VBA/EXCEL: AN ALTERNATIVE COMPUTER PROGRAMMING TOOL FOR ENGINEERING FRESHMAN

Mohammad H. N. Naraghi
Department of Mechanical Engineering
Manhattan College
Riverdale, NY 10471

Abstract

VBA/EXCEL is incorporated in an Engineering Freshman course in computer programming. The course presents generic programming concepts such as variables, arrays, loops, decisions and sub procedures. VBA/EXCEL and Visual Basic are used as the language of choice. This course involves a semester-long project which forces students to research some engineering, mathematics and scientific concepts at the same time as learning computer programming skills. This paper presents a description of this course and two sample projects that are suitable for an engineering freshmen course in computer programming.

Introduction

Computer programming is an essential and integral part of any engineering program. Engineering students in their junior and senior years face the task of solving problems using numerical approaches. Good programming skills enable them to tackle those problems easily. Furthermore, a good knowledge of computer programming makes an engineering graduate more attractive to research oriented engineering employers as well as to graduate engineering programs. In order to enable students to use their programming skills during the four years of engineering education the best time for teaching programming is in the freshman year.

In the past, FORTRAN was the engineering and scientific programming language. During the 1960's, 70's and, to some extent, the 1980's, FORTRAN was the only language with scientific functions. With the emergence of object oriented programming languages (C++, Java and Visual Basic) more attractive alternatives to FORTRAN became available. All of the new object-oriented programming languages have a comprehensive scientific functions library. The question that every engineering program has to answer is "which one of these languages is appropriate for a freshman engineering programming course?" In order to use the object-oriented capabilities of Java and C++ and develop an interesting project students have to go through a long period of instruction, which often cannot be accomplished in one semester. In fact, we may end up losing students (retention of engineering freshman is a critical issue in most engineering programs).

Visual Basic (VB) and Visual Basic Application (VBA) with Excel are attractive alternatives to C++ or Java. VB, with its Control Objects make computer programming a very interesting subject. In fact, it helps the student to enjoy programming. Also, the availability of VB in EXCEL (VBA) enables students to develop sophisticated Excel macros. With a few weeks of instruction, students can develop relatively complex programs with a GUI (Graphic User Interface). Once they learn one programming language migrating to other languages becomes easy. Additionally, most practicing engineers use spreadsheets, specifically, Excel as a computation tool. Spreadsheets can also be used to solve complex engineering problems involving differential equation, such as, heat conduction and fluid flow problems[1].

The freshman engineering course "Computer Programming for Engineers" (ENGS116) at Manhattan College consists of ten weeks of Visual Basic and VBA/EXCEL programming. The remaining four weeks of the course covers some mathematical concepts that help students to understand programming concepts. One interesting feature of this course that makes it an attractive course for our freshmen is its semester long group projects. The original version of this course involved development of computer games, which require some engineering and scientific knowledge. Some of these "game" projects are discussed in [2]. The new course involves development of engineering and scientific projects. This paper presents a description of the course and two sample projects.

Description of the Course

The course is designed to expose students to new methods of analysis (and to reinforce material they may have already been exposed to) in the solution of engineering-related problems. One of the methods of analysis is an introduction to structured programming using the Visual Basic programming language. The summary of topics covered in the course is as follows:

1. Basic concepts in mathematical analysis as applied to engineering problems
2. Use of EXCEL as a tool for engineering analysis
3. Introduction to structured programming concepts using Visual Basic
4. Engineering applications using structured programming

A small portion of the course is devoted to items 1 and 2. The coverage of the basic concepts of mathematical analysis is limited to the application of math in engineering. Homework and quizzes in this part of the course are practical problems that the students have to set the governing equations. Later in the course, students learn to use VBA/EXCEL to solve those equations.

The second part of the course "Use of EXCEL as a tool for engineering analysis," which is only six sessions, covers simple topics, such as, basic data entry, creating formulas and graphing. Other topics in EXCEL including: statistical tools, Goal Seeker, matrix operations and solution of non-linear system of equations using EXCEL's Solver.

The third item, the programming module, is the longest part of the course (more than ten weeks). In this module Visual Basic and VBA/EXCEL are used as programming languages. The emphasis, however, is placed on generic programming. The topics covered in this part of the course are: variables, input/output, decisions, loops, arrays, and sub-procedures and functions. The students see a number of examples related to these topics. Most of the homework and example problems are taken from the course textbooks[3, 4].

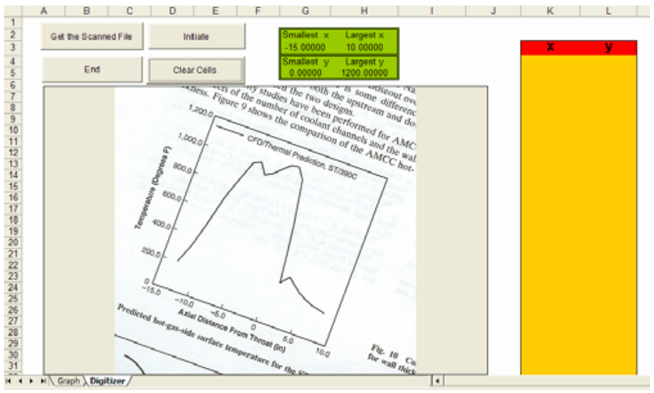
An interesting part of this course is a set of semester long projects. These projects involve some engineering and scientific topics, and require some research in programming as well as math and science. The students enrolled in the course are broken into groups of two or three students. The groups are given a list of possible projects. Some sample projects are: Mesh Generator, Graph Digitizer, and Graphic User Interface for a Program, Indoor Air Quality, Risk of Cancer, Projectile Motion, Random Walk, and Two Arm Robot. Each group is given the option of coming up with its own project, as long as the project has some engineering or scientific value. Students who have some programming background are given more challenging projects.

The following sections present two sample projects which demonstrate some interesting use of VBA/EXCEL. These projects are: digitizing a graph and finite element mesh These projects, along with other student projects can be downloaded from the following web site:

home.manhattan.edu/~mohammad.naraghi/engs116/projects06/

Graph Digitizer

Often, in our engineering and scientific work, we run into graphs in books or journals from which we need to extract numbers. What we normally do is to use a ruler and draw vertical and horizontal lines to read numbers off the graph. This can be a tedious and inaccurate task. The purpose of this project was to develop a VBA/EXCEL program for recording coordinates of points of graphs. A user can load a scanned version of a graph into the VBA/EXCEL image control object, then by defining the scale of the graph and clicking on various points on the graph record the coordinates of those points.



Smallest x	Largest x
-15.00000	10.00000
Smallest y	Largest y
0.00000	1200.00000

Figure 1: GUI of the graph digitizer showing a scanned graph loaded into its Image Control Object.

Figure 1 shows the Graphic User Interface (GUI) for this program which consists of several control objects located in an EXCEL Sheet called “Digitizer.” The largest control object in this interface is an Image named “Image1.” The image in Figure 1 is a chart scanned from an article. The program is designed to correct for any misalignment (axes tilt) of a chart during scanning. As shown in the figure the x-axis of the graph is not horizontal (it is intentionally tilted to check the accuracy of the program in correcting coordinates for the tilt angle). A user can load any image by clicking on the command button with caption “Get the Scanned File.” The name of this command button is cmdGetImage and its code is given below:

```
Private Sub cmdGetImage_Click()
   Dim ImageFilename As Variant, GifGraph
As String
   Dim JpecGraph As String, BitmapGraph As
String
   Dim MetafileGraph As String
   GifGraph = "Graphic Interchange Format
(*.gif), .gif,"
   JpecGraph = "JPEC File (*.jpg), *.jpg,"
   BitmapGraph = " Windows bitmap (*.bmp),
*.bmp,"
   MetafileGraph = " Windows Metafile
(*.wmf),    *.wmf"
   ImageFilename =
Application.GetOpenFilename( _
 GifGraph & JpecGraph & BitmapGraph &
MetafileGraph)
   If ImageFilename = False Then
      MsgBox "No file specified"
      End
   Else
      Image1.Picture =
LoadPicture(ImageFilename)
   End If
End Sub
```

When this program is executed an Open File prompt similar to that shown in Figure 2 appears on the screen. The user then should browse and load the graph file containing the image. Note that four types of graphic files are accommodated in the command button program (gif, jpg, bmp, wmf). Other types of graphic files can be easily incorporated, they are not included here to keep the list short. If the user does not specify a file the program sends a message stating “No file specified” and ends without loading the file to the Image1 control object.

After the scanned graph is loaded the limits of the x and y-axes (i.e., their minimum and maximum) are typed into cells that are highlighted in Figure 1. The program can be initiated by clicking on the command button captioned “Initiate.” The code for this command button “cmdInitiate” is given below:

```
Private Sub cmdInitiate_Click()
   actx0 = Cells(3, 7)
   actxmax = Cells(3, 8)
   acty0 = Cells(5, 7)
   actymax = Cells(5, 8)
   MsgBox "Click on the origin of the
coordinates"
   counter=0
End Sub
```

This part of the code reads the minimum and maximum values of x and y-axes and sends a message, asking the user to click on the origin of the coordinates.

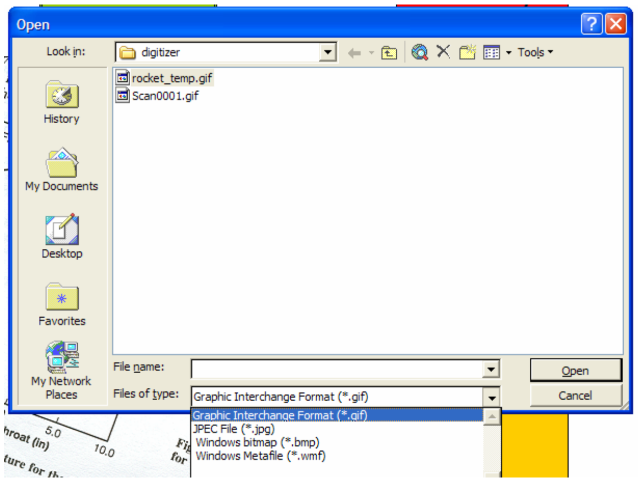


Figure 2: Open file prompt requesting scanned image.

The image "Image1" control object's Event Procedure is "MouseDown," which is coded as:

```
Private Sub Image1_MouseDown(ByVal Button
As Integer, ByVal Shift As Integer, _
ByVal x As Single, ByVal y As Single)
   Dim d As Single, l As Single, X1 As Single,
Y1 As Single
   y = 329.25 - y
   'counts clicks
   counter = counter + 1
   If counter = 2 Then
      d = y - y0
      l = Sqr(((x0 - x) ^ 2) + ((y0 - y) ^ 2))
      Cells(100, 1) = d / l
      angle = Cells(101, 1)
   End If
   If counter = 1 Then
      x0 = x
      y0 = y
      MsgBox "Click on the largest x point"
   ElseIf counter = 2 Then
      x = x - x0
      y = y - y0
      Call RotateCoordinates(x, y, xxmax,
xymax)
      MsgBox "Click on the largest y point"
   ElseIf counter = 3 Then
      x = x - x0
      y = y - y0
      Call RotateCoordinates(x, y, yxmax,
yymax)
      MsgBox "Click on the points that you want
to record Their coordiates"
   End If
      If counter > 3 Then
         x = x - x0
         y = y - y0
         Call RotateCoordinates(x, y, X1, Y1)
         x = (actxmax - actx0) / (xxmax) * X1 +
actx0
         y = (actymax - acty0) / (yymax) * Y1 +
acty0
         Cells(counter, 11) = x
         Cells(counter, 12) = y
      End If
End Sub
Sub RotateCoordinates(xold As Single, yold As
Single,
xnew As Single, ynew As Single)
   xnew = xold * Cos(angle) + yold * Sin(angle)
   ynew = -xold * Sin(angle) + yold *
Cos(angle)
End Sub
```

Each time the user clicks on the graph its x- and y- coordinates will be assigned to variable "x" and "y" in the argument of the above event procedure. It should be noted that the "y" variable in the VB is measured from the top edge of the image. The statement "y=329.25 – y" is used to convert the y coordinate such that it is measured from the bottom edge of the image. Note that 329.25 in y-coordinate conversion equation is the height of the Image. The variable "counter" counts click number. When counter is 1 the mouse click records the coordinate of the origin, when it is 2 the mouse click records the maximum x, and when it is 3 the mouse click records maximum y. For counter values greater than 3 the mouse click records coordinates of the points on the graph.

The variable "angle" calculates the tilt angle of the x-axis. Then the sub procedure "RotateCoordinates," which is simply a rotation of the coordinate equation, is used to account for the tilt angle. Finally, the coordinates are scaled based on the maximum and minimum values of x and y.

It should be noted that most of the variables used in the Image1_MouseDown and cmdInitiate_click event procedures are declared in the common declaration area. Hence, the variables, such as count and angle, keep their values between clicks.

As the user clicks on the points of the graph the coordinates of the points are recorded on the two columns labeled “X” and “Y” (see Figure 1). In order to get an accurate reading of the coordinates, the user may zoom into the graph using EXCEL’s scale. The recording of points can be ended by clicking on the “End” command button shown in Figure 1. The graph of the digitized points is drawn in a separate sheet called “Graph.” Figure 3 shows a reproduced graph of the digitized points for the scanned graph. The accuracy of the digitized coordinates depends on how accurate one can click on the points.

This simple VBA/EXCEL program is very useful for engineering faculty and students. A commercial program for digitizing graphs is UN-SCAN-IT software, which costs hundreds of dollars.

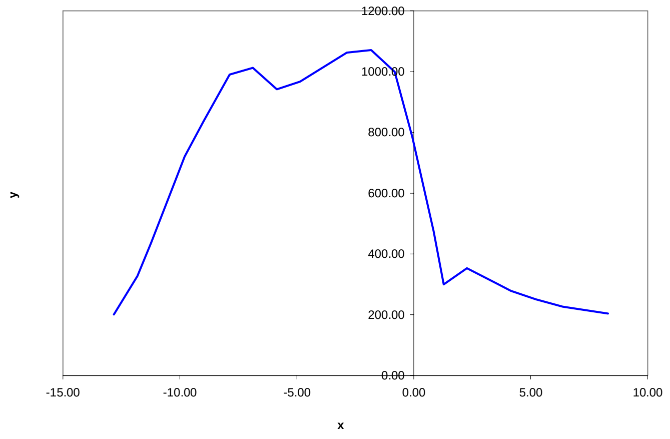
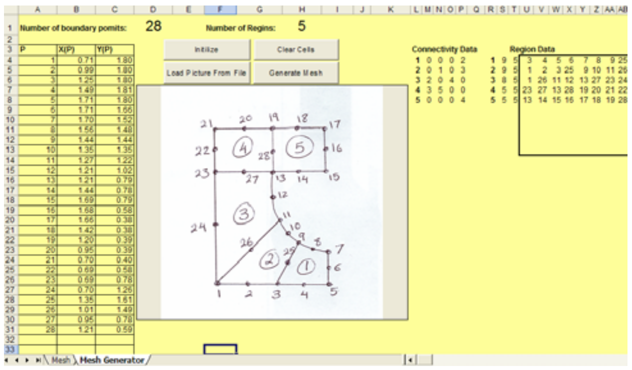


Figure 3 : Digitized graph of the graph shown in Figure 1.

Mesh Generator

Finite element analysis is an integral part of the Mechanical and Civil Engineering curriculum. The first step in the Finite Element analysis is the mesh generation. In this project a student group developed a simple two-dimensional finite element mesh generator based on the algorithm presented in [5]. In this algorithm, the meshing field is subdivided into a number of regions. Each region is specified by eight boundary points.



Number of boundary pomits: 28   Number of Regins: 5

P	X(P)	Y(P)
1	0.71	1.80
2	0.99	1.80
3	1.25	1.80
4	1.49	1.81
5	1.71	1.80
6	1.71	1.66
7	1.70	1.52
8	1.56	1.48
9	1.44	1.44
10	1.35	1.35
11	1.27	1.22
12	1.21	1.02
13	1.21	0.79
14	1.44	0.78
15	1.69	0.79
16	1.68	0.58
17	1.66	0.38
18	1.42	0.38
19	1.20	0.39
20	0.95	0.39
21	0.70	0.40
22	0.69	0.58
23	0.69	0.78
24	0.70	1.26
25	1.35	1.61
26	1.01	1.49
27	0.95	0.78
28	1.21	0.59

Connectivity Data

1	0	0	0	2
2	0	1	0	3
3	2	0	4	0
4	3	5	0	0
5	0	0	0	4

Region Data

1	9	5	3	4	5	6	7	8	9	25
2	9	5	1	2	3	25	9	10	11	26
3	8	5	1	26	11	12	13	27	23	24
4	5	5	23	27	13	28	19	20	21	22
5	5	5	13	14	15	16	17	18	19	28

Figure 4: Interface of the mesh generator program.

Figure 4 shows the VBA/EXCEL interface for the mesh generator program. A hand drawn two-dimensional field with five regions, marked 1 through 5 is loaded into the image box shown. Each region has eight boundary points, and the total number of boundary points for the image shown is 28. The number of regions and boundary points are entered into cells H1 and D1, respectively. The code for loading the field picture into the image control object is the same as that of the digitizer program described in the previous section.

By clicking on the “Initiate” command button the program reads the number of regions and the number of boundary points. Then the program sends a message asking the user to click on the first point of the scale. The sub program for the “Initiate” command button is:

```
Private Sub cmdIntialize_Click()
   INRG = Cells(1, 8)
   INBP = Cells(1, 4)
   Range("D1").Select
   Max = ActiveCell.Value
   counter = -2
   Range("A4").Select
   MsgBox "Click on the first point of the scale"
End Sub
```

Note that in the above program the variable “Max” is set to the number of boundary points and the variable “counter” is set to -2. After initializing the variables, as instructed by the message box, the user left clicks on the first point of the images scale. Then the Image1_MouseDown event procedure given below is executed.

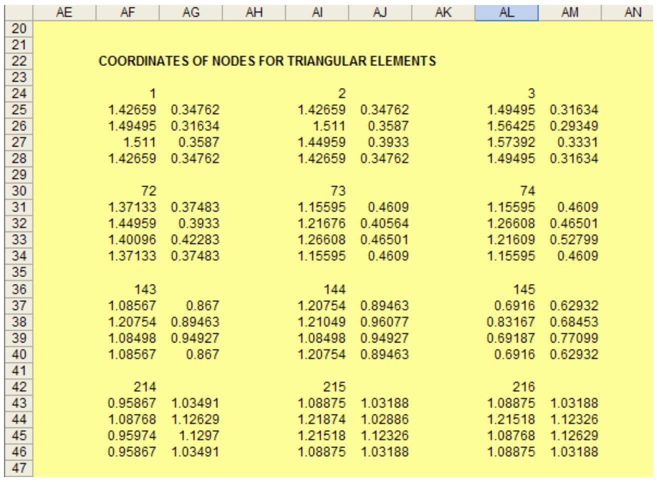
```
Private Sub Image1_MouseDown(ByVal Button
As Integer,
ByVal Shift As Integer, ByVal X As Single,
ByVal Y As Single)
   counter = counter + 1
   If counter = -1 Then
      x1 = X
      y1 = Y
      MsgBox "Click on the second point of the
scale"
   ElseIf counter = 0 Then
      MsgBox "Now click on the boundary
points in the numbered sequence"
      L = Sqr((X - x1) ^ 2 + (Y - y1) ^ 2)
   ElseIf counter > 0 And counter <= Max Then
      ActiveCell.Value = counter
      ActiveCell.Offset(0, 1).Select
      XP(counter) = X / L
      ActiveCell.Value = X / L
      ActiveCell.Offset(0, 1).Select
      YP(counter) = Y / L
      ActiveCell.Value = Y / L
      ActiveCell.Offset(1, -2).Select
   End If
End Sub
```

Each time that the user clicks on the image the variable “counter” increases by one unit. The first two clicks (counter=-1 and counter=0) record the two ends of the scale. When counter=0 the length of scale L is calculated. When counter > 0 the left clicks on the boundary points record coordinates of those points. These coordinates are scaled by dividing their values by L. The resulting coordinates are inserted in the table shown in the left side of Figure 4. Note that when counter > Max the Image1_MouseDown the event procedure stops recording the coordinates, even if the user keeps clicking on the image.

The region connectivity and region data are entered in the upper top part of the EXCEL interface shown in Figure 4. The region connectivity simply specifies what regions are connected to a given region. The region connectivity is entered in a counter clockwise order starting from the bottom side. For example, for the region one shown no region is attached to the bottom, right and top sides. Only region 2 is attached to its left side. Therefore, three zeros are entered for the first three numbers, and a two is entered for the last number.

For the “Region Data” the first number gives the region number followed by the number of nodes in the x- and y-directions for the region. The last eight numbers in the Region Data, which are in a highlighted box, give the eight boundary points corresponding to each region in a counterclockwise order, starting with the lower left point.

By clicking on the “Generate Mesh” button the mesh generating algorithm given in [5] is executed. The VB program for this algorithm is relatively long; hence it is not presented here. The mesh generator program prints element numbers and coordinates of their nodes in a tabular format in the cells of the EXCEL interface. In addition to this table, for each element, coordinates of the nodes are printed into cells in the format shown in Figure 5. The element number is printed on top followed by x and y-coordinates of the nodes of the element. These coordinates are used to plot elements. A chart is set to automatically plot a graph of each element and produce the layout of the mesh. Figure 6 shows the layout mesh generate for the field shown in Figure 4. Note that in Figure 6 each triangular element is represented by a graph line, hence they are shown in different colors. Also, EXCEL can only plot 255 line graphs in one chart. Although this program can generate as many elements as the dimensions of the variables allow, plotting the elements with EXCEL is limited to 255 elements. Other plotting software, such as TecPlot can be used to plot the mesh when the number of elements is more than 255.



	AE	AF	AG	AH	AI	AJ	AK	AL	AM	AN
20										
21										
22		COORDINATES OF NODES FOR TRIANGULAR ELEMENTS								
23										
24		1			2			3		
25		1.42659	0.34762		1.42659	0.34762		1.49495	0.31634	
26		1.49495	0.31634		1.511	0.3587		1.56425	0.29349	
27		1.511	0.3587		1.44959	0.3933		1.57392	0.3331	
28		1.42659	0.34762		1.42659	0.34762		1.49495	0.31634	
29										
30		72			73			74		
31		1.37133	0.37483		1.15595	0.4609		1.15595	0.4609	
32		1.44959	0.3933		1.21676	0.40564		1.26608	0.46501	
33		1.40096	0.42283		1.26608	0.46501		1.21609	0.52799	
34		1.37133	0.37483		1.15595	0.4609		1.15595	0.4609	
35										
36		143			144			145		
37		1.08567	0.867		1.20754	0.89463		0.6916	0.62932	
38		1.20754	0.89463		1.21049	0.96077		0.83167	0.68453	
39		1.08498	0.94927		1.08498	0.94927		0.69187	0.77099	
40		1.08567	0.867		1.20754	0.89463		0.6916	0.62932	
41										
42		214			215			216		
43		0.95867	1.03491		1.08875	1.03188		1.08875	1.03188	
44		1.08768	1.12629		1.21874	1.02886		1.21518	1.12326	
45		0.95974	1.1297		1.21518	1.12326		1.08768	1.12629	
46		0.95867	1.03491		1.08875	1.03188		1.08875	1.03188	
47										

Figure 5: Coordinates of some nodal points for triangular elements for plotting.

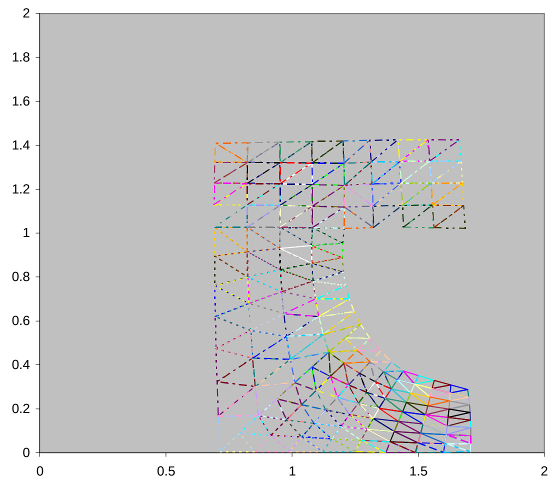


Figure 6: Two-dimensional finite element mesh generated by the Excel/VBA mesh generator.

Concluding Remarks

Incorporating VBA/EXCEL in a Freshman Computer Programming course (ENGS116) is presented. The course in addition to coverage of generic computer programming uses VBA/EXCEL as a programming language. Since we have started this format of the course, our students have developed a number of interesting projects, such as the ones presented in this paper. The programming skills that our students acquired in this course enable them to learn other languages, such as FORTRAN and C++, very quickly, sometimes via self-study. Additionally, their ability to manipulate EXCEL prepares our students for industrial jobs where spreadsheets are a popular means of computations.

References

1. Antar, M.A., “A Simplified Solution Technique for Some Fluid Flow and Heat Transfer Problems,” International Journal of Energy Research, Vol. 22, pp. 1291–1298 (1998).

2. Naraghi, M.H.N., and Litkouhi, B., “An Effective Approach for Teaching Computer Programming to Freshman Engineering Students,” Computers in Education Journal, Vol. XII, No. 2, pp.2-9, 2002.

3. Schneider, D.I., An Introduction to Programming Using Visual Basic 6.0, 4th edition, Prentice Hall, 2004.

4. Chapra, S.C., Power Programming with VBA/EXCEL, Prentice Hall, 2003.

5. Segerlind, L., Applied Finite Element Analysis, 2nd Edition, John Wiley & Sons, Inc., 1984.

Biographical Information

Dr. Mohammad Naraghi is a Professor of Mechanical Engineering at Manhattan College. Prior to joining Manhattan College, he was a Visiting Assistant Professor of Mechanical Engineering at University of Akron where he received his Ph.D. in Mechanical Engineering. Dr. Naraghi worked closely with NASA Glenn Research Center, through research grants and a number of Summer Faculty Fellowships, to develop a comprehensive Rocket Thermal Evaluation code (RTE). Because of this code, he received a certificate of recognition from NASA for the creative development of technically significant software which has been accepted and approved for dissemination to the public by NASA. Since the first release of RTE through NASA's COSMIC library (July 1991), Dr. Naraghi's research is in Thermal/Fluids area and he has published more than sixty articles in ASME, AIAA and international journals and conferences. He is recipient of a number of research grants from NASA and Air Force. Dr. Naraghi is a Fellow of ASME and a member of AIAA’s Liquid Propulsion Technical Committee.