Using Matlab Modeling in Cody to teach Teletraffic Engineering

Robin Braun and Zenon Chaczko
Faculty of Engineering and IT
University of Technology Sydney, Australia
Email: {Robin.Braun,Zenon.Chaczko}@uts.edu.au

Abstract—**Engineering students are having to model increasingly complex constructed environments at the same time that their traditional mathematical skills are decreasing[?]. New approaches to modeling are needed. We started with the premise that students find mathematical modeling of Teletraffic problems challenging, and that this can be replaced through modeling in a "computational" environment such as Matlab.. In this paper, we show how the Teletraffic Engineering part of a Network Management subject can be made more interesting and accessible. We plot the student satisfaction scores to show the effectiveness of the approach.**

I. Introduction

Modeling and abstraction are essential tools for engineers. Engineers Australia list them as competencies 2.1 to 2.4[4]. University of Technology Sydney lists hem as competencies C1 to C3 Abstraction and Modeling. In particular, Teletraffic engineering is a mathematics "heavy" discipline. Very often, students studying the subject take away a set of "tools" to solve common problems without a "real" understanding of the workings of the systems. This is because the systems are buried in so called "theory" which comprises of lists of mathematical expressions. However this is just a "language" for describing the theory. The theory is the actual things that happen in the system. The mathematics is also used to abstract and model the system and draw conclusions about its aggregated performance. We suggest that Matlab, and the online tool Matlab Cody can be used to "remove" the "mathematics", and leave the student enough "space" to understand underlying system.

In addition, the process matches very well with the UTS Model of Learning[1]. In our previous work[2], we described the notion of "flipping" the classroom, which also falls squarely into the approach described here.

II. Our propositionUsing Matlab Modeling in Cody to teach Teletraffic Engineering

We propose that students can learn mathematically intensive subject such as Teletraffic Engineering through the development of Computational Models in a computational modeling tool such as Matlab.

Traditionally students would have learned Teletraffic Engineering through the application of previously developed Equation Based Models. These models were first introduced by Agner Krarup Erlang[3] in the early 1900's. These models were only capable of showing the behavior of the systems they modeled in the "aggregate". They allowed us to calculate expectations of what may happen in the systems under changing conditions.

New approaches such as Agent Based Modeling[8] more closely emulate the exact behavior of the agents (clients, servers, etc.) in the system. This is a very valid approach, and warrants future investigation as a learning tool for students. However, that is not what we are describing in this paper.

What we are describing is a better way of internalizing, and making usable Equation Based Models of the system. We are suggesting that a computational environment where the mathematical rules are expressed as a set of natural language type phrases is a more usable expression of the Equation Based Models required.

III. The Modeling Tree

A. Where modeling fits in to the big picture from simulation to programming

1) Simulation. This is where you create something that behaves like the real thing, but you know is is not actually the real thing. Examples are:
   a) Medical simulation
   b) Flight simulation
   c) Computer Network simulation
2) Emulation. This is where you create something that behaves like the real thing. But it is used to better understand the real thing. Examples are:
   a) Build a small version of a bridge to make sure it still stands in the wind
   b) Use a wind tunnel to understand the behavior of an aircraft wing
3) Modeling. Create some abstract system that can be used to better understand the real thing. Examples are:
   a) Equation Based Modeling
   b) Agent Based Modeling
4) Programming This where modeling becomes embedded in and part of the real thing. It is the real world.

B. A useful view of what we mean by modeling

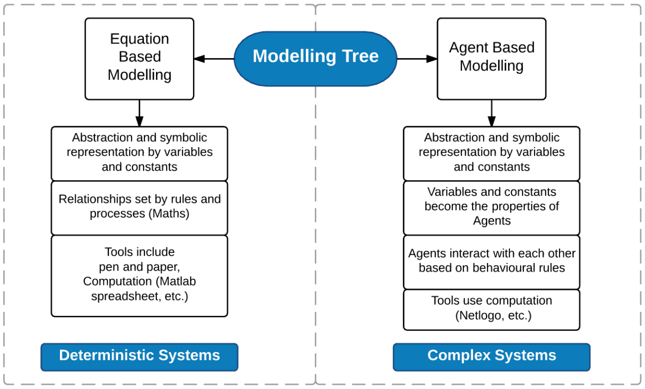

Equation Based Modelling
Modelling Tree
Agent Based Modelling
Abstraction and symbolic representation by variables and constants
Relationships set by rules and processes (Maths)
Tools include pen and paper, Computation (Matlab spreadsheet, etc.)
Deterministic Systems
Abstraction and symbolic representation by variables and constants
Variables and constants become the properties of Agents
Agents interact with each other based on behavioural rules
Tools use computation (Netlogo, etc.)
Complex Systems

Figure 1. A representation of the modeling "tree"

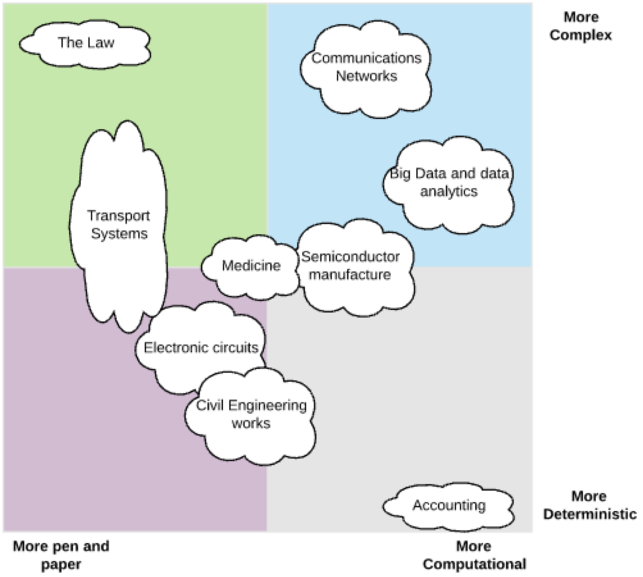


Figure 2. Areas of application

A useful view of the modeling tree is shown in Figure 1. Areas of application are suggested in Figure 2.

IV. The modeling tasks to be learned by the student

A. The context

The context is the subject 48750 offered at University of Technology Sydney. In particular, the first 6 sessions cover the basics of Teletraffic Engineering through Traffic Sources, Queues and Markov Modeling, Blocking Systems, Delaying Systems to Packet Based Systems. The purpose of the subject is

".. for students to develop competencies needed for the planning and management of communications networks, particularly in the areas of traffic source modeling, performance analysis, dimensioning, simulation and management. Students acquire a foundation in: modeling of traffic including packet-based systems; dimensioning of fixed networks and mobile networks; performance analysis of networks based on queuing theory; circuit switching networks; and packet switching networks."

B. An example

One of the problems to be used is the classical Grade of Service (GoS) problem. This typically yields to linear Markov Modeling, which formulates into Equation Based Models as follows:

1) It can be shown that the likelihood of any particular state in the Markov model is

$$P(j) = \frac{A^j}{j!}P(0) \qquad P(0) = \frac{1}{\sum_{x=0}^{N} \frac{A^j}{j!}}$$

Therefore

$$P(j) = \frac{\frac{A^j}{j!}}{\sum_{k=0}^{N} \frac{A^k}{k!}} \qquad B = \frac{\frac{A^N}{N!}}{\sum_{k=0}^{N} \frac{A^k}{k!}}$$

$$When\, j = N$$

2) Buried in each of these steps is a physical model that should be understood before the Equation Based mathematical model shown above makes any sense. This includes concepts such as Traffic Sources.
   a) From the definition of an Erlang, the Traffic carried by a circuit can be calculated from $A = \frac{\mu h}{T}$, where A is traffic offered in Erlangs, μ is average number of calls arriving during time T and h average call holding time.

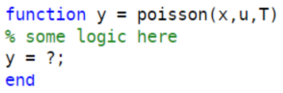
V. Matlab models in Matlab Cody

Matlab[9] is a widely used and very capable computational modeling environment supplied by The Mathworks. Matlab Cody is an online tool provided by the Mathworks for teaching coding in Matlab. We are using it to do that, but also to teach modeling. The Cody exercises we used are shown below, and are done in teams of 2 students[5][6][7].

- Session 01: Traffic and its Sources
- Session 02: Queues and Markov Modeling
- Session 03: Network Resources and Lost requests for service
- Session 04 - Network Resources and Delayed requests for service
- Session 05 - Application to Packet Based Systems
- Session 06 - Tightly-Coupled Multiprocessor

Each of the sessions had multiple problems to be solved inside Matlab Cody. For example, Session 01 is shown below:

1) Calculate Traffic
2) Average traffic to a website



```
function y = poisson(x,u,T)
% some logic here
y = ?;
end
```

Figure 3. Solution Template

3) Poisson Distribution
4) Binomial Distribution
5) Binomial vs Poisson
6) Error rate
7) Defects per wafer

A. Computational Models as apposed to Equation Based Models

An important point is that at no time were the students exposed to the Equations, or the Equation Based Models until after they had created the Computational Models. The students basis was a description of the Physical Model of the system. As mentioned above, the Computational Model is just a restatement of the Equation Based Model in natural language type Matlab statements, thus avoiding the arcane Greek symbols, etc.

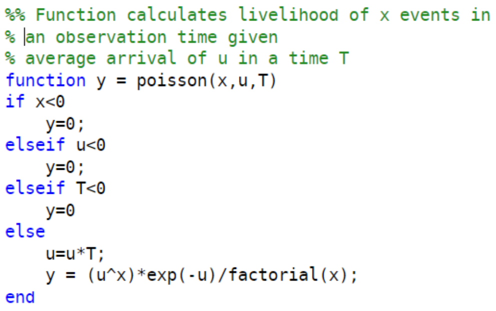
B. Statement of the model in Computational terms

The students were presented with a description as shown below:

Create a Matlab function that calculates the likelihood that x events will occur in an observation period, given that the source is Poisson distributed. The inputs to the function are the observation time T in seconds and the average occurrence of events per second u . The function must cope with invalid inputs by returning zero.

The student is presented with solution template as shown in Figure 3:

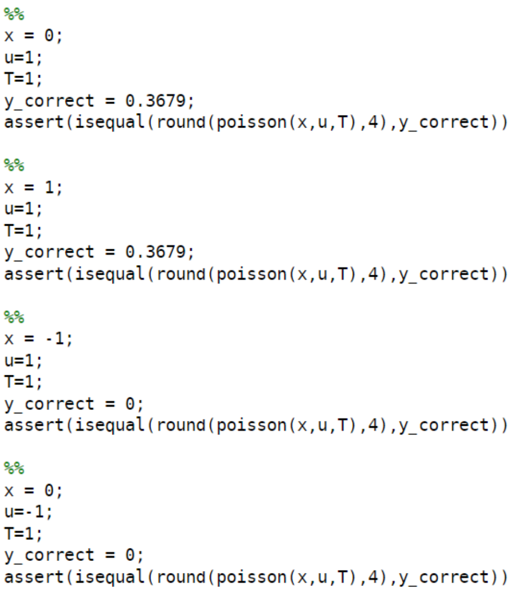
The instructor creates a reference solution and a set of tests in a test suite as shown in Figure 4:



```
%% Function calculates livelihood of x events in
% an observation time given
% average arrival of u in a time T
function y = poisson(x,u,T)
if x<0
    y=0;
elseif u<0
    y=0;
elseif T<0
    y=0
else
    u=u*T;
    y = (u^x)*exp(-u)/factorial(x);
end
```

Figure 4. Reference Solution

The test suite is shown in Figure



```
%%
x = 0;
u=1;
T=1;
y_correct = 0.3679;
assert(isequal(round(poisson(x,u,T),4),y_correct))

%%
x = 1;
u=1;
T=1;
y_correct = 0.3679;
assert(isequal(round(poisson(x,u,T),4),y_correct))

%%
x = -1;
u=1;
T=1;
y_correct = 0;
assert(isequal(round(poisson(x,u,T),4),y_correct))

%%
x = 0;
u=-1;
T=1;
y_correct = 0;
assert(isequal(round(poisson(x,u,T),4),y_correct))
```

Figure 5. Test Suite for the first problem

VI. Examples from Session 6

Session 6 concerned Tightly-Coupled processors. The background to the exercises is as follows:

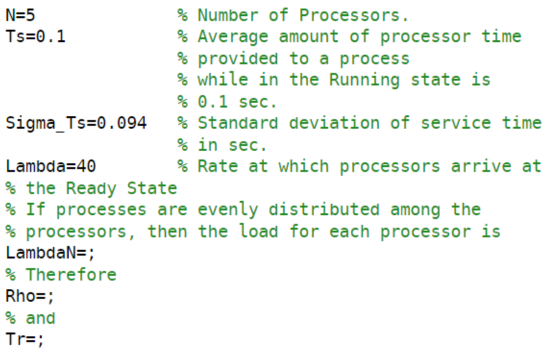
Let us consider the use of multiple tightly-coupled processors in a single computer system. One of the design decisions had to do with whether processes are dedicated to processors.

- If a process is permanently assigned to one processor from activation until its completion, then a separate short-term queue is kept for each processor.
- In this case, one processor can be idle, with an empty queue, while another processor has a backlog.
- To prevent this situation, a common queue can be used.
- All processes go into one queue and are scheduled to any available processor.
- Thus, over the life of a process, the process may be executed on different processors at different times.

The exercise for this Session is to try to get a feel for the performance speed-up to be achieved by using a common queue.

- Consider a system with five processors and that the average amount of processor time provided to a process while in the Running state is 0.1 sec.
- Assume that the standard deviation of service time is observed to be 0.094 sec.
- Because the standard deviation is close to the mean, we will assume exponential service time.
- Also assume that processes are arriving at the Ready state at the rate of 40 per second.

There were 4 problems associated with this scenario.



```
N=5              % Number of Processors.
Ts=0.1           % Average amount of processor time
                 % provided to a process
                 % while in the Running state is
                 % 0.1 sec.
Sigma_Ts=0.094   % Standard deviation of service time
                 % in sec.
Lambda=40        % Rate at which processors arrive at
% the Ready State
% If processes are evenly distributed among the
% processors, then the load for each processor is
LambdaN=;
% Therefore
Rho=;
% and
Tr=;
```

Figure 6. Problem 1 for Session 6

1) Single-Server Approach
2) Multi-server approach - Residence Time
3) Multi-server approach - Waiting Time
4) LAN Router situation

The environment common to all was:

- **N=5 (Number of Processors.)**
- **Ts=0.1 (Average amount of processor time provided to a process while in the Running state is 0.1 sec.)**
- **Sigma_Ts=0.094 (Standard deviation of service time in sec.)**
- **Lambda=40 (Rate at which processors arrive at the Ready State)**

See Figure 6 for details on the question, and Figure 7.

The success of the students who attempted this section is illustrated in Figure 7.

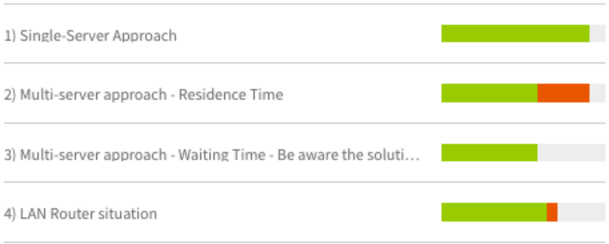


Figure 7. Success rate of students attempting Section 6. (Green is success. Red failure)

VII. Results of the Survey

The students were asked the following questions

1) Please give me an opinion about how useful the tutorials were in helping you to understand the very mathematical subject of Queues and Traffic. Do you agree or not that they were useful.
2) This is a bit of a complex question. Do you agree (or not) that using Matlab prompted you to think about subject matter, and hence understand it better.
3) Was your level of Matlab ability high enough to tackle the questions?
4) Do you agree that tackling the questions helped you raise your level of confidence in Matlab.
5) Do you agree that this experience with Matlab will enable you to use it (or an equivalent) better in your career?
6) Do you agree (or not) that Matlab Cody was a good environment in which to do these tutorials.

The score was +2 to -2 where +2 was the positive outcome. In addition the students were asked to make comments about their experience.

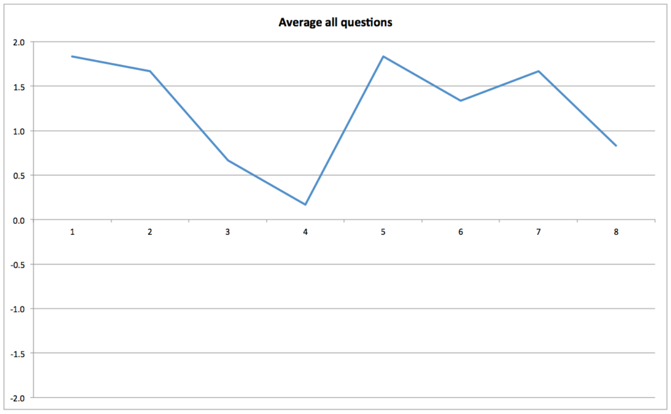


Figure 8. Average results

By way of analysis:

1) None of the responses were “below the line”. All were less or more positive
2) Most agreed that using Matlab enabled them to better understand the material
3) There was some ambivalence about their Matlab ability
4) Most agreed that they would be able to use this experience of computational modeling going forward in their careers

Here are a sample of the comments from the students.

1) *Really enjoyed the Cody coursework - interesting way to learn and practice the content*
2) *The tutorials were helpful at the start with good support, however from 3rd week onwards the questions got much harder and tutorial didn’t provide enough support or help to help the students. Towards the end i had to copy some of the answers and hope they were correct as i had trouble answering it on my with little support. Some tutorial questions were also harder than the lecture we were doing that week, making any pre requiset study before the class completely useless.*
3) *Really liked the Tutorials with Matlab Cody - was interactive and engaging.*

VIII. Conclusions

The results of both the satisfaction and "did I learn" surveys are very conclusive in indicating that we met our objectives. We will repeat the experiment this year (2017), with updated exercises based on the feedback received. Future work also includes trying to use Agent Based Modeling as an alternative.

References

[1] Shirley Alexander. The UTS Model of Learning, 2016.

[2] Robin Braun and Zenon Chaczko. That flipping classroom-getting engineering students to be consciously competent on their own. In *Information Technology Based Higher Education and Training (ITHET), 2015 International Conference on*, pages 1–4. IEEE, 2015.

[3] E Brockmeyer, H L Halstrm, Arne Jensen, and Agner Krarup Erlang. The life and works of AK Erlang. 1948.

[4] Engineers Australia. STAGE 1 COMPETENCY STANDARD FOR PROFESSIONAL ENGINEER. Technical report.

[5] David W Johnson, Roger T Johnson, and Karl A Smith. Cooperative learning returns to college what evidence is there that it works? *Change: the magazine of higher learning*, 30(4):26–35, 1998.

[6] K A Smith, D W Johnson, and R T Johnson. The use of cooperative learning groups in engineering education. In *Proceedings Eleventh Annual Frontiers in Education Conference, Rapid City, SD, Washington: IEEE/ASEE*, pages 26–32, 1981.

[7] Karl A Smith, Sheri D Sheppard, David W Johnson, and Roger T Johnson. Pedagogies of engagement: Classroom-based practices. *Journal of engineering education*, 94(1):87–101, 2005.

[8] Uri Wilensky and William Rand. *An introduction to agent-based modeling: modeling natural, social, and engineered complex systems with NetLogo*. MIT Press, 2015.

[9] Michael A Wirth and Peter Kovesi. MATLAB as an introductory programming language. *Computer Applications in Engineering Education*, 14(1):20–30, 2006.