04 June 2019

Warsaw University of Technology – EiTI faculty

Solving ordinary differential equations

No. 38

Wiktor Bara

Course name: Numerical Methods

Advisor: Andrzej Miękina

Table of contents

Introduction

Solving ordinary differential equations can be conducted in different ways. The solution can be obtained using both types of methods. Implicit or explicit. The difference between those algorithms is simple but significant. The explicit method is much easier to implement because the every next value of result uses only the values that already computed. While in the implicit method every next element does not refers only to computed values but also refers to itself. Below is presented investigation that concerns both methods and shows the pros and cons on both sides.

Description of Numerical Algorithm & Methodology

The differential equation that is to be solved using algorithms below:

$$4y'' + 4y + 5y = 0 \quad for\ t \in [0{,}10], \qquad y(0) = 4\ and\ y'(0) = -2$$

The Algorithms below require matrix **A** that can be obtain in the way:

$$u_1 = y$$
$$u_2 = y'$$

after derivation

$$u'_1 = y' = u_2$$

$$u'_2 = y'' = -\frac{5}{4}y - y' = -\frac{5}{4}u_1 - u_2$$

the values included in matrix are the coefficients in the system of equations and are equal:

$$\mathbf{A} = \begin{bmatrix} 0 & 1 \\ -\frac{5}{4} & -1 \end{bmatrix}$$

Buthcer (Gauss-Legendre) implicit method

The method of order 6 that use the table below to obtain the approximated function from differential equation (K=3, p=6). That method use the matrix **A** that was computed above.

$$\begin{array}{c|ccc}
\frac{1}{2}-\frac{\sqrt{15}}{10} & \frac{5}{36} & \frac{2}{9}-\frac{\sqrt{15}}{15} & \frac{5}{36}-\frac{\sqrt{15}}{30} \\
\frac{1}{2} & \frac{5}{36}+\frac{\sqrt{15}}{24} & \frac{2}{9} & \frac{5}{36}-\frac{\sqrt{15}}{24} \\
\frac{1}{2}+\frac{\sqrt{15}}{10} & \frac{5}{36}+\frac{\sqrt{15}}{30} & \frac{2}{9}+\frac{\sqrt{15}}{15} & \frac{5}{36} \\
\hline
 & \frac{5}{18} & \frac{4}{9} & \frac{5}{18}
\end{array}$$

The description of algorithm of implicit method:

$$y_n = y_{n-1} + h(\frac{5}{18}f_1 + \frac{4}{9}f_2 + \frac{5}{18}f_3)$$

with:

$$f_1 = f(t_{n-1} + \left(\frac{1}{2} - \frac{\sqrt{15}}{10}\right)h, y_{n-1} + h(\frac{5}{36}f_1 + \left(\frac{2}{9} - \frac{\sqrt{15}}{15}\right)f_2 + (\frac{5}{36} - \frac{\sqrt{15}}{30})f_3)$$

$$f_2 = f(t_{n-1} + \frac{1}{2}h, y_{n-1} + h((\frac{5}{36} + \frac{\sqrt{15}}{24})f_1 + \frac{2}{9}f_2 + (\frac{5}{36} - \frac{\sqrt{15}}{24})f_3)$$

$$f_3 = f(t_{n-1} + \left(\frac{1}{2} + \frac{\sqrt{15}}{10}\right)h, y_{n-1} + h(\left(\frac{5}{36} + \frac{\sqrt{15}}{30}\right)f_1 + (\frac{2}{9} + \frac{\sqrt{15}}{15})f_2 + \frac{5}{36}f_3)$$

after application to ODE system:

$$\boldsymbol{u}'(t) = \boldsymbol{A} \cdot \boldsymbol{u}(t) + \boldsymbol{b} \cdot e(t) \qquad for\ t \in [0, T]$$

The sixth order Gauss-Legendre method yields:

$$\boldsymbol{u_n} = \boldsymbol{u_{n-1}} + h(h\left(\frac{5}{18}\boldsymbol{f_1} + \frac{4}{9}\boldsymbol{f_2} + \frac{5}{18}\boldsymbol{f_3}\right) \qquad for\ n = 1,2,\ldots\ and\ \boldsymbol{u_0} = \boldsymbol{u}(0)$$

with:

$$\boldsymbol{f}_1 = \boldsymbol{A} \cdot \left[\boldsymbol{u_{n-1}} + h(\frac{5}{36}\boldsymbol{f_1} + \left(\frac{2}{9} - \frac{\sqrt{15}}{15}\right)\boldsymbol{f_2} + (\frac{5}{36} - \frac{\sqrt{15}}{30})\boldsymbol{f_3})\right] + \boldsymbol{b} \cdot e(t_{n-1} + \left(\frac{1}{2} - \frac{\sqrt{15}}{10}\right)h)$$

$$\boldsymbol{f}_2 = \boldsymbol{A} \cdot \left[\boldsymbol{u_{n-1}} + h((\frac{5}{36} + \frac{\sqrt{15}}{24})\boldsymbol{f_1} + \frac{2}{9}\boldsymbol{f_2} + (\frac{5}{36} - \frac{\sqrt{15}}{24})\boldsymbol{f_3})\right] + \boldsymbol{b} \cdot e(t_{n-1} + \frac{1}{2}h,)$$

$$\boldsymbol{f}_3 = \boldsymbol{A}\cdot\left[\boldsymbol{u}_{n-1} + h(\left(\frac{5}{36}+\frac{\sqrt{15}}{30}\right)\boldsymbol{f}_1 + (\frac{2}{9}+\frac{\sqrt{15}}{15})\boldsymbol{f}_2 + \frac{5}{36}\boldsymbol{f}_3)\right] + \boldsymbol{b}\cdot e(t_{n-1} + \left(\frac{1}{2}+\frac{\sqrt{15}}{10}\right)h)$$

or:

$$\begin{bmatrix} \mathbf{I}-\frac{5}{36}\mathrm{h}\mathbf{A} & (\frac{2}{9}-\frac{\sqrt{15}}{15})\mathrm{h}\mathbf{A} & (\frac{5}{36}-\frac{\sqrt{15}}{30})\mathrm{h}\mathbf{A} \\ (\frac{5}{36}+\frac{\sqrt{15}}{24})\mathrm{h}\mathbf{A} & \mathbf{I}-\frac{2}{9}\mathrm{h}\mathbf{A} & (\frac{5}{36}-\frac{\sqrt{15}}{24})\mathrm{h}\mathbf{A} \\ (\frac{5}{36}+\frac{\sqrt{15}}{30})\mathrm{h}\mathbf{A} & (\frac{2}{9}+\frac{\sqrt{15}}{15})\mathrm{h}\mathbf{A} & \mathbf{I}-\frac{5}{36}\mathrm{h}\mathbf{A} \end{bmatrix}\cdot\begin{bmatrix}\boldsymbol{f}_1\\ \boldsymbol{f}_2\\ \boldsymbol{f}_3\end{bmatrix} = \begin{bmatrix} \boldsymbol{A}\cdot\boldsymbol{u}_{n-1} + \boldsymbol{b}\cdot e(t_{n-1} + \left(\frac{1}{2}-\frac{\sqrt{15}}{10}\right)h) \\ \boldsymbol{A}\cdot\boldsymbol{u}_{n-1} + \boldsymbol{b}\cdot e(t_{n-1} + \frac{1}{2}h) \\ \boldsymbol{A}\cdot\boldsymbol{u}_{n-1} + \boldsymbol{b}\cdot e(t_{n-1} + \left(\frac{1}{2}+\frac{\sqrt{15}}{10}\right)h) \end{bmatrix}$$

Euler explicit method

That method also uses the matrix **A** but the fact that the method is explicit the computation is much easier:

$$y_n = y_{n-1} + h_{n-1} f(t_{n-1}, y_{n-1})$$

after application to ODE system:

$$\boldsymbol{u}_n = \boldsymbol{u}_{n-1} + h\cdot\boldsymbol{A}\cdot\boldsymbol{u}_{n-1}$$

The built in function **ode113** is used as a method to obtain reference values of computed differential equations. That one of the input arguments had to be modified to provide ability to compute differential equations of order 2. The function multiply the input values of y byu matrix **A.**

The procedure to obtain the root-mean-squared error and the maximum error.

That procedure compares the functions obtained using the Gauss-Legendre method as well as Euler explicit method in comparison to built in method **ode113**. The error is computed according to step h.

$$\delta_2(h) = \frac{\left\|\hat{\dot{\mathbf{y}}}(t;h) - \dot{\mathbf{y}}(t,h)\right\|_2}{\left\|\dot{\mathbf{y}}(t,h)\right\|_2} \quad \text{(the root-mean-square error)}$$

$$\delta_\infty(h) = \frac{\left\|\hat{\dot{\mathbf{y}}}(t;h) - \dot{\mathbf{y}}(t,h)\right\|_\infty}{\left\|\dot{\mathbf{y}}(t,h)\right\|_\infty} \quad \text{(the maximum error)}$$

Results

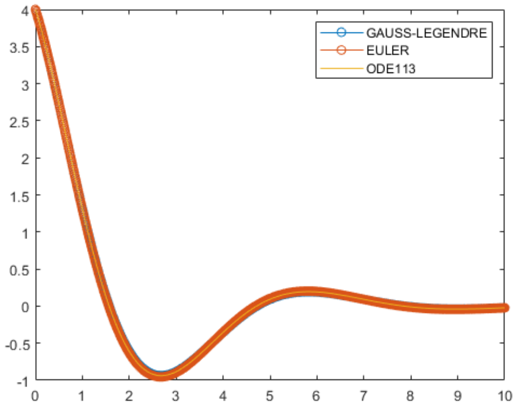

Figure 1. Solution h = 0.01

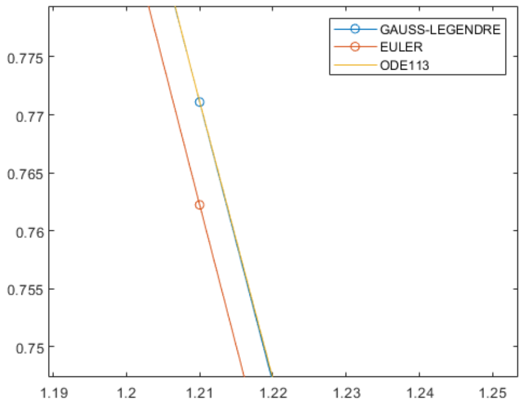

Figure 2. Solution h = 0.01 zoom

The solution of differential equation that concerns h = 0.01 is presented on the Fig. 1 and Fig. 2 As we can see in the default scale the difference between obtained outcome is unnoticeable and to see the disparity we have to zoom the plot several times. As we can observe on the zoomed graph the difference between Gauss-Legendre method and the built in method **ode113** is almost none while the Euler method shows slightly deviation.

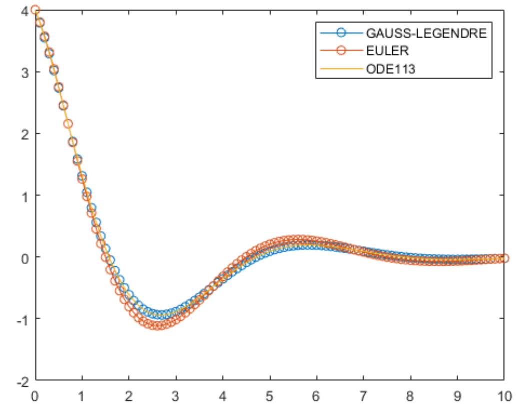

Figure 3. Solution h = 0.1

In the case when investigation concerns h=0.1 (Fig. 3) the both Euler and Gauss Legendre method are quite similar, but in the Euler one the error occurred is significant while the Gauss-Legendre one is almost the same as the built in function.

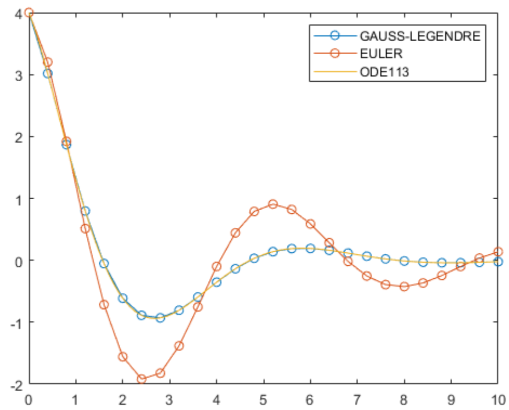

Figure 4. Solution h = 0.4

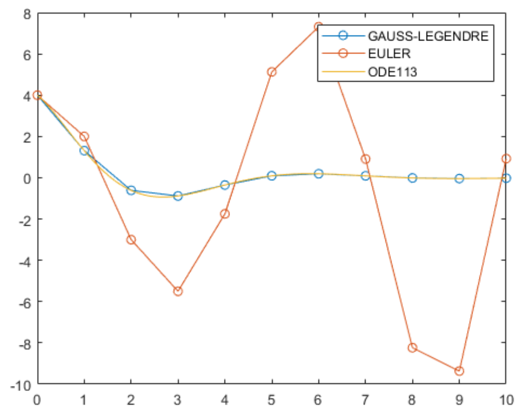

Figure 5. Solution h = 1

After further investigation and taking into consideration h = 0.4 in Fig. 4 and h = 1 in Fig. 5 the errors in case Euler method are going bigger end bigger while the Gauss-Legendre method is still almost perfect.

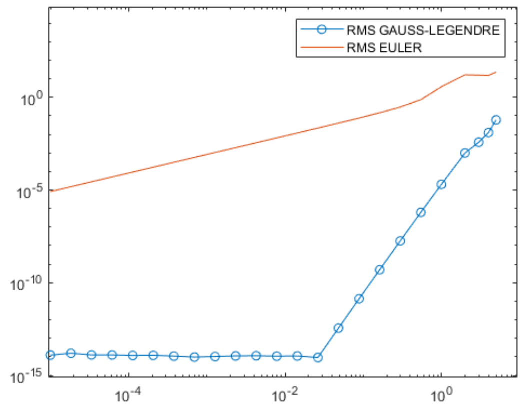

Figure 7. Root-mean-square error

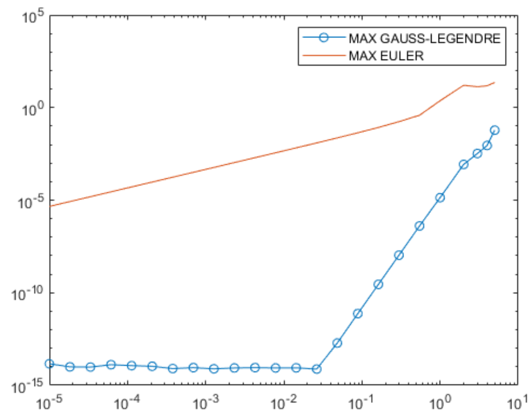

Figure 6. Maximum error

The investigation of thr root-mean-square error (RMS) and the maximum error (MAX) is presented on Fig. 6 and Fig. 7 The differences are significant and is obvious that the Gauss-Legendre method is much more accurate than Euler method in consideration the h of the same value. The errors in case Gauss-Legendre method are in case of very small h are stable not because of algorithm but because of the fact that the method **RelTol** and **AbsTol** are limited to 10^{-13} or 10^{-14} and the consideration of lower values is unnecessary due to the fact that there is limitation of computing power of computer (eps). The significant change in the graph in point ($\sim 10^{-1,5}$) when the errors grown and the next breakpoint is in the area of 1 and bigger values when the error is growing almost exponentially. The errors of Euler method are much bigger than the errors of the Gauss-Legendre algorithm. Beside the fact that the there is difference in several orders of magnitude. The breakpoint that changes accuracy significantly is close to h = 0.4 and we can see on the Fig Z that the changes of the algorithm are substantial. After exceeding the point h=1 the Euler method is no longer acceptable as we can see on Fig. 4.

Conclusions

Basing on the results we can assume that the algorithms above have their own advantages and disadvantages.

The Gauss-Legendre is much more accurate than the Euler method, but that fact is achieved at the expense of the much harder implementation and a little bit longer computing on every step.

The Euler method is significantly less accurate and in some cases is almost useless (h> 0.4), but the implementation is much less problematic and the computing time of one step is shorter.

After comparison we can however assume that beside the fact of a little bit longer computing of one step the Gauss-Legendre method is much quicker in solving equation because the satisfying outcome can be obtained using much bigger value of step (h) and as the time of computing is linearly dependent on the value of h the time can be shorten significantly.

List of References

[1] https://www.mathworks.com,

[2] Lecture notes ENUME 2019, Roman Morawski

SOURCE CODE

```
clear
close all

A = [0, 1;-1.25 , -1];
h = 0.01;
y0 = [4; -2];
ye(:,1)  =  [4 -2];
u(:,1) = [4 -2];

I = eye(2);
Bu = [I-(5/36)*h*A, ((sqrt(15)/15)-
(2/9))*h*A, ((sqrt(15)/30)-5/36)*h*A;
      -(5/36 + sqrt(15)/24)*h*A, I-(2/9)*h*A,
((sqrt(15)/24)-5/36)*h*A;
      -(5/36 + sqrt(15)/30)*h*A, -(2/9 +
sqrt(15)/15)*h*A, I-(5/36)*h*A];


opts = odeset('Reltol',10^(-
13),'AbsTol',10^(-14),'Stats' ,'on');
t = 0:h:10;
[ode_t, ode_y] = ode113(@fun,[0
10],y0',opts);




for i=2:length(t)
    F(:,i) = Bu\[A*u(:,i-1);A*u(:,i-1)
;A*u(:,i-1)];
    u(:,i) = u(:,i-1) + h*((5/18)*F(1:2,i) +
(4/9)*F(3:4,i) +(5/18)*F(5:6,i));

    ye(:,i) = ye(:,i-1) + A*h*ye(:,i-1);

end

figure('name','1')
plot(t,u(1,:),'-o');
hold on
plot(t,ye(1,:),'-o');
hold on
plot(ode_t, ode_y(:,1),'-');
hold off
legend('GAUSS-LEGENDRE','EULER','ODE113')


%---------------------------------------------
hspace = [logspace(-5,0,20) 2 3 4 5];
iter = 1;
for h=hspace
    clear u1 ode1_y t1 ode1_t F ye1;

    u1(:,1) = [4 -2];
    ye1(:,1) = [4 -2];
    Bu = [I-(5/36)*h*A, ((sqrt(15)/15)-
(2/9))*h*A, ((sqrt(15)/30)-5/36)*h*A;
      -(5/36 + sqrt(15)/24)*h*A, I-(2/9)*h*A,
((sqrt(15)/24)-5/36)*h*A;
      -(5/36 + sqrt(15)/30)*h*A, -(2/9 +
sqrt(15)/15)*h*A, I-(5/36)*h*A];


    t1 = 0:h:10;
    [ode1_t, ode1_y] =
ode113(@fun,t1,y0',opts);



    for i=2:length(t1)
        F(:,i) = Bu\[A*u1(:,i-1);A*u1(:,i-1)
;A*u1(:,i-1)];
        u1(:,i) = u1(:,i-1) +
h*((5/18)*F(1:2,i) + (4/9)*F(3:4,i)
+(5/18)*F(5:6,i));

        ye1(:,i) = ye1(:,i-1) + A*h*ye1(:,i-
1);

    end

    rmse(iter) = norm(ye1(1,:)'-
ode1_y(:,1),2)/norm(ode1_y(:,1),2);
    maxere(iter) = norm(ye1(1,:)'-
ode1_y(:,1),inf)/norm(ode1_y(:,1),inf);


    rms(iter) = norm(u1(1,:)'-
ode1_y(:,1),2)/norm(ode1_y(:,1),2);
    maxer(iter) = norm(u1(1,:)'-
ode1_y(:,1),inf)/norm(ode1_y(:,1),inf);
    iter = iter +1
end

figure('name','RMS');
loglog(hspace, rms,'-o');
hold on
loglog(hspace, rmse);
legend('RMS GAUSS-LEGENDRE','RMS EULER')

figure('name','MAX ERROR');
loglog(hspace, maxer,'-o');
hold on
loglog(hspace, maxere);
legend('MAX GAUSS-LEGENDRE','MAX EULER')


function dydt = fun(t,y)
dydt = [y(2);-1.25*y(1) - y(2)];
end
```