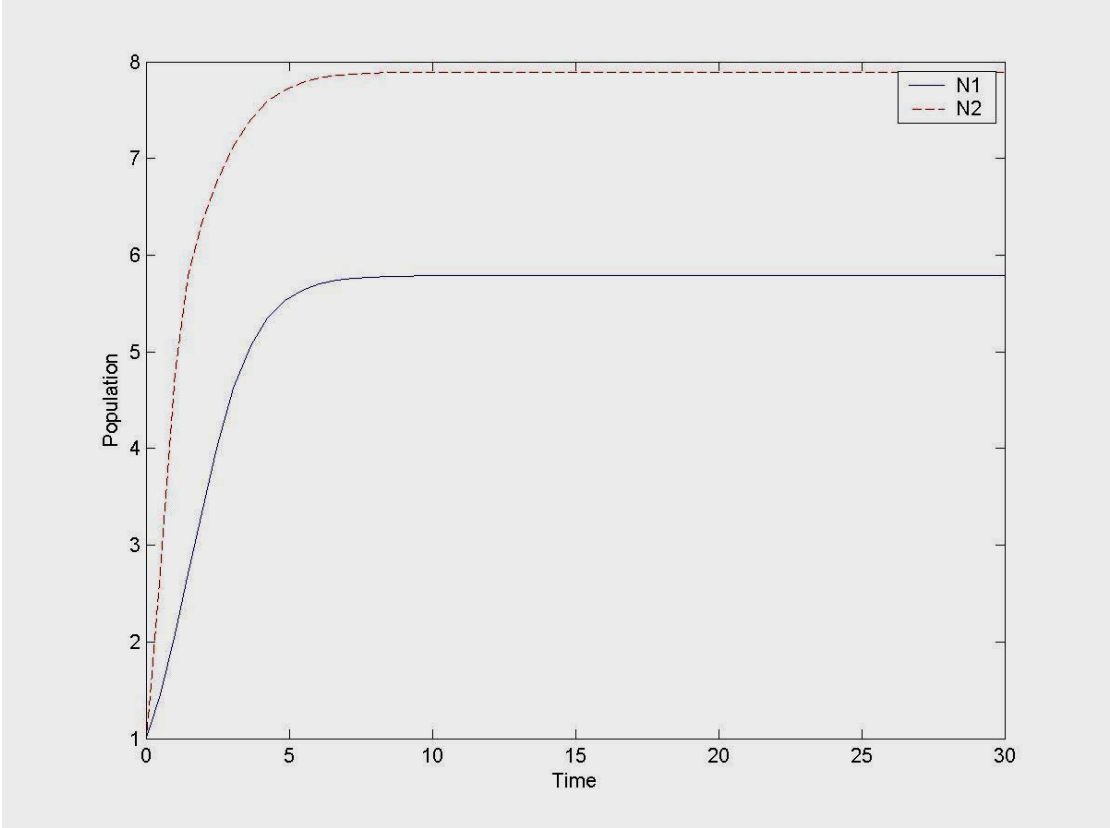


Fig 2.2 Typical simulation result revealing the symbiotic relationship among the variables.

From the simulation result, N_1 and N_2 steady state values exceed each species individual carrying capacity (taken as 5 in this simulation). Species help each other reaching higher levels in a symbiosis model. What most students did not do is some algebraic hand calculations, to see how the number of equilibrium points and their type depend on the coefficients. Overall understanding must sometimes require an algebraic approach.

Appendix E: Understanding what Linearization means

Some students, when asked to assess the validity of a linearized model, look at both the nonlinear model and its linearized model, side-by-side, forgetting that needs to be compared are the variable deviations. The model shown below (for a logistic model) does it right – the simulated nonlinear model is that of the nonlinear deviations from the equilibrium state.

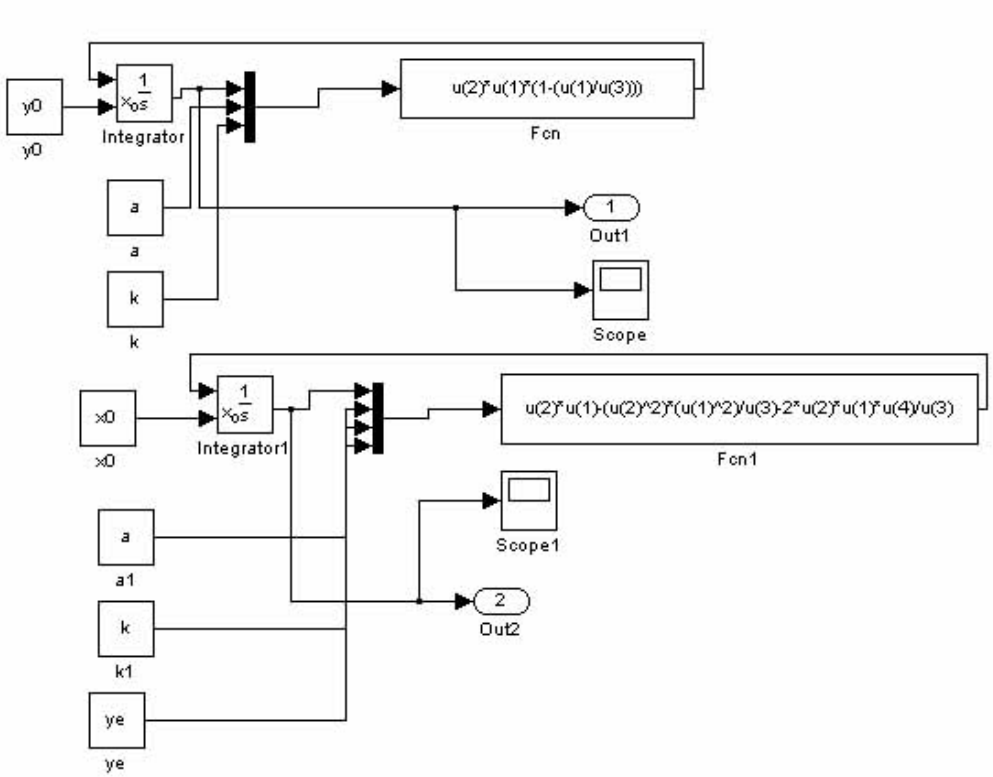


Fig E.1 Model for assessing the validity of of a logistic model linearization.

MATLAB file

```
clc;
clear all;
e=0.5;
a=1;
k=5;
i=1;
for e=0.01:0.01:0.1
% Case 1 x=y-K
ye=k-e;
x0=-e;
y0=k-e;
sim('hw4_3',10);
plot(tout,yout(:,1)-k+e,'--',tout,yout(:,2)+e);
devi(i)=((yout(12,1)-k+e)-(yout(12,2)+e))/(yout(12,1)-k+e)*100;
i=i+1;
end;
x=(yout(12,1)-k+e)
y=yout(12,2)+e
t=tout;
figure;
e=0.01:0.01:0.1;
plot(e,devi);
% Case 2 x=y-0
i=1;
figure;
for e=0.01:0.01:0.1
ye=e;
x0=e;
y0=e;
sim('hw4_3',20);
```

```
plot(tout,yout(:,1),'*',tout,yout(:,2));
devx(i)=(yout(7,1)-(yout(7,2)))/(yout(7,1))*100;
i=i+1;
end;
figure;
e=0.01:0.01:0.1;
plot(e,devx);
```

Around the point y=K:

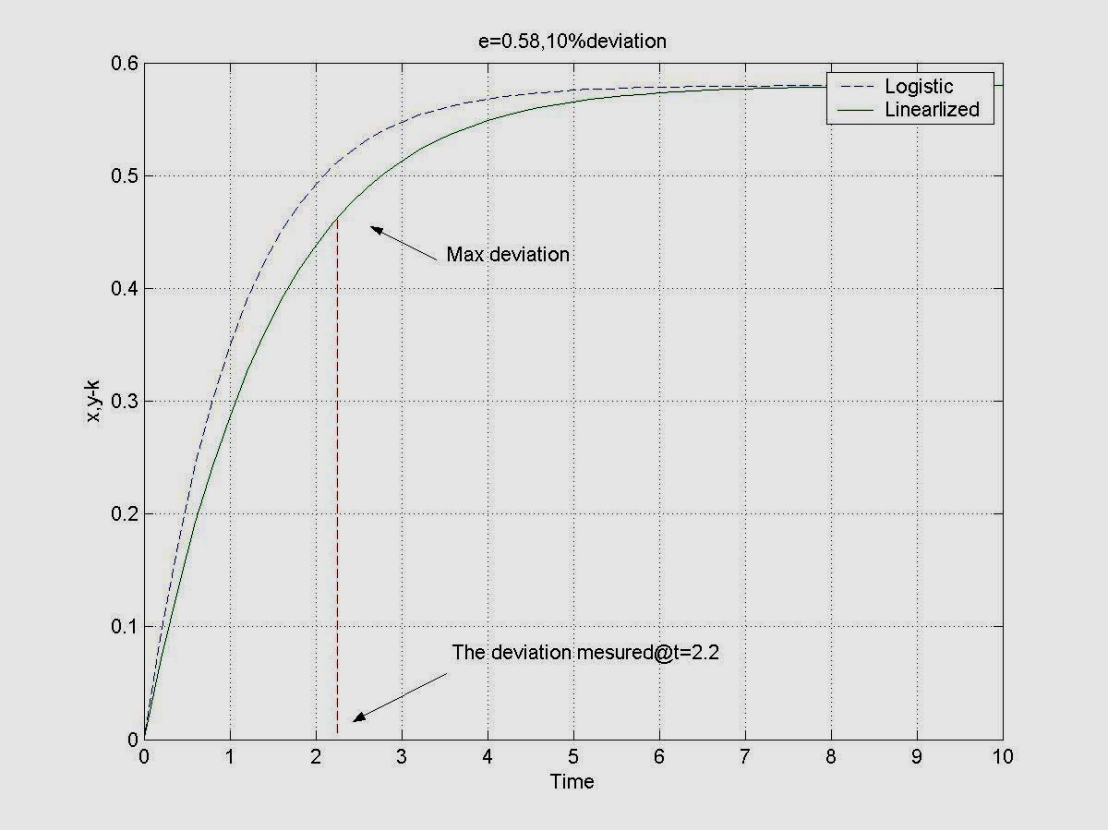


Fig E.2 Deviation of linearized model when ε = 0.58 (about 10% deviation @t=2.2)

Arbitrarily using a 10% deviation as a threshold, the maximum initial condition deviation value ε before maximum error exceeds the threshold is 0.58. K- 0.58≤y≤K+0.58 is the range around y=K for which linearization predicts well the shape of the time response.

Appendix F: Multi-species model with no constraint

A problem suggested in [4] asked students to create a three species model – say, Wolves, Sheep and Grass. The model below shows the full three-species model version. Students however had to run reduced models (those when wolves are not present, or ones in which sheep are not present) to fully understand the role played by each of the species (i.e. how the presence of wolves help the grass reach a higher steady-state value)

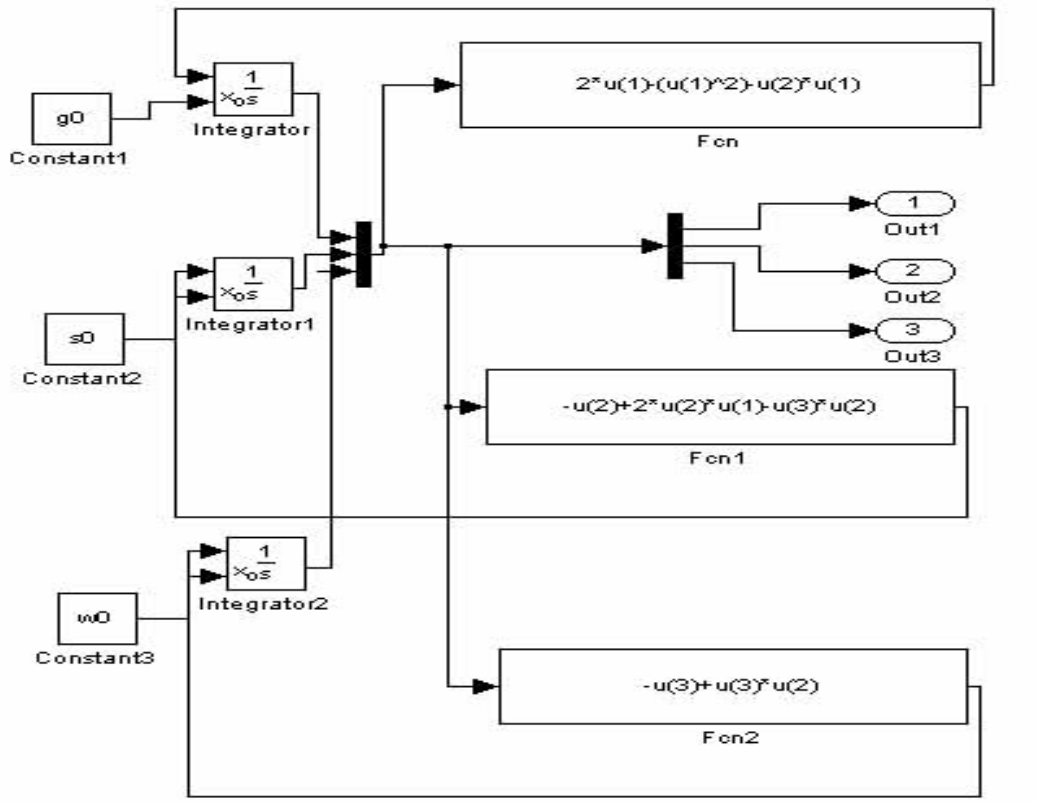


Figure F.1. A wolves-sheep-grass model.

MATLAB m-file.

```
clc;
clear;
s0=1;
g0=1;
w0=1;
color=['r' 'g' 'b' 'c' 'y' 'm'];
cindex=1;
for w0=1:2:5
   sim('hw3_12',17);
   plot(tout,yout(:,1),color(1),tout,yout(:,2),color(2),tout,yout(:,3),color(2+cindex));
   cindex=cindex+1;
   hold on;
end;
figure;
s0=1;
g0=1;
w0=3;
grid on;
sim('hw3_12',40);
plot3(yout(:,1),yout(:,2),yout(:,3),color(cindex));
grid on;
```

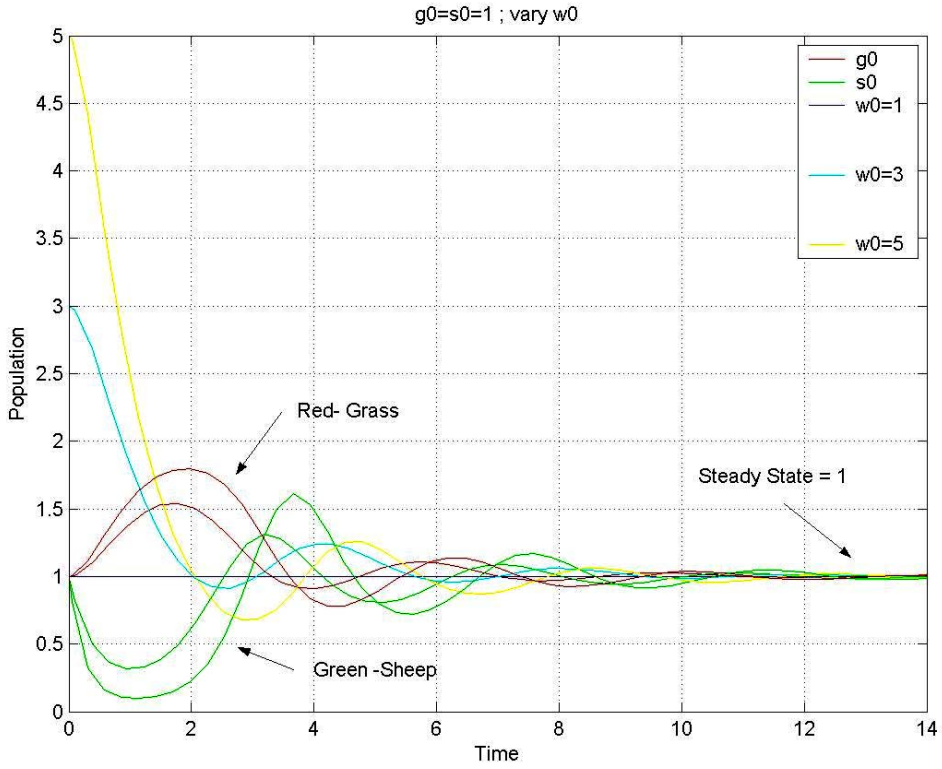


Fig F.2. Initial population of sheep and grass = 1; vary initial wolves population

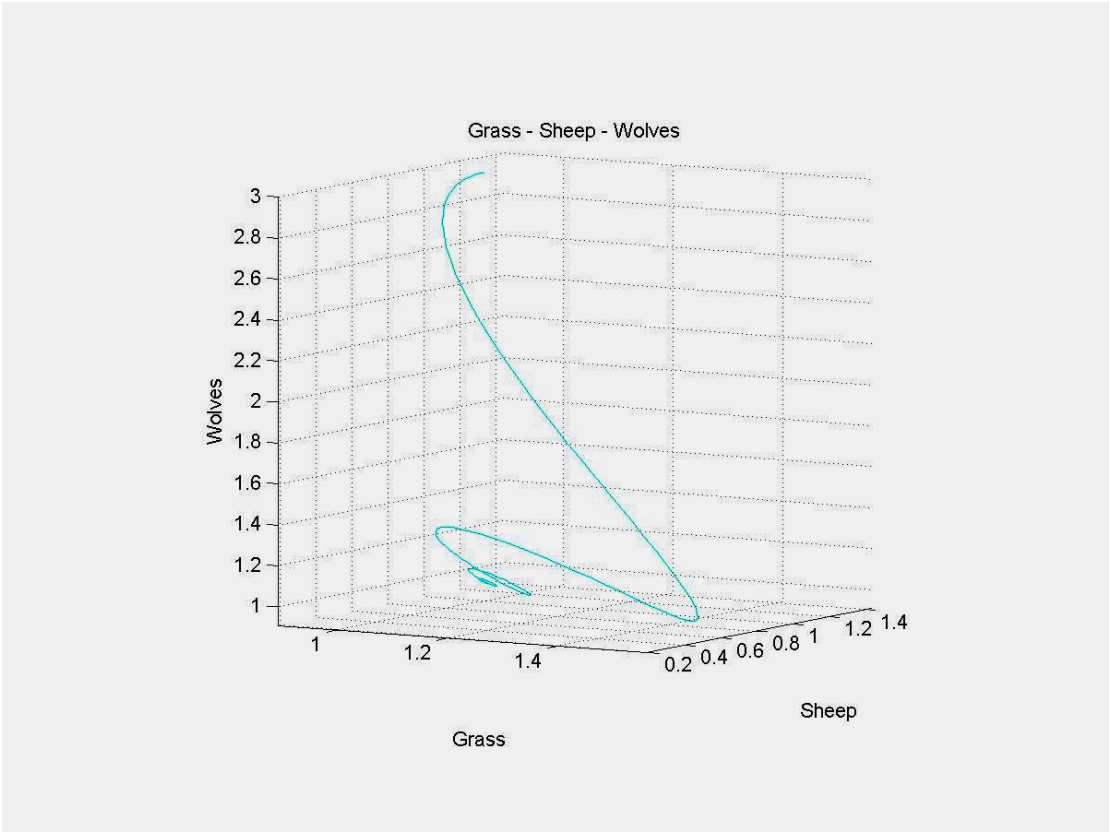


Fig F.3 Grass – Sheep – Wolves 3D plot (axes may be rotated for best view of trajectories).

Appendix G: Models with Constraints

Shown below is the Susceptible-Infected-Recovered (S-I-R) explained in the lectures. Note the algebraic constraint block that now replaces one of the integrators in the model.

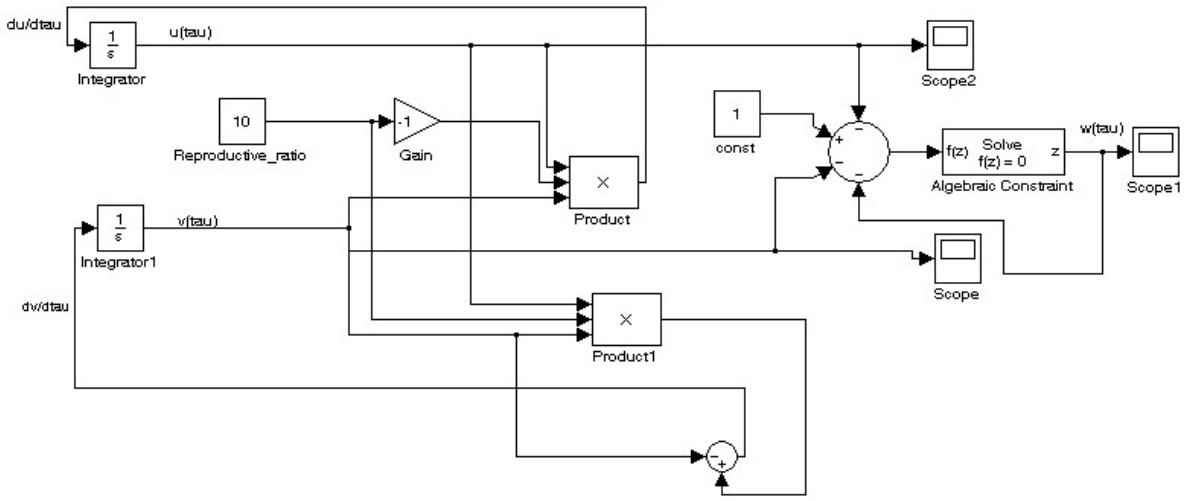


Fig. G.1. S-I-R model featuring an algebraic constraint block.

The S-I-R model extended to include population annual growth, vaccination strategy model fitting to published data about the last case of smallpox outbreak, was assigned as a homework project.

$$\frac{dS}{dt} = -\beta IS + rN - cS \qquad \frac{dI}{dt} = \beta IS - \gamma I \qquad \frac{dR}{dt} = \gamma I + cS$$

The initial populations and desired results are tabulated in the following table. Total elimination of the virus is dependant on the ability of the S-I-R model to drive the number of susceptible and infected people to zero.

Table #1

	Initiation of Smallpox	Eradication of Smallpox
Population	$N_0 = 3{,}486{,}218{,}303$	$N_{10} = 3{,}486{,}218{,}303 \times 1.02^{10}$
Susceptible	$S_0 = 3{,}476{,}218{,}303$	$S_{10} = 0$
Infected	$I_0 = 10{,}000{,}000$	$I_{10} = 0$
Recovered	$R_0 = 0$	$R_{10} = 3{,}486{,}218{,}303 \times 1.02^{10}$

At the beginning of the outbreak, ten million (10,000,000) infected people were recorded. Therefore, the number of susceptible people can be determined subtracting the number of infected from the entire population ($S_0 = N_0 - I_0$). The second table shows the same information scaled by the initial total population.

Table #2

	Initiation of Smallpox	Eradication of Smallpox
Population	$N_0 = 1$	$N_{10} = 1.22$
Susceptible	$S_0 = 0.9971$	$S_{10} = 0$
Infected	$I_0 = 0.0029$	$I_{10} = 0$
Recovered	$R_0 = 0$	$R_{10} = 1.22$

The parameters ß – Infection rate, γ – Removal rate, r – Overall population growth rate and c – Vaccination rate, are all percentage values that are unaffected by any population scaling.

The removal rate γ, had to be determined using the facts that the smallpox infection period lasted 0.065 years and the fatality rate was 30%. In this model, γ was randomly assigned to a value of 60%.
Estimation of ß and c was completed by initially fixing c at c=0.1, and conducting multiple runs with many β values. It was found that ß = 0.05 drives the infected population to zero in 10 years, as given in the data. Using this value of β, students could then investigate how c could have been increased to eradicate the disease in a shorter than 10 years time span.

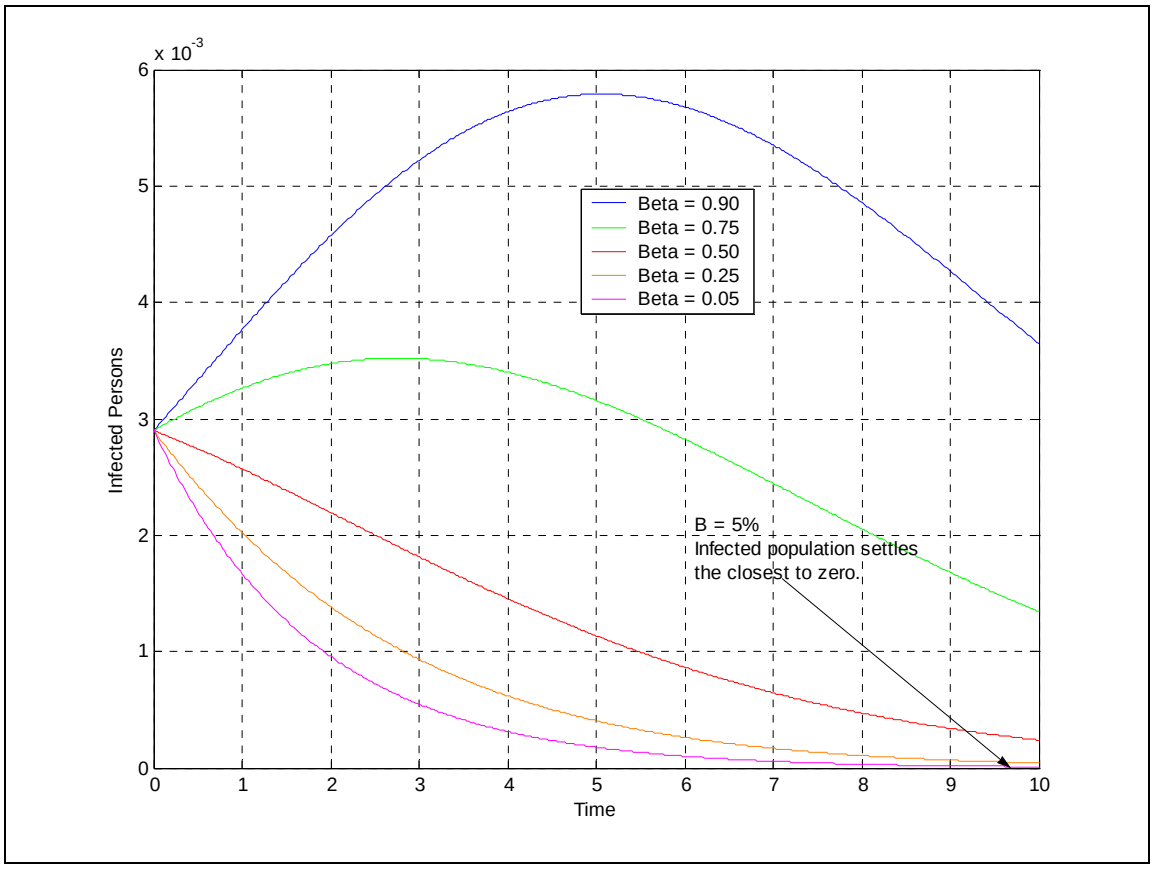


Figure G.2: Smallpox model with various Beta parameters.

Appendix H: Stiff Models

Students were asked to simulate the basic enzyme kinetics model, for various initial concentrations of the substrate and the enzyme.

$$S + E \underset{k_{-1}}{\overset{k_1}{\rightleftarrows}} C \xrightarrow{k_2} P + E$$

One of the key challenges was to determine when Simulink's default integration method "ODE45" may produce erroneous results. The two plots below show the problem's two time scales. What appears to be constant "steady-state" values for c(t) and s(t) are actually Michaelis-Menten quasi-equilibrium conditions, as shown in the longer time scale plot (the two variables decay together while maintaining an algebraic constraint relationship). If the final conditions of the fast time scale picture are not equal to the "initial conditions" in the slow time scale picture, the system is "stiff", and a stiff numerical integration method may be needed. The smaller is the enzyme to substrate initial concentration the larger is the disparity between the two time scales.

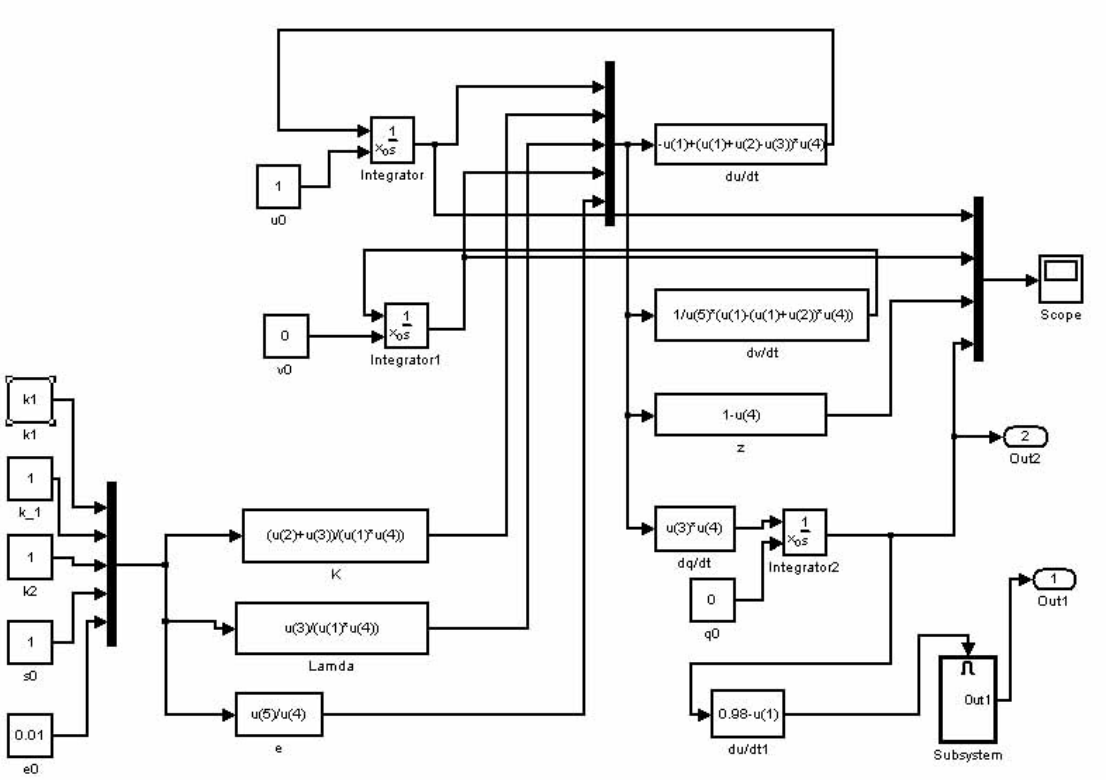


Fig H.1. Basic enzyme kinetics model with product formation time measurement

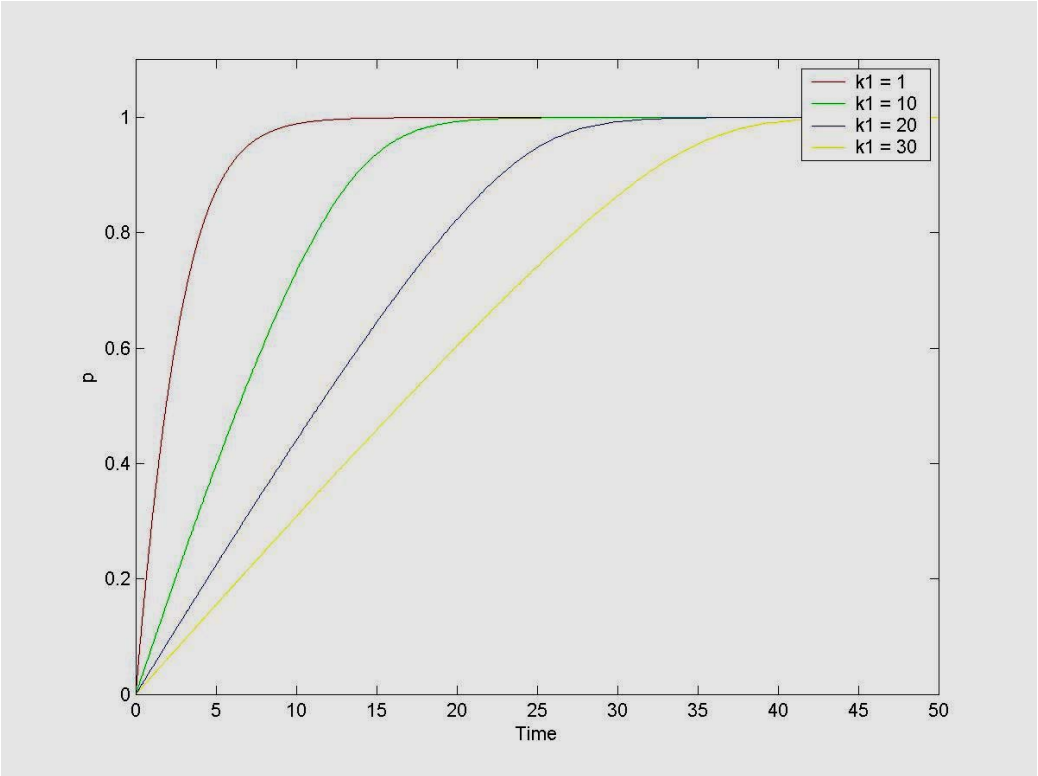


Fig H.2 Simulation Results of p(t) for k1 = 1,10,20,30

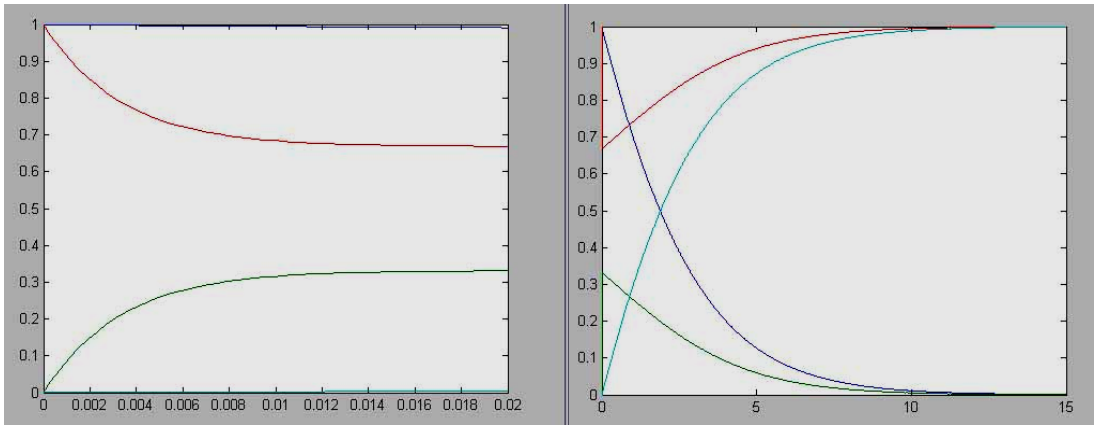


Fig H.3 Fast-time scale and slow time scale simulations

Appendix I: Dymola® Simulation

The following electrical-circuit analog was built based on a respiratory mechanics example in [1].
Students were asked to select reasonable ventilator parameters, and it was easy enough to assure that no negative pressures develop and that the air flow be symmetric. What unexpectedly proved more challenging, to almost all students in the class, is the calibration of the ventilator's amplitude to assure that the incoming air volume is realistic (i.e. around half a liter). Unlike Simulink® where all you have to do is to integrate the flow variable to obtain the volume variable, Dymola® has a different structure. The basic simulation program consists of at least one "package" and at least one "model" contained in a package. Parts of the system that have different physical nature are expected to reside in separate models. In this example, the "electrical" system does not directly allow mathematical integration of one of the variables. This must be done in a different model, that receives its input from the elextrical model. Of course, volume estimate can be done indirectly via checking the input capacitance's "voltage".

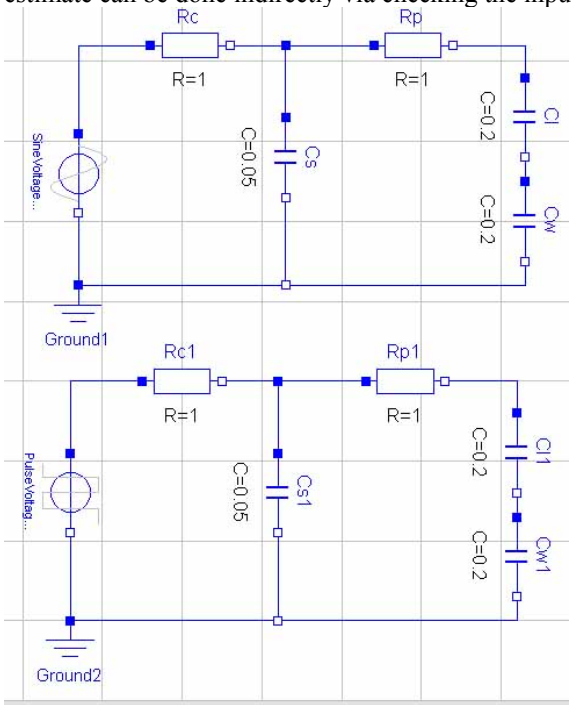


Fig I.1 Respiratory mechanics simulation featuring two types of ventilators.

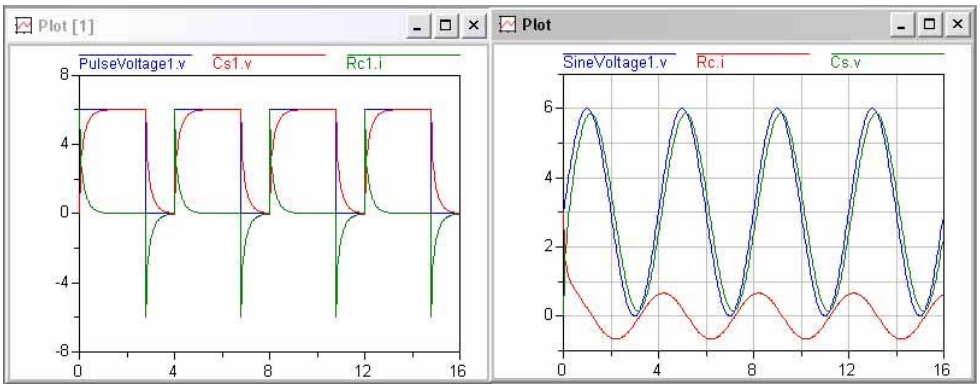


Fig I.2 Simulation results showing the incoming air flow and pressure in the lungs.

Appendix J: Heart and Blood Circulation Simulator [2]

The beauty of the simulator is in their allowing the students to play "what-if" games: The systemic arterial compliance C_{sa} may, for example, be adjusted to obtain a systemic arterial blood pressure of 120/80 mmHg.

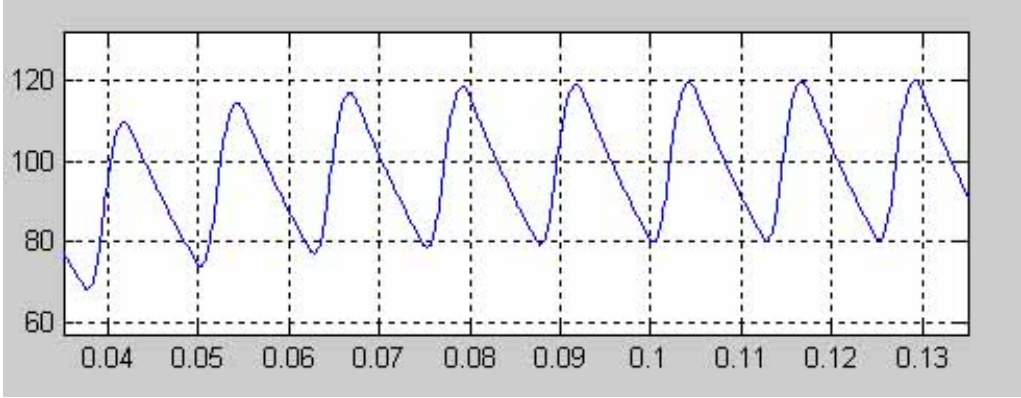


Fig J.1 The arterial blood pressure when $C_{sa} = 0.00112$

We next show sample curves obtained by the students, as they varied model parameters:

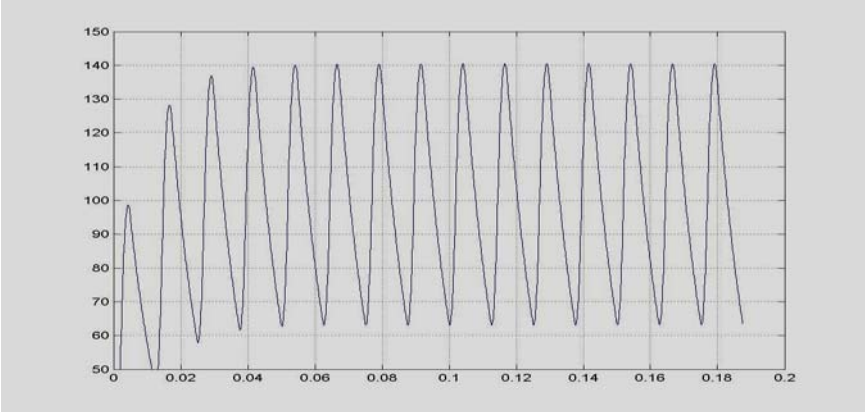


Fig J.2 Arterial blood pressure when C_{sa} is halved (due to atherosclerosis)

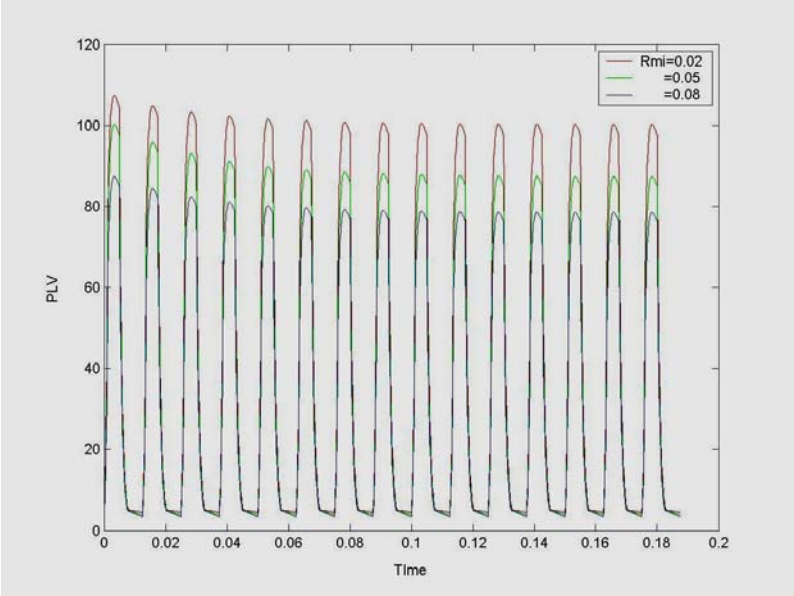


Fig J.3. Left ventricle pressure for various degrees of mitral valve stenosis.

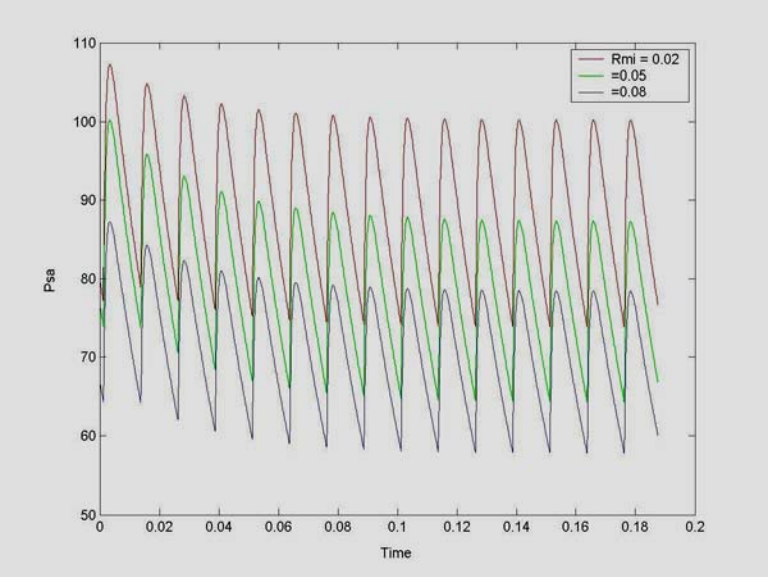


Fig J.4 Uncontrolled and uncompensated arterial pressures for various degrees of mitral valve stenosis.

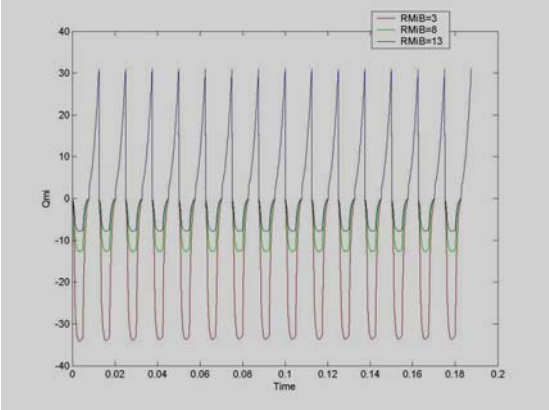
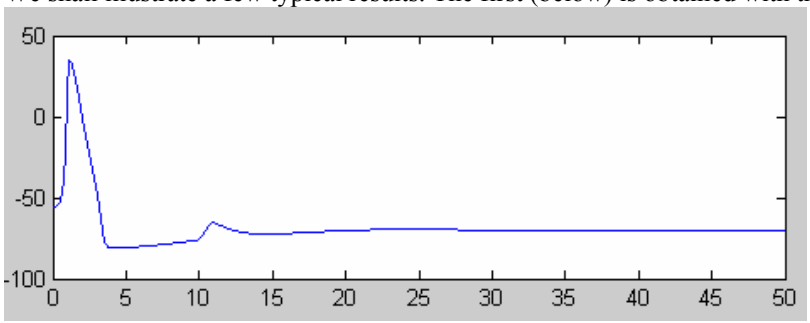


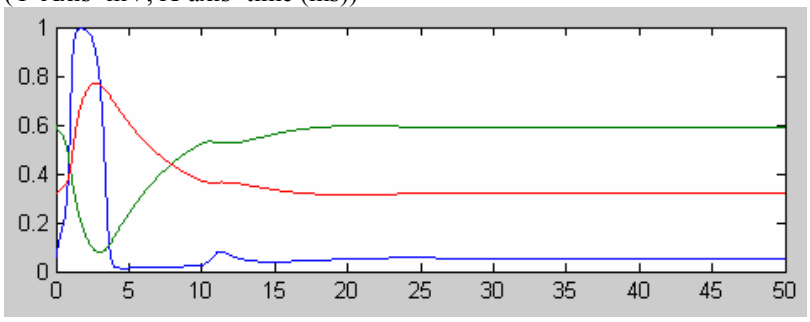
Fig J.5. Backflow through regurgitated mitral valve.

Appendix K: Hodgkin-Huxley Model for Nerve Cell Action Potential Simulator [2]

We shall illustrate a few typical results. The first (below) is obtained with the program default values.

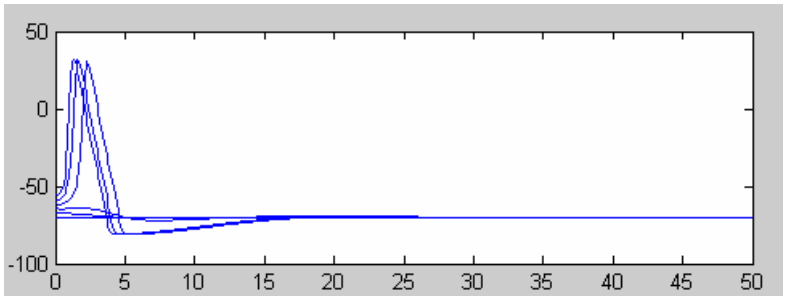


(Y-Axis=mV, X-axis=time (ms))

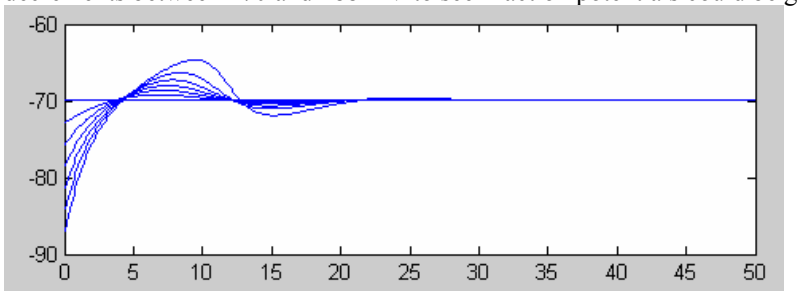


(Y-Axis=Fraction of open gates, X-Axis=time (ms))
(Blue=m gates, Red=h gates, Green=n gates)

Fig. K.1. Action potential (above) and sodium and potassium channels gating functions (below) with small pulse current excitation.

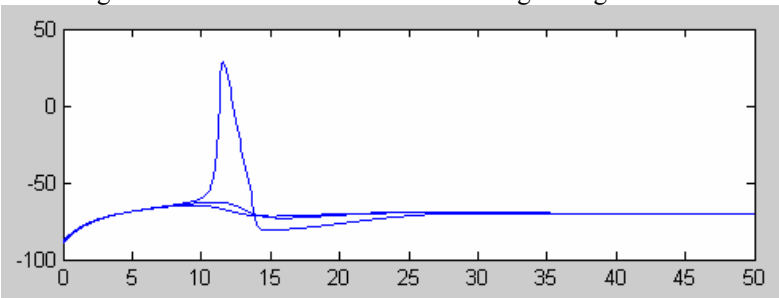


(Y-Axis=mV, X-axis=time (ms))
Fig. K.2. Search for the minimum excitation level threshold.
Next is the examination of the effect of hyperpolarizing shocks where voltages are applied in –3mV decrements between –70 and –88 mV to see if action potentials could be generated.

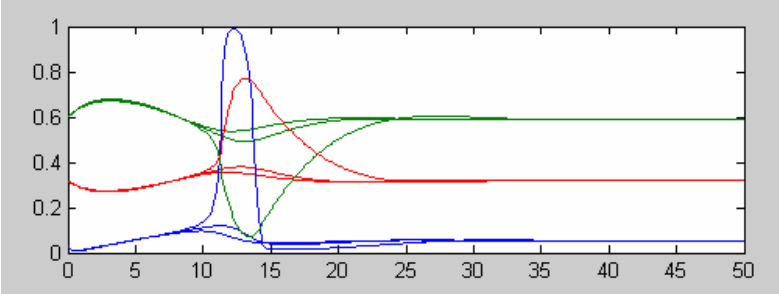


(Y-Axis=mV, X-axis=time (ms))
Fig. K.3. Hyperpolarizing shocks effect – Part 1.

Extending the search to the –88 and –90 mV range using –1mv decrements :



(Y-Axis=mV, X-axis=time (ms))
Fig. K.4. Hyperpolarizing shocks effect – Part 2.
Here we can see that –90mV is the hyperpolarization excitation threshold for the modeled neuron. Looking at the fraction of open gates graphs (below) sheds light about the inner mechanism of the pulse creation.



(Y-Axis=Fraction of open gates, X-Axis=time (ms))
(Blue=m gates, Red=h gates, Green=n gates)
Fig. K.5. Gating functions for previous figure scenario.

Appendix L: Basic Compartmental Modeling

One of the students chose to study a simplified model of Type I Diabetes. We shall quote large segments of this solution. In this scenario, the pancreas does not produce enough insulin to regulate the glucose level in the blood.
The model should describe that when glucose level increases, insulin is released, which increases the liver's ability to absorb glucose and amino acids. Inside the liver, the amount of glycogen and protein synthesis is increased during this process. The glycogen breaks down to glucose inside the liver to be drawn back by the blood if blood sugar levels drops.
A "Two-compartment drug and two-compartment-metabolite" model is explored.
The parameters for this model are:
V1 = Total blood volume = 1. (This was chosen arbitrarily as reference)
V2 = Volume of blood around liver = 0.2 (rough estimate)
V3 = Volume of glycogen in liver = 0.3 (estimate)
V4 = Volume of glucose in liver = 0.2 (estimate)
K_{21} = Diffusion constant from blood to liver cells (0.25)
K_{12} = Diffusion constant from liver cells to blood (.01) much less than K_{21}
K_{31} = Diffusion constant (rate that insulin increases glycogen production). 0.75
K_{43} = Diffusion constant (rate that glycogen is turned into glucose inside of liver). = .05
K_{34} = Diffusion constant (rate that glucose breaks down into glycogen inside of liver). = .001 Should be much less than above.
Initial conditions:
X1 = 0.1
X2 = .05
X3 = 0.25
X4 = 0.5

Picking up values that seemed reasonable resulted in the model shown in Figure L.1. b1u1(t) rate of injecting insulin = 0.31. This value needed to be greater than X1. The variable that needs to be regulated is the insulin concentration in the body, which can be measured directly or indirectly by measuring the glucose level in the blood. The drug injection rate was altered to get a reasonable looking plot.

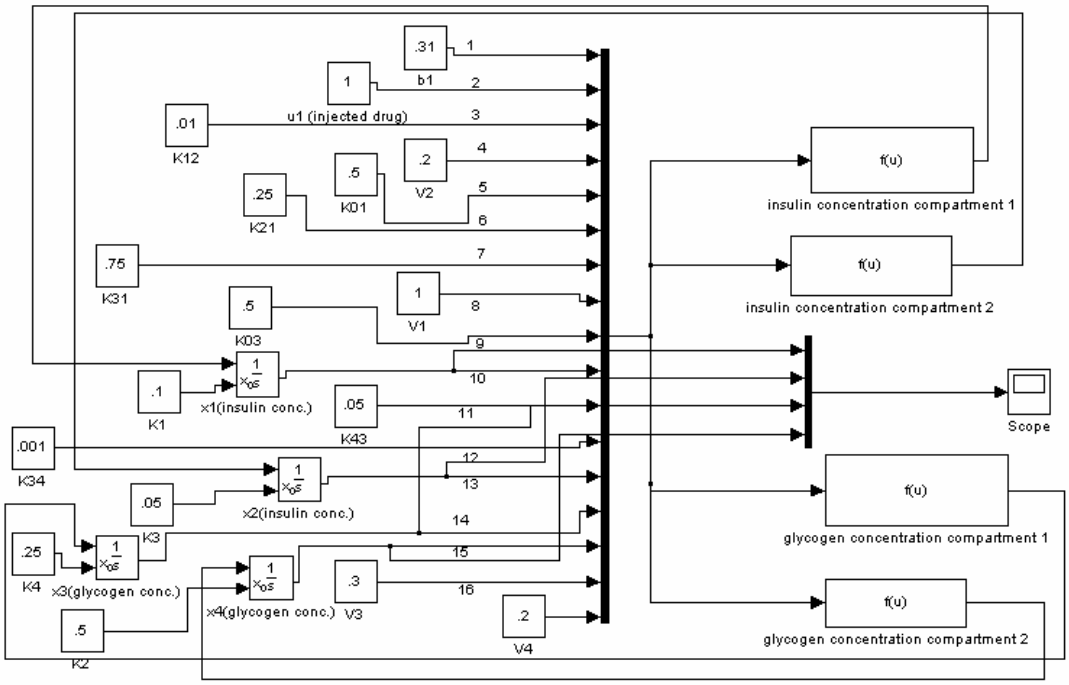


Fig. L.1. Simulink model for the four-compartment model.

The gain b_1 was then changed to 0.23 and the plot of Figure L.2. was obtained.

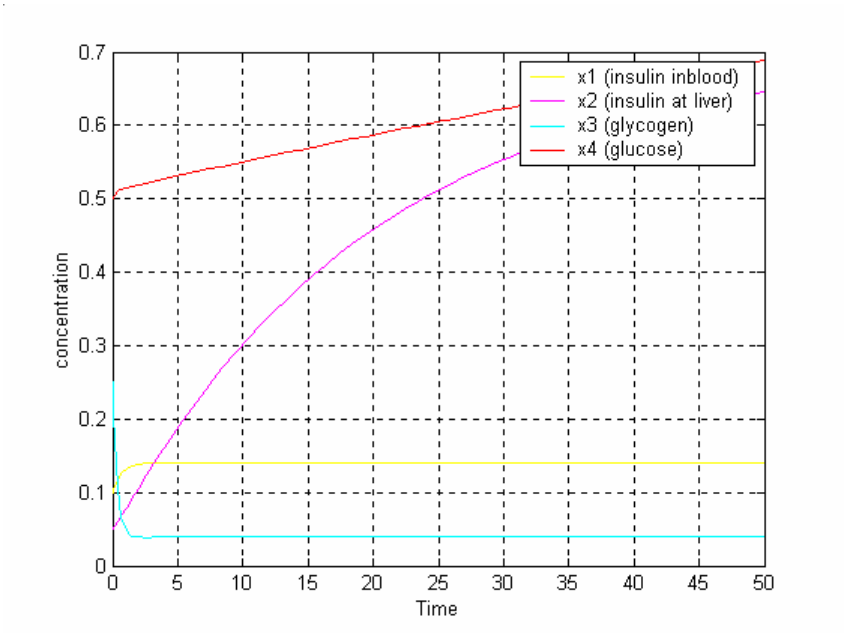


Figure L.2. Model output, featuring incorrect glycogen concentration behavior.

In this example the blood insulin level levels off to a value that is somewhat less than the rate of injection indicating that it is being consumed somewhere in the system. There is a problem with the glycogen level (it was decreasing) and the problem is traced to the value of K_{03}= 0.5. This value was set too high. When K_{03} was reduced to .02 the results shown in Figure L.3 were obtained:

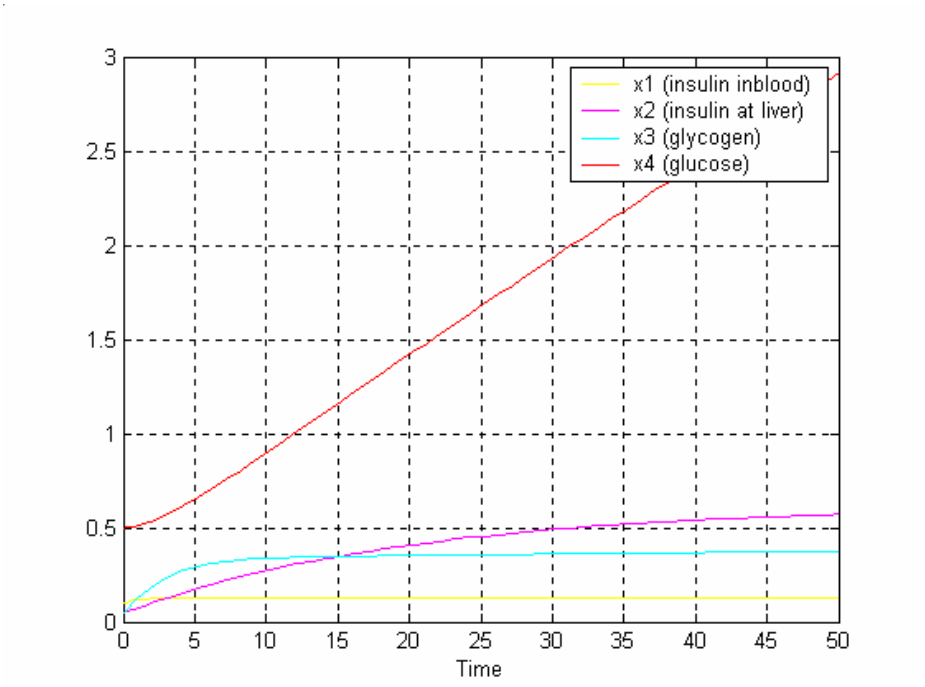


Figure L.3. Corrected model output

This looks more like what one would expect. When a longer simulation was done (2000 points), the glucose value levels off as shown in Figure L.4. The final glucose level is directly dependent on what the rate of insulin injection is.

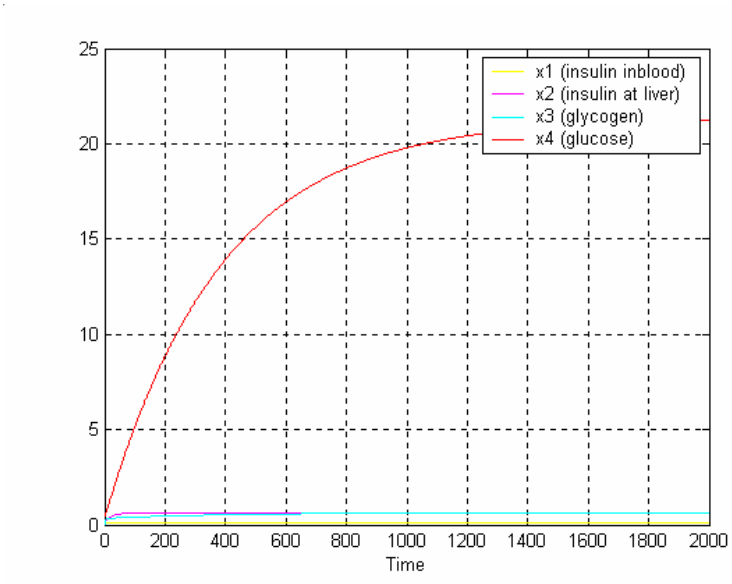


Figure L.4. Long term view of model response.

Model is next modified as shown in Figure L.5 to investigate what happens when insulin is not given to the patient.

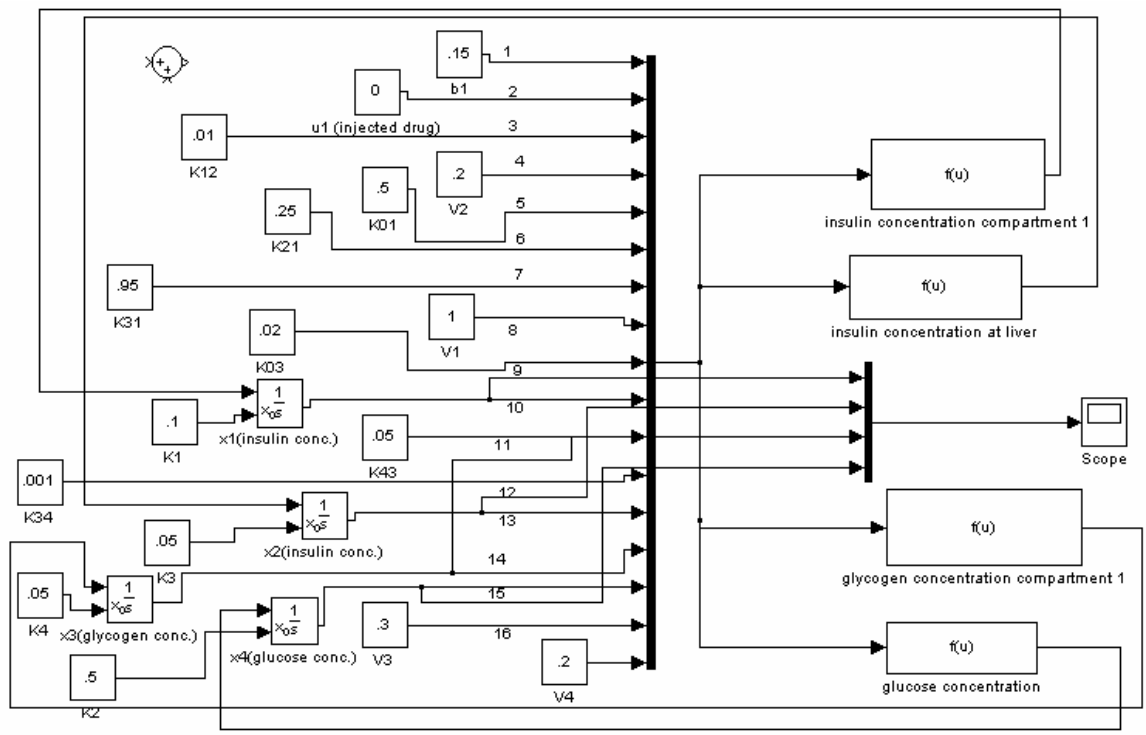


Figure L.5. Model when insulin is withheld
The resulting plot is:

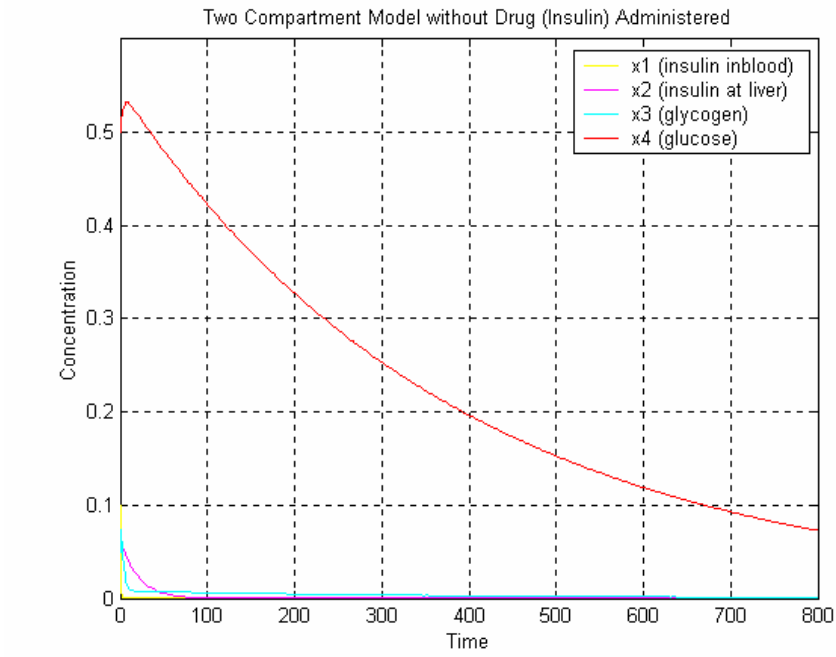


Figure L.6. Glucose concentration decreasing.

We can see here that when the insulin is withheld, the glucose level which starts out high decreases to zero because without insulin, glycogen cannot be produced in the liver which in turn produces glucose that is released back into the bloodstream.
Next the model is modified to add a feedback loop from the glucose concentration comparing it against a given "set point". The output of this comparator was amplified and then fed back to adjust the injected insulin level. The new model with the new parameters is shown below:

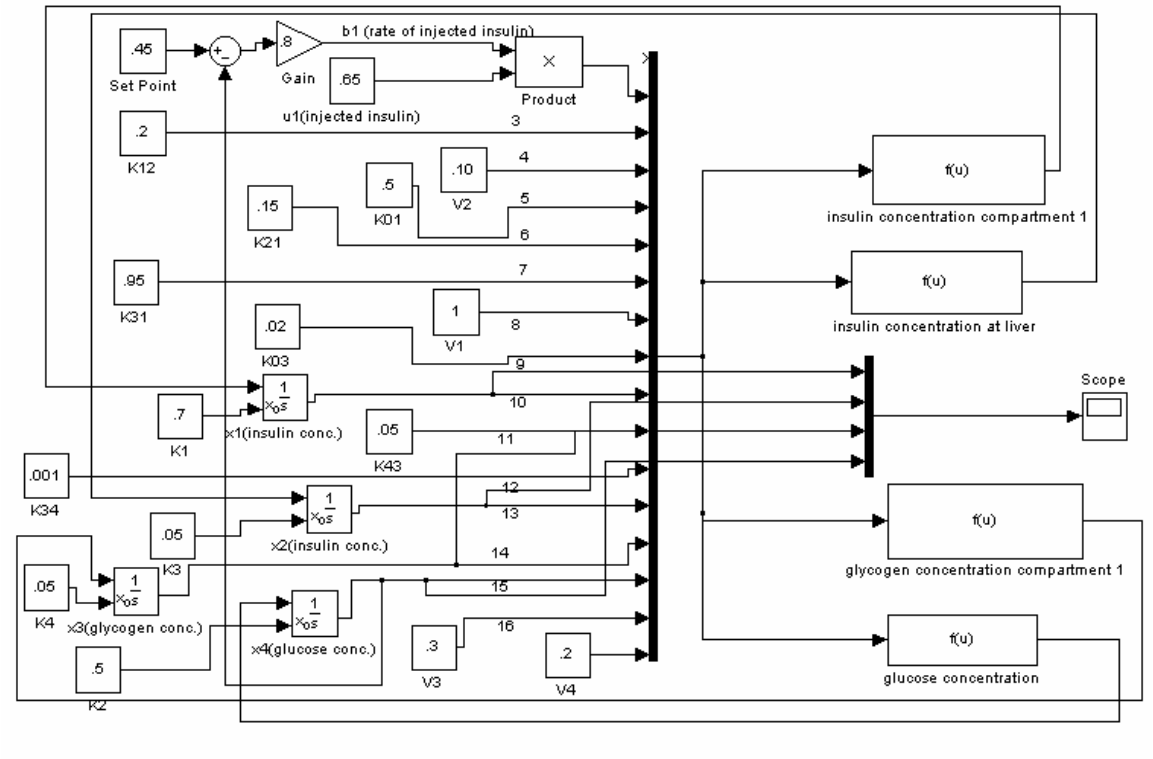


Figure L.7. Feedback control of glucose concentration level
With these parameters, the plot below was obtained:

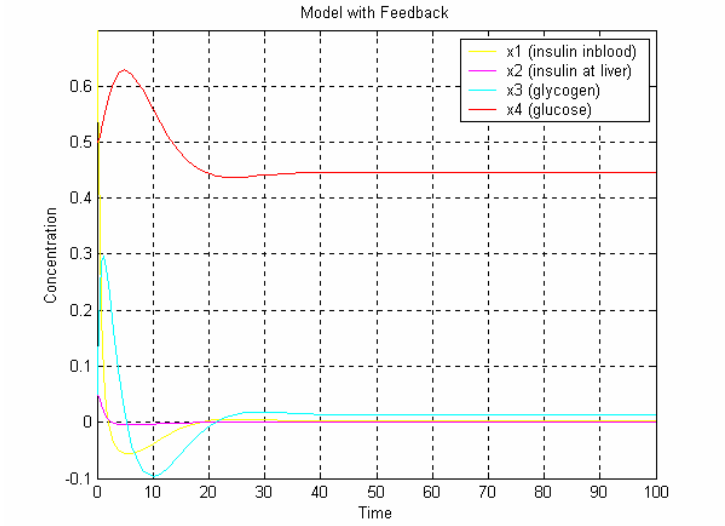


Figure L.8. Closed loop control results – more work is needed…

From this plot we can see that the glucose level does come close to the predefined set point. The insulin level settles at a level just above zero. Because of the gain stage, there are a few instances where the concentrations dropped below zero. He was able to find a few combinations that did not do this, but this affected the ability of the feedback loop to maintain the proper glucose level.

This example is not typical to what most students in the class were able to do with compartmental modeling. Most had no sufficient controls know-how to succeed in this elective assignment. More work is needed to improve the course's controls contents.