# Using VBA As An Alternative

**by**
***Edward Rosen, EMR Technology***

## Introduction

Many routine calculations carried out using stand-alone systems such as MathCad (1) can be carried out as well in Excel using its macro language Visual Basic for Applications (VBA). The advantages of doing this are discussed in (2). This communication reviews (3) some of the details involved in a VBA implementation of the integration of an n-dimensional system of differential equations.

## Earth's Carbon Cycle

A model of the earth's carbon cycle is discussed in (4). The differential equations describing the model together with their initial values are given in Tables 1 and 2. A plot of deforestation ($F_d(t)$) and fossil fuels data ($F_f(t)$) is given in graphical form in Figure 1 (taken from Reference 4). The task is to integrate the differential equations from the year 1850 to 1990. Of particular interest is the growth of atmospheric carbon dioxide over that period of time.

## The Excel Spreadsheet

The spreadsheet of the model is given in Figure 2. The step size for the integration (h) and the number of equations to be solved (n) are specified under System Parameters (C5, C6). The model parameters starting in C12 are defined to the left of the main table. The main table begins in location F12 with the year 1850. The initial values of the eight dependent variables are specified on the same line. The value of the $CO_2$ (P12) concentration is calculated from the value of M1.

The spreadsheet invokes the array function procedure Intc (Figure 3) starting in the year (1850 + h). The initial invocation with the output selected in H13:O13 is

= Intc ( $D$6,$D$7,$G13,$H13:$O13,$D$13)

Ctrl+Shift+Enter is pressed.

This corresponds to the general calling sequence

= Intc ( h, n, x, y, prm)

where

h = step size
n = number of equations
x = independent variable
y = dependent variable vector
prm = parameter vector

Entries for the following years are copied from the entry for 1850 + h..

**The VBA Integration and Interpolation Routines**

VBA has access to a large number of routines but lacks two that are needed for this application: an n-dimensional integration routine and a one-dimensional interpolation routine that can be extrapolated.

For the integration, the array function Intc invokes the rk4a function. This is a $4^{th}$ order Runge-Kutta routine utilizing a fixed step size, h (Figure 4). The routine was modified by Pedro L. Claveria (5) from the EMR Technology Library (6). The parameter vector prm was passed from the spreadsheet to rk4a via the Intc function.

Data for $F_f(t)$ and $F_d(t)$ (at 1850, 1860, 1870, etc) was read directly from the plot of Figure 2 in Reference (4). The data was entered using the array function of VBA (7) and is used in the function dydx (Figure 5) which defines the right hand sides of the differential equations. The $F_r(t)$ data (for the reforestation) was taken as zero (Reference (4)).

The interpolation function Interp (Figure 6) was used to interpolate quadratically within the arrays yr, AFFt, AFDt and AFRt

The general calling sequence is

= Interp (nl, x, fx, arg)

where

nl = 1 for linear interpolation, 2 for quadratic interpolation
x = the independent variable vector
fx = the dependent variable vector
arg = the independent variable vector

The years 1991 to 2000 in Figure 2 represents an extrapolation.

## Executing the Spreadsheet

The reader may download the spreadsheet

Earth Carbon Cycle.xls

which is available in this issue of CACHE News. The VBA may be brought up by hitting Alt+F11. By modifying the data in dydx for $F_f(t)$, $F_d(t)$ and $F_r(t)$ (AFFt,AFDt,AFRt for years 1850 to 1990) different atmospheric carbon dioxide levels can be predicted for future years (as suggested in Reference (4)).

## Results and Conclusions

Table 3 compares the results of the spreadsheet/VBA computations with that given in Reference (4). The results compare favorably considering the different ways the data for $F_f(t)$ and $F_d(t)$ are handled. (In this study interpolation is used on data read from a chart. In Reference (4) curve fitting was employed on original data). Another source of difference may be due to the different integration routines used in this study and that used from MathCad (1).

Excel offers a viable alternative for carrying out routine calculations such as integrating initial value differential equations. This may be done with VBA code or other languages such as FORTRAN (8,9).

## References


1. MathCad, http://www.mathcad.com

2. Rosen, E. M. "On the Choice of VBA", *CACHE News*, No 56 Spring 2003

3. Rosen, E. M. "The Case for Excel and Visual Basic for Applications", *CACHE News*, No 46 Spring 1998 p 5

4. Schmitz, R. A. "The Earth's Carbon Cycle", *Chemical Engineering Education*, Vol 36, No. 4 Fall, 2002, pp 296

5. Pedro Luis Claveria Vila, Private Communication (circa, April 2002)

6. EMR Technology Group Library, CACHE Corporation

7. Walkenbach, John *Excel 2000 Power Programming with VBA* IDG Books Worldwide, Inc , Foster City, CA. 1999 p 255

8. Rosen, E. M. "Calling FORTRAN Subroutines from Excel 7.0" *CACHE News*, No 47, Fall 1998

*9.* Rosen, E. M. “On the Use of IMSL Routines in Excel 7.0” *CACHE News* No 49, Fall 1999

Table 1

Earth’s Carbon Cycle Differential Equations

$$dM_1/dt = -(k_{12}+k_{13})M_1 - k_{15}M_8\frac{M_1-g}{M_1+\Gamma} + k_{21}M_2^{b_2} + k_{31}M_3^{b_3} + k_{51}M_5 + k_{61}M_6 + F_f(t) + F_d(t) - F_r(t)$$

$$dM_2/dt = k_{12}M_1 - (k_{23}+k_{24})M_2 - k_{21}M_2^{b_2} + k_{42}M_4$$

$$dM_3/dt = k_{13}M_1 + k_{23}M_2 - k_{34}M_3 - k_{31}M_3^{b_3} + k_{43}M_4$$

$$dM_4/dt = k_{24}M_2 + k_{34}M_3 - (k_{42}+k_{43})M_4$$

$$dM_5/dt = k_{15}M_8\frac{M_1-g}{M_1+\Gamma} - (k_{51}+k_{56})M_5 - F_d(t) + F_r(t)$$

$$dM_6/dt = k_{56}M_5 - k_{61}M_6$$

$$dM_7/dt = -F_f(t)$$

$$dM_8/dt = \frac{-\left[k_d F_d(t) - k_r F_r(t)\right]}{M_{5,ref}}$$

**Table 2**

**Numerical Values and Units for Model Constants**

| Symbol | Value | Units |
|---|---|---|
| $k_{12}$ | 0.0931 | $y^{-1}$ |
| $k_{13}$ | 0.0311 | $y^{-1}$ |
| $k_{15}$ | 147 | $y^{-1}$ |
| $k_{21}$ | 7.05006E-26 | $PgC^{(1-ß_2)}y^{-1}$ |
| $k_{23}$ | **0.0781** | $y^{-1}$ |
| $k_{24}$ | **0.0164** | $y^{-1}$ |
| $k_{31}$ | 2.31615E-21 | $PgC^{(1-ß_3)}y^{-1}$ |
| $k_{34}$ | **0.714** | $y^{-1}$ |
| $k_{42}$ | **0.00189** | $y^{-1}$ |
| $k_{43}$ | **0.00114** | $y^{-1}$ |
| $k_{51}$ | **0.0862** | $y^{-1}$ |
| $k_{56}$ | **0.0.0862** | $y^{-1}$ |
| $k_{61}$ | **0.0333** | $y^{-1}$ |
| $ß_2$ | **9.4** | |
| $ß_3$ | **10.2** | |
| ? | 62 | PgC* |
| G | 198 | PgC |
| $k_d$ | **0.230** | |
| $k_r$ | 1.0 | |
| $M_{5,ref}$ | 580 | |

***PgC, 1 Pg = $10^{15}$ g** **Conversion Factor: 2.218 PgC/ppmv**

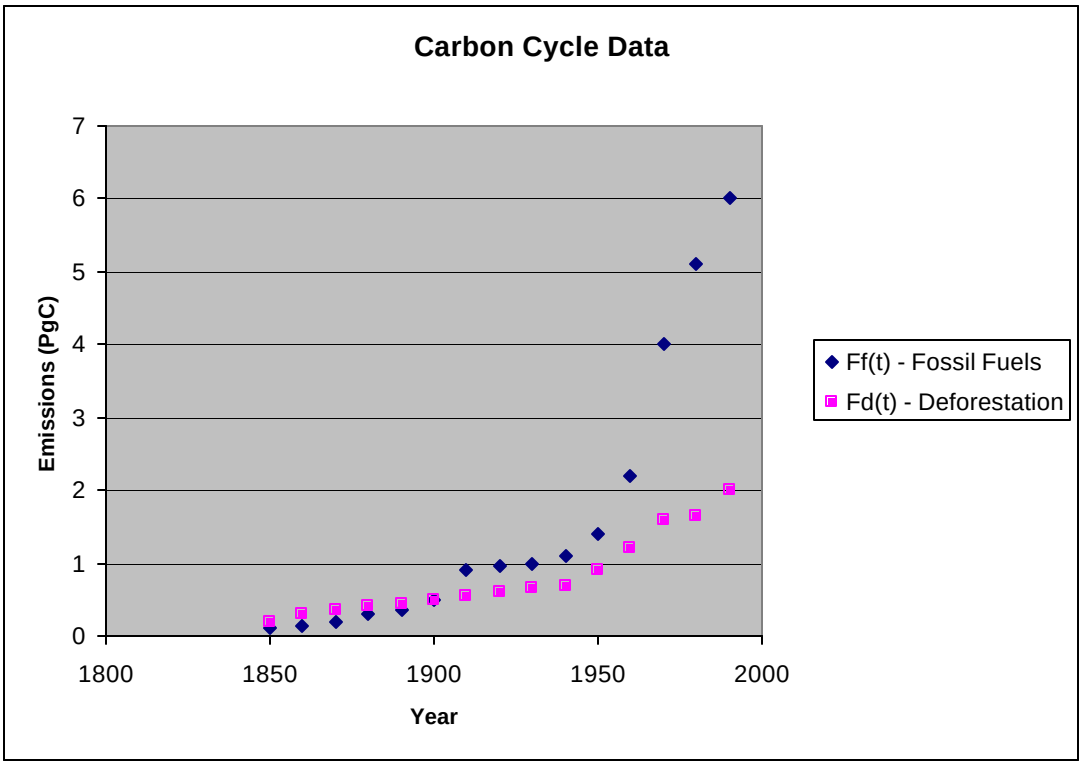


Figure 1
Fossil Fuels and Deforestation Emissions

| A/1 | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 2 | | Earth's Carbon Cycle from Schnitz CEE Vol 36 No 4 Fall 2002 | | | | | | | | | | | | | | |
| 3 | | | | | | | | | | | | | | | | |
| 4 | | | | | | | | | | | | | | | | |
| 5 | | System Parameters | | Units | | | | | | | | | | | | |
| 6 | | h | | 1 years | | | | | | | | | | | | |
| 7 | | n | | 8 Number of Equations | | | | | | | | | | | | |
| 8 | | | | | | | | | | | | | | | | |
| 9 | | Parameters | | | | | | | | | | | | | | |
| 10 | | | | | | | | | | | | | | | | |
| 11 | prm | Symbol | Value | Units | | Year | M1 | M2 | M3 | M4 | M5 | M6 | M7 | M8 | | CO2 (ppmv) |
| 12 | | | | | | | | | | | | | | | | |
| 13 | | 1 k12 | 0.0931 | y^-1 | | 1850 | 612.0 | 730.0 | 140.0 | 37000.0 | 580.0 | 1500.0 | 5300.0 | 1.0000000 | | 287.6 |
| 14 | | 2 k13 | 0.0311 | y^-1 | | 1851 | 612.5 | 730.0 | 140.1 | 36999.9 | 579.7 | 1500.0 | 5299.9 | 0.9999182 | | 287.8 |
| 15 | | 3 k15 | 147 | y^-1 | | 1852 | 612.9 | 730.0 | 140.1 | 36999.8 | 579.4 | 1500.0 | 5299.8 | 0.9998317 | | 272.3 |
| 16 | | 4 k21 | 7.05E-26 | PgC^(1-beta2) /y | | 1853 | 613.3 | 730.0 | 140.1 | 36999.7 | 579.1 | 1500.0 | 5299.7 | 0.9997407 | | 288.2 |
| 17 | | 5 k23 | 0.0781 | y^-1 | | 1854 | 613.7 | 730.0 | 140.2 | 36999.6 | 578.9 | 1500.0 | 5299.6 | 0.9996452 | | 288.4 |
| 18 | | 6 k34 | 0.0164 | y^-1 | | 1855 | 614.0 | 730.1 | 140.2 | 36999.6 | 578.8 | 1499.9 | 5299.4 | 0.9995456 | | 288.5 |
| 19 | | 7 k31 | 2.32E-21 | PgC^(1-beta2) /y | | 1856 | 614.4 | 730.1 | 140.2 | 36999.5 | 578.6 | 1499.9 | 5299.3 | 0.9994420 | | 288.7 |
| 20 | | 8 k34 | 0.714 | y^-1 | | 1857 | 614.7 | 730.2 | 140.2 | 36999.5 | 578.5 | 1499.8 | 5299.2 | 0.9993347 | | 288.9 |
| 21 | | 9 k42 | 0.00189 | y^-1 | | 1858 | 615.0 | 730.2 | 140.2 | 36999.4 | 578.4 | 1499.7 | 5299.0 | 0.9992238 | | 289.0 |
| 22 | | 10 k43 | 0.00114 | y^-1 | | 1859 | 615.3 | 730.2 | 140.2 | 36999.4 | 578.3 | 1499.6 | 5298.9 | 0.9991095 | | 289.2 |
| 23 | | 11 k51 | 0.0862 | y^-1 | | 1860 | 615.6 | 730.3 | 140.2 | 36999.4 | 578.3 | 1499.5 | 5298.8 | 0.9989921 | | 289.3 |
| 24 | | 12 k56 | 0.0862 | y^-1 | | 1861 | 615.9 | 730.3 | 140.2 | 36999.3 | 578.2 | 1499.5 | 5298.6 | 0.9988721 | | 289.4 |
| 25 | | 13 k61 | 0.0333 | y^-1 | | 1862 | 616.2 | 730.3 | 140.2 | 36999.3 | 578.1 | 1499.4 | 5298.4 | 0.9987502 | | 289.6 |
| 26 | | 14 beta2 | 9.4 | y^-1 | | 1863 | 616.5 | 730.4 | 140.2 | 36999.3 | 578.1 | 1499.3 | 5298.3 | 0.9986262 | | 289.7 |
| 27 | | 15 beta3 | 10.2 | y^-1 | | 1864 | 616.8 | 730.4 | 140.2 | 36999.3 | 578.1 | 1499.2 | 5298.1 | 0.9985003 | | 289.8 |
| 28 | | 16 gamma | 62 | PgC | | 1865 | 617.0 | 730.4 | 140.2 | 36999.2 | 578.0 | 1499.1 | 5298.0 | 0.9983724 | | 290.0 |
| 29 | | 17 lambda | 198 | PgC | | 1866 | 617.3 | 730.5 | 140.2 | 36999.2 | 578.0 | 1499.0 | 5297.8 | 0.9982426 | | 290.1 |
| 30 | | 18 kd | 0.23 | | | 1867 | 617.6 | 730.5 | 140.2 | 36999.2 | 578.0 | 1498.9 | 5297.6 | 0.9981107 | | 290.2 |
| 31 | | 19 kr | 1 | | | 1868 | 617.8 | 730.5 | 140.2 | 36999.2 | 577.9 | 1498.8 | 5297.4 | 0.9979769 | | 290.3 |
| 32 | | 20 M5, ref | 580 | | | 1869 | 618.1 | 730.5 | 140.2 | 36999.2 | 577.9 | 1498.7 | 5297.2 | 0.9978411 | | 290.5 |
| 33 | | | | | | 1870 | 618.4 | 730.6 | 140.2 | 36999.2 | 577.9 | 1498.6 | 5297.0 | 0.9977033 | | 290.6 |
| | | | | | | | | | | | | | | | | |
| | | | | | | 1986 | 738.1 | 742.7 | 142.4 | 37053.9 | 579.3 | 1493.7 | 5111.9 | 0.9597859 | | 346.9 |
| | | | | | | 1987 | 741.4 | 743.1 | 142.5 | 37055.5 | 579.5 | 1493.9 | 5106.2 | 0.9590549 | | 348.4 |
| | | | | | | 1988 | 744.8 | 743.4 | 142.5 | 37057.2 | 579.7 | 1494.1 | 5100.4 | 0.9583076 | | 350.0 |
| | | | | | | 1989 | 748.2 | 743.7 | 142.6 | 37058.9 | 579.8 | 1494.3 | 5094.5 | 0.9575428 | | 351.6 |
| | | | | | | 1990 | 751.6 | 744 | 142.7 | 37060.6 | 580.0 | 1494.5 | 5088.5 | 0.9567595 | | 353.2 |
| | | | | | | | | | | | | | | | | |
| | | | | | | 1991 | 755.1 | 744.4 | 142.7 | 37062.4 | 580.1 | 1494.7 | 5082.5 | 0.9559562 | | 354.8 |
| | | | | | | 1992 | 758.6 | 744.7 | 142.8 | 37064.3 | 580.2 | 1495.0 | 5076.4 | 0.9551320 | | 356.5 |
| | | | | | | 1993 | 762.2 | 745.1 | 142.8 | 37066.1 | 580.3 | 1495.2 | 5070.2 | 0.9542856 | | 358.2 |
| | | | | | | 1994 | 765.9 | 745.4 | 142.9 | 37068.1 | 580.4 | 1495.4 | 5063.9 | 0.9534158 | | 359.9 |
| | | | | | | 1995 | 769.5 | 745.7 | 143.0 | 37070.0 | 580.5 | 1495.7 | 5057.6 | 0.9525213 | | 361.6 |
| | | | | | | 1996 | 773.3 | 746.1 | 143.0 | 37072.0 | 580.5 | 1495.9 | 5051.2 | 0.9516011 | | 363.4 |
| | | | | | | 1997 | 777.1 | 746.5 | 143.1 | 37074.1 | 580.5 | 1496.1 | 5044.7 | 0.9506540 | | 365.2 |
| | | | | | | 1998 | 780.9 | 746.8 | 143.2 | 37076.2 | 580.4 | 1496.3 | 5038.2 | 0.9496787 | | 367.0 |
| | | | | | | 1999 | 784.8 | 747.2 | 143.2 | 37078.3 | 580.4 | 1496.5 | 5031.5 | 0.9486740 | | 368.8 |
| | | | | | | 2000 | 788.7 | 747.5 | 143.3 | 37080.5 | 580.3 | 1496.7 | 5024.9 | 0.9476388 | | 370.6 |

Figure 2
Spreadsheet for Earth
Carbon Cycle

```
     Option Base 1
Public Function Intc(h, n, x, y, prm)


Dim xx As Single
Dim Irtn As Integer
Dim I As Integer

nn = n + 1

ReDim yy(1 To n) As Single
ReDim ddd(1 To nn) As Single
ReDim fff(1 To n)

xx = x

For I = 1 To n
yy(I) = y(I)
Next I

Irtn = rk4a(n, h, xx, yy, prm)

xx = xx + h

ddd(1) = xx

For I = 2 To nn
ddd(I) = yy(I - 1)
Next I

Intc = ddd

End Function
```

**Figure 3**

**Listing of VBA Array Function Intc**

```
                Public Function rk4a(n, h, x, y, prm)
'
'Modified from Pedro L. Claveria abril/2002
'based in EMR Technology Group Library
'
'
'n = number of equations
'h  = step size for integration
'x = independent variable
'y = vector of dependent variables
'prm = vector parameters

ReDim ccc(n), fff(n)
ReDim k1(n), k2(n), k3(n), k4(n)
ReDim y2(n), y3(n), y4(n)

'Calculation of k1
muda1 = dydx(x, y, prm, fff)
For I = 1 To n: k1(I) = fff(I): Next
'Calculation of k2
For I = 1 To n: y2(I) = y(I) + 0.5 * h * k1(I): Next
muda2 = dydx(x + h / 2, y2, prm, fff)
For I = 1 To n: k2(I) = fff(I): Next
'Calculation of k3
For I = 1 To n: y3(I) = y(I) + 0.5 * h * k2(I): Next
muda3 = dydx(x + h / 2, y3, prm, fff)
For I = 1 To n: k3(I) = fff(I): Next
'Calculation of k4
For I = 1 To n: y4(I) = y(I) + h * k3(I): Next
muda4 = dydx(x + h, y4, prm, fff)
For I = 1 To n: k4(I) = fff(I): Next

'New values of the dependent variables
For I = 1 To n
    ccc(I) = y(I) + (h / 6) * (k1(I) + 2 * k2(I) + 2 * k3(I) + k4(I))
Next I

For I = 1 To n
    y(I) = ccc(I)
Next I

rk4a = 0

End Function
```

**Figure 4**

**Listing of VBA Function Procedure rk4a**

```
Public Function dydx(x, y, prm, fff)

'x = independent variable
'y = vector of dependent variables
'prm = parameter vector
'fff = dy/dx

Dim AFFt As Variant
Dim AFDt As Variant
Dim AFRt As Variant
Dim Ayr  As Variant

Dim NL As Integer
Dim xx As Single

Dim t1, t2, t3, t4 As Single


yr = Array(1850,1860,1870,1880,1890,1900,1910,1920,1930,1940, _
            1950, 1960, 1970, 1980, 1990)

AFFt = Array(0.1, 0.15, 0.2, 0.3, 0.35, 0.5, 0.9, 0.95, 1#, 1.1, _
            1.4, 2.2, 4#, 5.1, 6#)

AFDt = Array(0.2, 0.3, 0.35, 0.4, 0.45, 0.5, 0.55, 0.6, 0.65, 0.7, _
            0.9, 1.2, 1.6, 1.65, 2#)

AFRt = Array(0#, 0#, 0#, 0#, 0#, 0#, 0#, 0#, 0#, 0#, 0#, 0#, _
            0#, 0#, 0#)

NL = 2
xx = x


FFt = Interp(NL, yr, AFFt, xx)
FDt = Interp(NL, yr, AFDt, xx)
FRt = Interp(NL, yr, AFRt, xx)

t1 = -(prm(1) + prm(2)) * y(1) - prm(3) * y(8) * (y(1) - prm(16)) _
       / (y(1) + prm(17))
t2 = prm(4) * y(2) ^ prm(14)
t3 = prm(7) * (y(3) ^ prm(15)) + prm(11) * y(5) + prm(13) * y(6)
t4 = FFt + FDt - FRt

fff(1) = t1 + t2 + t3 + t4

t1 = prm(1) * y(1) - (prm(5) + prm(6)) * y(2)
t2 = -prm(4) * y(2) ^ prm(14) + prm(9) * y(4)

fff(2) = t1 + t2
```

**Figure 5**

**Listing of VBS Function Procedure dy/dx (page 1)**

```
t1 = prm(2) * y(1) + prm(5) * y(2) - prm(8) * y(3)
t2 = -prm(7) * y(3) ^ prm(15) + prm(10) * y(4)

fff(3) = t1 + t2

fff(4) = prm(6) * y(2) + prm(8) * y(3) - (prm(9) + prm(10)) * y(4)

t1 = prm(3) * y(8) * (y(1) - prm(16)) / (y(1) + prm(17))
t2 = -(prm(11) + prm(12)) * y(5) - FDt + FRt

fff(5) = t1 + t2

fff(6) = prm(12) * y(5) - prm(13) * y(6)

fff(7) = -FFt

fff(8) = -(prm(18) * FDt - prm(19) * FRt) / prm(20)



dydx = 0
End Function
```

**Figure 5**
**Listing of VBS Function Procedure dy/dx (page 2)**

Public Function Interp(nl, x, fx, arg)

```
Public Function Interp(n, x, fx, arg)

'Written by EMRosen  6/19/97
'Copyright (c) by EMR Technology Group

'Modified 8/15/97 For Extrapolation

'  nl is the order of interpolation  -1 for Linear, 2 for second order
'  x is the x array of numbers to be used -
'            the name for the range is used in the call
' fx is the array corresponding values of f(x) -
'            the the first element of the range may be used
'arg is the x argument
' The return is the value of f(arg)

xarg = arg

'Nc is the total array length
 Nc = Application.Count(x)
'MsgBox "Value of Nc=" & Nc

If (xarg < x(1)) Then
       Num = 1
       GoTo Setn:
 End If

If (xarg >= x(Nc) And n = 1) Then
        Num = Nc - 1
        GoTo Setn:
 End If

 If (xarg >= x(Nc) And n = 2) Then
        Num = Nc - 2
        GoTo Setn:
 End If

'Num is the index of the element in the range less than or equal to arg
Num = Application.Match(arg, x, 1)

Setn:


If (n = 2 And Nc < 3) Then
      Interp = 0
      Exit Function
End If

 If (n = 1 And Nc < 2) Then
     Interp = 0
     Exit Function
End If
```

**Figure 6**
**Listing of VBS Function Procedure Interp (Page 1)**

```
xarg = arg

x1 = x(Num)
fx1 = fx(Num)

x2 = x(Num + 1)
fx2 = fx(Num + 1)

If (n = 2 And Nc >= Num + 2) Then GoTo Second:

If (n = 2 And Nc < Num + 2) Then GoTo Third:

Term1 = (xarg - x2) / (x1 - x2)
Term2 = (xarg - x1) / (x2 - x1)

Interp = Term1 * fx1 + Term2 * fx2

Exit Function
Second:

x3 = x(Num + 2)
fx3 = fx(Num + 2)
GoTo Continue:


Third:

x1 = x(Nc - 2)
fx1 = fx(Nc - 2)

x2 = x(Nc - 1)
fx2 = fx(Nc - 1)

x3 = x(Nc)
fx3 = fx(Nc)

Continue:
Term1 = (xarg - x2) * (xarg - x3) / ((x1 - x2) * (x1 - x3))
Term2 = (xarg - x1) * (xarg - x3) / ((x2 - x1) * (x2 - x3))
Term3 = (xarg - x1) * (xarg - x2) / ((x3 - x1) * (x3 - x2))

Interp = Term1 * fx1 + Term2 * fx2 + Term3 * fx3

End Function
```

**Figure 6**
**Listing of VBS Function Procedure Interp (Page 2)**

Table 3

Comparison of Solutions at 1990

| | **Reference 4** | **This Work** |
|---|---|---|
| $M_1$ | 753 | 751.6 |
| $M_2$ | 744 | 744.0 |
| $M_3$ | 143 | 142.7 |
| $M_4$ | 37071 | 37060.5 |
| $M_5$ | 577 | 580.0 |
| $M_6$ | 1489 | 1494.5 |
| $M_7$ | 5086 | 5088.5 |
| $M_8$ | 0.952 | 0.9568 |