UDC: 517.925.7:519.62
Professional paper

SOLVING INITIAL-VALUE DIFFERENTIAL PROBLEMS USING NUMERICAL MULTISTEP METHOD

Ylldrita Seferi[1*], Flamure Sadiki[1], Alit Ibraimi[1], Krutan Rasimi[1]

[1*]*Department of Mathematics, University of Tetova, North Macedonia*

[*]*Corresponding author-mail: ylldrita.seferi@unite.edu.mk*

Abstract

Differential equations are used to model problems in science and engineering that involve the change of some variable with respect to another. In real-life problems, the differential equation that models the problem is too complicated to solve exactly for this reason, in recent years much attention has been devoted to deriving numerical methods for approximating their solution. In particular, in this paper we consider the use of Adams-Bashforth like a multi-step method. Adams-Bashforth method of different steps is constructed, and then we use the fourth-step one, on Mathematica Package for numerical approaches. The numerical results are compared with analytical ones, shown in different ways, tables and graphics, accompanied by examples of their use.

Keywords: Differential Equation, Initial-Value Problem, Adams-Bashforth Method, Numerical Approach, Analytical Solution

1. Introduction

Most of these problems require the solution of an initial-value problem, that is, the solution to a differential equation that satisfies a given initial condition. In common real-life situations, the differential equation that models the problem is too complicated to solve exactly and one of two approaches is taken to approximate the solution.

One-step methods works with the approximation for the mesh point t_{i+1} involves information from only one of the previous mesh points, t_i. Although these methods might use function evaluation information at points between t_i and t_{i+1}, they do not retain that information for direct use in future approximations [1,2,3]. All the information used by these methods are obtained within the subinterval over which the solution is being approximated. The approximate solution is available at each of the mesh points t_0, t_1, ..., t_i before the approximation at t_{i+1} is obtained, and because the error $\left|w_j - y(t_j)\right|$ tends to increase with j, so it seems reasonable to develop methods that use these more accurate previous data when approximating the solution at t_{i+1}. Methods using the approximation at more than one previous mesh point to determine the approximation at the next point are called multistep methods. The precise definition of these methods follows, together with the definition of the two types of multistep methods [1,4,5].

Definition 1.1. The $m-$ step multistep method for solving the initial-value problem

$$y' = f(t, y) \quad a \le t \le b, \quad y(a) = \alpha \tag{1.1}$$

has a difference equation for finding the approximation w_{i+1} at the mesh point t_{i+1} represented by the following equation, where *m* is an integer greater than 1:

$$w_{i+1} = a_{m-1}w_i + a_{m-2}w_{i-1} + ... + a_0 w_{i+1-m} + h\left[b_m f\left(t_{i+1}, w_{i+1}\right) + b_{m-1} f\left(t_i, w_i\right) + ... + b_0 f\left(t_{i+1-m}, w_{i+1-m}\right)\right]$$

for $i = m-1, m, ..., N-1$, where $h = \frac{b-a}{N}$. $a_0, a_1, ..., a_{m-1}$ and $b_0, b_1, ..., b_m$ are constants, and the starting values $w_0 = \alpha_o$, $w_1 = \alpha_1$, ..., $w_{m-1} = \alpha_{m-1}$ are specified [5].For $b_m = 0$, the method is called explicit, or open,when $b_m \neq 0$ the method is called implicit, or closed [5,6].

2. Adams-Bashforth Method

The methods introduced by Astronomer Royal John Couch Adams in 1855 were the first of high order that use only function evaluations, and in their current variable-order, variable step-size implementations they are among the most efficient methods for general non stiff initial value problems.Adams was particularly interested in the using his ability for accurate numerical calculations to investigate the orbits of the planets. He predicted the existence of Neptune by analyzing the irregularities in the planet Uranus, and developed various numerical integration techniques to assist in the approximation of the solution of differential equations [5,7].

Let $y' = f(t, y(t))$ be an initial-value problem, with initial condition $y(a) = \alpha$ for $a \le t \le b$. If integrated overthe interval $[t_i, t_{i+1}]$has the property that:

$$\int_{t_i}^{t_{i+1}} y'(t)dt = \int_{t_i}^{t_{i+1}} f(t, y(t))dt \Rightarrow y(t_{i+1}) - y(t_i) = \int_{t_i}^{t_{i+!}} f(t, y(t))dt \quad (2.1)$$

To derive an Adams-Bashforth explicit *m*-step technique, we form the backward difference polynomial through t_{i-1}, t_i given with,

$$p(t) = \frac{t - t_i}{t_{i-1} - t_i} f(t_{i-1}, y(t_{i-1})) + \frac{t - t_{i-1}}{t_i - t_{i-1}} f(t_i, y(t_i)) =$$
$$= \frac{t_i - t}{h} f(t_{i-1}, y(t_{i-1})) + \frac{t - t_{i-1}}{h} f(t_i, y(t_i))$$

the step size is $h = t_i - t_{i-1}$, also the integral taken from (2.1) can be transformed to:

$$\int_{t_i}^{t_{i+1}} f(t, y(t))dt \approx \int_{t_i}^{t_{i+!}} p(t)dt = \int_{t_i}^{t_{i+1}} \left[\frac{t_i - t}{h} f(t_{i-1}, y(t_{i-1})) + \frac{t - t_{i-1}}{h} f(t_i, y(t_i))\right]dt =$$
$$\left[f(t_{i-1}, y(t_{i-1}))\left(-\frac{1}{2}\right)\frac{(t_i - t)^2}{h} + f(t_i, y(t_i))\frac{(t - t_{i-1})^2}{2h}\right]\Bigg|_{t_i}^{t_{i+1}} =$$
$$= \frac{3h}{2} f(t_i, y(t_i)) - \frac{h}{2} f(t_{i-1}, y(t_{i-1}))$$

From $y(t_{i+1})-y(t_i)=\int\limits_{t_i}^{t_{i+1}} f(t,y(t))dt$ we have an numerical approach method given with:

$$y(t_{i+1})=y(t_i)+\frac{h}{2}[3f(t_i,y(t_i))-f(t_{i-1},y(t_{i-1}))] \tag{2.2}$$

named as two-step numerical method of Adams-Bashforth.

Example 2.1. Adams-Bashforth with polynomial of order 0:

$$y_{i+1}=y_i+\int\limits_{t_i}^{t_{i+1}} p_0(t)dt$$

$$p_0(t)=const=f(t_i,y_i)$$

$$y_{i+1}=y_i+\int\limits_{t_i}^{t_{i+1}} f(t_i,y_i)dt=y_i+f(t_i,y_i)\Bigg|_{t_i}^{t_{i+1}}=y_i+f(t_i,y_i)(t_{i+1}-t_i)=y_i+f(t_i,y_i)\cdot h$$

$$y_{i+1}=y_i+f(t_i,y_i)\cdot h \tag{2.3}$$

This means that this method, with the presented condition, is nothing but the Euler method with local error $o(h^2)$and global error $o(h)$. The Adams-Bashforth method with the polynomial of degree 0 is identified with the Euler method, the error in that case is the same as in the case of the identified method [6,7].

Example 2.2. Adams-Bashforth with polynomial of order 1:

$$y_{i+1}=y_i+\int\limits_{t_i}^{t_{i+1}} p_1(t)dt$$

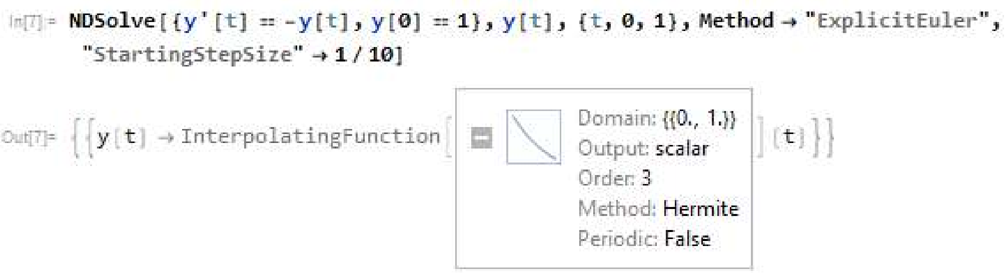
$$p_1(t)=f_i+\frac{f_i-f_{i-1}}{h}(t-t_i),\ f_i=f(t_i,y_i)$$

$$\int\limits_{t_i}^{t_{i+1}}\left(f_i+\left[\frac{f_i-f_{i-1}}{h}\right](t-t_i)\right)dt=f_i(t_{i+1}-t_i)+\frac{f_i-f_{i-1}}{h}\cdot\frac{(t_{i+1}-t_i)^2}{2}=f_ih+\frac{f_i-f_{i-1}}{h}\cdot\frac{h^2}{2}$$

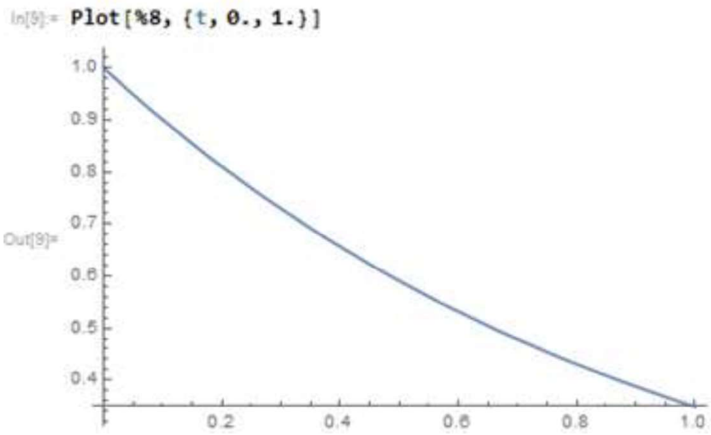
$$y_{i+1}=y_i+\frac{h}{2}(3f_i-f_{i-1}) \tag{2.4}$$

This means that this method, with the presented condition, is nothing but the method of Trapezoid with local error $o(h^3)$and global error $o(h^2)$. The Adams-Bashforth method with the polynomial of degree 1 is identified with the method of Trapezoid, the error in that case is the same as in the case of the identified method [5,7].

Example 2.3. The Euler method is implemented on Mathematica, like “ExplicitEuler”. We give an example of solving an ordinary differential equation using NDSolve of Mathematica package. The differential equation taken in this example is: $y'(t) = -y(t)$ on the segment $[0,1]$ with step size $h = \frac{1}{10}$.



```
In[7]:= NDSolve[{y'[t] == -y[t], y[0] == 1}, y[t], {t, 0, 1}, Method → "ExplicitEuler",
          "StartingStepSize" → 1 / 10]

Out[7]= {{y[t] → InterpolatingFunction[ Domain: {{0., 1.}}
                                        Output: scalar
                                        Order: 3
                                        Method: Hermite
                                        Periodic: False ][t]}}
```



```
In[9]:= Plot[%8, {t, 0., 1.}]

Out[9]=
```

However we can’t integrate $f(t, y(t))$ without knowing $y(t)$, the solution to the problem, so we instead integrate an interpolating polynomial $p(t)$ to $f(t, y(t))$, one that is determined by some of the previously obtained data points $(t_0, w_0), (t_1, w_1), ..., (t_i, w_i)$. When we assume, in addition, that $y(t_i) \approx w_i$, equation (1.1) becomes $y(t_{i+1}) \approx w_i + \int_{t_{i+1}}^{t_i} p(t)dt$.

To derive an Adams-Bashforth explicit *m*-step technique, we form the backward difference polynomial $p_{m-1}(t)$ through $(t_i, f(t_i, y(t_i))), (t_{i-1}, f(t_{i-1}, y(t_{i-1}))), ..., (t_{i-m+1}, f(t_{i-m+1}, y(t_{i-m+1})))$ some numbers $\mu_i \in (t_{i-m+1}, t_i)$ exist with:

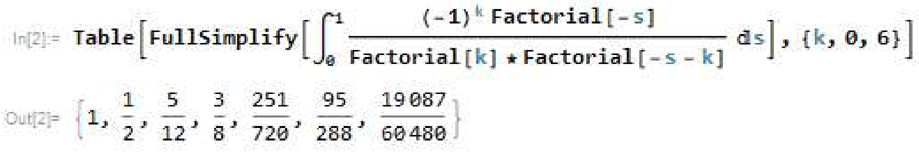
$$f(t, y(t)) = p_{m-1}(t) + \frac{f^{(m)}(\mu_i, y(\mu_i))}{m!}(t-t_i)(t-t_{i-1})\cdot\ldots\cdot(t-t_{i-m+1})$$

Introducing the variable substitution $t = t_i + sh$ with $dt = hds$ into $p_{m-1}(t)$ and the error term implies that

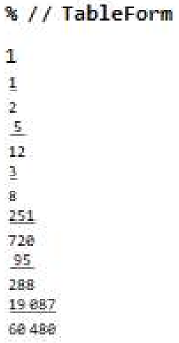
$$\int_{t_i}^{t_{i+1}} f(t, y(t))dt = \int_{t_i}^{t_{i+1}} \sum_{k=0}^{m-1} (-1)^k \binom{-s}{k} \nabla^k f(t_i, y(t_i))dt +$$
$$+ \int_{t_i}^{t_{i+1}} \frac{f^{(m)}(\mu_i, y(\mu_i))}{m!}(t-t_i)(t-t_{i-1})\cdot\ldots\cdot(t-t_{i-m+1})dt = \sum_{k=0}^{m-1} \nabla^k f(t_i, y(t_i))h(-1)^k \int_0^1 \binom{-s}{k} ds +$$
$$+ \frac{h^{m+1}}{m!} \int_0^1 s(s+1)\cdot\ldots\cdot(s+m-1) f^{(m)}(\mu_i, y(\mu_i))ds$$

$$\int_{t_i}^{t_{i+1}} f(t, y(t))dt = \sum_{k=0}^{m-1} \nabla^k f(t_i, y(t_i))h(-1)^k \int_0^1 \binom{-s}{k} ds + \frac{h^{m+1}}{m!} \int_0^1 s(s+1)\cdot\ldots\cdot(s+m-1) f^{(m)}(\mu_i, y(\mu_i))ds$$
(2.4)

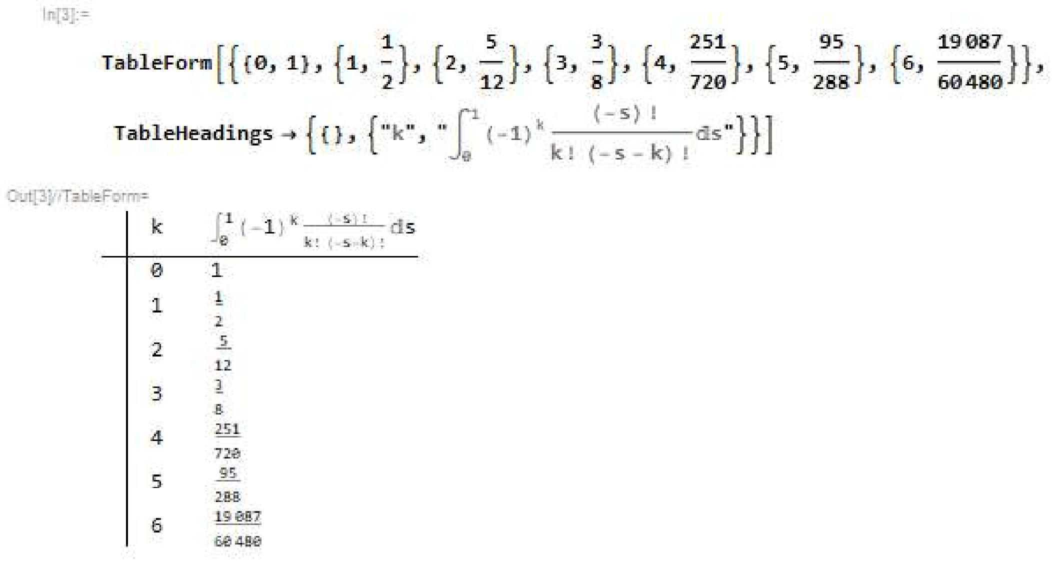
The integral $(-1)^k \int_0^1 \binom{-s}{k} ds$ for various values of k are easily evaluated and are listed using Mathematica package [8,9].



```
In[2]:= Table[FullSimplify[∫₀¹ ((-1)^k Factorial[-s]) / (Factorial[k] * Factorial[-s - k]) ds], {k, 0, 6}]

Out[2]= {1, 1/2, 5/12, 3/8, 251/720, 95/288, 19087/60480}
```



```
% // TableForm

1
1
2
5
12
3
8
251
720
95
288
19087
60480
```



```
In[3]:=
TableForm[{{0, 1}, {1, 1/2}, {2, 5/12}, {3, 3/8}, {4, 251/720}, {5, 95/288}, {6, 19087/60480}},
  TableHeadings -> {{}, {"k", "∫_0^1 (-1)^k (-s)!/(k! (-s - k)!) ds"}}]

Out[3]//TableForm=
  k   ∫_0^1 (-1)^k (-s)!/(k! (-s-k)!) ds
  0   1
  1   1/2
  2   5/12
  3   3/8
  4   251/720
  5   95/288
  6   19087/60480
```

Weighted Mean Value Theorem for Integrals can be used to deduce that for some number $\xi_i \in (t_{i+1-m}, t_{i+1})$, the error term in becomes

$$\frac{h^{m+1}}{m!}\int_0^1 s(s+1)\cdot\ldots\cdot(s+m-1)f^{(m)}(\mu_i, y(\mu_i))ds = \frac{h^{m+1}f^{(m)}(\xi_i, t(\xi_i))}{m!}\int_0^1 s(s+1)\cdot\ldots\cdot(s+m-1)ds =$$

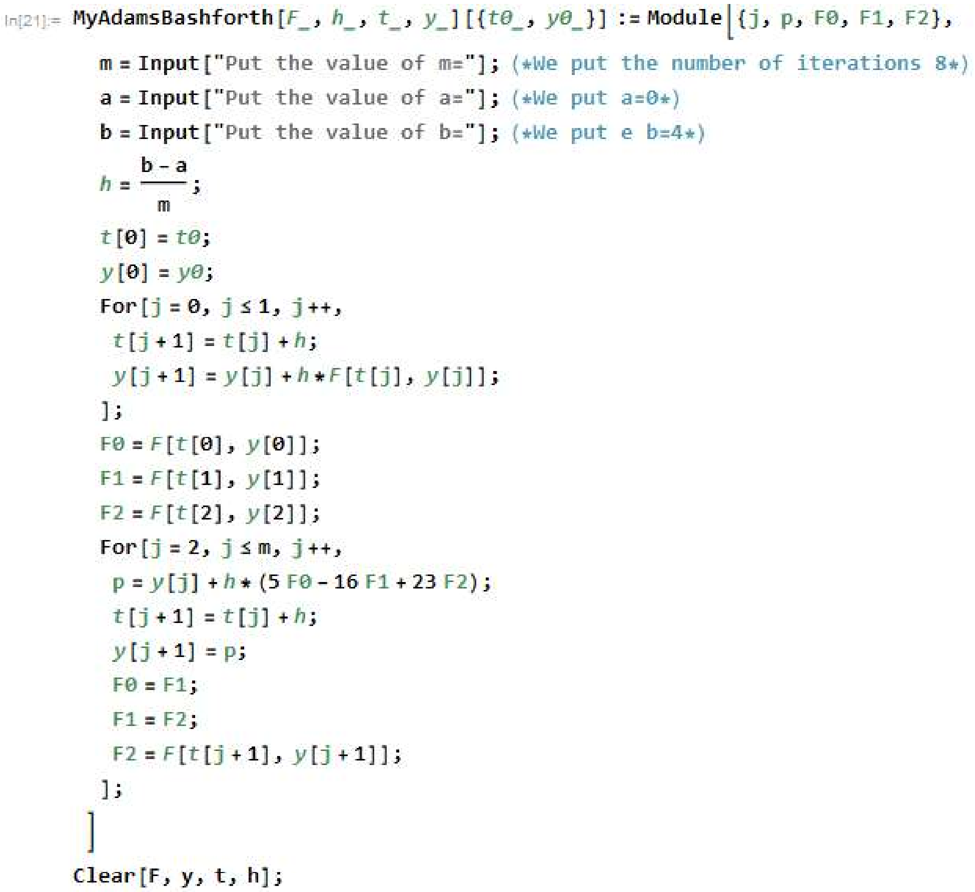
$$= h^{m+1}f^{(m)}(\xi_i, y(\xi_i))(-1)^m\int_0^1 \binom{s}{-m}ds$$

For different steps, the numerical method of Adams-Bashforth, have different formulas. Those different
forms are given in the next table [7].

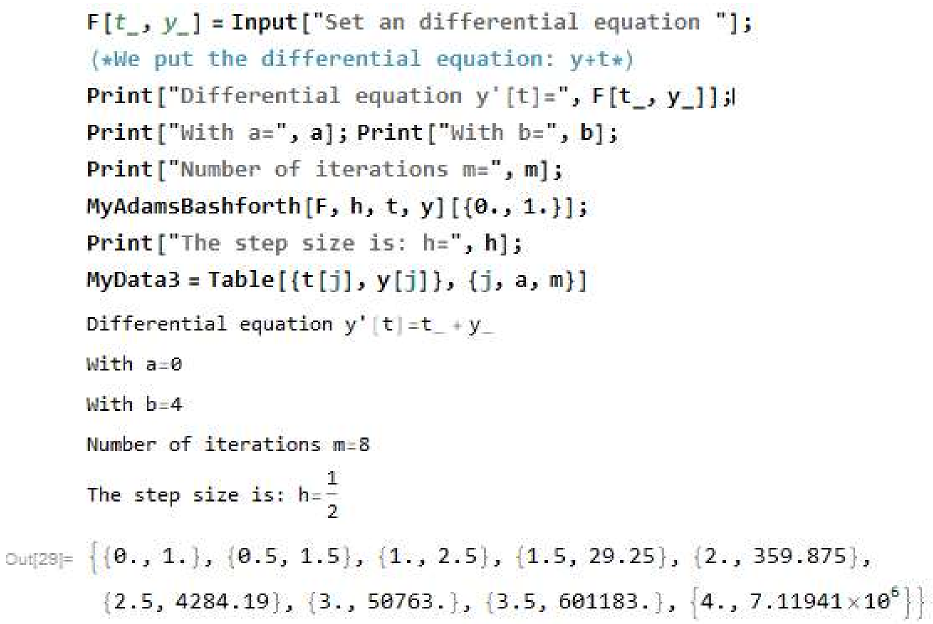
Step	Formula	Error
1	$y_{i+1} = y_i + hf_i$	$\frac{h}{2}y''(\xi)$
2	$y_{i+2} = y_{i+1} + \frac{h}{2}[3f_{i+1} - f_i]$	$\frac{5h^2}{12}y'''(\xi)$
3	$y_{i+3} = y_{i+2} + \frac{h}{12}[23f_{i+2} - 16f_{i+1} + f_i]$	$\frac{3h^3}{8}y^{(4)}(\xi)$
4	$y_{i+4} = y_{i+3} + \frac{h}{24}[55f_{i+3} - 59f_{i+2} + 37f_{i+1} - 9f_i]$	$\frac{251h^4}{720}y^{(5)}(\xi)$
5	$y_{i+5} = y_{i+4} + \frac{h}{720}[1901f_{i+4} - 2774f_{i+3} + 2616f_{i+2} - 1274f_{i+1} + 251f_i]$	$\frac{95h^5}{2888}y^{(6)}(\xi)$

3. Numerical examples

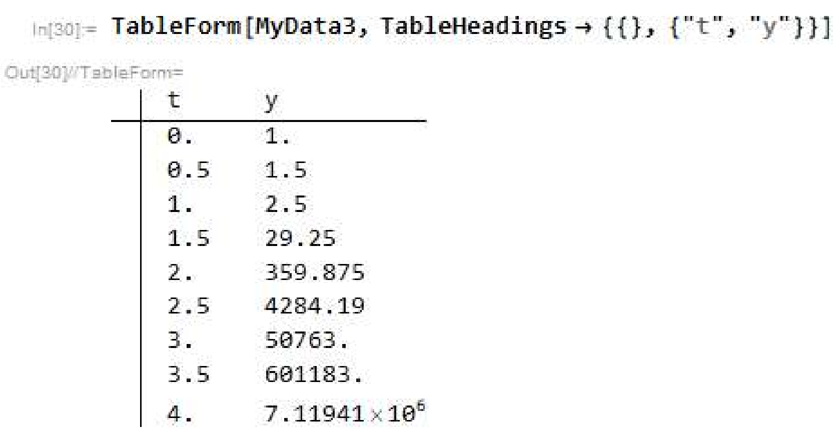
*Example 3.1.*An implementation of Adams-Bashforth method for solving differential equation using the package Mathematica [9]. The differential equation used in this experiment is $y = t - y(t)$ with initial value $y(0) = 0$. The results are shown by fourth step Adams-Bashforth Method [7,8], with steps-size $h = 0.5$ for $t \in [0, 4]$.



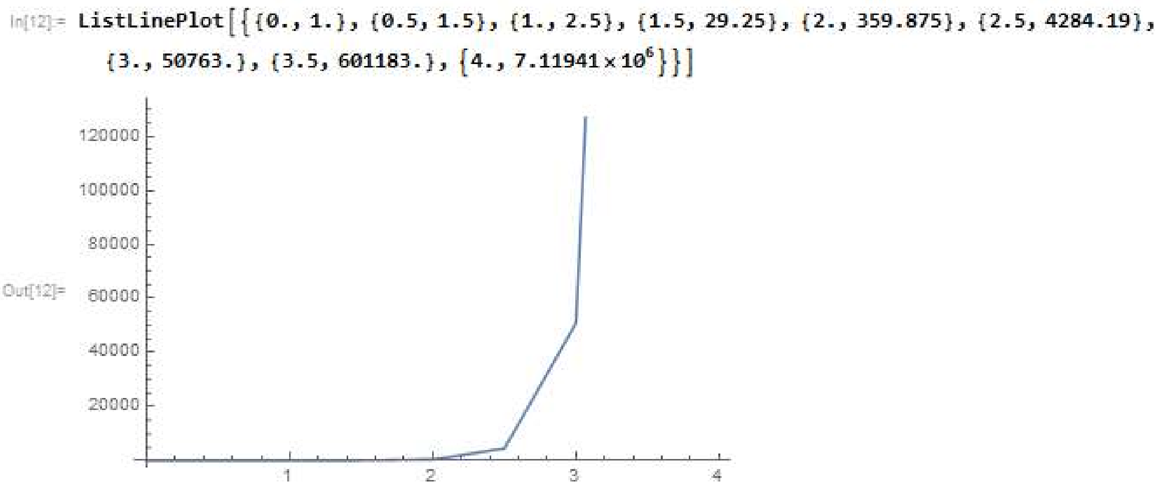
```
In[21]:= MyAdamsBashforth[F_, h_, t_, y_][{t0_, y0_}] := Module[{j, p, F0, F1, F2},

   m = Input["Put the value of m="]; (*We put the number of iterations 8*)
   a = Input["Put the value of a="]; (*We put a=0*)
   b = Input["Put the value of b="]; (*We put e b=4*)
        b - a
   h = ───── ;
          m
   t[0] = t0;
   y[0] = y0;
   For[j = 0, j ≤ 1, j++,
    t[j + 1] = t[j] + h;
    y[j + 1] = y[j] + h * F[t[j], y[j]];
   ];
   F0 = F[t[0], y[0]];
   F1 = F[t[1], y[1]];
   F2 = F[t[2], y[2]];
   For[j = 2, j ≤ m, j++,
    p = y[j] + h * (5 F0 - 16 F1 + 23 F2);
    t[j + 1] = t[j] + h;
    y[j + 1] = p;
    F0 = F1;
    F1 = F2;
    F2 = F[t[j + 1], y[j + 1]];
   ];
  ]
  Clear[F, y, t, h];
```



```
F[t_, y_] = Input["Set an differential equation "];
(*We put the differential equation: y+t*)
Print["Differential equation y'[t]=", F[t_, y_]];
Print["With a=", a]; Print["With b=", b];
Print["Number of iterations m=", m];
MyAdamsBashforth[F, h, t, y][{0., 1.}];
Print["The step size is: h=", h];
MyData3 = Table[{t[j], y[j]}, {j, a, m}]
Differential equation y'[t]=t_ + y_
With a=0
With b=4
Number of iterations m=8
The step size is: h=1/2
Out[29]= {{0., 1.}, {0.5, 1.5}, {1., 2.5}, {1.5, 29.25}, {2., 359.875},
 {2.5, 4284.19}, {3., 50763.}, {3.5, 601183.}, {4., 7.11941×10^6}}
```

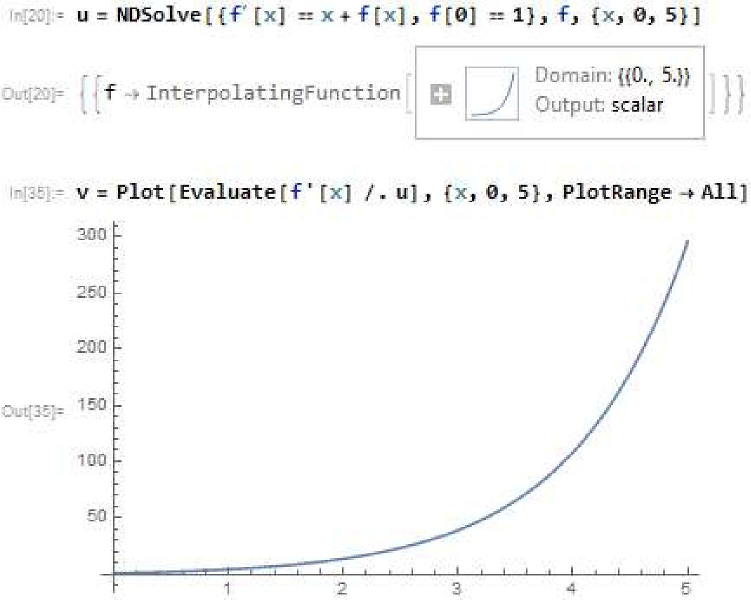


```
In[30]:= TableForm[MyData3, TableHeadings → {{}, {"t", "y"}}]
Out[30]//TableForm=
   t      y
   0.     1.
   0.5    1.5
   1.     2.5
   1.5    29.25
   2.     359.875
   2.5    4284.19
   3.     50763.
   3.5    601183.
   4.     7.11941×10^6
```



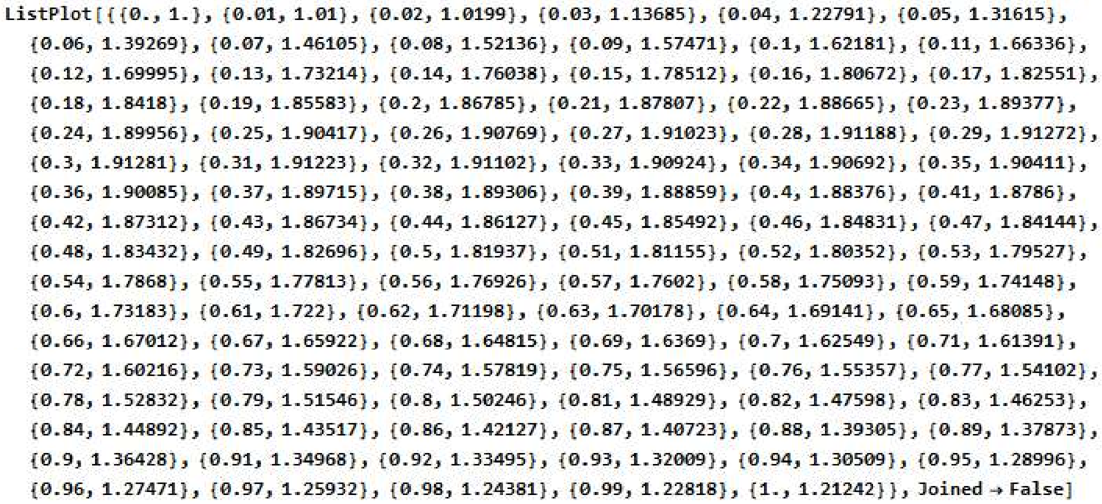
```
In[12]:= ListLinePlot[{{0., 1.}, {0.5, 1.5}, {1., 2.5}, {1.5, 29.25}, {2., 359.875}, {2.5, 4284.19},
   {3., 50763.}, {3.5, 601183.}, {4., 7.11941×10^6}}]
Out[12]=
```

The same equation is solved using NDSolve from the same package, an multi-step code for numerical approaches of differential equations which is very closed to exact solution [8,9]. The results are compared with the previous ones and show that the method is useful for our experiments.

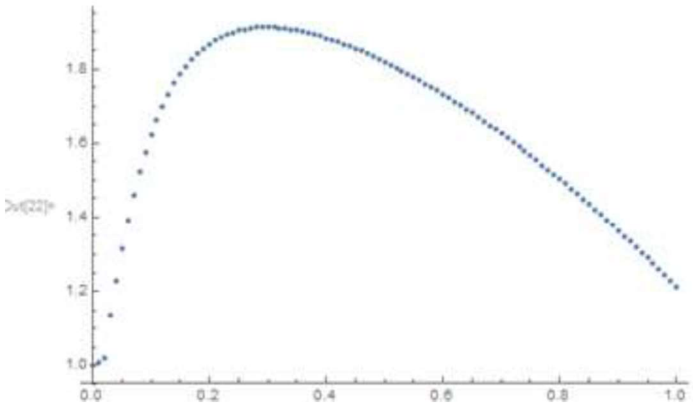


```
In[20]:= u = NDSolve[{f′[x] == x + f[x], f[0] == 1}, f, {x, 0, 5}]

Out[20]= {{f → InterpolatingFunction[ ⊞  Domain: {{0., 5.}}
                                          Output: scalar ]}}

In[35]:= v = Plot[Evaluate[f'[x] /. u], {x, 0, 5}, PlotRange → All]
```

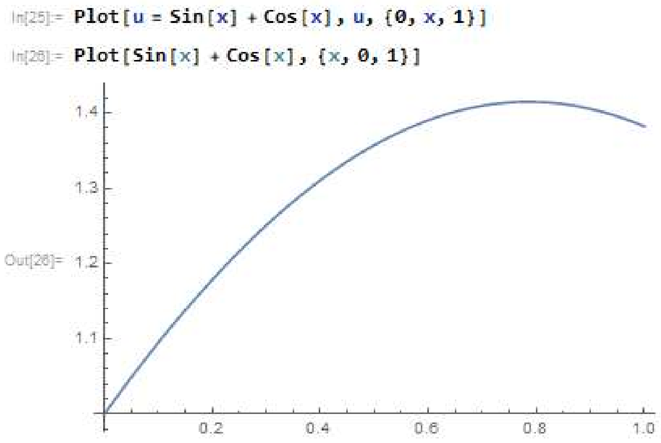
Example 3.2. We use Code [9] on Examle 3.1. to approach the solution of $y'(t) = -y(t) + 2\cos t$, with initial condition $y(0) = 1$, step size $h = 0.01$ on $[0, 1]$. The results are:



```
ListPlot[{{0., 1.}, {0.01, 1.01}, {0.02, 1.0199}, {0.03, 1.13685}, {0.04, 1.22791}, {0.05, 1.31615},
  {0.06, 1.39269}, {0.07, 1.46105}, {0.08, 1.52136}, {0.09, 1.57471}, {0.1, 1.62181}, {0.11, 1.66336},
  {0.12, 1.69995}, {0.13, 1.73214}, {0.14, 1.76038}, {0.15, 1.78512}, {0.16, 1.80672}, {0.17, 1.82551},
  {0.18, 1.8418}, {0.19, 1.85583}, {0.2, 1.86785}, {0.21, 1.87807}, {0.22, 1.88665}, {0.23, 1.89377},
  {0.24, 1.89956}, {0.25, 1.90417}, {0.26, 1.90769}, {0.27, 1.91023}, {0.28, 1.91188}, {0.29, 1.91272},
  {0.3, 1.91281}, {0.31, 1.91223}, {0.32, 1.91102}, {0.33, 1.90924}, {0.34, 1.90692}, {0.35, 1.90411},
  {0.36, 1.90085}, {0.37, 1.89715}, {0.38, 1.89306}, {0.39, 1.88859}, {0.4, 1.88376}, {0.41, 1.8786},
  {0.42, 1.87312}, {0.43, 1.86734}, {0.44, 1.86127}, {0.45, 1.85492}, {0.46, 1.84831}, {0.47, 1.84144},
  {0.48, 1.83432}, {0.49, 1.82696}, {0.5, 1.81937}, {0.51, 1.81155}, {0.52, 1.80352}, {0.53, 1.79527},
  {0.54, 1.7868}, {0.55, 1.77813}, {0.56, 1.76926}, {0.57, 1.7602}, {0.58, 1.75093}, {0.59, 1.74148},
  {0.6, 1.73183}, {0.61, 1.722}, {0.62, 1.71198}, {0.63, 1.70178}, {0.64, 1.69141}, {0.65, 1.68085},
  {0.66, 1.67012}, {0.67, 1.65922}, {0.68, 1.64815}, {0.69, 1.6369}, {0.7, 1.62549}, {0.71, 1.61391},
  {0.72, 1.60216}, {0.73, 1.59026}, {0.74, 1.57819}, {0.75, 1.56596}, {0.76, 1.55357}, {0.77, 1.54102},
  {0.78, 1.52832}, {0.79, 1.51546}, {0.8, 1.50246}, {0.81, 1.48929}, {0.82, 1.47598}, {0.83, 1.46253},
  {0.84, 1.44892}, {0.85, 1.43517}, {0.86, 1.42127}, {0.87, 1.40723}, {0.88, 1.39305}, {0.89, 1.37873},
  {0.9, 1.36428}, {0.91, 1.34968}, {0.92, 1.33495}, {0.93, 1.32009}, {0.94, 1.30509}, {0.95, 1.28996},
  {0.96, 1.27471}, {0.97, 1.25932}, {0.98, 1.24381}, {0.99, 1.22818}, {1., 1.21242}}, Joined → False]
```



The results are compared with exact solution of the current equation as:



```
In[25]:= Plot[u = Sin[x] + Cos[x], u, {0, x, 1}]
In[26]:= Plot[Sin[x] + Cos[x], {x, 0, 1}]
```

4. Conclusion

The particular mathematical method used is an Adams method, it is simple to understand and code, but it is efficient and robust enough for real work. Adams-Bashforth methods of order 4 is used to find numerical solutions of initial value problems. The effectiveness of this method for the treatment of stiff problems isshown on the bases of its attractive properties. Numerical experiments also are presented using different content of solving them, NDSolve or Analytical solve. The method, inside all the code have good stability properties, and the behavior on some difficult test problems, is good. In our implementation of this method we use Mathematica Package 11.0.

References

[1]. C.W. Gear, Numerical initial value problems in ordinary differential equations, *Prentice-Hall*, (1971).

[2]. J.D. Lambert, Numerical methods for ordinary differential equations, *John Wiley & Sons,* (1991).

[3]. D.F. Griffiths, D.J. Higham, Numerical methods for ordinary differential equations. Initial value problems, Springer *Undergraduate Mathematics Series*, (2010).

[4]. F. Ismail, Exploring Efficient Numerical Methods for Differential Equations*, University Putra Malaysia Press Serdang*, (2016).

[5]. L. Brugnano, K. Burrage, P. Burrage, Adams-Type Methods for the Numerical Solution of Stochastic Ordinary Differential Equations, BIT, *Springer Verlag*, (1998).

[6]. L. Brugnano, D. Trigiante, Solving Differential Problems by Multistep Initial and Boundary Value Methods, *Gordon and Breach Science Publishers*, Amsterdam, (1998).

[7]. R.L. Burden, J.D. Faires, Numerical Analysis, Ninth Edition, *United States Copyright Act*, (2010).

[8]. S. Wolfram, The Mathematica Book, Fifth Edition, *1988-2003 Wolfram Research*, (1993).

[9]. D. Eugene, Mathematica, Second Edition, *Schaum's Outlines*, (2009).