TI-84 Program Runge-Kutta

Re: text p.144 for TI-85 implementation

Re: text p.134 for TI-Nspire CX CAS implementation, 5th ed

\[ Y' = F(x,y) \]

Comment

PROGRAM: RKSTRING

:ClrHome

:Input "Y' =", Str1

:String->Equ(Str1, 1Y)

:Input "X_0 =", X

:Input "Y_0 =", Y

:Input "X MAX =", M

:Input "NO. OF STEPS =", N

:(M-X)/N -> H

:0 -> L

:ClrAllLists

:For(I,1,N)

:ClrHome

:L+1 -> L

:X -> L_1(L)

:Y -> L_2(L)

:1Y_1 -> A

:(L_1(L) + .5H) -> X

Midpoint
\[(L_2(L)) + 0.5A \rightarrow Y\]  
Midpoint predictor, i.e. add \(0.5 \times H \times A\) to \(L_2(L)\)

\[Y_1 \rightarrow B\]  
Assign the 1st updated function value of \(Y_1\) to \(B\), \(B\) is the 2nd slope \(K_2\)

\[(L_2(L)) + 0.5B \rightarrow Y\]  
Improved midpoint predictor, i.e. add \(0.5 \times H \times B\) to \(L_2(L)\)

\[Y_1 \rightarrow C\]  
Assign the 2nd updated function value of \(Y_1\) to \(C\), \(C\) is the 3rd slope \(K_3\)

\[(L_1(L)) + H \rightarrow X\]  
New \(X\) value

\[(L_2(L)) + HC \rightarrow Y\]  
Endpoint predictor

\[Y_1 \rightarrow D\]  
Assign the 3rd updated function value of \(Y_1\) to \(D\), \(D\) is the 4th slope \(K_4\)

\[(A + 2B + 2C + D)/6 \rightarrow K\]  
Average slope is \(K\)

\[(L_2(L)) + HK \rightarrow Y\]  
Updated approximate \(Y\) value using average slope and step size, i.e. \(H \times K\)

:Disp “\(X=\)”\(X\), “\(Y=\)”\(Y\)  
Display \(x\) and approximate \(y\) values

:Pause  
Requires user intervention to press the enter key

:End  
End of the loop

:L+1 \rightarrow L  
Update \(L\) counter

:X \rightarrow L_1(L)\]  
Add last \(X\) value to \(L_1\) list

:Y \rightarrow L_2(L)\]  
Add last \(Y\) value to \(L_2\) list

:DelVar \(Y_1\)  
Delete from memory the contents of \(Y_1\)

:PlotsOn  
Turn on all stat plots

:Plot1(Scatter, \(L_1\), \(L_2\))  
Define Plot1 as a scatter plot with lists \(L_1\) and \(L_2\)

:ZoomStat  
Redefine the viewing window to display all points

:DispGraph  
Display the graph

:Pause  
Requires user intervention to press the enter key
:Disp "X in \( L_1 \), "Y in \( L_2 \)\) In the graph, X values are in \( L_1 \) and Y values are in \( L_2 \)

:Stop Ends program execution and returns to home screen when “DONE” is printed

Notes:

1. Confirmation of the results when executing the above code can be seen on p.138 (4th ed) and p.128 (5th ed) in the text in figure 2.6.2. In this example, prompts for \( Y' \) requires the input \( X+Y \) (i.e. ALPHA \( X+\text{ALPHA} \ Y \)). The prompt for \( X' \) requires the input 0, the prompt for \( Y' \) requires the input 1, the X MAX prompt requires input of 1, and the NO. OF STEPS prompt requires input of 10. Whenever a pause occurs on the TI screen, press enter. At the end, a scatter plot of the \((x,y)\) values will appear, press enter and DONE will appear on the home screen indicating that program execution has ended.

2. Another example to try is: Consider \( 0 \leq x \leq 0.5 \) and we want only 2 steps with \( y'=y \) and \( y(0)=1 \). For this example the actual solution is \( y = Ce^x = e^x \). At the prompt for \( Y' \) enter \( Y \), at the prompt for \( X' \) enter 0, at the prompt for \( Y' \) enter 1, the X MAX prompt requires input of 0.5, and the NO. OF STEPS prompt requires input of 2. A summary of generated values for this Runge-Kutta problem follows:

<table>
<thead>
<tr>
<th>X</th>
<th>0</th>
<th>0.25</th>
<th>0.5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Y actual</td>
<td>1</td>
<td>1.28403</td>
<td>1.64872</td>
</tr>
<tr>
<td>Y approximate</td>
<td>1</td>
<td>1.28402</td>
<td>1.64870</td>
</tr>
</tbody>
</table>

Note that 5 place accuracy was used to show the excellent agreement between actual and approximate values. These results are better than those for the Euler method.

3. When creating the TI-84 program, use the PGRM key to access commands.

4. When done creating the program, 2nd Quit.

5. One has to press the enter key whenever the Pause instruction is encountered in the program. This is done purposely to allow the user to read all lines of output.

6. \( \rightarrow \) is created with STO key.

7. To add lines to the code:

a) I first added as many lines as needed with the entry INPUT, i.e. 2nd insert->PGRM->select INPUT under I/O->enter, then do this repeatedly for as many lines as needed

b) Then rewrite over the newly created INPUT lines

c) Note that the PlotsOn command is from 2nd catalog