A Visual Basic Program for Gauss-Jordan Elimination

On the next page is Visual Basic code that is designed to run inside Excel and solve systems of complex equations by Gauss-Jordan elimination. Follow these steps:

- Enter the code into Excel by following the instructions on page 32. (the first four bullets)
- Enter an augmented matrix in the upper, left corner of a spreadsheet.
- On the Developer tab in Excel click Macros and run the macro called Gauss_Jordan_Complex. A dialog box asks the size of the system. Then the program carries out the steps of the Gauss-Jordan method and replaces the original matrix with the row-reduced matrix.

```
Sub Gauss_Jordan_Complex()
   Dim Row, Col, K, N As Integer
   Dim PivR, PivI, AR, AI, BR, BI, MultR, MultI, Temp As Double

   N = InputBox("What is the system's size, N?")               'This problem has N equations in N unknowns.
                                                                'Col is the index of the present (pivot) column.
   For Col = 1 To N                                             'This loop runs over the columns.
      MsgBox "Start working on column " & Col
      For Row = Col To N                                        'Find a candidate
         If Cells(Row, 2 * Col - 1) <> 0 Or Cells(Row, 2 * Col) <> 0 Then GoTo 1   'for the pivot element.
      Next Row                                                  'When one is found we can get started.

      MsgBox "ERROR - No unique solution"                       'Quick exit:  The present column
      Exit Sub                                                  'is full of zeros, so there is no solution, so quit.

1:     If Row <> Col Then                                       'If the candidate pivot is not already on the
         MsgBox "Pivot up row " & Row                           ' diagonal then pivot up its row.
         For K = 1 To 2 * (N + 1)                               'i.e. do ERO # 3.
            Temp = Cells(Row, K)
            Cells(Row, K) = Cells(Col, K)
            Cells(Col, K) = Temp
         Next K
      End If
                                          'do ERO # 1 to put a 1 in the diagonal (pivot) element.
      PivR = Cells(Col, 2 * Col - 1):     PivI = Cells(Col, 2 * Col)

      If Not (PivR = 1 And PivI = 0) Then
         MsgBox "Apply ERO # 1 to row " & Col
         For K = 1 To N + 1                                     'Run through all the elements in the row.
            AR = Cells(Col, 2 * K - 1):     AI = Cells(Col, 2 * K)                  ' [ ERO#1 ]
            Cells(Col, 2 * K - 1) = (AR * PivR + AI * PivI) / (PivR ^ 2 + PivI ^ 2)
            Cells(Col, 2 * K) = (AI * PivR - AR * PivI) / (PivR ^ 2 + PivI ^ 2)
         Next K
      Else
         MsgBox "Can skip ERO # 1 on row " & Col
      End If
                           'do ERO # 2 to put 0's in each spot above and below the diagonal element.
      For Row = 1 To N
         If Row <> Col Then
            MultR = Cells(Row, 2 * Col - 1):     MultI = Cells(Row, 2 * Col)

            If MultR <> 0 Or MultI <> 0 Then
               MsgBox "Apply ERO # 2 to row " & Row
               For K = 1 To N + 1  'Run through all the elements in the row.
                  AR = Cells(Row, 2 * K - 1):     AI = Cells(Row, 2 * K)            ' [ ERO#2 ]
                  BR = Cells(Col, 2 * K - 1):     BI = Cells(Col, 2 * K)
                  Cells(Row, 2 * K - 1) = AR - MultR * BR + MultI * BI
                  Cells(Row, 2 * K) = AI - MultR * BI - MultI * BR
               Next K
            Else
               MsgBox "Can skip ERO # 2 on row " & Row
            End If
         End If
      Next Row
   Next Col

   MsgBox "Done!  Answers are in the last two columns."
End Sub
```

Compare this program with the Gauss-Jordan program for real numbers given on page 49. The differences are:

- Two columns of the spreadsheet are needed for each complex number. Odd columns (1, 3, 5, 7, …) hold the real parts and even columns (2, 4, 6, 8, …) hold the imaginary parts.

- ERO #1: The division of a complex number such as $AR+j\,AI$ by a complex pivot $PivR+j\,PivI$ is done as follows:

$$\frac{AR+j\,AI}{PivR+j\,PivI} = \frac{(AR+j\,AI)\cdot(PivR-j\,PivI)}{(PivR+j\,PivI)\cdot(PivR-j\,PivI)}$$

$$= \frac{AR\cdot PivR+AI\cdot PivI}{PivR^2+PivI^2} + j\,\frac{AI\cdot PivR-AR\cdot PivI}{PivR^2+PivI^2}$$

  The real part of the result is put in an odd column of the spreadsheet and the imaginary part into an even column.

- ERO #2: Suppose row 2 contains an element $AR+j\,AI$ and row 1 contains $BR+j\,BI$. The elementary row operation $R_2 \leftarrow R_2 - Mult\cdot R_1$ is done as follows (assuming that $Mult = MultR + j\,MultI$):

$$AR+j\,AI \leftarrow (AR+j\,AI)-(MultR+j\,MultI)(BR+j\,BI)$$

or

$$AR+j\,AI \leftarrow (AR-MultR\cdot BR+MultI\cdot BI)+j(AI-MultR\cdot BI-MultI\cdot BR)$$

  This is actually two equations, one for the real part,

$$AR \leftarrow AR-MultR\cdot BR+MultI\cdot BI,$$

  and one for the imaginary part,

$$AI \leftarrow AI-MultR\cdot BI-MultI\cdot BR.$$

The real part of the result is put in an odd column of the spreadsheet and the imaginary part into an even column.
