ANNALS OF "DUNAREA DE JOS" UNIVERSITY OF GALATI
MATHEMATICS, PHYSICS, THEORETICAL MECHANICS
FASCICLE II, YEAR XV (XLVI) 2023, No. 1
DOI: https://doi.org/10.35219/ann-ugal-math-phys-mec.2023

Numerical solutions of the differential equations

Nina N. Cazacu[1]

[1] *High School of Arts, United Principates Street,* 030167, *Bucharest, Romania*
[*]*Corresponding author:* cazanina@yahoo.com

Abstract

In this paper, we propose to study the numerical integration of Cauchy's problem (PC-in short) for a system of differential equations of the first order by the method of successive approximations and the method of the polygonal line, the best known in practice. For example, the methods are applied to some particular functions. The novelty of this paper consists, mainly, in the translation of existing approximation methods into C++ code, in order to visualize the results including the graphic image of both approximate and exact values trajectories, for comparison.

Keywords: algorithm, approximate solutions, Cauchy's problem

1. INTRODUCTION

Following the well-known theory of numerical integration for Cauchy's problem of the first order systems of differential equations, we are going to codify some of the best-known methods of approximating the solutions, and obtain as accurate values as possible.

Consider **PC** for a normal differential system [2][3]:

$$y' = f(x,y), y(x_0) = y_0, \tag{1}$$

where $f : \Re^{1+m} \to \Re^m$, $m \geq 1$, is a continuous function in a domain $D \in \Re^{1+m}$, which contains the given point $[x_0, y_0]$. Since D is an open set, contains a neighborhood $P := I \times V$, with

$$I := \{x \in \Re; | x - x_0 | < \alpha\}, V := \{y \in \Re^m; || y - y_0 || < \beta\},$$

where **PC**(1) is equivalent to the integral equation

$$y(x) = y_0 + \int_{x_0}^{x} f(t, y(t))dt, x \in I. \tag{2}$$

If we note with $(Ty)(x)$ 2nd term in (2), we arrive at a fixed point problem, $y = Ty$, to which we can apply the principle of Picard-Banach contractions. We get thus

Theorem. (**Picard**) Let $f \in C(P)$ satisfying Lipschitz's condition in y, i.e.,

$$|| f(x,y) - f(x,z) || \leq L || y - z ||, \forall [x,y], [x,z] \in P.$$

2. APPROXIMATION METHODS: MAS AND EULER'S POLYGONAL LINE

The **PC**(1) admits a unique solution, $y = y(x)$ ([1][2][3]) defined and continuous on the interval $(x_0 - k, x_0 + k)$, where $k := \min(\alpha, \beta / M)$ with $M := \sup\{|| f(x,y) ||; [x,y] \in P\}$, to which the *method of successive approximations (**MAS**-in short)* converges:

$$y_0 := y(x_0), y_n := Ty_{n-1}, \forall n \geq 1.$$

Before moving on, some remarks on the assumptions of the theorem are necessary. Thus, to fulfill **Lipschitz's condition**, it is enough to assume that $f'_y \in C(P)$, because, P being a relatively

compact set, exists $L := \sup\{\| f'_y \|; [x,y] \in P\}$, and, with Lagrange formula, we have:

$$\| f(x,y_1) - f(x,y_2) \| \le \sup\{\| f'_y(x,y)(y_1 - y_2) \|; [x,y] \in P\} \le L \| y_1 - y_2 \|.$$

Changing x with y, that is, looking for **form solutions** like $x = x(y)$, we can ask for a **Lipschitz condition** in ***x*** with the same effect. Moreover, we can establish the existence (sometimes uniqueness) without ***f*** fulfilling a **Lipschitz condition** in one of the arguments. Still, it is necessary to ask ***f*** for more than simple continuity (see **Peano's theorem**).

Finally, the interval of existence and uniqueness, $(x_0 - k, x_0 + k)$, is not a maximal interval, **MAS** being able to converge on larger intervals as it is revealed from

Example 1. PC: $y' = 1 + y^2, y(0) = 0$ has the solution $y = tgx$ and it can be shown that **MAS** converges for $|x| < \frac{\pi}{2}$. However, from the theorem we have: $|f(x,y)| := |1 + y^2| \le 1 + \beta^2 =: M$ so that $k = \frac{\beta}{1+\beta^2}$, which has a maximum, $k_{max} = \frac{1}{2}$ for $\beta = 1$. Thus, the theorem guarantees the convergence of **MAS** only for $|x| < \frac{1}{2}$. In practice, **MAS** is used only in extreme cases because its numerical realization requires the calculation of integrals, and **its convergence is generally slow**.

Example 2. PC: $y' + y = 2e^x, y(0) = 1$ has also solution. Integrated as a linear **PC** of the order I we get the solution $y = e^x$. We apply **MAS** with the initial approximation $Y_0(x) = 1$ and show that $\lim Y_n(x) = y(x)$, in which $Y_n(x)$ is the explicit solution obtained at step $n \ge 1$.

We are lead to:

$$Y_n = 1 + \int_0^x (2e^t - Y_{n-1}(t))dt = 2e^x - 1 - \int_0^x Y_{n-1}(t)dt = 2e^x - \sum_{i=0}^{n} \frac{x}{i!}, \text{ n=2p-1.}$$

It results: $limY_n = e^x = y(x)$, uniform on $\mathcal{R}$.

To establish **the existence and uniqueness of the solution on an interval** $[a,b] \subset \Re$, which contains x_0, the **MAS** theory is completed by ***Euler**'s polygonal line method*, which constructs a function $Y(x)$ uniformly approximating the exact solution([2][3]).

It is considered an equidistant division of the interval $J(=[x_0,b])$ on right of x_0, i.e., $x_i := x_0 + ih$, $0 \le i \le n$, with $h > 0$ small enough; (for the interval H to the left of x_0 it is recommended the variable change $x = -t$ and the similar procedure). Note with $y := y(x_i)$, the values of the exact solution $y(x)$ in the nodes x_i, and with Y_i the values of an approximate solution calculated in these nodes. Approximating the derivative with the incremental ratio we have

$$y'(x) \approx \frac{y(x+h) - y(x)}{h} = f(x, y(x))$$

and it is led to define the iteration: $Y_0 := y_0, Y_{i+1} := Y_i + hf(x_i, Y_i), 0 \le i \le n-1,$ (3)

which gives us the values of the approximate solution $Y(x)$ in the nodes x_i.

In the other points $x \ne x_i$ it is defined the approximate solution $Y(x)$ as a linear function that joins the points $[x_i, Y_i]$ şi $[x_{i+1}, Y_{i+1}]$, meaning:

$$Y(x) = Y_i + \frac{x - x_i}{x_{i+1} - x_i}(Y_{i+1} - Y_i) \equiv Y_i + \frac{Y_{i+1} - Y_i}{h}(x - x_i), x \in [x_i, x_{i+1}]. \quad (4)$$

Let $\varepsilon > 0$ chosen small enough to have

$$B_\varepsilon := \{[x,y] \in \Re^{1+m}; \| y - y(x) \| < \varepsilon\} \subseteq D.$$

Assuming there is a solution $y(x)$ if is proven that $\| y(x)-Y(x) \|< \varepsilon$, $\forall x \in J$, it results the the existence of $Y(x)$, which was shown in the previous construction, which implies the existence of the exact solution on J.

Let $L := \sup\{\| f'_y(x,y) \|; [x,y] \in B_\varepsilon$ and $\delta > 0$ so that:

$$\frac{\delta}{L}(e^{(b-x_0)L} - 1) < \varepsilon. \tag{5}$$

Because the function $\varphi(x) := f(x, y(x))$ is uniformly continuous on J, there exists $\eta(\delta) > 0$ such that $x', x'' \in J$, and $| x' - x'' | < \eta$ determine $\| \varphi(x') - \varphi(x'') \| < \delta$. The next sentence is also known in the specialty literature:

"If $h < \eta$ ***, then*** $[x_i, Y_i] \in B_\varepsilon$ **,** $0 \le i \le n$ ***and*** $\| y(x) - Y(x) \| < \varepsilon$ **,** $\forall x \in J$ ***"***

For the proof, the procedure is the induction on $i \ge 0$. We start with $i = 0$. It results $[x_0, Y_0] \in B_\varepsilon$ because $Y_0 := y_0 = y(x_0)$. Assuming that $[x_0, Y_0], \ldots, [x_i, Y_i] \in B_\varepsilon$ and $x_{n+1} \le b$, for compare y_{i+1} with Y_{i+1} it can be calculated the difference

$$y_{i+1} - y_i = \int_{x_i}^{x_{i+1}} f(t, y(t))dt = hf(x_i, y_i) + r_i,$$

with:

$$\| r_i \| \le \int_{x_i}^{x_{i+1}} \| f(t, y(t)) - f(x_i, y_i) \| dt \le \delta h.$$

So: $Y_{i+1} - y_{i+1} = Y_i - y_i + h[f(x_i, Y_i) - f(x_i, y_i)] - r_i$ and, with the Lagrange formula, we are led to:

$$\| Y_{i+1} - y_{i+1} \| \le \| Y_i - y_i \| + h \| f'_y(x_i, \xi_i) \| \cdot \| Y_i, y_i \| + \| r_i \| \le (1 + hL) \| Y_i - y_i \| + \delta h.$$

By repeated applications: $\| Y_{n+1} - y_{n+1} \| \le (1 + hL)^{n+1} \| Y_0 - y_0 \| + \delta h \sum_{i=0}^{n} (1 + hL)^i \le \delta h \frac{(1 + hL)^{n+1} - 1}{hL}$.

Taking into account that $e^{hL} \ge 1 + hL$ and $x_{n+1} = x_0 + (n+1)h \le b$, so $(n+1) \le \frac{b - x_0}{h}$, it results:

$$(1 + hL)^{n+1} \le e^{(b-x_0)L}.$$

Consequently:

$$\| Y_{n+1} - y_{n+1} \| \le \frac{\delta}{L}(e^{(b-x_0)L} - 1) < \varepsilon,$$

which involves $[x_{n+1}, Y_{n+1}] \in B_\varepsilon$. It was thus shown that for any $x_n \le b$ we have $\| Y(x) - y(x) \| < \varepsilon$, $\forall x \in J$, knowing that Y and y are uniformly continuous functions on J.

3. CODIFICATION OF EULER'S METHOD

We will present next, a codification for the application of Euler's method, previously explained. Applying **Euler's method** on the interval [0,1] , or or any interval included in it, we can approximate the solution of the PCs $y' + y = 2e^x, y(0) = 1$. Being integrable by quadratics, **PC** has the exact solution for the equation: $y(x) := e^x - x^2 - 2x - 2$.

For the application of Euler's method and codification of it on the interval J=[0;0,5], for example, it is noted : $f(x, y) := x^2 + y$, $f'_y = 1$.

The calculation code for the concrete example is made in C++ language, having two

subroutines. The two subordinates are: **f(a,b)**, which calculates the expression: $\mathbf{a^2+b}$, and **y(x)**, which calculates the values of the expression $\mathbf{e^x-x^2-2x-2}$.

The precision is noted: $\mathbf{e}=10^{-4}$, and the initial conditions of the variables are: X[0]=0,Y[0]=1. The framing interval is [0;0,5] with **b**=0,05 and **n** = 48. The maximum value of the derivative in relation to y is calculated: **L**=1. For $\mathbf{h}=10^{-3}$, according to the theory, $\delta=(e*L)/e^{(b-X[0])*L}-1)-0,0001$, so: $|\delta|< e=0,0001$.

The source code based on the previous theoretical considerations:

```
include<iostream.h>
#include<math.h>
#include<iomanip.h>
#include<graphics.h>
#include<dos.h>
#include<conio.h>
long double h,X[100],Y[100];int n=48, i;
long double f(float a,float b) {return a*a+b;}
long double y(float x) {return(exp(x)-pow(x,2)-2*x-2);}
void main()
{clrscr();
 float aux=1,b=0.05; float L=1;X[0]=0;Y[0]=1;
 long double e=pow(10,-4),z,h,delta=(e*L)/(exp((b-X[0])*L)-1)-0.001;
 float x=0;z=aux+delta; float pas=pow(10,-3);
 do{aux=z;x+=pas;z=f(x,y(x));}
 while(abs(z-aux)>=delta);
 h=pas;
 cout<<"n="<<n<<' '<<"h"<<' '<<h<<endl;
 for(i=1;i<=n;i++)X[i]=X[i-1]+h;
 for(i=1;i<=n;i++)Y[i]=Y[i-1]+h*f(X[i-1],Y[i-1]);
 cout<<"  X[i]   Y[i]   y(X[i])    Y[i+1]"<<endl;
 int j=1;for(int i=1;i<=n;i++){j++;if(j%24==0) delay(5000);
 cout<<setw(5)<<X[i]<<setw(10)<<Y[i]<<setw(10)<<(-y(X[i]))<<setw(10)<<Y[i+1]<<endl; }}
```

In the penultimate iteration, the results will coincide to the first four decimal places after the comma, so up to the 5th, exclusively, as in figure 1. The difference δ in value between Y[i] and y(X[i]), so between columns 2 and 3, is: 1.049182-1.049133=0.000049, approximately 0.00005 which verifies the relationship established by calculation for agreed numerical data 0.0001: 0.00005<0.0001.

Assigning numerical values for ε and δ it could determine h (or any other combination), so that the approximate solution has a prescribed precision.

A better precision can be obtained if we use at each step more values of the function in the so-called *prediction-correction method*. The method consists of a prediction stage when we use Euler's method:

$$x_{i+1} := x_i + h, Z_{i+1} := Y_i + hf(x_i, Y_i),$$

to obtain a first approximation for the value of the solution y in the $x_i + h$ point, which is then corrected in the following sense:

$$Y_{i+1} := Y_i + h\frac{f(x_i, Y_i) + f(x_i, Z_i)}{2}.$$

The source code will be completed with the following sequence:

```
for(i=1;i<=n-1;i++){
 X[i+1]=X[i]+h;Z[i+1]=Y[i]+h*f(X[i],Y[i]);
 Y[i+1]=Y[i]+h*(f(X[i],Y[i])+f(X[i],Z[i]))/2;}
 cout<<"  X[i]   Y[i]   f(X[i],Y[i])    f(X[i],Z[i])    Y[i+1]    Z[i+1]"<<endl;
 j=1;for(i=1;i<=n;i++){j++;if(j%24==0) delay(5000);
 cout<<setw(5)<<X[i]<<setw(10)<<Y[i]<<setw(10)<<f(X[i],Y[i])<<
 setw(10)<<f(X[i],Z[i])<<setw(10)<<Y[i+1]<<setw(10)<<Y[i+1]<<endl; }
```

4. RESULTS AND DISCUSSION

The vector Y[i] is displayed, with 48 components, approximate values in the 2nd column, and also 48 values, but "exact", calculated in the vector y(X[i]).(see Table 1)

Table 1. The listed values of the approximate and exact solutions, using the *polygonal line* method (written in two columns)

n=48 h= 0.001

X[i]	Y[i]	y(X[i])	Y[i+1]
0.001	1.001	1.001	1.002001
0.002	1.002001	1.002002	1.003003
0.003	1.003003	1.003004	1.004006
0.004	1.004006	1.004008	1.00501
0.005	1.00501	1.005012	1.006015
0.006	1.006015	1.006018	1.007021
0.007	1.007021	1.007024	1.008028
0.008	1.008028	1.008032	1.009036
0.009	1.009036	1.00904	1.010045
0.01	1.010045	1.01005	1.011056
0.011	1.011056	1.01106	1.012067
0.012	1.012067	1.012072	1.013079
0.013	1.013079	1.013084	1.014092
0.014	1.014092	1.014098	1.015106
0.015	1.015106	1.015112	1.016122
0.016	1.016122	1.016127	1.017138
0.017	1.017138	1.017144	1.018156
0.018	1.018156	1.018161	1.019174
0.019	1.019174	1.019179	1.020194
0.02	1.020194	1.020199	1.021214
0.021	1.021214	1.021219	1.022236
0.022	1.022236	1.02224	1.023259
0.023	1.023259	1.023262	1.024282
0.024	1.024282	1.024286	1.025307
0.025	1.025307	1.02531	1.026333
0.026	1.026333	1.026335	1.02736
0.027	1.02736	1.027361	1.028388
0.028	1.028388	1.028388	1.029417
0.029	1.029417	1.029416	1.030448
0.03	1.030448	1.030445	1.031479
0.031	1.031479	1.031475	1.032511
0.032	1.032511	1.032506	1.033545
0.033	1.033545	1.033538	1.03458
0.034	1.03458	1.034571	1.035615
0.035	1.035615	1.035605	1.036652
0.036	1.036652	1.03664	1.03769
0.037	1.03769	1.037676	1.038729
0.038	1.038729	1.038713	1.039769
0.039	1.039769	1.039751	1.040811
0.04	1.040811	1.040789	1.041853
0.041	1.041853	1.041829	1.042897
0.042	1.042897	1.04287	1.043941
0.043	1.043941	1.043911	1.044987
0.044	1.044987	1.044954	1.046034
0.045	1.046034	1.045997	1.047082
0.046	1.047082	1.047042	1.048131
0.048	1.049182	1.049133	0

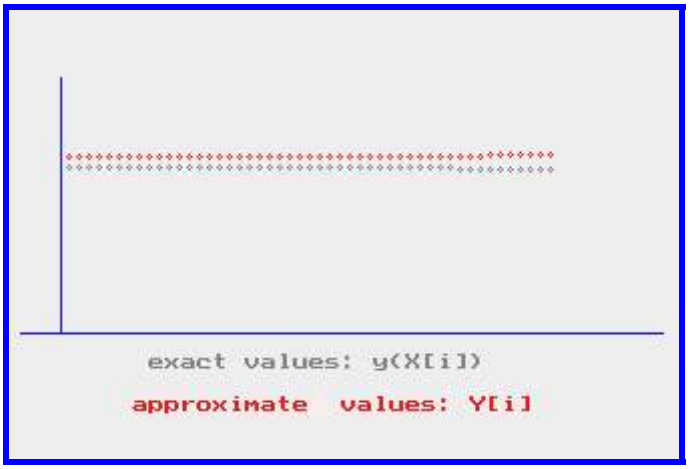

Fig. 1 Graphical representation of the exact and approximate PCs solutions

After applying the *prediction-correction* method, the accuracy increases visibly. In the 3rd and 4th columns are listed the 48 values of the vector f(X[i], Y[i]) and respectively of the vector f(X[i], Z[i]), the identical values starting with the 3rd component. Also, due to this addition, the components of the vector Y[i+1] and respectively their approximations, components of the vector Z[i+1], are identical after the first iteration. So, applying this method to the previous example, we obtain acceptable results, meaning that, in this case, the values in columns 3 and 4 coincide up to the 6th decimal, as do those in the columns 5 and 6 (Table 2).

Table 2. The listed values of the approximate and exact solutions, using the *prediction-correction* method (written in two columns)

X[i]	Y[i]	f(X[i],Y[i])	f(X[i],Z[i])	Y[i+1]	Z[i+1]
0.001	1.001	1.001001	1e-06	1.001501	1.001501
0.002	1.001501	1.001504	1.002005	1.002502	1.002502
0.003	1.002502	1.002511	1.002511	1.003505	1.003505
0.004	1.003505	1.003521	1.003521	1.004508	1.004508
0.005	1.004508	1.004533	1.004533	1.005513	1.005513
0.006	1.005513	1.005549	1.005549	1.006518	1.006518
0.007	1.006518	1.006567	1.006567	1.007525	1.007525
0.008	1.007525	1.007589	1.007589	1.008533	1.008533
0.009	1.008533	1.008614	1.008614	1.009541	1.009541
0.01	1.009541	1.009641	1.009641	1.010551	1.010551
0.011	1.010551	1.010672	1.010672	1.011561	1.011561
0.012	1.011561	1.011705	1.011705	1.012573	1.012573
0.013	1.012573	1.012742	1.012742	1.013586	1.013586
0.014	1.013586	1.013782	1.013782	1.0146	1.0146
0.015	1.0146	1.014825	1.014825	1.015615	1.015615
0.016	1.015615	1.015871	1.015871	1.01663	1.01663
0.017	1.01663	1.016919	1.016919	1.017647	1.017647
0.018	1.017647	1.017971	1.017971	1.018665	1.018665
0.019	1.018665	1.019026	1.019026	1.019684	1.019684
0.02	1.019684	1.020084	1.020084	1.020704	1.020704
0.021	1.020704	1.021145	1.021145	1.021726	1.021726
0.022	1.021726	1.02221	1.02221	1.022748	1.022748
0.023	1.022748	1.023277	1.023277	1.023771	1.023771
0.024	1.023771	1.024347	1.024347	1.024795	1.024795
0.025	1.024795	1.02542	1.02542	1.025821	1.025821
0.026	1.025821	1.026497	1.026497	1.026847	1.026847
0.027	1.026847	1.027576	1.027576	1.027875	1.027875
0.028	1.027875	1.028659	1.028659	1.028904	1.028904
0.029	1.028904	1.029744	1.029744	1.029933	1.029933
0.03	1.029933	1.030833	1.030833	1.030964	1.030964
0.031	1.030964	1.031925	1.031925	1.031996	1.031996
0.032	1.031996	1.03302	1.03302	1.033029	1.033029
0.033	1.033029	1.034118	1.034118	1.034063	1.034063
0.034	1.034063	1.035219	1.035219	1.035098	1.035098
0.035	1.035098	1.036323	1.036323	1.036135	1.036135
0.036	1.036135	1.037431	1.037431	1.037172	1.037172
0.037	1.037172	1.038541	1.038541	1.038211	1.038211
0.038	1.038211	1.039655	1.039655	1.03925	1.03925
0.039	1.03925	1.040771	1.040771	1.040291	1.040291
0.04	1.040291	1.041891	1.041891	1.041333	1.041333
0.041	1.041333	1.043014	1.043014	1.042376	1.042376
0.042	1.042376	1.04414	1.04414	1.04342	1.04342
0.043	1.04342	1.045269	1.045269	1.044465	1.044465
0.044	1.044465	1.046401	1.046401	1.045512	1.045512
0.045	1.045512	1.047537	1.047537	1.046559	1.046559
0.046	1.046559	1.048675	1.048675	1.047608	1.047608

5. CONCLUSIONS

Departing from the existing theory, we have applied *MAS* and *EULER's* methods in order to numerical solving a system of differential equations, the last method being perfected through the *prediction-correction* process. We have completed the presentation with the C++ code that allowed listing the sets of exact and approximate values of the solutions, as well as graphic visualization. The theory codification is the particular novelty of the paper. The article refers only to a narrow section of the methods for numerical solutions of the differential systems, namely, the successive approximations and the polygonal line method which are the most used for the first-order equations.

References

1. Arnold V.I., Ordinary differential equations, Scientific and Enciclopedic Publishing House, Bucharest (1978).
2. Craiu M., Rosculet N.M., Applicable differential equations, Bucharest, E.D.P. (1971).
3. Mirica St., Differential and integral equations, Bucharest University Publishing House (2000).
4. Mirica St., Differential equations and equations with partial derivatives, Bucharest University Publishing House (1999).