Spreadsheets in Education (eJSiE)

Volume 1 | Issue 3 Article 4

10-5-2005

Spreadsheet Solution of Systems of Nonlinear Differential Equations

Ali El-Hajj
American University of Beirut, Lebanon, elhajj@aub.edu.lb

Sami Karaki
American University of Beirut, Lebanon

Mohammed Al-Husseini
American University of Beirut, Lebanon

Karim Y. Kabalan
American University of Beirut, Lebanon

Recommended Citation

El-Hajj, Ali; Karaki, Sami; Al-Husseini, Mohammed; and Kabalan, Karim Y. (2005) Spreadsheet Solution of Systems of Nonlinear Differential Equations, *Spreadsheets in Education (eJSiE)*: Vol. 1: Iss. 3, Article 4.
Available at: http://epublications.bond.edu.au/ejsie/vol1/iss3/4

Spreadsheet Solution of Systems of Nonlinear Differential Equations

Abstract

This paper presents a method for obtaining numerical approximation to solutions of systems of nonlinear differential equations of one variable using spreadsheets. This solution method is simply based on effecting an integration formula in a block. The input of the block is equal to a given variable and its output is equal to the integral of that variable. The block allows the user to set the initial condition for differential equation. A nonlinear differential equation of order n is solved through a cascaded interconnection of these integration blocks to obtain the solution function and all its successive derivatives. This approach can also be used to solve a system of nonlinear differential equations with initial conditions for the functions and their derivatives. The method is easy to learn and allows a step-by-step study of the solution, and an investigation of the influence of changing one or more parameters on this solution. The graphical capabilities of spreadsheets gives the method a visual feel and allows easy comparisons between exact and numerical solutions. The method is illustrated through several examples of non-linear differential equations which demonstrate its accuracy, flexibility and simplicity.

Keywords

Spreadsheet Simulation, Differential Equations, Non-Linear Systems, Numerical Integration

Spreadsheet Solution of Systems of Nonlinear Differential Equations

Ali El-Hajj, Sami Karaki,
Mohammed Al-Husseini, and Karim Y. Kabalan
Electrical and Computer Engineering Department,
Faculty of Engineering and Architecture,
American University of Beirut, Lebanon.
elhajj@aub.edu.lb

October 5, 2005

Abstract

This paper presents a method for obtaining numerical approximation to solutions of systems of nonlinear differential equations of one variable using spreadsheets. This solution method is simply based on effecting an integration formula in a block. The input of the block is equal to a given variable and its output is equal to the integral of that variable. The block allows the user to set the initial condition for differential equation. A nonlinear differential equation of order n is solved through a cascaded interconnection of these integration blocks to obtain the solution function and all its successive derivatives. This approach can also be used to solve a system of nonlinear differential equations with initial conditions for the functions and their derivatives. The method is easy to learn and allows a step-by-step study of the solution, and an investigation of the influence of changing one or more parameters on this solution. The graphical capabilities of spreadsheets gives the method a visual feel and allows easy comparisons between exact and numerical solutions. The method is illustrated through several examples of non-linear differential equations which demonstrate its accuracy, flexibility and simplicity.

Submitted September 2004; revised and accepted November 2004.

Keywords: Spreadsheet Simulation, Differential Equations, Non-Linear Systems, Numerical Integration.

1 Introduction

Spreadsheet programs are used in education to solve problems in electromagnetic fields, fluid flow, electric circuits, logic circuits and others. Hagler [1] uses them to solve partial differential equations in two independent variables based on the computational molecule approach. Crow [2] uses spreadsheets to solve electrostatic problems by Laplace's

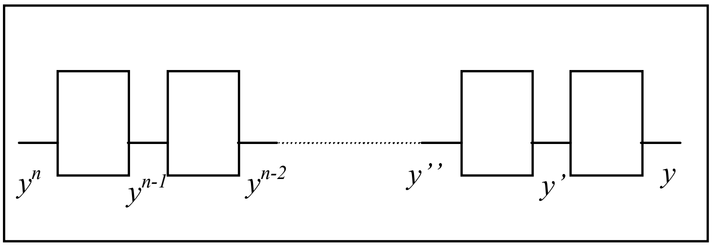


Figure 1: Cascading of integration blocks for an order n differential equation.

equation for up to three-dimensional systems with two dielectrics present in charge-free regions. Shapiro [3] also uses spreadsheets to carry out the numerical calculation of the potential distribution in a pn diode. Kharab [4] employs macro-based programs to obtain the solution of hyperbolic partial differential equations. El-Hajj and Hazim [5] simulate logic networks using a spreadsheet with a customized user interface based on toolbars, menus, help, dialog sheets, and other advanced spreadsheet features.

This work develops a solution methodology of systems of nonlinear differential equations of one variable using spreadsheets. The method allows solving any initial value problem (IVP) of the form:

$$\begin{aligned} f(y^{(n)}, y^{(n-1)}, ..., y', y, x) &= 0 \\ y^{(n-1)}(0) &= y_{n-1} \\ &\cdots \\ y(0) &= y_0 \end{aligned} \tag{1}$$

Where $y^{(n)}$ represents the n^{th} derivative of the function $y(x)$ with respect to $x.$ The solution to (1) uses an integration block with its output equal to the integral of its input. A variable $y(x)$ and its successive derivatives are obtained through a cascaded interconnection of these integration blocks as shown in Figure 1. It is assumed that no discontinuities are present over the integration interval of interest. This type of integration was used in [6] to simulate linear control systems. This approach will also be used to numerically solve a system of m nonlinear differential equations with initial conditions for the functions and their derivative. The differential equations are functions of the same variable x and have the following the form:

$$g_i(y_m^{(n)}, y_m^{(n-1)}, ..., y_m, y_{m-1}^{(n)}, ..., y_{m-1}, ..., y_1^{(n)}, ..., y_1, x) = 0;\ i = 1, m \tag{2}$$

This method is highly accessible and is affordable since spreadsheets are basic software packages that are widely available. Furthermore, it is easy to learn and allows a step-by-step study of the solution, and the investigation of the influence of changing one or more parameters on this solution. The graphical capabilities of spreadsheets give the method a visual bonus. It can actually be used by students to verify worked out solutions by comparing them to the numerical one obtained using this spreadsheet.

Section 2 first presents the "Differential Equations" Toolbar which was developed to solve the nonlinear differential equations. The usage of the method is presented in section 3 using illustrative examples. Section 4 extends the method for a system of nonlinear differential equations. Implementation details of the "Differential Equations" Toolbar are given in section 5. A discussion of the method and concluding remarks are finally given in section 6.

2 The *Differential Equations* Toolbar

In the solution process, a worksheet calculation is considered as one numerical iteration. For this purpose, the spreadsheet calculation mode is set to "Manual" and the number of iterations to be performed in one run is specified. A worksheet range is prepared to capture data calculated throughout the iterations, and save it for plotting. The "Differential Equations" toolbar shown in Figure 2 was developed to initialize or reset a worksheet, to generate cascaded integration blocks, and to calculate a solution. This is done using three buttons: Reset, Integrator, and Run. Upon clicking any of these buttons, an underlying macro causes some related actions to execute, which are described as follows:

2.1 The *Reset* actions

In the reset process, the worksheet is prepared to start the integration process by generating or resetting different parameters. As shown in Figure 2, an initialization flag is generated and stored in cell `B1`. It is set to 0 to indicate that no calculation is required in the worksheet, and it will be set to 1 when calculation is required. A counter stored in cell `B2` displays the number of iterations, or spreadsheet calculations, that are performed. This counter is initialized to zero and incremented for each worksheet calculation. The integration step is stored in cell `B3`, and the value of the variable, which is the product of the integration step and the iteration number, is calculated and stored in cell `B4`. The number of worksheet calculations to be performed in a single run (Iterations) is stored in cell `F1`. In order to capture data calculated in different iterations, the columns "Count", "Var", and "Ycal" are generated (See Figure 2). The column with header "Count" indicates the iteration numbers for which data should be captured for displaying or plotting. By default, data is to be displayed for the first three iterations. The user may copy cell `A13` to range `A14:A`x to display more values (where x is the row number of the last value to display). Moreover, it is possible to set an interval different than unity between two adjacent values by changing cell `A12`, formatted bold for this purpose, to the desired iteration number following iteration 0. The column with header "Var" indicates the values of the variable for different iteration counts. It is obvious that this value is the product of the count and step values. Cell `B11` can be copied to range `B14:B`x for more values to be displayed. The column with header "Ycal" indicates the values of the output for different counts. This output is initially set to `$B$4` and should be edited in the formula in cell `C11` to reflect another output, and then copied to range `C12:C`x. Other outputs can be displayed by generating in column D data similar to C12:Cx but for a different output. The copy command can be used for this purpose by copying cell C11 into cell D11, editing cell D11 to reflect the different output, and then copying D11 into D12:Dx.

2.2 The *Run* actions

Before starting the run process, it is possible to change the default values of the integration step (cell B3) and the "Iterations" number (cell F1) to any desired values. It is also possible to control the data displayed, the number of points, and the interval by manually changing and/or adding information to the sub-columns with headers "Count", "Var" and "Ycal" as previously described. In the run process, the flag is automatically set to 1, and the worksheet is calculated a number of times indicated by the "Iterations" number. The data calculated during the run process is saved in the sub-columns "Var" and "Ycalc". It is possible to "Run" the worksheet (i.e., click the "Run" button) many consecutive times using different number of "iterations", in which case the most recent run continues from where the previous one stopped. In order to repeat the calculations for different parameters such as a new step, the worksheet is "Reset" (i.e., click the "Reset" button), the new parameters entered, then click the "Run" button.

2.3 The *Integrator* button

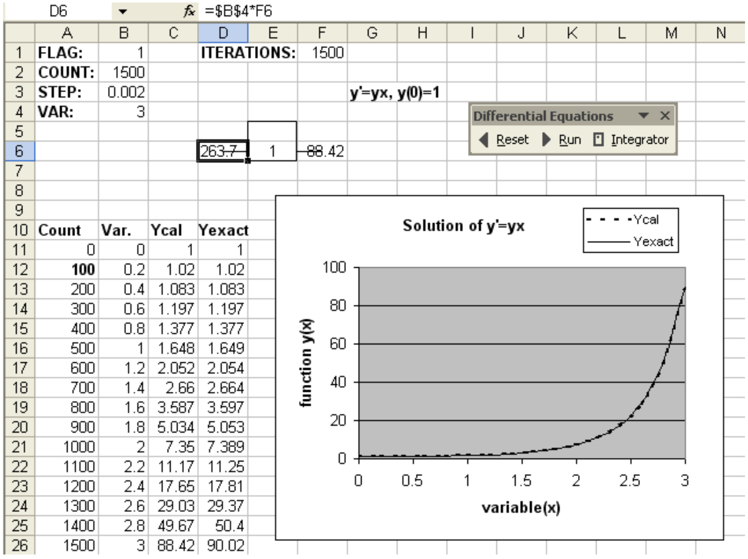
As indicated in the Introduction, an integrator is a block with its output equal to the integral of its input. In order to generate an integrator at a certain location, a cell is selected and the "Integrator" button is clicked. The integrator generated will have the selected cell as its input, and the output will be automatically selected to facilitate the generation of other integrators in cascade. By successively clicking a number of times the "Integrator" button, a number of integrators will be generated in which the output of one is input to the next to help implement a cascaded system as shown in Figure 1. The initial value of an integrator output is in the lower cell of the integrator body and is assigned a default value of 0. As an example, the integrator shown in Figure 2 is generated by selecting cell D6 and clicking at the "Integrator" button. The initial value of the output, which is assumed equal to one, is next written in cell E6 by overriding its default value.

In order to illustrate and test this numerical integration process, examples with known analytical solution were tried and the numerical integration results were found to be in close agreement as presented in the next section.

3 The Solution Process

Consider the differential equation given in (1), which can be written as:

$$y^{(n)} = g(y^{(n-1)}, y^{(n-2)}, ..., y', y, x) \tag{3}$$



D6 =$B$4*F6

	A	B	C	D	E	F	G
1	**FLAG:**	1		**ITERATIONS:**		1500	
2	**COUNT:**	1500					
3	**STEP:**	0.002					**y'=yx, y(0)=1**
4	**VAR:**	3					
5							
6				263.7	1	88.42	
7							
8							
9							
10	**Count**	**Var.**	**Ycal**	**Yexact**			
11	0	0	1	1			
12	**100**	0.2	1.02	1.02			
13	200	0.4	1.083	1.083			
14	300	0.6	1.197	1.197			
15	400	0.8	1.377	1.377			
16	500	1	1.648	1.649			
17	600	1.2	2.052	2.054			
18	700	1.4	2.66	2.664			
19	800	1.6	3.587	3.597			
20	900	1.8	5.034	5.053			
21	1000	2	7.35	7.389			
22	1100	2.2	11.17	11.25			
23	1200	2.4	17.65	17.81			
24	1300	2.6	29.03	29.37			
25	1400	2.8	49.67	50.4			
26	1500	3	88.42	90.02			

Figure 2: Integration tools and solution of example 1.

Assuming that eq (3) has no discontinuity over the integration interval, this equation is implemented by a sequence of integration blocks as in Figure 1, which are easily inserted by clicking the "Integrator" button n times. Any two consecutive blocks are automatically connected since the first block's output is the new active cell, which becomes the input of the second block generated by the second click. So the n outputs of the successive blocks represent the functions $y^{(n-1)}, y^{(n-2)}, ..., y'$, and y. The Excel equation editor is then used to enter formula (3) at the input of the first block using the cell addresses of different block outputs as values for $y^{(n-1)}, y^{(n-2)}, ..., y'$, and y. The initial values of the variable and its derivatives are finally written in their assigned cells, and the worksheet is then prepared to execute with the required step, number of iterations, and output to display and/or plot. To illustrate the method, two examples with known exact solutions are given and the results are compared with those obtained analytically.

Example 1. Consider the following equation:

$$y' = \frac{dy}{dx} = yx, \qquad (4)$$
$$y(0) = 1$$

In order to solve this equation starting from an empty sheet, the "Reset" button is first clicked which generates various elements indicated in the previous section, cell `D6` is next selected and the "Integrator" button is clicked which generates the integrator with input y' (cell `D6`) and output y (Cell `F6`) as shown in Figure 2. The initial value $y(0)$ is entered in cell `E6`. The right-hand side of (4) is entered in cell `D6` as: `=$B$4*F6`. The integration step is changed to 0.002, the iteration number is changed to 600, the "Run" button is clicked to run the worksheet for 600 iterations. The numerical results show good agreement with those based on the exact solution $y = e^{x^2/2}$ in range `D11:D26`.

In order to plot the output `F6` using points at intervals of 100 iterations, cell `A12` is changed to 100, and the variable `$B$4` in the formula generated in cell `C11` is changed to `$F$6`. Cell `A13` is copied to `A14:A26`, cell `B13` is copied to `B14:B26`, and cell `C11` is copied to `C12:C26`. The initial value of the output (for $Count = 0$, $y(0) = 1$) is next written at cell `C11` since it will not be captured in the first iteration in which Count is 1. The value which corresponds to the exact solution is stored in column `D` by writing in `D11` the equation `=EXP(0.5*B11^2)` and by copying it to cells `D12:D26`. The worksheet is “Reset” then “Run” under these new conditions and the Excel “Chart wizard” used to plot of the numerically calculated solution and that obtained from the exact one as shown in Figure 2.

Students can use this implementation to study the influence of one or more parameters on the solution. For example, it is possible to change the value of the step, “Reset” then “Run” the worksheet and see how the graph is recalculated and how the accuracy of the solution is affected. Some parameters such as an initial value or an equation coefficient may be changed to investigate the sensitivity of the solution with respect to such changes. Also, the iteration number can be set to one in order to follow the calculations step by step. It is also possible to unhide cells `D5` and `E5` which contain intermediate results and observe further details on how the integration is performed step by step. It is also possible to increase the number of iterations to conclude on how long the agreement with the exact solution remains valid. Figure 2 shows that in the last few iterations, the error due to the approximations used in the method accumulate and become apparent.

Example 2. Consider the following differential equation:

$$y'' + 4y = x^2 \sin 2x \tag{5}$$

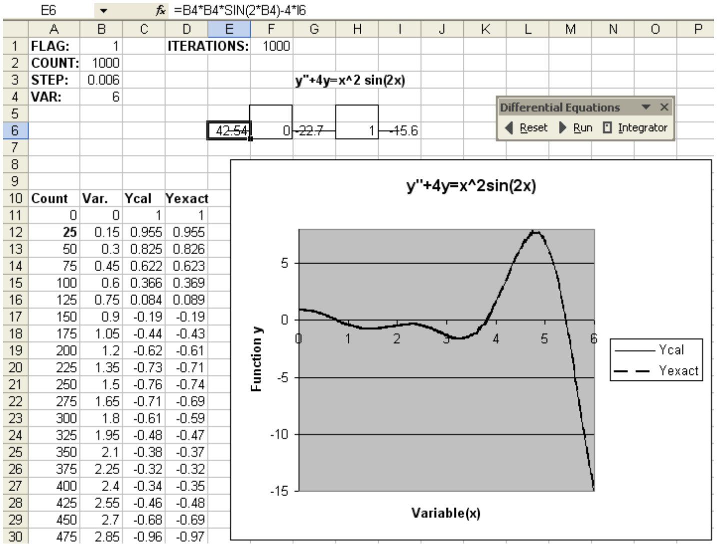
The initial conditions are $y(0) = 1$ and $y'(0) = 0$. The above equation is solved by cascading two integrators in series as shown in Figure 3. The input and output of the first one are at cells `E6` and `G6`, respectively; those of the second one are at cells `G6` and `I6`, respectively. The input equation of the first integrator, given by `=B4*B4*SIN(2*B4)-4*I6`, is entered in cell `E6` and the initial conditions are in cells `F6` and `H6`. The exact solution is given by:

$$y(x) = \left(y(0) - \frac{x^3}{12} + \frac{x}{32}\right)\cos 2x + \left(y'(0) + \frac{x^2}{16} - \frac{2}{128}\right)\sin 2x \tag{6}$$

The values of the function as numerically calculated and the exact ones are calculated as for Example 1. The plots of the exact and calculated values are shown in Figure 3.

4 Systems of Differential Equations

Consider the following system of m differential equations:



E6 =B4*B4*SIN(2*B4)-4*I6

	A	B	C	D	E	F	G	H	I
1	FLAG:	1		ITERATIONS:		1000			
2	COUNT:	1000							
3	STEP:	0.006					y"+4y=x^2 sin(2x)		
4	VAR:	6							
5									
6					42.54	0	-22.7	1	-15.6
7									
8									
9									
10	Count	Var.	Ycal	Yexact					
11	0	0	1	1					
12	**25**	0.15	0.955	0.955					
13	50	0.3	0.825	0.826					
14	75	0.45	0.622	0.623					
15	100	0.6	0.366	0.369					
16	125	0.75	0.084	0.089					
17	150	0.9	-0.19	-0.19					
18	175	1.05	-0.44	-0.43					
19	200	1.2	-0.62	-0.61					
20	225	1.35	-0.73	-0.71					
21	250	1.5	-0.76	-0.74					
22	275	1.65	-0.71	-0.69					
23	300	1.8	-0.61	-0.59					
24	325	1.95	-0.48	-0.47					
25	350	2.1	-0.38	-0.37					
26	375	2.25	-0.32	-0.32					
27	400	2.4	-0.34	-0.35					
28	425	2.55	-0.46	-0.48					
29	450	2.7	-0.68	-0.69					
30	475	2.85	-0.96	-0.97					

Figure 3: Spreadsheet solution of example 2.

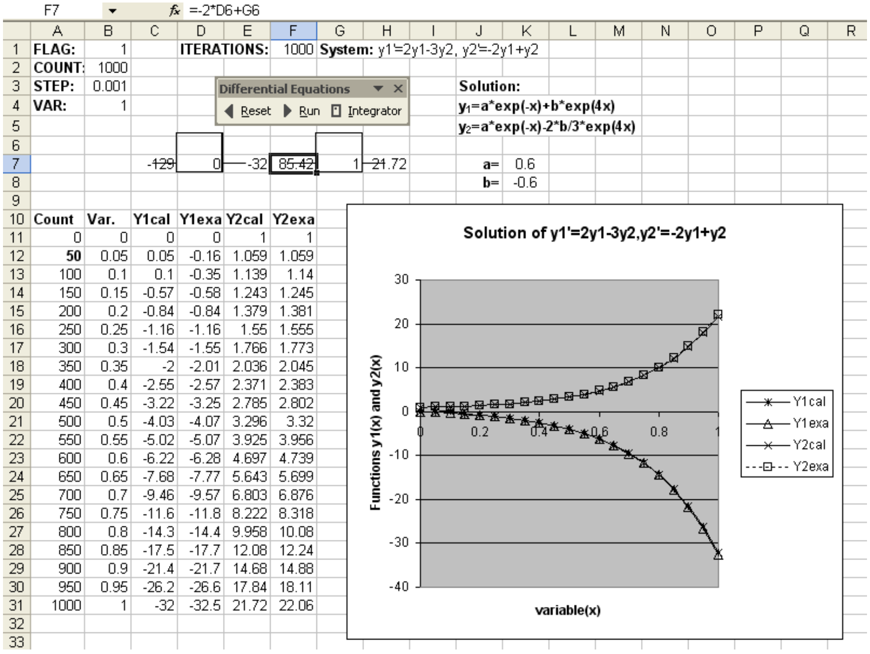
$$
\begin{aligned}
y_1^{(n)} &= g_1(y_1^{(n-1)}, y_2^{(n-1)}, ..., y_1^{'}, y_2^{'}, ..., y_1, y_2, ..., x) \\
y_2^{(n)} &= g_2(y_1^{(n-1)}, y_2^{(n-1)}, ..., y_1^{'}, y_2^{'}, ..., y_1, y_2, ..., x) \\
&\dots \\
y_m^{(n)} &= g_m(y_1^{(n-1)}, y_2^{(n-1)}, ..., y_1^{'}, y_2^{'}, ..., y_1, y_2, ..., x)
\end{aligned}
$$

To obtain a solution, each equation is simulated using the method described in the previous section. However, the spreadsheet row-wise calculation must be taken into consideration when entering the equations. When an equation refers to a variable calculated in preceding cells, the value of the variable and its derivatives in the preceding iteration must be used. For this purpose, the integrator output obtained in the preceding iteration is saved in the upper cell of the integrator body but formatted as hidden for convenience in the display (see section 6). In this case, referring to the upper cell of an integrator rather than its output (i.e., `E5` instead of `F6` as in Figure 2) solves the row-wise calculation constraint. When an equation refers to a variable located in a forward cell, the variable value is obtained directly from this cell as in previous examples.

Example 3. Consider the following system of differential equations:

$$y_1' = 2y_1 - 3y_2 \tag{7}$$

$$y_2' = -2y_1 + y_2 \tag{8}$$



F7 =-2*D6+G6

FLAG: 1 ITERATIONS: 1000 System: y1'=2y1-3y2, y2'=-2y1+y2

COUNT: 1000

STEP: 0.001 Solution:

VAR: 1 y1=a*exp(-x)+b*exp(4x)

y2=a*exp(-x)-2*b/3*exp(4x)

-129 0 -32 85.42 1 21.72 a= 0.6

b= -0.6

Count	Var.	Y1cal	Y1exa	Y2cal	Y2exa
0	0	0	0	1	1
50	0.05	0.05	-0.16	1.059	1.059
100	0.1	0.1	-0.35	1.139	1.14
150	0.15	-0.57	-0.58	1.243	1.245
200	0.2	-0.84	-0.84	1.379	1.381
250	0.25	-1.16	-1.16	1.55	1.555
300	0.3	-1.54	-1.55	1.766	1.773
350	0.35	-2	-2.01	2.036	2.045
400	0.4	-2.55	-2.57	2.371	2.383
450	0.45	-3.22	-3.25	2.785	2.802
500	0.5	-4.03	-4.07	3.296	3.32
550	0.55	-5.02	-5.07	3.925	3.956
600	0.6	-6.22	-6.28	4.697	4.739
650	0.65	-7.68	-7.77	5.643	5.699
700	0.7	-9.46	-9.57	6.803	6.876
750	0.75	-11.6	-11.8	8.222	8.318
800	0.8	-14.3	-14.4	9.958	10.08
850	0.85	-17.5	-17.7	12.08	12.24
900	0.9	-21.4	-21.7	14.68	14.88
950	0.95	-26.2	-26.6	17.84	18.11
1000	1	-32	-32.5	21.72	22.06

Figure 4: Spreadsheet solution of example 3.

The initial conditions are $y_1(0) = 0$ and $y_2(0) = 1$. The above equations are solved by two integrators in row formation as shown in Figure 4. The first integrator has `C7` and `E7` as input and output cells, and those of the second one are `F7` and `H7`. The input equations for `C7` and `F7` simulate the right-hand-side (RHS) of (7) and (8), respectively by the formulae `=2*E7-3*H7` and `=-2*D6+H7`. The exact solution is:

$$y_1 = ae^{-x} + be^{4x} \tag{9}$$

$$y_2 = ae^{-x} - 2be^{4x}/3$$

The coefficients a and b are calculated from the initial conditions as $a = (2y_1(0) + 3y_2(0))/5 = 3/5$ and $b = 3(y_1(0) - y_2(0))/5 = -3/5$. Figure 4 shows the plots of the exact and calculated solutions, which are in good agreement.

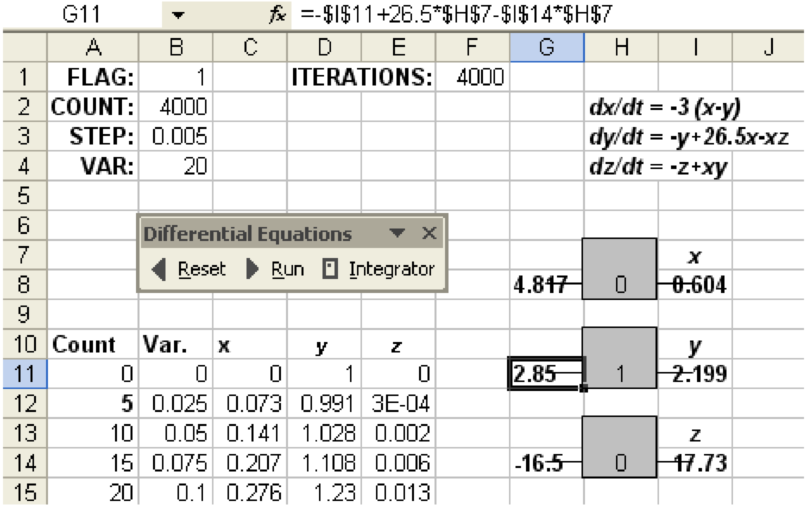
Example 4. Consider the Lorentz Attractor which is described by the following system of nonlinear differential equations:

$$x'(t) = -3[x(t) - y(t)] \tag{10}$$

$$y'(t) = -x(t)z(t) + 26.5x(t) - y(t) \tag{11}$$

$$z'(t) = x(t)y(t) - z(t) \tag{12}$$

The initial conditions are $x(0) = z(0) = 0$, $y(0) = 1$. The above equations are solved by using three integrators as shown in Figure 5. The three integrators represent (11) (12)



G11 =-$I$11+26.5*$H$7-$I$14*$H$7

	A	B	C	D	E	F	G	H	I	J
1	FLAG:	1		ITERATIONS:		4000				
2	COUNT:	4000						dx/dt = -3 (x-y)		
3	STEP:	0.005						dy/dt = -y+26.5x-xz		
4	VAR:	20						dz/dt = -z+xy		
5										
6										
7									x	
8							4.817	0	0.604	
9										
10	Count	Var.	x	y	z				y	
11	0	0	0	1	0		2.85	1	2.199	
12	5	0.025	0.073	0.991	3E-04					
13	10	0.05	0.141	1.028	0.002				z	
14	15	0.075	0.207	1.108	0.006		-16.5	0	17.73	
15	20	0.1	0.276	1.23	0.013					

Figure 5: Spreadsheet solution of example 4.

and (13) with their input cells being G8, G11 and G14 and their output cells I8, I11, and I14, respectively. The input of each integrator represent the RHS of one of the above equations. For example, the input of the second integrator second integrator is the RHS of (11) and is implemented in cell G11 by the formula =-$I$11+26.5*$H$7-$I$14*$H$7, as shown in Figure 5. The plots of $y(x)$ and $z(y)$ are given in Figures 6 and 7 and are in close agreement with those obtained using Mathematica.

5 Toolbar Implementation Details

In this section, the implementation of the macros linked to the three toolbar buttons is explained. In each case, the actions related to the macro are recorded, then the visual basic code generated by the macro recorder is edited, and the macro is linked to the toolbar button. In editing a macro, some references are made relative, and some unnecessary code which is often inserted by the Visual Basic editor is eliminated. This development approach reduces the programming effort that is required when the macros are written from scratch using the Visual Basic language. Circular references are needed as they are present in the method concept: the output of an integrator is a function of the input, which, in many cases, is a function of the output calculated during the previous iteration. Many authors have solved differential equations [1-4] without using circular references. In this kind of solution, an iteration or calculation step corresponds to a set of cells, and these cells will contain numbers and formulas representing intermediate and final results in successive iterations, which require the user to write and edit the integration formulas used in these cells. However, such a solution approach cannot be easily extended to solve large systems of equations with a large number of iterations in which the user, having investigated the details, is now to

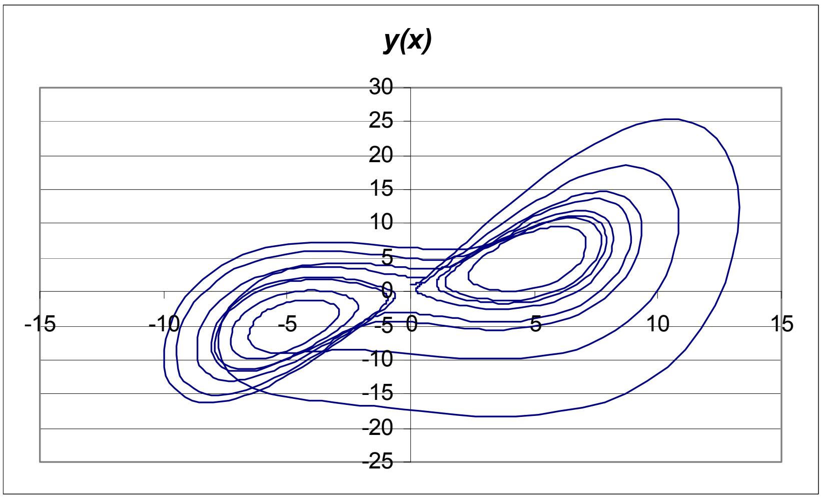


Figure 6: Plot of $y(x)$ in the Lorentz attractor system of example 4.

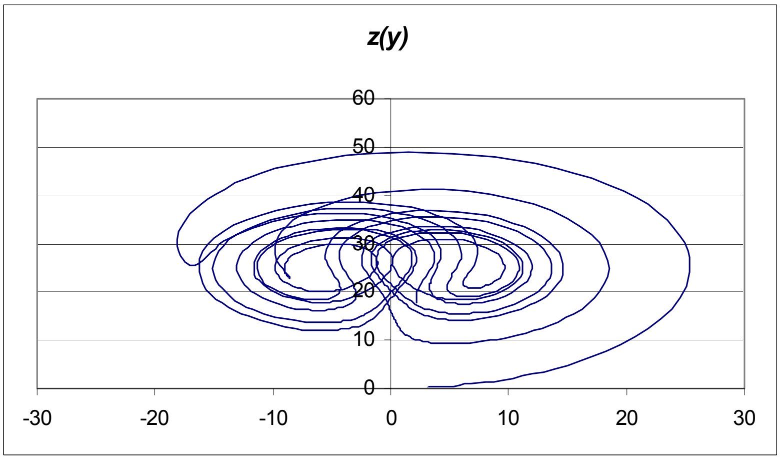


Figure 7: Plot of $z(y)$ in the Lorentz attractor system of example 4.

focus on the overall systems behavior. In this proposed approach intermediate results in earlier iterations are successively overwritten and the required results displayed to help the user focus her/his attention on the system as a whole.

In the case of the *Reset* macro, the calculation mode is set to manual with a default number of iterations equal to one, and the column width is set to a default fixed value. All information in ranges `A1:B4`, `E1`, `F1`, `A10:C13`, is next written according to the specifications described in Section 2. For example, the initialization flag, stored in cell `B1`, is set to 0. The counter stored in cell `B2` is initialized to zero and incremented for each worksheet calculation to indicates the number of iterations. This is done by writing at cell `B2` the conditional formula: `=IF(B1=0,0,B2+1)`. The value of the variable, which is the product of the integration step and the iteration number, is calculated and stored in cell `B4` using the formula: `=B2*B3`. In order to set an interval different than unity between two adjacent values, the interval value is assumed equal to `A12-A11`, and the following formula is written in `A13:  =A12+$A$12-$A$11`. The variable value at cell `B11` is equal to `=A11*$B$3` which is copied into cells `B12:B13`. Capturing the data for display (the default is `$B$4`) is done by writing in cell `C11` the formula

```
=IF($B$1=0,0,IF($A11=$B$2,$B$4,C11))          (13)
```

This formula is copied into cells `C12:C13`.

In the case of the *Run* macro, the recorded events are: Flag `B1` is set to 1, the number of iterations is set to the value indicated in cell `F1`, the recalculation mode is set to manual, and the worksheet calculates.

In the case of the integration block with input equal to a function $f(x)$ and output equal to the integral $y(x)$ of this function, this integral is obtained by solving the equation:

$$\frac{dy}{dx} = f(x) \tag{14}$$

Equation 14 is assumed to have an initial value $y(0) = y_0$. Given an input sequence x_i with $i = 0, 1, \ldots$, the corresponding output sequence y_i can be obtained using a numerical integration formula [7]. We will illustrate our method using the trapezoidal rule but higher order rules can readily be used:

$$y_{i+1} = y_i + \frac{h}{2}\left(f(x_i) + f(x_{i+1})\right) \tag{15}$$

The macro *Integrator* is recorded while constructing an integrator with input `D6` and output `F6` as shown in Figure 2. The drawing toolbar is first use to draw this integrator. The initial value of y is stored in cell `E6` and assigned zero as a default value. The input cell `D6` of the integrator implements $f(x_{i+1})$ which is the right-hand side of the differential equation being evaluated at the $i + 1$ worksheet calculation. For a correct operation of the integrator, $f(x_i)$ and y_i must be updated by reading their values from the previous worksheet calculation. Due to the row-wise calculation of Excel worksheets (row-wise calculation mode must be used in other spreadsheet programs), this is made possible by storing $f(x_i)$ and y_i in cells calculated prior to `D6` and `F6`, (i.e., cells `D5` and E5). The previous input value $f(x_i)$ is stored in cell D5 by writing in this cell the formula: `=IF($B$1=0,0,D6)`. The previous output value y_i is stored in cell E5 by writing in this cell the formula: `=IF($B$1=0,E6,F6)`. The output y_{i+1} is obtained at cell F6 by implementing 15 as follows: `=IF($B$1=0,E6,E5+0.5*$B$3*(D6+D5))`. For convenience in the display, cells D5 and E5 which contain such intermediate calculation are hidden and can be unhidden again using the format/hide-unhide commands. The macro *Integrator* is next edited to ensure that all references and drawing are relative to the input cell in the current active worksheet. It is finally attached to the *Integrator* toolbar button. As shown in the previous sections, when the "Integrator" toolbar button is clicked, an integration block is generated at the current active cell that becomes its input, its integration formulae are automatically inserted, and its output cell is made the new active cell.

6 Conclusion

A method for solving nonlinear systems of differential equations using modern spreadsheet programs has been presented. The nonlinear differential equations are of one variable and assumed to have no discontinuities over the integration interval. It was shown how this solution is achieved in a very simple and modular approach using a single integration block having as input a given variable and as output the integral of the variable. It was demonstrated how to solve higher order equations by simply cascading these blocks. The method is highly accessible and of low cost since spreadsheets are widely available. The method presented has its advantages and limitations when compared with other methods. It is particularly useful in educational settings where budgets are limited and students may not have means to access sophisticated packages, or time to write a sophisticated program. It uses a basic software package widely available and familiar to a large class of computer users which is good for obtaining quick solution using all available spreadsheet features. It is easy to use since solving an equation requires transforming this equation into cascaded blocks, in addition to a click on *Reset* button and a number of clicks on an "Integrator" button equal to the equation order, and entering parameters and the equation to the first block.

This method is very useful for educational purposes when one solves an equation and analyzes the results, step by step, recording the successive values in the worksheet, then repeating the calculations with different parameter values, and plotting the results, and finally taking full advantage of various spreadsheet features to process and analyze the results. This implementation can be used as well by students taking a course in Numerical Analysis who can understand the trapezoidal rule implementation and follow a similar approach to implement and compare other integration formulas.

This implementation is sufficiently accurate for a large number of practical applications since Excel stores numbers and performs calculations using 15 digits of precision. To increase accuracy, it is possible to reduce the integration step or use more sophisticated formulas in simulating integration blocks. For example, the trapezoidal rule used to simulate the integrator may be replaced by a higher order rule [7].

References

[1] M. Hagler, “Spreadsheet Solution of Partial Differential Equations”, IEEE Trans. on Education, Vol. 30, No. 3, August, 1987, pp. 130-134.

[2] T. T. Crow, “Solutions to Laplace’s equation using spreadsheets on a personal computer”, American Journal of Physics, 55 (9), September, 1987, pp. 817-823.

[3] F. R. Shapiro, “The Numerical Solution of Poisson’s Equation in a pn Diode Using a Spreadsheet”, IEEE Trans. on Education, Vol. 38, No. 4, November, 1995, pp. 380-384.

[4] A. Kharab and R. Kharab, “Spreadsheet Solution of Hyperbolic Partial Differential Equations”, IEEE Trans. on Education, Vol. 40, No. 1, February, 1997, pp. 103-110.

[5] A. El-Hajj and M. Hazim, “On Using Spreadsheets for Logic Networks Simulation”, IEEE Trans. on Education, Vol. 41, No.4, pp.317-320, November 1998.

[6] A.El-Hajj, "Functional Simulation Using Spreadsheets", Simulation, Vol. 73, No.2, pp.80-90, August 1999.

[7] S.C.Chapra and R.P.Canale, Numerical Methods for Engineers, 2nd Ed., New York: McGraw-Hill, 1990, pp. 478-482.