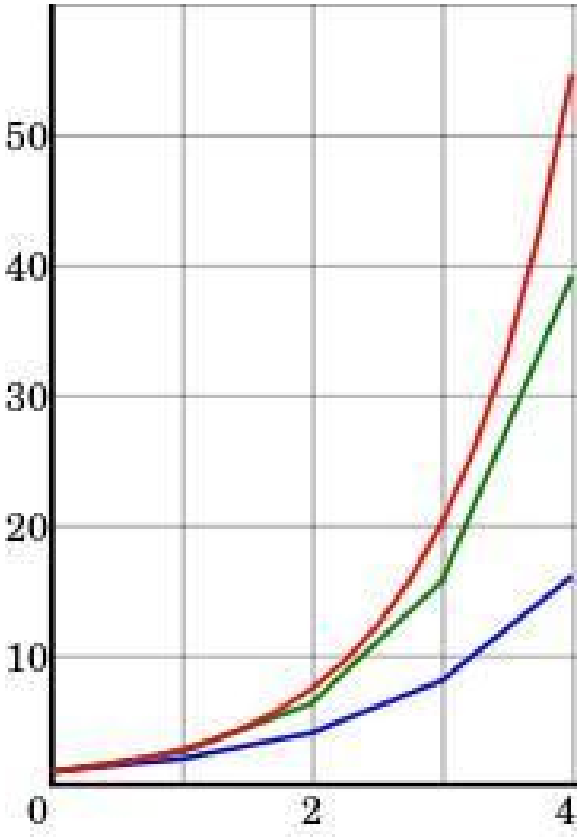
NUMERICAL METHODS

REPORT

Finding approximate solution and comparison with exact



Anton Krylov

03.11.2018

BS17-3

Variant 11

TECHNOLOGIES

Assignment have been implementing with **Python** programming language.

Library for computations: **Numpy**.

Library for GUI: **PyQt5**.

Library for plotting: **matplotlib**.

STRUCTURE

All source code can be found here.

The final program includes one class ***num_methods*** and *main* method. The class contains all needed information about function and methods for it approximate computation:

Technical:

1. *globalfield* Sets initial conditions and elements/events for graphical user interface.

2. *main* method. Creates window for graphical application, add plots, labels for changing initial conditions.

3. *update* method. Self-defined. Triggers recompute for numerical methods and updates graphs on plots.
4. *submitx* method. Recompute for new initial value problem x0
5. *sumbity* method. Recompute for new initial value problem y0

Math:

1. *f* method. Returns the result of y`=f(x,y) function.
2. *f_ex* method. Returns the result of analitically solved differential equation for IVP.

3. *euler* method. Returns the y values for given interval using Euler's method.
4. *up_euler* method. Returns the y values for given interval using Improved Euler's method

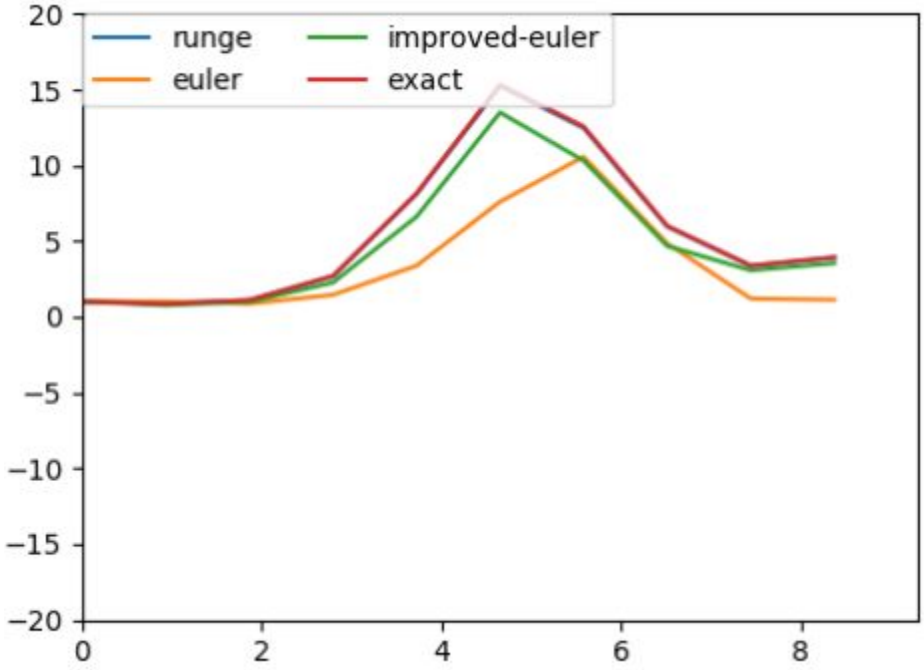
5. *runge* method. Returns the y values for given interval using Runge-Kutta's method.

Main method creates all needed functions, textbox and slider, connect all code together, and runs the application.

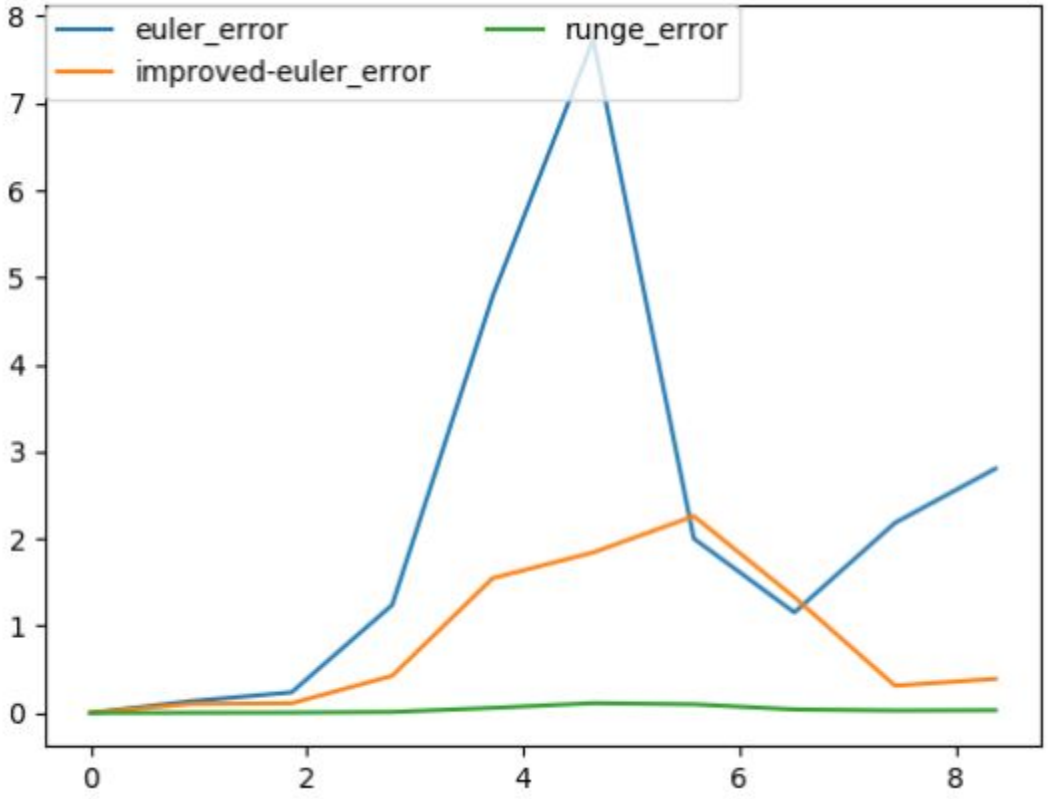
IVP

The given differential equation is $y' = e^{-sinx} - ycosx$
For this equation the general solution is:
$y(x) = c_1 e(-sin(x)) + x\, e\hat{}(-sin(x))$

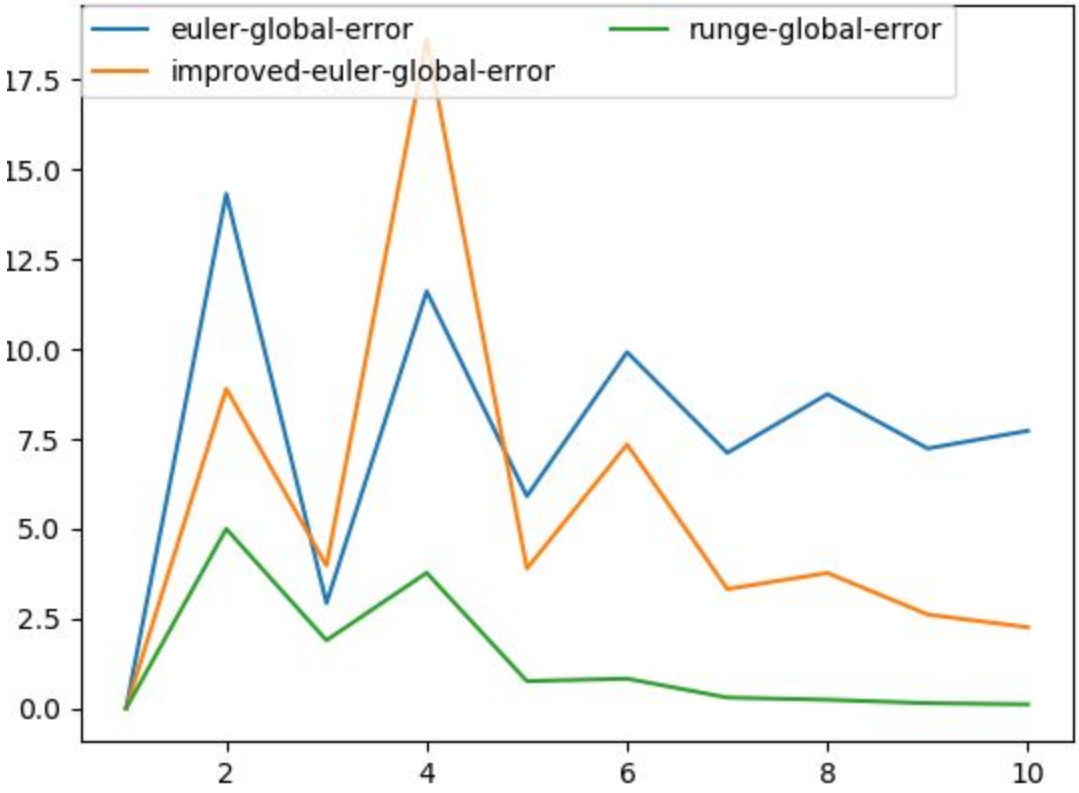
For given x = 0, y = 1 => c = 1 and N=10, and function will be look like that:



and local errors will be

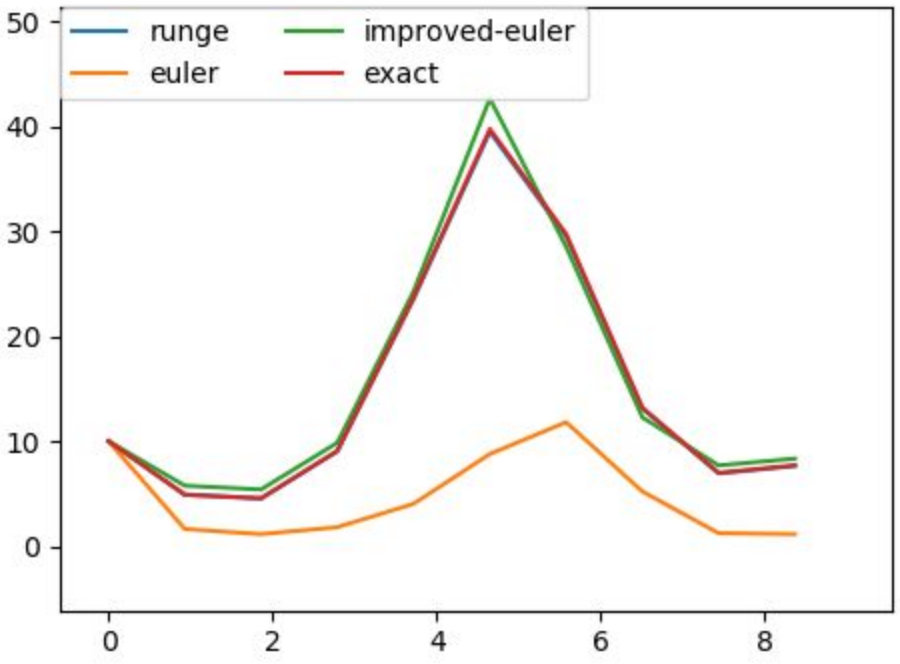


And global error from 1 to 10 will be

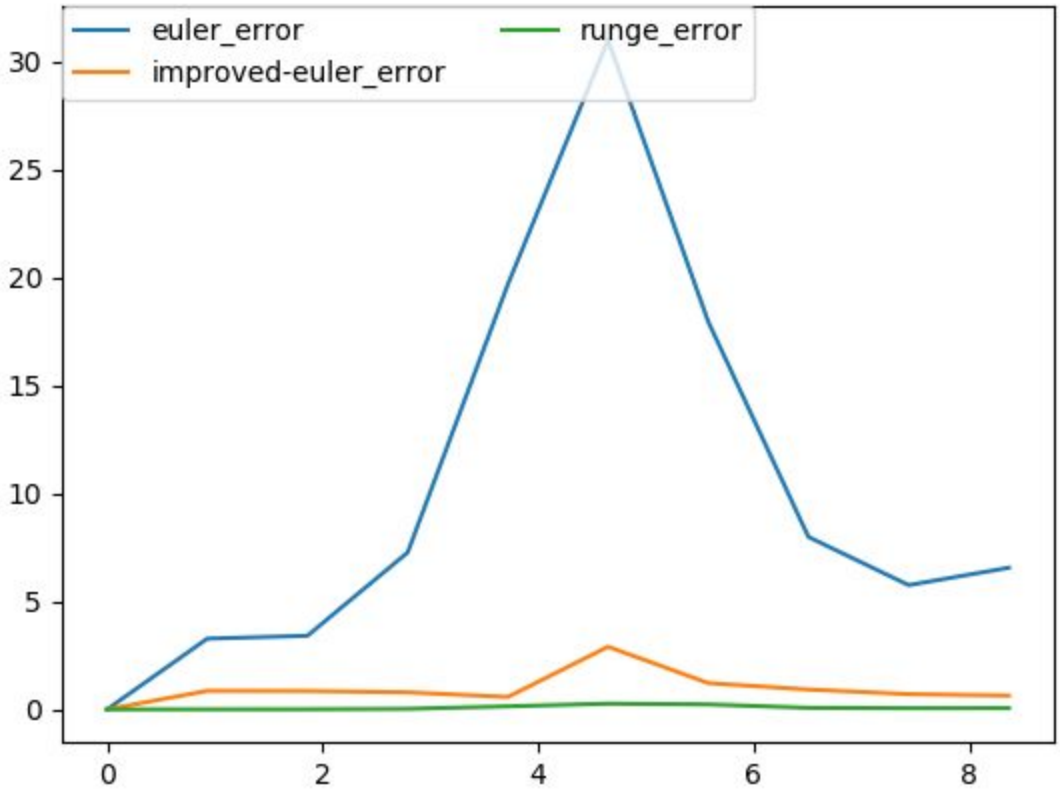


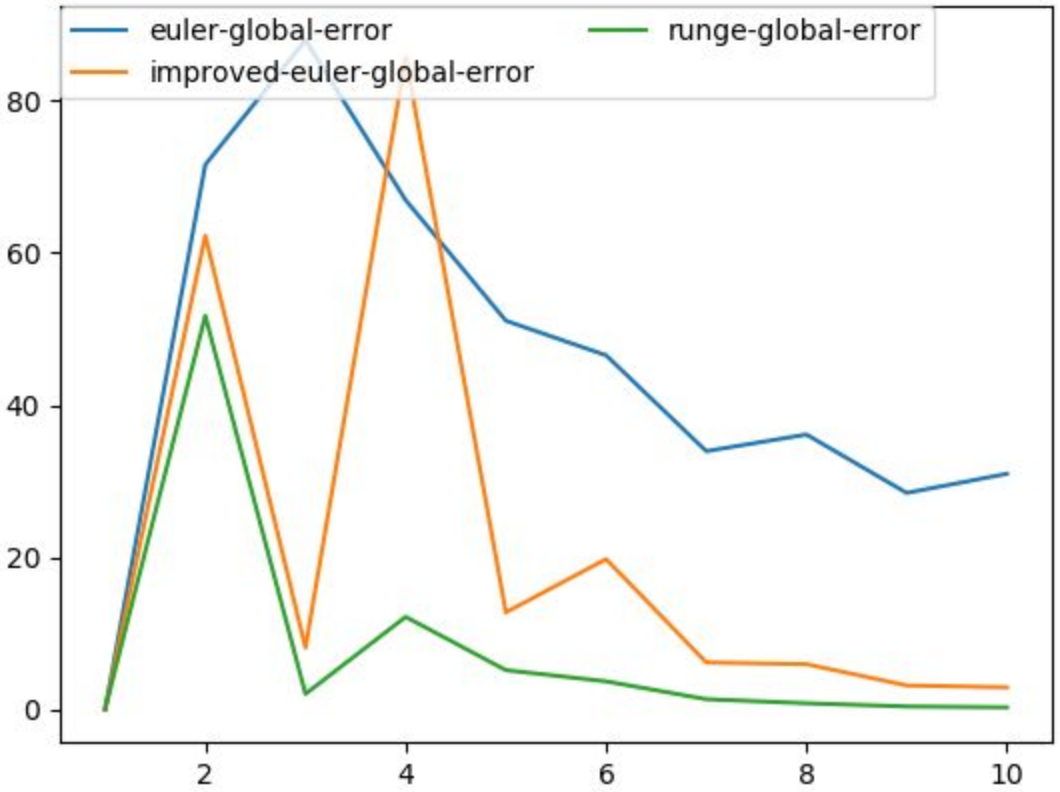
To show the difference between precision in methods we have to change the initial conditions.

For example, let y(0) = 10. With N = 10

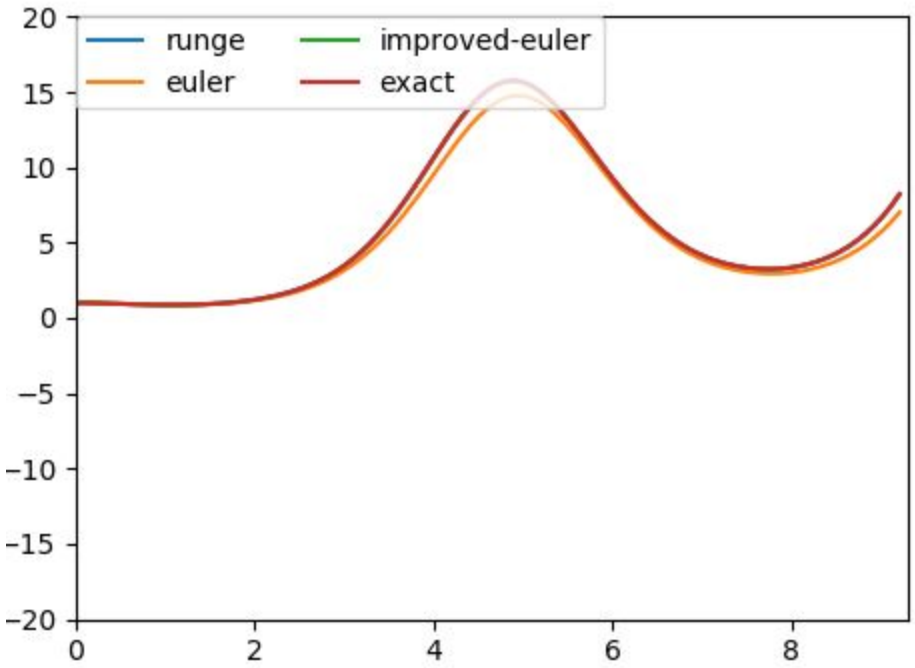


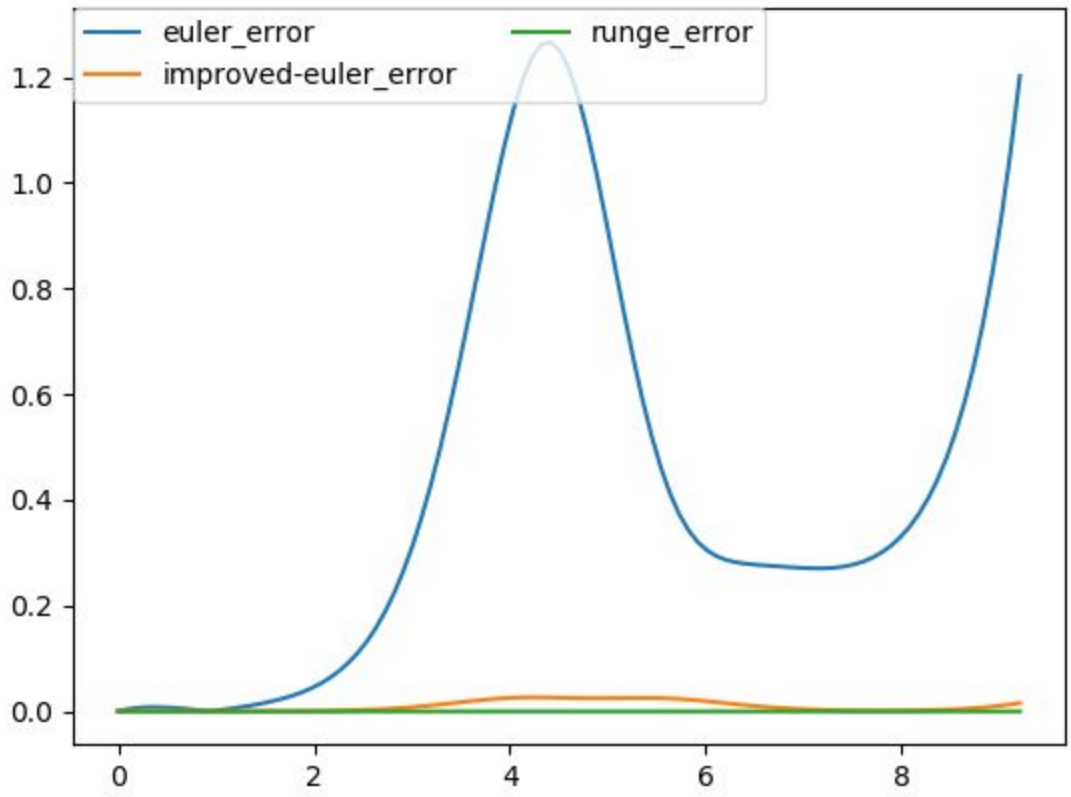
:

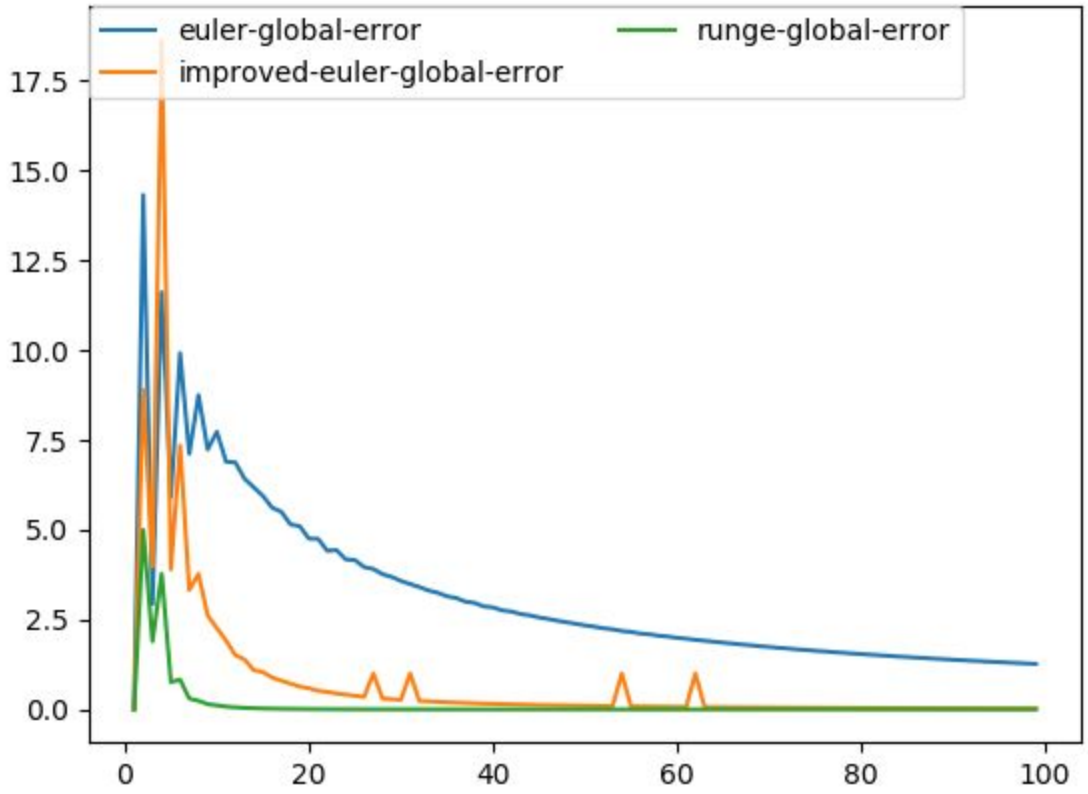




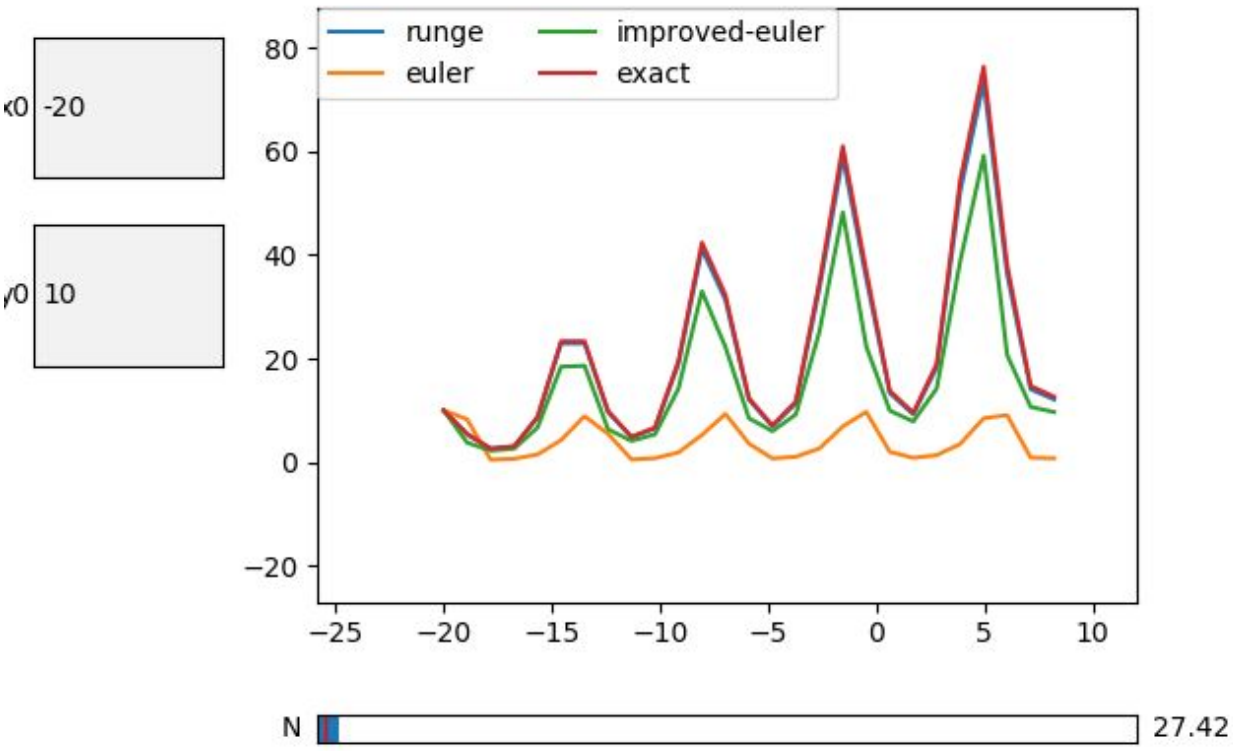
But if we increase the amount of step, the difference between methods becomes smaller. With N = 100 graphs are:

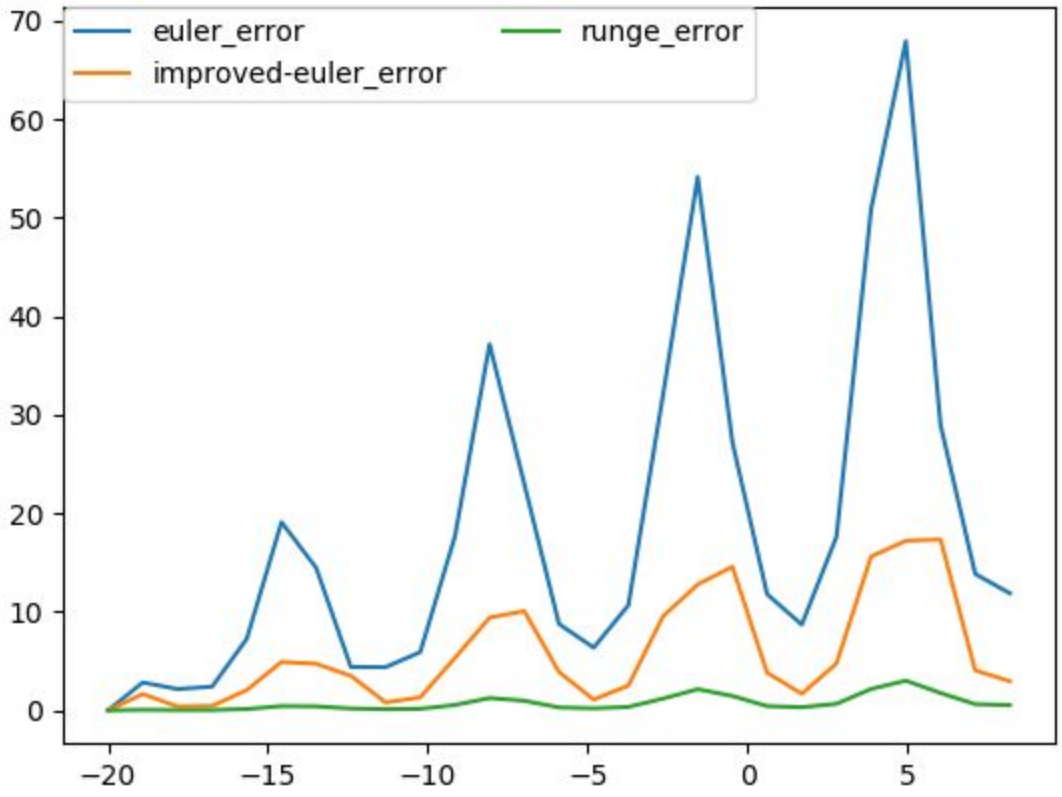



euler_error
improved-euler_error
runge_error
1.2
1.0
0.8
0.6
0.4
0.2
0.0
0
2
4
6
8


euler-global-error
improved-euler-global-error
runge-global-error
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
0
20
40
60
80
100

As we can see, Euler’s method shows the worst results, the amount of error go like periodic function(and also the graph look like periodic function)





CONCLUSION

Among given three numerical methods the one that shows the best results is Runge-Kutta's method. But it requires the computation of function for 4 times on each step, so for the problems where range is very large and accurate precision doesn't required the better choice is improved Euler's method with good balance of precision and steps required to achieve that precision.