

# An Introductory Course of Modeling and Computation for Chemical Engineers

MORDECHAI SHACHAM

*Department of Chemical Engineering, Ben-Gurion University of the Negev, Beer-Sheva 84105, Israel*

Received 24 January 2004; accepted 15 November 2004

**ABSTRACT:** A new introductory course for modeling and computation, with main emphasis on computer-based problem solving has been developed. Advanced features for enhancing learning effectiveness included in the course are: the use of real life engineering problems as starting points for learning new material, multistage problems, where the problem difficulty and the complexity of the computational tool are gradually increased. The programming assignments involve mainly modification of examples, and self-grading with immediate feedback ensures mastery level completion of the assignments. The website of the course is [http://www.bgu.ac.il/chem\\_eng/pages/Courses/shacham%20courses/intoduction.htm](http://www.bgu.ac.il/chem_eng/pages/Courses/shacham%20courses/intoduction.htm). © 2005 Wiley Periodicals, Inc. *Comput Appl Eng Educ* 13: 137–145, 2005; Published online in Wiley InterScience ([www.interscience.wiley.com](http://www.interscience.wiley.com)); DOI 10.1002/cae.20039

**Keywords:** modeling and computation; problem-based learning; programming; self-grading

## INTRODUCTION

It is generally accepted that computer-based (or computer-enhanced) problem solving (CBPS) is a very important or possibly the most important application of the computer in engineering education and practice. In spite of that, the penetration of CPBS in the various engineering disciplines has been disappointingly slow and of limited extends (Kantor and Edgar [1], Jones [2]). The paradox is that the long-term emphasis on the study of computer programming can be partly blamed for the impediment in the spread of CPBS. Programming in FORTRAN, Pascal, or C is often being taught without establishing any connec-

tion with CPBS in a particular engineering discipline. Faculty, who learned programming this way, envisions CPBS as being a little bit of mathematical modeling and mostly coding and debugging a computer program. This form of CPBS contributes very little to the understanding of the course material, while most of the time is spent on the technical details of programming. This combined with the fact that very few practicing engineers do any programming (in a survey that was conducted by Davis et al. [3] and included engineering management and recent chemical engineering graduates 92% said that they never or seldom program in FORTRAN or another computer language) explain the lack of enthusiasm on behalf of engineering educators to use CPBS more widely.

In order to change this situation CPBS and not a programming language must be put in the center of

---

Correspondence to: M. Shacham ([shacham@bgumail.bgu.ac.il](mailto:shacham@bgumail.bgu.ac.il)).  
© 2005 Wiley Periodicals Inc.

the introductory computer course for engineers. We have developed such course for chemical engineers where the emphasis is on CPBS and it covers several types of software packages of various complexity, flexibility, and user friendliness. The packages learned in this course include POLYMATH<sup>1</sup>, Excel<sup>2</sup>, and MATLAB<sup>3</sup>. POLYMATH is used in several textbooks (Cutlip and Shacham [4], Fogler [5], and Himmelblau [6] and Kyle [7]) for numerical problem solving. Excel is a computational tool that has a very wide range of capabilities and it is extensively used by practicing engineers. MATLAB is used in some undergraduate and graduate engineering courses, for its wide range of capabilities and because it can also be considered as a programming language. The three packages are learned in a sequence where the concepts are build up gradually, making the learning of programming with MATLAB considerably easier.

To enhance the learning effectiveness the following advanced features are included in the course.

### Based on Real-Life Problems

The study of each subject starts by presenting a real life problem, which requires numerical solution.

### Multi-Stage Problems

The problems require computer solution in several stages. Preliminary stages are solvable with the easy to use software package POLYMATH while advanced stages require the use of Excel (Shacham et al. [8]) or MATLAB programming (Shacham et al. [9])

### Programming by Modification

Solved examples are provided. The students are encouraged to modify and extend the solved examples to solve their assignments.

### Self-Grading

Students check and grade their own Internet-based homework assignments and exams. The self-test program provides immediate feedback on errors made (Shacham [10]).

The course is divided into six chapters according to the mathematical/numerical method that has to be used for solving the problems:

1. Complex consecutive calculations
2. Iterative solution of a nonlinear equation
3. Matrix operations and solution of systems of linear equations
4. Multiple-linear, polynomial and nonlinear regression
5. Introduction to solution of systems of nonlinear algebraic equations
6. Introduction to solution of systems of ordinary differential equations

The course is given to freshmen chemical engineering students who have already taken an introductory Material and Energy Balance course or the two courses are run in parallel.

In the following sections chapters 1, 2, and 4 which cover about 75% of the course material will be described in detail. An Internet-based homework assignment together with its self-test will be demonstrated and computer-based exams will be briefly described. Finally some preliminary results of the course evaluation will be provided and conclusions will be drawn. Most of the course material (except the computer-based exams) is available at [http://www.bgu.ac.il/chem\\_eng/pages/Courses/shacham%20-courses/introduction.htm](http://www.bgu.ac.il/chem_eng/pages/Courses/shacham%20-courses/introduction.htm)

## COMPLEX CONSECUTIVE CALCULATIONS

The contents of this chapter are shown in Table 1. The principal example used is the “Calculation of the compressibility factor and derived thermodynamic properties using a cubic equation of state” (Shacham et al. [8] and [9]). This example is of practical importance in several respects; the one that is most easily explained to first year engineering students is that the compressibility factor is used to convert measured volume of natural gas delivered to a customer to units of weight where the payment is done according to the weight delivered. This problem does not require iterative solution but the calculations are complex enough to demonstrate the need for computer solution. In addition to the basic arithmetic operations the calculations also require math functions such as logarithm, square and cubic roots, the trigonometric functions cosine, and arccosine, if conditions and logical expressions. Cubic equations of state have three roots one real root is always a physically correct solution, related to the gas phase. The other two roots may be real or complex, one of the additional real roots can also be physically correct (related to the liquid phase), the second real root or the two complex roots are physically meaningless. This helps to demonstrate, in

<sup>1</sup>POLYMATH is copyrighted by M. Shacham, M. B. Cutlip, and M. Elly (<http://www.polymath-software.com/>).

<sup>2</sup>Excel is a trademark of Microsoft Corporation (<http://www.microsoft.com>).

<sup>3</sup>MATLAB is a trademark of The Math Works, Inc. (<http://www.mathworks.com>).

**Table 1** Contents of Chapter 1, "Complex Consecutive Calculations"

---

|  |  |
|--|--|
| Chapter 1: Complex Consecutive Calculations  |  |
| Mathematical and numerical concepts demonstrated   |  |
| Analytical solution of cubic equations, multiple roots, physically feasible and infeasible solutions   |  |
| Problems used in the examples, assignments, and exams  |  |
| Calculation of thermodynamic properties using a cubic equation of state (Shacham et al, 2003 [8])  |  |
| Calculation of adiabatic flame temperature (Shacham, 1998 [10], problem 1.12 in Cutlip and Shacham [4])  |  |
| POLYMATH program   |  |
| Nonlinear equations solver   |  |
| Excel options and functions  |  |
| Definition of constants and arithmetic formulas, arithmetic functions, creating series, absolute and relative addressing, if statements and logical functions, two-input data tables, XY (Scatter) plot  |  |
| MATLAB options and functions   |  |
| Interactive execution and m-files, definition of scalars and arrays, clearing variables, arithmetic operations, math, logical and trigonometric functions, command window control functions: <i>clc</i> , <i>format</i> and <i>disp</i> , graphing functions: <i>plot</i> , <i>loglog</i> , <i>semilogx</i> , <i>semilogy</i> , <i>title</i> , <i>xlabel</i> and <i>ylabel</i> , control flow functions: <i>for</i> , <i>if</i> , <i>else</i> , and <i>end</i> , comments. |  |

---

such an early stage of studying computation that results obtained from the computer should never be blindly accepted but must be critically examined for physical feasibility.

The example is built up in stages. At first, the calculations are carried out for one set of input data: temperature and pressure. In the second step, the calculations are carried out for about a hundred different combinations of temperature and pressure, and in the third step the results of the calculations are presented in tabular and graphical forms.

The mathematical model of the example is first set up using the "Nonlinear Equation Solver" program of the POLYMATH package. POLYMATH is the easiest to use for this purpose because it requires minimal modification of the equations and the variable names. The equations can be entered in the same order as they appear in the problem definition, even if the calculation order is different, since POLYMATH reorders the equations in the right calculation sequence. The program also issues warnings for undefined variables, so that common, difficult to detect errors, like using 0 (zero) in the variable name in one place and the letter "o" somewhere else, can be easily detected.

The POLYMATH model of the problem is solved for a few selected values of temperatures and pressures and the results are compared with literature data, in order to verify the correctness of the model. After that we proceed to convert the model for solution by Excel and MATLAB.

The conversion to Excel involves "pasting" the POLYMATH model into two adjacent columns of an Excel worksheet. In the first column the POLYMATH model is retained as is, for documentation purposes. On the second column the equations are converted to Excel formulas, essentially by changing the variable

names to cell addresses and changing the definitions of the *if* statements and some intrinsic functions. The conversion proceeds line by line where the correctness of every finished line can be verified by comparing the resultant numerical value with the POLYMATH result.

The conversion to MATLAB is done first in an interactive mode. The ordered equations are copied one by one from the POLYMATH solution report and pasted into the MATLAB command window, for one set of temperature and pressure values. The intrinsic function names and syntax are changed as needed. This, again, enables line by line verification of the MATLAB model. The finished model can be copied from the command window and pasted into the MATLAB editor to create an m-file. This concludes the first part of the example, which includes the preparation and verification of the mathematical model of the problem with POLYMATH, Excel, and MATLAB.

In the second part, the calculations are carried out for about a hundred different combinations of temperature and pressure. Only Excel and MATLAB are used in this part. In Excel the column containing the formulas of the model can be duplicated as many times as needed while providing the addresses of the pertinent temperature and pressure values. However, the most convenient and effective way to carry out the calculations is by using the "Two-Input Data Table."

In MATLAB, several new concepts must be learned in order to carry out repetitive calculations while changing the parameter values. The m-file containing the model of the problem must be converted into a user-defined function. A separate main program (m-file) must be written where the parameters are stored in one-dimensional arrays and the function is called repeatedly with different parameter values by

means of *for* commands. The results are stored in a two-dimensional array and various display and plot commands are used to display the results in tabular and graphic forms (linear and logarithmic scales).

In this first chapter, special emphasis is put on good engineering and programming practices. The students are expected to describe all the variables, including the units used next to the equation in which the variable is calculated and also when displaying its numerical value in the results. In the graphs prepared, the students are expected to mark clearly the axes, to use scaling appropriate to the order of magnitude of the variables, to remove unused graph areas that often generated by the automatic scaling routines of the software, and to use right proportion between the graph and the letter sizes. In MATLAB, good programming practice requires clearing the workspace and the command window before starting the execution and explicitly defining the preferred format for printing the results.

The Redlich–Kwong equation of state is used in all the class examples presented. The homework assignments may include different cubic equations of state (Soave–Redlich–Kwong or Peng–Robinson) or calculation of the adiabatic flame temperature ([4], this problem can be represented by a cubic equation if the heat capacities are correlated with 2nd degree polynomials).

This chapter represents, for most of the students, the first introduction to the use of the POLYMATH package, to computer (MATLAB) programming, and to the use of the three packages for engineering problem solving. Most of the basic material of the course is included in this chapter and it takes about one third

of the course's time (12–15 h of lectures and recitation sessions) to complete it.

## ITERATIVE SOLUTION OF A NONLINEAR EQUATION

The objective of this chapter is to introduce the students to the concept of iterative, numerical solution of problems. The contents of this chapter are presented in Table 2. The two class examples used in this chapter are the calculation of flow rate in a pipeline (Cutlip and Shacham [4]) and the calculation of outlet temperatures in a countercurrent flow-heat exchanger. The first example involves the solution of the Bernoulli equation for the fluid velocity where the pressure drop, the elevation difference, the length, and the diameter of the pipe are specified. The POLYMATH nonlinear equation solver is used for setting up the problem and solving the equations for a few selected combinations of pipe lengths and diameters. At this point the concept of *initial estimate*, which is essential when iterative solution techniques are used, is introduced and the selection of a physically sensible initial estimate is emphasized.

After that the model equation are converted to Excel formulas and at first, the “Goal Seek” tool of Excel is used for solving the nonlinear equation. But the use of Goal Seek is tedious if the calculation has to be repeated many times for different parameter (pipe length and diameter, in this case) values. For repetitive calculations the *successive substitution* method is introduced. Using this method requires rewriting the Bernoulli equation so as to generate a converging

**Table 2** Contents of Chapter 2, “Iterative Solution of a Nonlinear Equation”

---

|   |   |
|---|---|
| Chapter 2. Iterative Solution of a Nonlinear Equation |   |
| Mathematical and numerical concepts demonstrated      |   |
|   | Iterative methods of solution: successive substitution and secant, selection of the initial guess, stopping criterion, error tolerance, attainable accuracy |
| Problems used in the examples, assignments, and exams |   |
|   | Calculation of the flow rate in a pipeline (problem 5.10 in Cutlip and Shacham [4])   |
|   | Terminal velocity of falling particles (problem 5.6 in Cutlip and Shacham [4])  |
|   | Gas volume calculations using various equations of state (problem 1.8 in Cutlip and Shacham [4])  |
|   | Bubble point, dew point and isothermal flash of an ideal multi-component mixture (problems 1.9, 1.10, 4.8 and 4.9 in Cutlip and Shacham [4])                |
|   | Countercurrent heat exchanger   |
|   | Adiabatic flame temperature in combustion (Shacham, 1998 [10] problem 1.12 in Cutlip and Shacham [4])   |
| POLYMATH program                                      |   |
|   | Nonlinear equations solver  |
| Excel options and functions                           |   |
|   | Goal seek   |
| MATLAB options and functions                          |   |
|   | User defined functions, global variables, control flow function: <i>while</i>   |

---

sequence of iterations. The *attainable accuracy* in iterative solutions is discussed and the concept of the *stopping criterion* based on an *error tolerance* is introduced.

The new material learned in MATLAB includes the *global variables*, which are used to transfer the current parameter values to the user defined function and the *while* statement which is used to stop the iterations when the stopping criterion is satisfied or the maximum allowed number of iterations exceeded.

The countercurrent heat exchanger problem involves the calculation of the outlet temperatures and the heat exchanger duty when the inlet temperatures, fluid properties, and heat transfer area and coefficients are specified. The problem is converted into the form of a single nonlinear equation by solving the cold fluid and hot fluids heat balance equations for the unknown temperatures and the iterations are carried out on the heat duty only.

The determination of an initial guess for the heat duty represents a special challenge in this case and it provides an excellent example of physical and mathematical feasibility of the initial guess. If too high heat duty is specified as an initial guess (or reached during the iterations) that may lead to higher exit temperature of the cold fluid than the inlet temperature of the hot fluid which is physically impossible. At the same time such temperatures require calculation of the logarithm of a negative number (in the expression of the logarithmic mean temperature difference) which is computationally impossible.

For most combinations of the input data values the successive substitution technique diverges and this provides the opportunity to introduce the *secant* method. Similarly to successive substitution, this method is easy to program and to use and it converges

reasonably fast for all the examples and assignments that included in this chapter.

The homework assignments and exams for this chapter include solving various equations of state (cubic or higher order) for the molar volume, calculating terminal velocity of falling particles, bubble point, dew point and isothermal flash calculations for ideal multi-component mixtures, and calculating adiabatic flame temperature in combustion.

### MULTIPLE-LINEAR, POLYNOMIAL, AND NONLINEAR REGRESSION

The contents of this chapter are shown in Table 3. The objective of the chapter is to enable the students to use POLYMATH, Excel, and MATLAB to fit a straight line, a polynomial, and a linear or a linearizable nonlinear regression model to data and to analyze the quality of the fit using the variance, the standard deviation, the correlation coefficient, the 95% confidence intervals on the parameters and the residual plot. This chapter is built around three examples “Fitting a Straight Line to Thermal Conductivity Data,” “Heat Evolved During the Hardening of Portland Cement” (Daniels and Wood, [11]), and “Fitting Polynomials and Correlation Equations to Vapor Pressure Data” (Shacham et al. [12,4]).

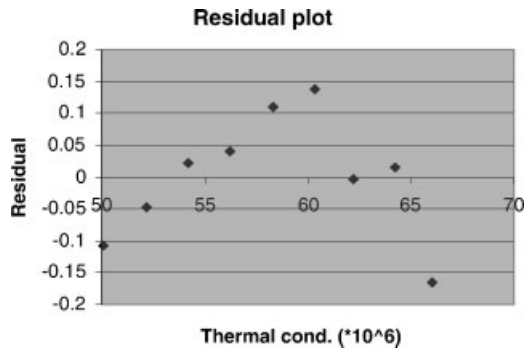
Solving the straight-line example by POLYMATH serves as an introduction to the basic concepts: model parameters, confidence intervals, the correlation coefficient, the variance, and the residual plot. All of those are automatically prepared and displayed by POLYMATH. In the thermal conductivity example all the indicators show excellent fit (the correlation coefficient is 0.9997) however, the residual plot

**Table 3** Contents of Chapter 4, “Multiple-Linear, Polynomial and Nonlinear Regression”

---

|   |  |
|---|--|
| Chapter 4. Multiple-Linear, Polynomial and Nonlinear Regression |  |
| Mathematical and numerical concepts demonstrated                |  |
|   | Representing the least squares regression problem by the normal equations, linearization of nonlinear correlations by transformation of variables, using confidence intervals and residual plots for assessing the goodness of the fit |
| Problems used in the examples, assignments, and exams           |  |
|   | Correlation of thermodynamic and physical properties of selected compounds (problems 2.3 and 2.4 in Cutlip and Shacham [4])  |
|   | Fitting polynomials and correlation equations to vapor pressure data (problems 1.3 and 1.4 in Cutlip and Shacham [4])  |
|   | Heat of hardening of cement (Daniel and Wood [11])   |
| POLYMATH program  |  |
|   | Regression   |
| Excel options and functions                                     |  |
|   | <i>Linest</i> , <i>data analysis package</i> , <i>regression</i> , <i>trendline</i>  |
| MATLAB options and functions                                    |  |
|   | Row-wise computation of the normal matrix, the functions <i>mean</i> , <i>num2str</i> , <i>int2str</i>   |

---



**Figure 1** Residual plot for a straight line fit to thermal conductivity data.

(shown in Fig. 1) exhibits a curvature that cannot be explained by a straight line. This demonstrates for the students the importance of the residual plot in the analysis of the goodness of the fit.

In Excel the LINEST function is used for obtaining the coefficient values, their standard deviation, and the standard error of the dependent variable (thermal conductivity). Those are converted to the confidence intervals and to the variance, respectively, for the sake of consistency between the various software packages.

To solve by MATLAB, the normal equations are first derived and solved using Cramer's rule. The equations for calculating the variance, the correlation coefficient and the residuals are provided and the complete set is entered as an m-file.

The "Heat of Hardening" example involves multiple linear regression where the independent variables are the weight percents of the various components of the Portland cement and the dependent variable is the heat released during the hardening process. This problem has been extensively used in the statistical literature because of the interesting point that when attempting to fit a linear model that includes all four independent variables and a free parameter, the resultant model is ill conditioned (all 95% confidence intervals are larger, in absolute value, than the respective parameter values). Physical considerations dictate to set the free parameter at zero. When the free parameter is set at zero a well behaving, valid model is obtained. This demonstrates, for the students, the importance of the confidence intervals in the analysis of the goodness of the fit.

The solution process in POLYMATH and Excel is not significantly different than that for the straight line. In MATLAB the normal matrix is inverted using the *inv* intrinsic function instead of using Cramer's rule, as in the first example.

The third example that involves correlation of vapor pressure versus temperature data demonstrates

the concepts of polynomial regression and the linearization of nonlinear regression models so that the coefficients can be obtained using multiple linear regression. In polynomial regression the students are taught to select the most adequate polynomial degree by consecutive regressions using increasing degree polynomials until randomly distributed residuals obtained while making sure that all the confidence intervals are smaller in absolute value than the respective parameter values. Using too many polynomial terms demonstrate the concepts of over-fitting and ill-conditioning of the regression problem, which results in extremely wide confidence intervals. Fitting Clapeyron's and Riedel's equations to the data requires transformation of the dependent variable (using logarithm of the pressure) and transformations of the independent variable (reciprocal and logarithm of the absolute temperature). More details of the vapor pressure correlation example are available in Reference [12].

## INTERNET-BASED HOMEWORK ASSIGNMENTS

The problem definition for a typical Internet-based homework assignment is shown in Figure 2. In this assignment the bubble and dew point temperatures of a multi-component mixture have to be calculated at different pressures. Some of the data for the assignment (the mole fractions of the various components) are generated randomly so that different students get slightly different assignments and a student can redo the assignment with different data. The assignment involves repeated solutions of a nonlinear equation and it has to be solved using POLYMATH, and then continuing either with MATLAB or Excel.

After solving the assignment the student enters the self-test, using his identification number, in order to obtain a feedback and the grade for the assignment. The first part of the self-test is shown in Figure 3. The student is asked to input the calculated bubble and dew point temperatures at a particular pressure. If his answers are correct he receives the highest grade for this part of the self-test and can go on to the second part. The grade is shown to the student and it is also sent to a database and added to his record. If one (or more) of the answers is incorrect, follow-up questions are presented (see Fig. 4). The feedback provided in most cases is the correct numerical values. Those values give the student the information needed to enable him to find the mistake he made and correct it. After correcting the mistake he can redo the self-test, but this time he will receive a different set of pressure values to prevent him from inputting the feedback he

Introduction to Personal Computers

Homework Assignment No. 4

Bubble Point and Dew Point for an Ideal Multi-component Mixture

For the four component mixture whose composition (mole fractions,  $z_j$ ) is given below

|           |            |          |           |           |
|-----------|------------|----------|-----------|-----------|
| Methane   | Ethane     | Propane  | n-Butane  | n-Pentane |
| 0.0       | 0.0        | 0.0      | 0.0       | 0.1       |
| n- Hexane | n- Heptane | n-Octane | n- Nonane | n- Decane |
| 0.2       | 0.3        | 0.4      | 0.0       | 0.0       |

1. Calculate the bubble point temperature ( $T_b$ ) and the dew point temperature ( $T_d$ ) at the following pressures: P (atm) = 1,5,10,15, 20, 30, 50, 70, 100 and 150. Use the secant method to solve the nonlinear equations. As initial estimates use the boiling point temperature at the specified pressure of the most volatile and least volatile compounds. Stop the iterations when the error  $\epsilon = |T_{i+1} - T_i|$  (where  $i$  is the iteration number) becomes smaller than  $10^{-5}$  or the number of iterations exceeds 20.
2. Summarize the results in a tabular form and prepare a graph of  $T_b$  and  $T_d$  versus pressure. In preparing the graphs pay attention to proper labeling of the axes (variable name, symbol and units), proper size ratio of the plot area and the width of the lines and the letters, setting minimal and maximal values for the axes so that there are no unused graph areas and using logarithmic scale whenever necessary.

Figure 2 Problem definition for an Internet-based homework assignment.

received in the previous trial. This cycle of self-test, feedback, and correction can be repeated as many times as needed, to ensure that the student has full understanding of the problem and knows how to solve it correctly. After arriving at the correct solution he submits the report on the assignment, including the programs that were used for solving it, electronically, so that the non-numerical aspects of the report can also be checked.

The homework assignments represent only 20% of the total grade of the course, yet most students invest considerable effort to get all homework assignments perfect.

MIDTERM AND FINAL EXAMS

The midterm and the final exams of the course are held in a computer laboratory, under supervision, where the students have to present their identity card to take the exam. The duration of the midterm exam is 2 h and the final exam is 3 h. The questions presented in the exams are similar to those of the homework

assignments and every student gets a different set of numerical data for the exam. The problems are solved using the software package specified in the problem statement and the student grades his own solution using a self-test utility. The feedback provided by the utility is similar to that of the homework assignments but the student is allowed to take the self-test for a particular question only twice (if time permits). The report on the solution is submitted electronically and some of the exams are checked and re-graded manually, to ensure that the grade given by the self-test utility is fair and proper credit is given to the various questions even if an incorrect numerical value is carried over to the subsequent answers.

DISCUSSION AND CONCLUSIONS

The particular structure of the introductory course in modeling and computation has evolved over several years. One of the latest revisions took place 2 years ago when it was decided to extend the course by

**Self Test for Homework Assignment No.4**  
Introduction to Personal Computers Exit Self Test

---

Question 1

Question 2

**Bubble Point and Dew Point for an Ideal Multi-component Mixture**

Component 1: n-Butane                      Mole fraction: .25  
 Component 2: n-Pentane                    Mole fraction: .35  
 Component 3: n-Hexane                     Mole fraction: .25  
 Component 4: n-Heptane                   Mole fraction: .15  
 Pressure (atm): 20

|     |                               | Value Entered | Correct Value | Correct/Incorrect |
|-----|-------------------------------|---------------|---------------|-------------------|
| 1.2 | Bubble point temperature (°C) | 164           | 164.39701     | Correct           |
|     | Dew point temperature (°C)    | 193           | 193.24769     | Correct           |

Your grade for question 1 is 50 out of 50

Figure 3 Self-test for the homework assignment.

|     |   | Value Entered | Correct Value | Correct/Incorrect |
|-----|---|---------------|---------------|-------------------|
| 1.1 | Boiling point temperature of n-Butane (°C)      | 118           | 117.5704      | Correct           |
|     | The value of f(Tb) at this point                | -0.2          | -.54315       | Incorrect         |
|     | Boiling point temperature of n-Heptane (°C)     | 249           | 249.36181     | Correct           |
|     | The value of f(Tb) at this point                | 1             | 1.85316       | Incorrect         |
|     | The first temp. calculated by the secant method | 130           | 147.44201     | Incorrect         |

Your grade for question 1 is 4 out of 50

Figure 4 Follow-up questions in the self-test.

adding MATLAB to its contents and simultaneously to remove the required, 3 h/week FORTRAN programming course ChE from the program. At that year MATLAB was taught after finishing the study of POLYMATH and Excel and the similarities between the different packages were not exploited in order to enhance the learning effectiveness. The midterm and final exams showed high success rates when POLYMATH and Excel were used but very low success rate in using MATLAB.

Consequently, the course was revised this last year, adding the *multi-stage* and *programming by modification* aspects. The course was given to a class of 60 students in the spring semester of 2003. The final exam of the course included the use of the “secant” method for solution of the adiabatic flame temperature problem (Shacham, [10]) with MATLAB programming. The students had an hour and a half to complete this assignment and 75% of them (40 students) managed to get a working program that yielded the correct solution. In our experience this is an unusually high rate of success in MATLAB programming for chemical engineering students.

No formal evaluation of the course has been carried out yet and it is probably too early to arrive at definite conclusions. However, the results of the exams and informal discussion with the students indicate that this form of the course was very successful in enabling them to use the three CBPS packages with the wide range of capabilities. In addition to acquiring the ability to use the packages they were also able to select a particular package that is most suitable for carrying out a particular task and became aware of the importance of CBPS.

It is to be hoped that by correcting the false believe that CBPS requires mostly coding and debugging its application can be considerably extended in engineering education and practice.

## REFERENCES

- [1] J. C. Kantor and T. Edgar, Computing skills in the chemical engineering curriculum. In: B. Carnahan, editor. Computers in chemical engineering education, CACHE, Austin, TX, 1996, pp 9–20.
- [2] J. B. Jones, The non-use of computers in undergraduate engineering science courses, *J Eng Educ* 88 (1998), 11–14.
- [3] J. F. Davis, G. E. Blau, and G. V. Reklaitis, Computers in undergraduate chemical engineering education: A perspective in training and application, *Chem Eng Educ* 29 (1999), 26–31.
- [4] M. B. Cutlip and M. Shacham, Problem solving in chemical engineering with numerical methods, Prentice Hall, Upper Saddle River, NJ, 1999.
- [5] H. S. Fogler, Elements of chemical reaction engineering, 3rd ed., Prentice Hall, Upper Saddle River, NJ, 1999.
- [6] D. M. Himmelblau, Basic principles and calculations in chemical engineering, 6th ed., Prentice Hall, Upper Saddle River, NJ, 1999.
- [7] B. G. Kyle, Chemical and process thermodynamics, 3rd ed., Prentice Hall, Upper Saddle River, NJ, 1999.
- [8] M. Shacham, N. Brauner, and M. B. Cutlip, Efficiently solve complex calculations, *Chem Eng Prog* 99 (2003), 56–61.
- [9] M. Shacham, N. Brauner, and M. B. Cutlip, An exercise for practicing programming in the ChE curriculum—Calculation of thermodynamic Properties using the Redlich–Kwong equation of state, *Chem Eng Educ* 27 (2003), 148–152.
- [10] M. Shacham, Computer based exams in undergraduate engineering courses, *Comput Appl Eng Educ* 6 (1998), 201–209.
- [11] C. Daniel and F. S. Wood, Fitting equations to data, 2nd ed., Wiley, New York, 1980.
- [12] M. Shacham, N. Brauner, and M. B. Cutlip, Replacing the graph paper by interactive software in modeling and analysis of experimental data, *Comput Appl Eng Educ* 4 (1996), 241–251.

**BIOGRAPHY**

**Mordechai Shacham** received his BSc (1969) and DSc (1973) from the Technion, Israel Institute of Technology. He is currently a professor of chemical engineering and chairman of the committee for promoting e-learning at the Ben-Gurion University of the Negev, Beer-Sheva, Israel, where he has served since 1974 at every academic level including two 4-year terms as department head. His research interests include analysis, modeling, and regression of data, applied numerical methods, computer-aided instruction, and process simulation, design, and optimization. He is coauthor of the POLYMATH numerical software package and the textbook *Problem Solving in Chemical Engineering with Numerical Methods*.