
AC 2011-1615: TEACHING SPREADSHEET-BASED NUMERICAL ANALYSIS WITH VISUAL BASIC FOR APPLICATIONS AND VIRTUAL INSTRUMENTS

Nikunja Swain, South Carolina State University

Dr. Swain is currently a Professor at the South Carolina State University. Dr. Swain has 25+ years of experience as an engineer and educator. He has more than 50 publications in journals and conference proceedings, has procured research and development grants from the NSF, NASA, DOT, DOD, and DOE and reviewed number of books on computer related areas. He is also a reviewer for ACM Computing Reviews, IJAMT, CIT, ASEE, and other conferences and journals. He is a registered Professional Engineer in South Carolina.

Teaching Spreadsheet-Based Numerical Analysis with Visual Basic for Applications and Virtual Instruments

Abstract

LabVIEW, EXCEL and VBA are currently used in a number of engineering schools and industries for simulation and analysis. By introducing virtual instrumentation (LabVIEW) and EXCEL/VBA to the existing laboratory facilities and course(s) the students can be well trained with the latest design techniques and computer aided instrumentation, design and process control used throughout industry. This will also allow the students greater interaction with the subject matter and improve his/her skills in the use of number of applied engineering software packages. This paper will discuss design and development of interactive instructional modules for Numerical Analysis Course using LabVIEW, EXCEL and VBA.

Introduction

The students' over reliance upon formulas and routine use of technique in problem solving too often lead to poor performance in advanced courses and a high attrition rate in the engineering, technology, and science programs. The students' lack of comprehension of mathematical concepts results in time wastage during laboratory experiments, misinterpretations of lab data and underachievement in standardized science and engineering tests that stress the fundamentals^{1, 2, 3, 4}. This problem can be effectively addressed by improving the student's conceptual understanding and comprehension of the topics covered in introductory science and technology courses. One way to achieve this is through interactive learning and teaching and upgrading the existing laboratories with modern equipment. This will require increased funding and resources. But in recent years there is a decrease in resource allocation making it increasingly difficult to modernize the laboratories to provide adequate levels of laboratory and course work and universities are under pressure to look for alternative cost effective methods. One way to achieve this is through interactive learning and teaching through the use of software packages like LabVIEW (Virtual Instruments), Excel and Visual Basic for Applications (VBA). These programs along with MatLAB, Mathematica and Maple are used to teach courses such as Numerical Analysis and Engineering Problem Solving. This paper discusses some of the numerical analysis instructional modules using LabVIEW, EXCEL and VBA.

Numerical Analysis Instructional Modules using LabVIEW

Approximately 10 to 15 years ago, National Instruments Corporation introduced a new program called LabVIEW. The acronym stands for Laboratory Virtual Instrumentation Engineering Work Bench. Originally designed for test and measurement applications, the program has been modified over the years to design and analyze various complex systems. LabVIEW is a graphical programming environment and is based on the concept of data flow programming. The data flow programming concept is different from the sequential nature of traditional programming languages, and it cuts down on the design and development time of an application. It is widely accepted by industry, academia, and research laboratories around the world as a standard for data acquisition and instrument control software. Since LabVIEW is based on graphical programming, users can build instrumentation called virtual instruments (VIs) using software objects. With proper hardware, these VIs can be used for remote data acquisition,

analysis, design, and distributed control. The built-in library of LabVIEW has a number of VIs that can be used to design and develop any system. LabVIEW can be used to address the needs of various courses in a technology and science curriculum^{6, 7, 8, 9}.

LabVIEW Application Areas

LabVIEW is extremely flexible and some of the application areas of LabVIEW are Simulation, Data Acquisition, and Data Processing. The Data Processing library includes signal generation, digital signal processing (DSP), measurement, filters, windows, curve fitting, probability and statistics, linear algebra, numerical methods, instrument control, program development, control systems, and fuzzy logic. These features of LabVIEW will help provide an interdisciplinary, integrated teaching and learning experience that integrates team-oriented, hands-on learning experiences throughout the engineering technology and sciences curriculum, engaging students in the design and analysis process beginning with their first year.

The Mathematics VIs of LabVIEW are located on the **Functions»Analyze»Mathematics** palette. These VIs can be used to perform many different kinds of mathematical calculations. The following is a listing of VIs in LabVIEW:

1D and 2D Evaluation VIs, Linear Algebra VIs, Array Operations VIs, Numeric Functions VIs
Calculus VIs, Optimization VIs, Curve Fitting VIs Probability and Statistics VIs

Formula VIs, and Zeroes VIs. The user has to provide appropriate inputs and outputs in the LabVIEW Control Panel and make required connections in the LabVIEW diagram panel to simulate these VIs.

Example VI to solve system of linear equations

This VI solves the following linear equations:

$$\begin{aligned}5x_1 + x_2 + 3x_3 &= 5 \\2x_1 + 7x_2 + 9x_3 &= 4 \\8x_1 + 6x_2 + 4x_3 &= 9\end{aligned}$$

The linear equations are written in Matrix Form ($Ax = B$) form and then A and B (known vector) are supplied as inputs to the VI. The VI solves for the roots and displays the results as shown in Figure 1. The VI is flexible and can be easily modified to accommodate more number of equations by simply changing the dimension of A, B, and solution vector.

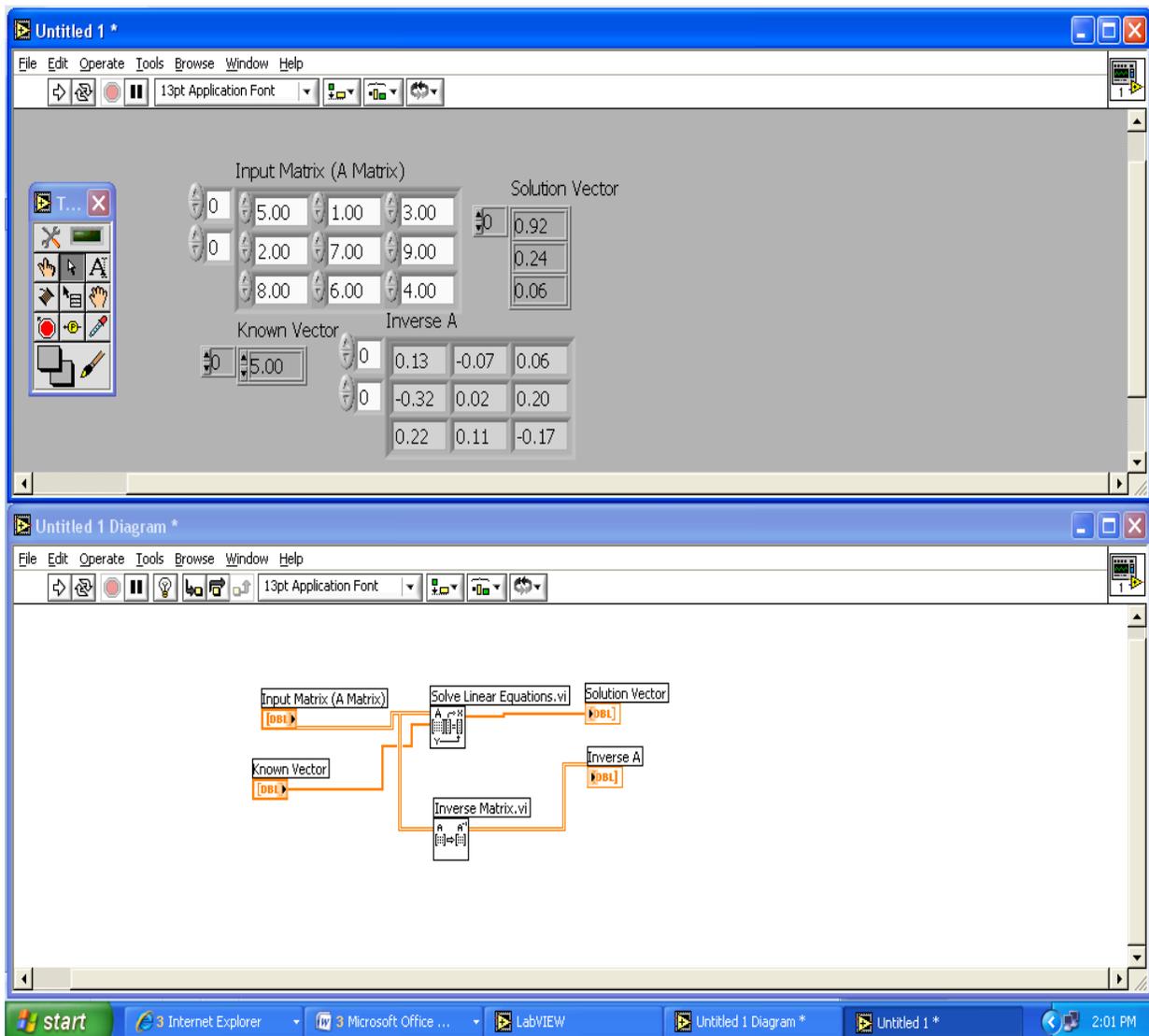


Figure 1 – VI to solve linear equations

Example VI to perform LU Decomposition

LU decomposition method has its place in the solution of simultaneous linear equations⁵. Research shows that LU decomposition method is computationally more efficient than Gaussian elimination. LU decomposition takes less computational time to find the inverse of a matrix. Typical values of the ratio of the computational time for different values of n are given in Table 1.

Table 1 Comparing computational times of finding inverse of a matrix using LU decomposition and Gaussian elimination.

n	10	100	1000	10000
$CT _{inverseGE} / CT _{inverseLU}$	3.28	25.83	250.8	2501

Figure 2 below presents the LU decomposition VI. This VI is found in Linear Algebra VI section of Mathematics palette of LabVIEW. The user has to provide the A matrix in control panel of LabVIEW and make appropriate connections in the diagram panel.

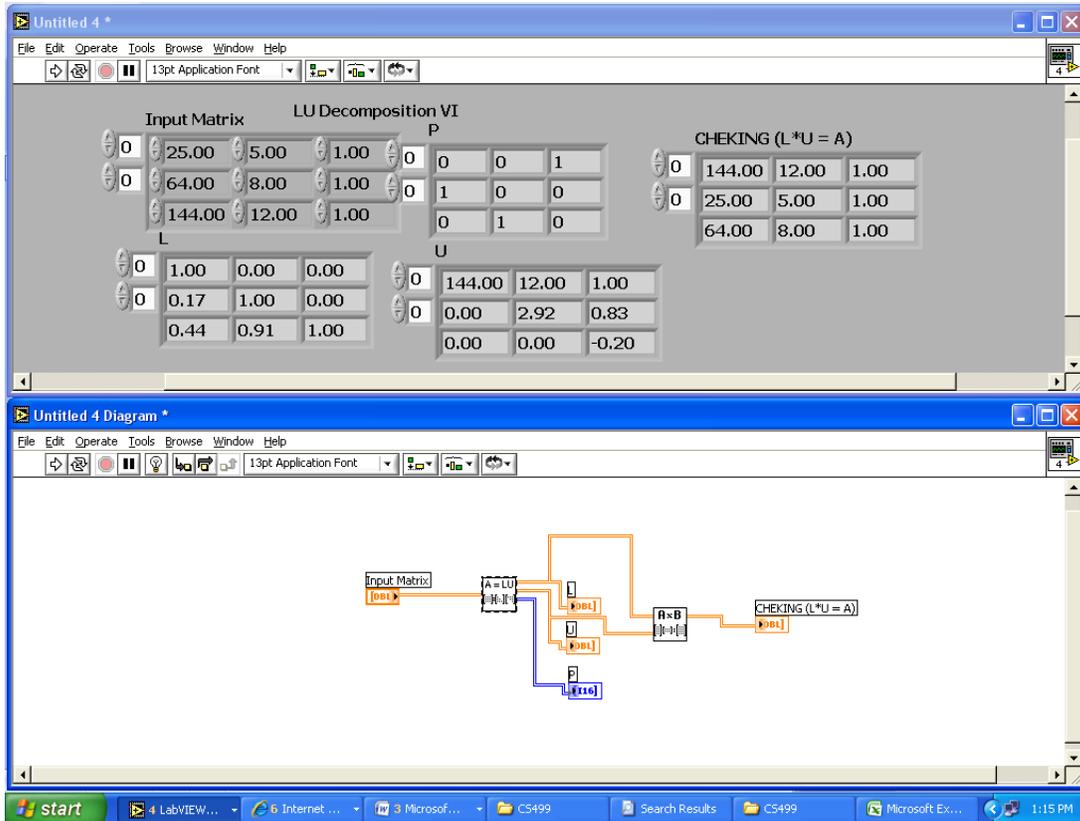


Figure 2 – LU Decomposition VI

Numerical Analysis Instructional Module using Visual Basic for Applications (VBA)

The Newton-Raphson method is based on the principle that if the initial guess of the root of $f(x) = 0$ is at x_i , then if one draws the tangent to the curve at $f(x_i)$, the point x_{i+1} where the tangent crosses the x -axis is an improved estimate of the root⁵.

Using the definition of the slope of a function, at $x = x_i$

$$f'(x_i) = \tan \theta$$

$$= \frac{f(x_i) - 0}{x_i - x_{i+1}},$$

which gives

$$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)} \tag{1}$$

Equation (1) is called the Newton-Raphson formula for solving nonlinear equations of the form $f(x)=0$. So starting with an initial guess, x_i , one can find the next guess, x_{i+1} , by using Equation (1). One can repeat this process until one finds the root within a desirable tolerance.

Algorithm

The steps of the Newton-Raphson method to find the root of an equation $f(x)=0$ are

1. Evaluate $f'(x)$ symbolically
2. Use an initial guess of the root, x_i , to estimate the new value of the root, x_{i+1} , as

$$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)}$$

3. Find the absolute relative approximate error $|\epsilon_a|$ as

$$|\epsilon_a| = \left| \frac{x_{i+1} - x_i}{x_{i+1}} \right| \times 100$$

4. Compare the absolute relative approximate error with the pre-specified relative error tolerance, ϵ_s . If $|\epsilon_a| > \epsilon_s$, then go to Step 2, else stop the algorithm. Also, check if the number of iterations has exceeded the maximum number of iterations allowed. If so, one needs to terminate the algorithm and notify the user.

Figure 3 represents the VBA screen shot for solving $f(x) = 3x^3 + 3x - 1$ using Newton's method, and Figure 4 represents the simulation results. The data for the simulation is entered in EXCEL spread sheet. The VBA module gets the input data from the EXCEL spreadsheet, performs the iterations, and displays the result in Excel spread sheet

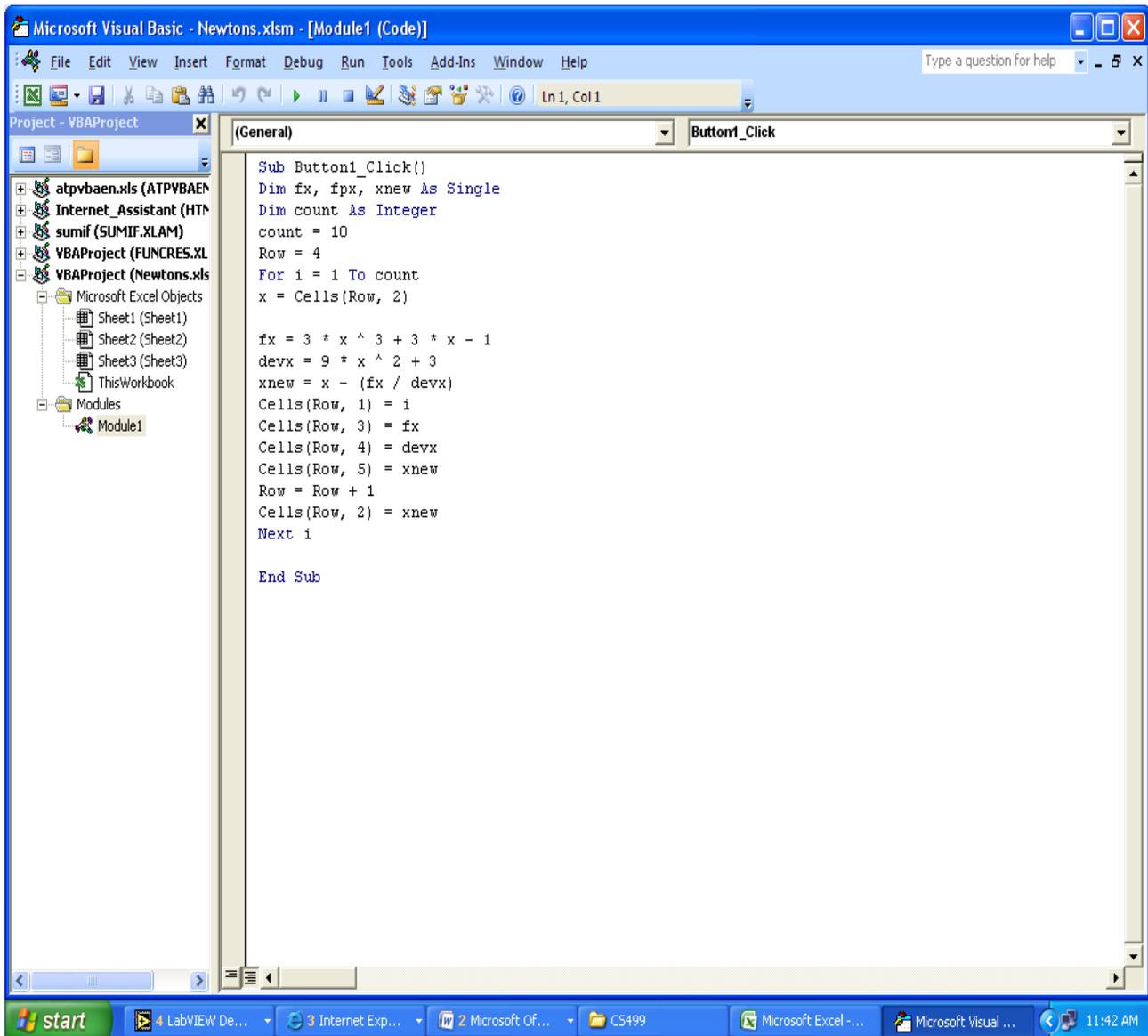


Figure 3 – VBA code for Newton method

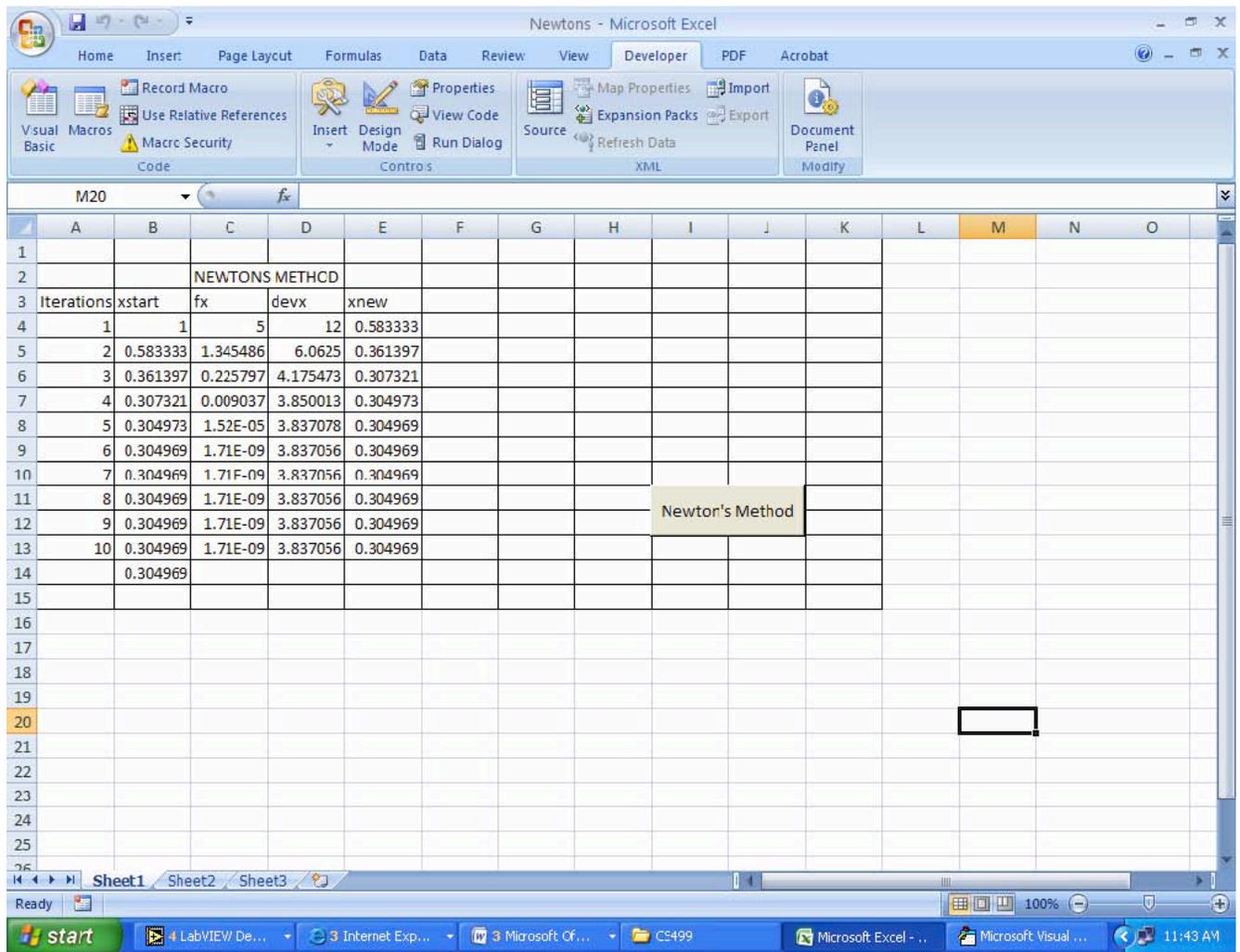


Figure 4 – Results of VBA simulation

Numerical Analysis Instructional Module using EXCEL

Many a times, a function $y = f(x)$ is given only at discrete points such as $(x_0, y_0), (x_1, y_1), \dots, (x_{n-1}, y_{n-1}), (x_n, y_n)$. How does one find the value of y at any other value of x ? Well, a continuous function $f(x)$ may be used to represent the $n + 1$ data values with $f(x)$ passing through the $n + 1$ points. Then one can find the value of y at any other value of x . This is called interpolation. Of course, if x falls outside the range of x for which the data is given, it is no longer interpolation but instead is called extrapolation. Of course, if x falls outside the range of x for which the data is given, it is no longer interpolation but instead is called *extrapolation*. So what kind of function should one choose? A polynomial is a common choice for an interpolating function because polynomials are easy to (A) evaluate, (B) differentiate, and (C) integrate, relative to other choices such as a trigonometric and exponential series. Polynomial interpolation involves finding a polynomial of order n that passes through the $n + 1$ data points. One of the methods used to find this polynomial is called the Lagrangian method of interpolation. Other methods include Newton's divided difference polynomial method and the direct method.

Figures 5 and 6 demonstrate the use of Direct Interpolation and Lagrangian method of interpolation in Excel. The results agree with the theoretical solution provided in ⁵. The problem selected for this example is form ⁵ mentioned above and is as follows:

The upward velocity of a rocket is given as a function of time in Table 1. Determine the value of the velocity at $t = 16$ seconds using (a) Direct Method and (b) Lagrange polynomial method. .

Table 1: Velocity as a function of time.

t (s)	$v(t)$ (m/s)
0	0
10	227.04
15	362.78
20	517.35
22.5	602.97
30	901.67

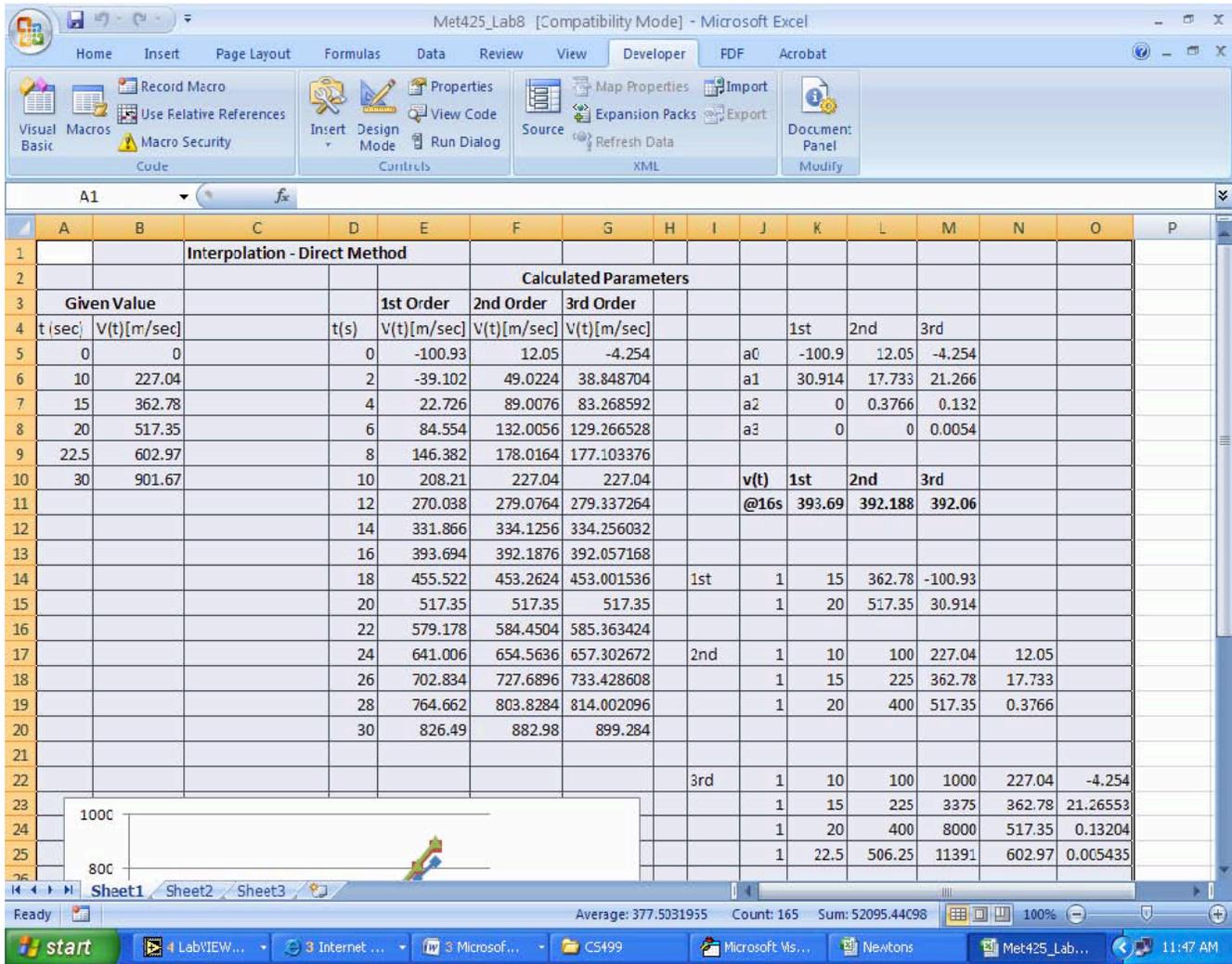


Figure 5 – Excel spreadsheet results for Direct Method

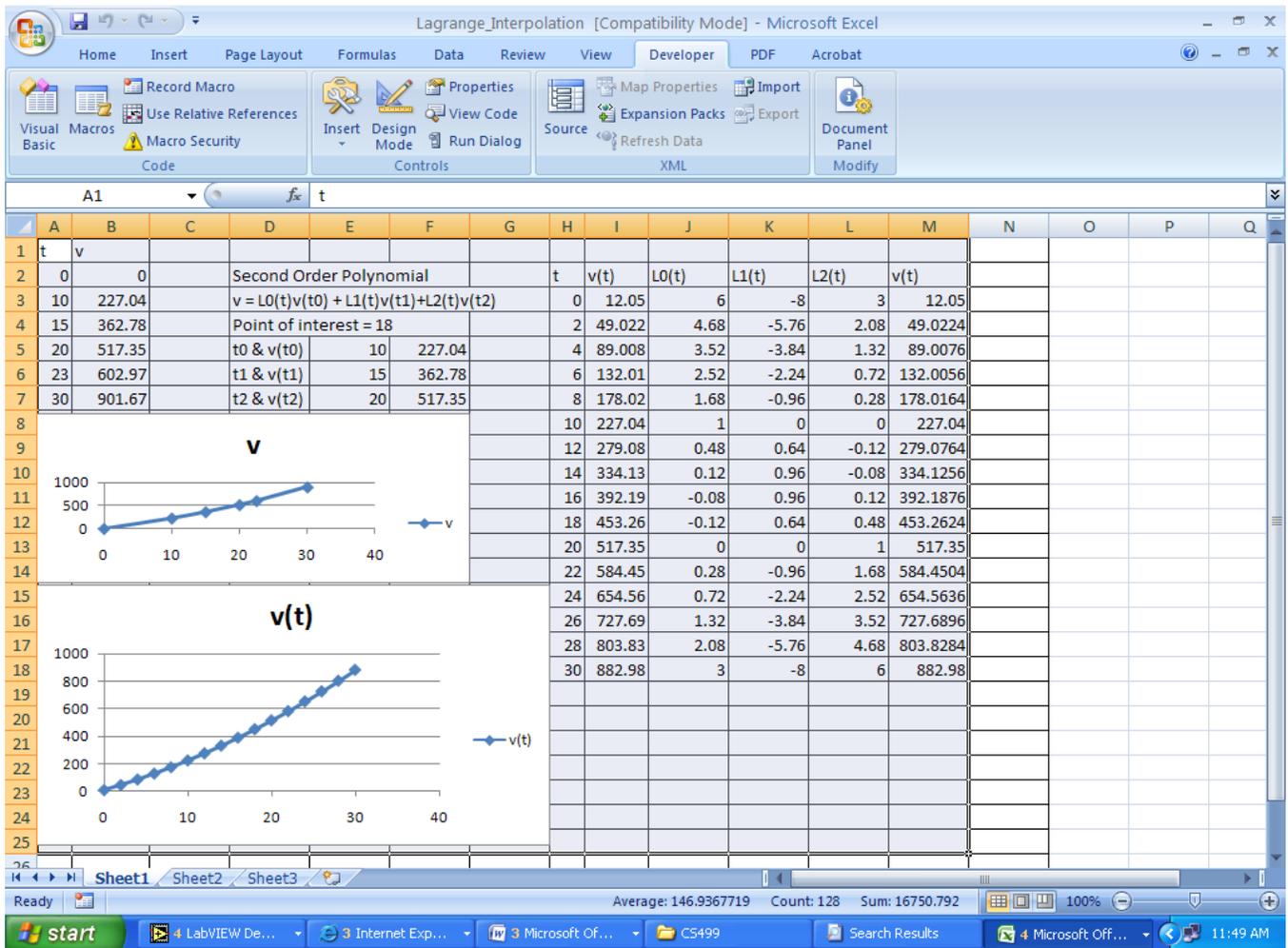


Figure 6 – Excel Spreadsheet results for Lagrange polynomial.

Summary and Conclusions

We have developed number of modules using EXCEL, VBA, and LabVIEW for numerical analysis and engineering problem solving courses. The sample modules presented above were user friendly and performed satisfactorily under various input conditions. These and other modules (Bisection Method, Runge-Kutta method, Secant Method, Numerical Integration, etc) helped the students to understand the concepts in more detail. These modules can be used in conjunction with other teaching aids to enhance student learning in various courses and will provide a truly modern environment in which students and faculty members can study engineering, technology, and sciences at a level of detail.

Acknowledgement

This work was funded in part by a grant from the NSF-HBCU-UP/RISC grant. We are thankful to the NSF for providing us with this help..

References

1. Swain, N. K., Korrapati, R., Anderson, J. A. (1999) "Revitalizing Undergraduate Engineering, Technology, and Science Education Through Virtual Instrumentation", NI Week Conference, Austin, TX..
2. Elaine L., Mack, Lynn G. (2001), "Developing and Implementing an Integrated Problem-based Engineering Technology Curriculum in an American Technical College System" Community College Journal of Research and Practice, Vol. 25, No. 5-6, pp. 425-439.
3. Buniyamin, N, Mohamad, Z., 2000 "Engineering Curriculum Development: Balancing Employer Needs and National Interest--A Case Study" – Retrieved from ERIC database.
4. Kellie, Andrew C., And Others. (1984), "Experience with Computer-Assisted Instruction in Engineering Technology", Engineering Education, Vol. 74, No. 8, pp712-715.
5. URL: <http://numericalmethods.eng.usf.edu/>
6. Anderson, J. A., Korrapati. R. B., & Swain. N. K., "Digital signal processing using virtual instrumentation". *Proceedings of SPIE Vol. 4052*.
7. Korrapati, R. B. & Swain. N. K., "Study of Modulation using Virtual Instruments". *Proceedings of National Conference on Allied Academies, Spring 2000*.
8. Swain, N. K., Anderson, J. A., & Korrapati. R. B. "Computer based virtual engineering Laboratory (CBVEL) and Engineering Technology Education". *2000 Annual ASEE Conference Proceedings*.
9. Lisa Wells and Jeferey Travis, LabVIEW for Everyone, Graphical Programming Even Made Easier, Prentice Hall, NJ 07458, 1997.