Innopolis University

DIFFERENTIAL EQUATIONS ASSIGNMENT REPORT

BY MAXIM KORSUNOV

2019

Introduction

Task

The task of the assignment was to create the solution to first-order differential equation using numerical methods such as Euler, Improved Euler and Runge-Kutta. Except the solution, we had to present visual graphs of these methods, local and total approximation errors.

During the implementation of the assignment, I decided to do not only my variant, which was the 11th, but all of them. That is why report will show the assignment as a whole, not only one specific differential equation.

Demo

The assignment was created as a web application, so it has a demo published in Github pages. You can access it using the link.

Exact solution

In order to compare the numerical solutions and find local and total approximation errors, there is a need in a proper exact solution that can help us see the graph of the equation.

The equation is:

$$y' = xy - xy^3$$

Initial conditions are

$$y_0 = \sqrt{\frac{1}{2}},\ x_0 = 0,\ \ x_{max} = 3$$

This is the example of Bernoulli equation that can be solved with the changing of variable:

1) Subtract xy from both sides and divide by $-{}^{1}/_{2}\, y^3$ gives:

$$y' - xy = -xy^3$$

$$-\frac{2y'}{y^3} + \frac{2x}{y^2} = 2x$$

2) Changing variables: let $v = \frac{1}{y^2}$, so

$$\frac{dv}{dx} = -\frac{2y'}{y^3} => \frac{dv}{dx} + 2vx = 2x$$

3) Find integrating factor $\mu = e^{\int 2xdx} = e^{x^2}$, that we should multiply to previous equation

$$e^{x^2}\frac{dv}{dx} + v\left(2xe^{x^2}\right) = 2xe^{x^2}$$

4) Apply the reverse product rule to express v:

$$\int (e^{x^2}v)\,dx = \int 2e^{x^2}xdx$$

$$e^{x^2}v = e^{x^2} + c$$

$$v(x) = ce^{-x^2} + 1$$

5) Solve for $y(x)$:

$$y(x) = \frac{e^{x^2/2}}{\sqrt{e^{x^2} + c}}$$

6) In order to easily find constant in the future, it is better to express it:

$$c = \frac{e^{\frac{1}{4}x_0^4}}{y_0^2} - e^{x^2}$$

Implementation

Technology

For the fast development with the best quality *JavaScript* language was chosen as prior to make good user interface and connect *Chart.js* library to construct graphs of solutions and errors.

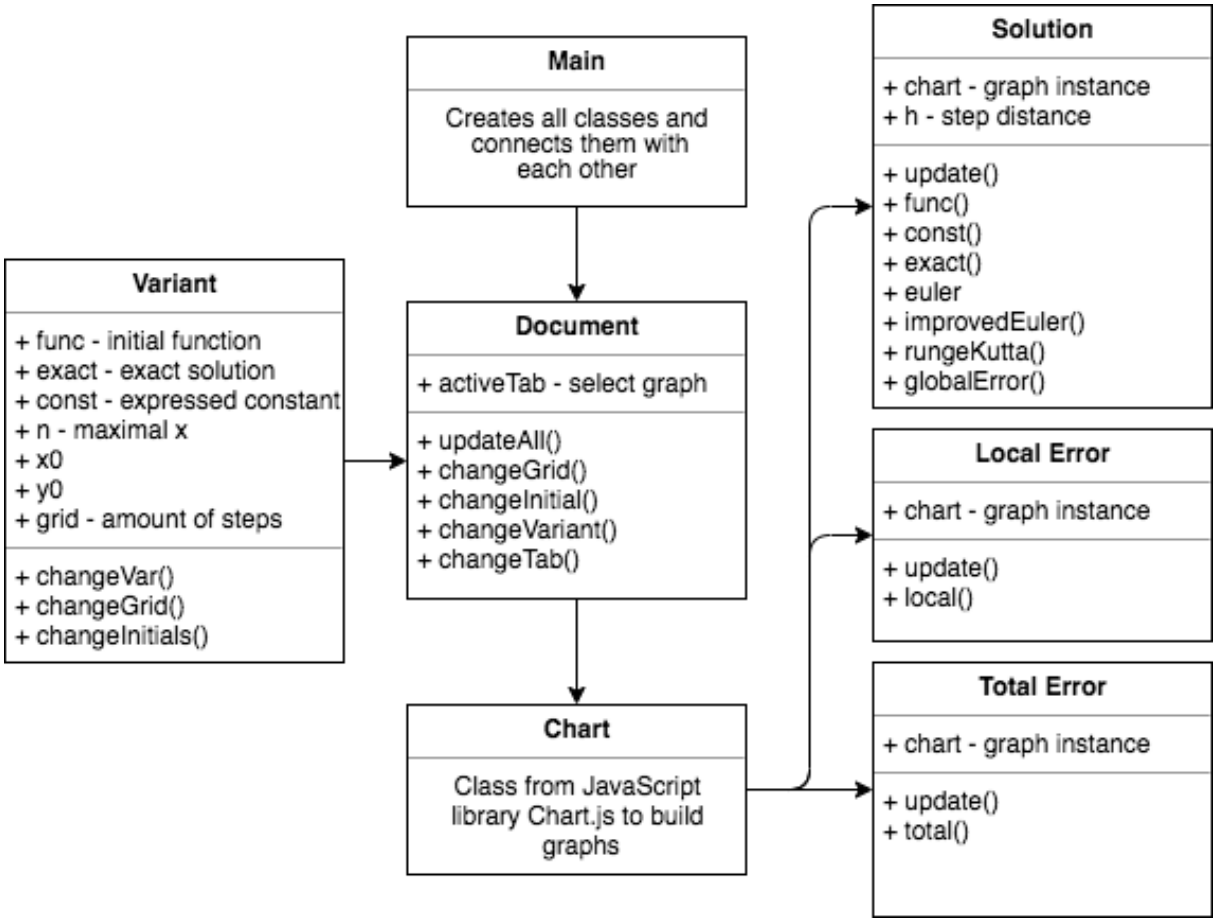
The code is build using OOP and follows SOLID principles to be readable and understandable by the most part of programming community.

Structure

As far as the application is object oriented, it can be divided into classes. JavaScript approach requires working with webpage's document model and, since the application implements all variants, it has following classes:

• Main • Variant • Document • Chart • Solution • Local Error • Total Error •

These classes are represented on the following diagram.



Variant

The class works with the data that manages the most important properties of the graphs: initial values, function declarations and functions of exact solutions solved by hand. All variants are structured in an array in configuration file like this:

```
11: {
  func: 'x*y - x*Math.pow(y, 3)',
  exact: 'Math.exp(1/2 * x*x) / Math.sqrt(c + Math.exp(x*x))',
  const: 'Math.exp(1/4 * Math.pow(x, 4)) / Math.pow(y, 2) - Math.exp(x*x)',
  x: 0,
  y: Math.sqrt(1/2),
  n: 3,
},
```

Document

Document class presents the webpage and its interactions with user. It changes tabs (what graph is to display), handles inputs of new initial values, slider of the size of the step and the change of the variant. The most important feature *updateAll()* rerenders all charts on the page with some speed optimization.

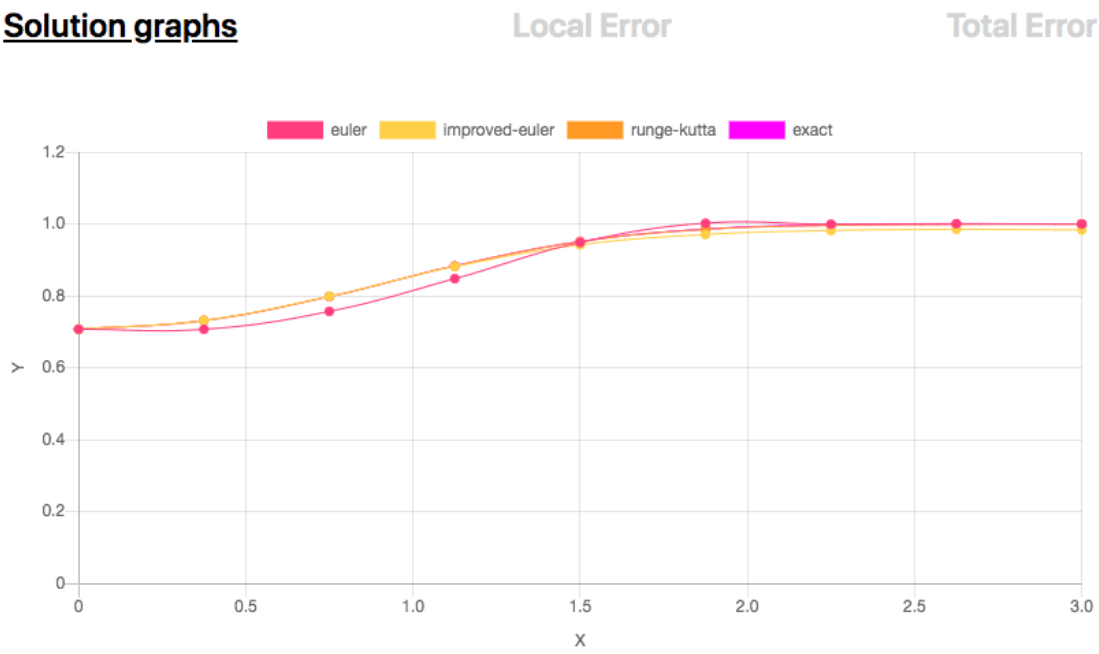
Solution, Local and Total Error

These classes operate with formulas and data that is passed down to the class **Chart.** They have mostly functions-calculators of different numerical methods that return the array of points which is passed to the chart to be plotted.

Results

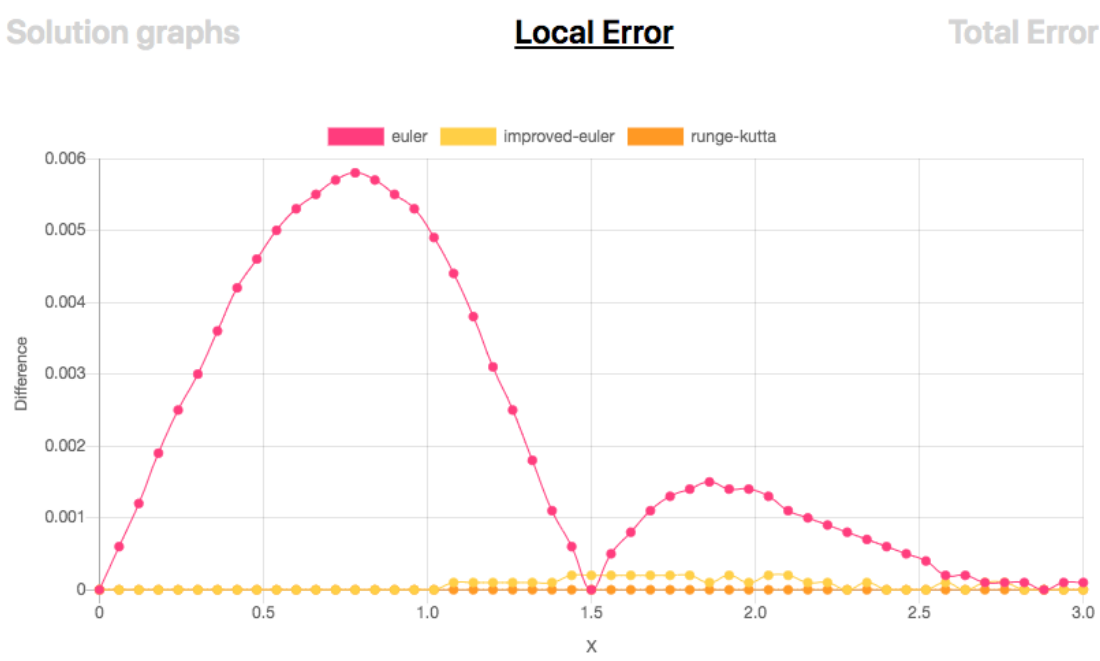
Numerical methods

In case of 11th variant, the graph shows that all methods are very close to exact solution, but Runge-Kutta is the most optimized numerical methods that is nearly the same as the exact solution.



Local Error

Local approximation error gives the new look at the difference between methods. Here Runge-Kutta has no difference with exact solution, but Euler method if far from the truth.



Total Error

The graph is needed to show how the number steps affect our picture and how big mistake could be. In this case, there is almost no error after we try to take 9 steps.

Conclusion

As it was shown, numerical solutions are easy to understand and very helpful in case if you need numbers to operate with your task. Runge-Kutta method gives the best accuracy comparing with 2 other methods, so it has to be the most preferable numerical method in everyday calculations.