Report of Project 4: Compact SPICE for ODE Solution

Generic ODE Solvers:

Within this project, ODE solvers using Forward Euler, RK4, RK34(with time adaption) methods are designed to simulate the RC and CS Amplifier circuits. Solvers are designed to be able to take in any arbitrary dimensions of input vector, and able to interface with any ODE functions by indicating the function type. The input interface of ODE solvers is:

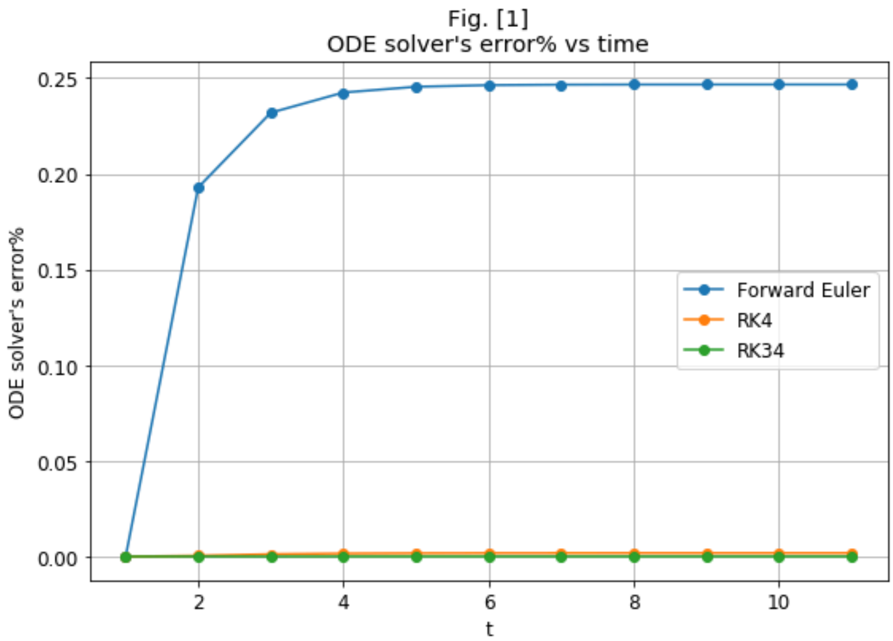
`Vector x`: the variables to be calculated;
`double t`: the time-stamp for each step's estimation;
`double h`: the step length;
`String[] fType`: the ODE function types matching `Vector x`.

Task 1, 2: Design and Inheritance

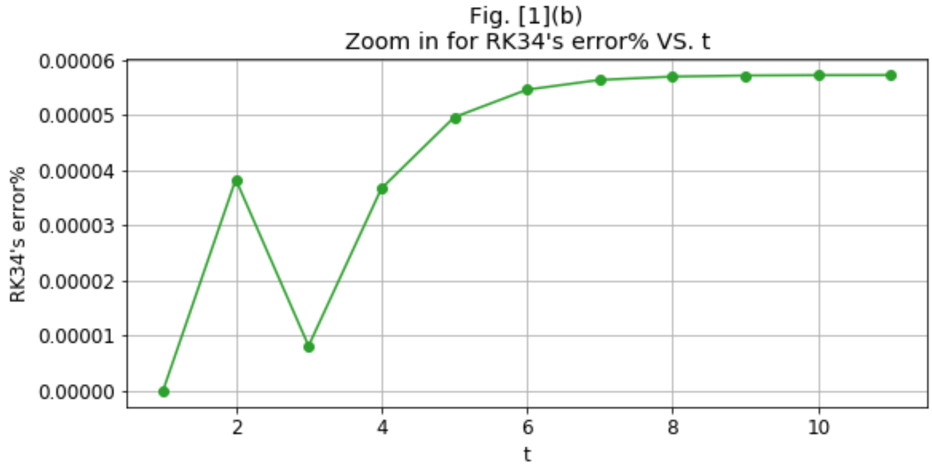
The program breaks down into 4 java scripts (Solvers.java, Simulation.java, Test.java, and Main.java) and inherits 2 previous-written helper classes (Vector.java, FileIO.java), which are elaborated in detail in Git-submission's ReadMe.md file. The inherited classes' path was built together with this project to ensure they can run interchangeably. Also, the 1-d generic ODE solver from previous Hacker Practice is modified into the version that takes input of any arbitrary Vector.

Task 3: Validation for Generic ODE Solvers

The ODE solvers are tested with the example given in Note 6, page 23, where the ground-truth is provided to help compute the normalized relative error $|\varepsilon_x|$. Marching from $t = 0s$ to $t = 11s$, $timestep = 1s$, and compute the $|\varepsilon_x|$ within each time step:



Observation: The $|\varepsilon_x|$ of Forward Euler and RK4 methods accumulates with time (increasing), and it could be seen that forward Euler has worse accuracy compared to RK34 (with time adaption) and RK4, because Forward Euler is only first-order accurate, while RK4 and RK34 are forth-order accurate. The $|\varepsilon_x|$ of RK34 in Fig. [1] is too small to tell the trend, so zoom it in as:



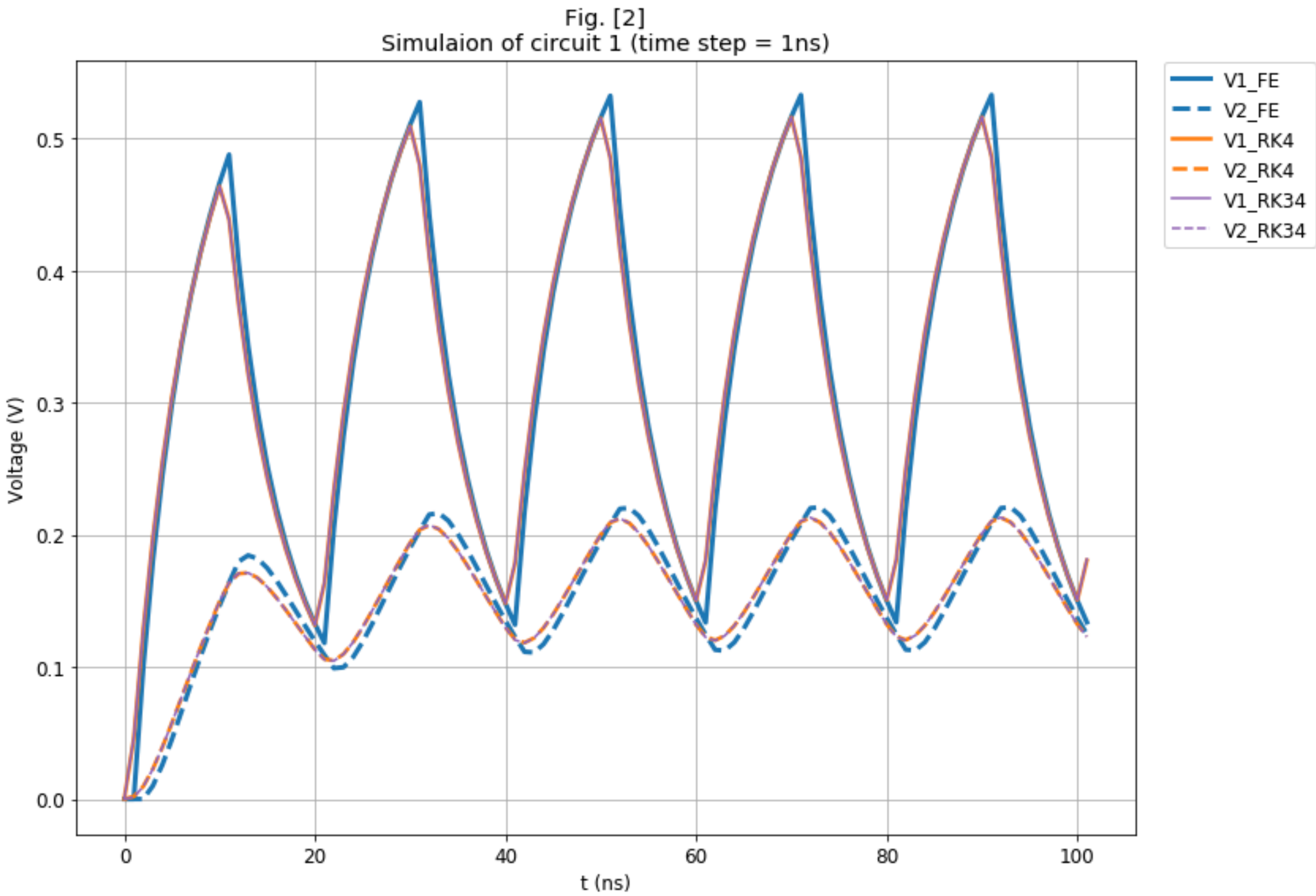
It could be seen that RK34's error as a whole is also accumulated (increasing) with time.

- ## Task 4: Simulation for RC Circuit's V_1, V_2:

The voltages at node 1 and node 2 of RC circuit in handout's Fig. 3, (V_1, V_2) could be expressed as:

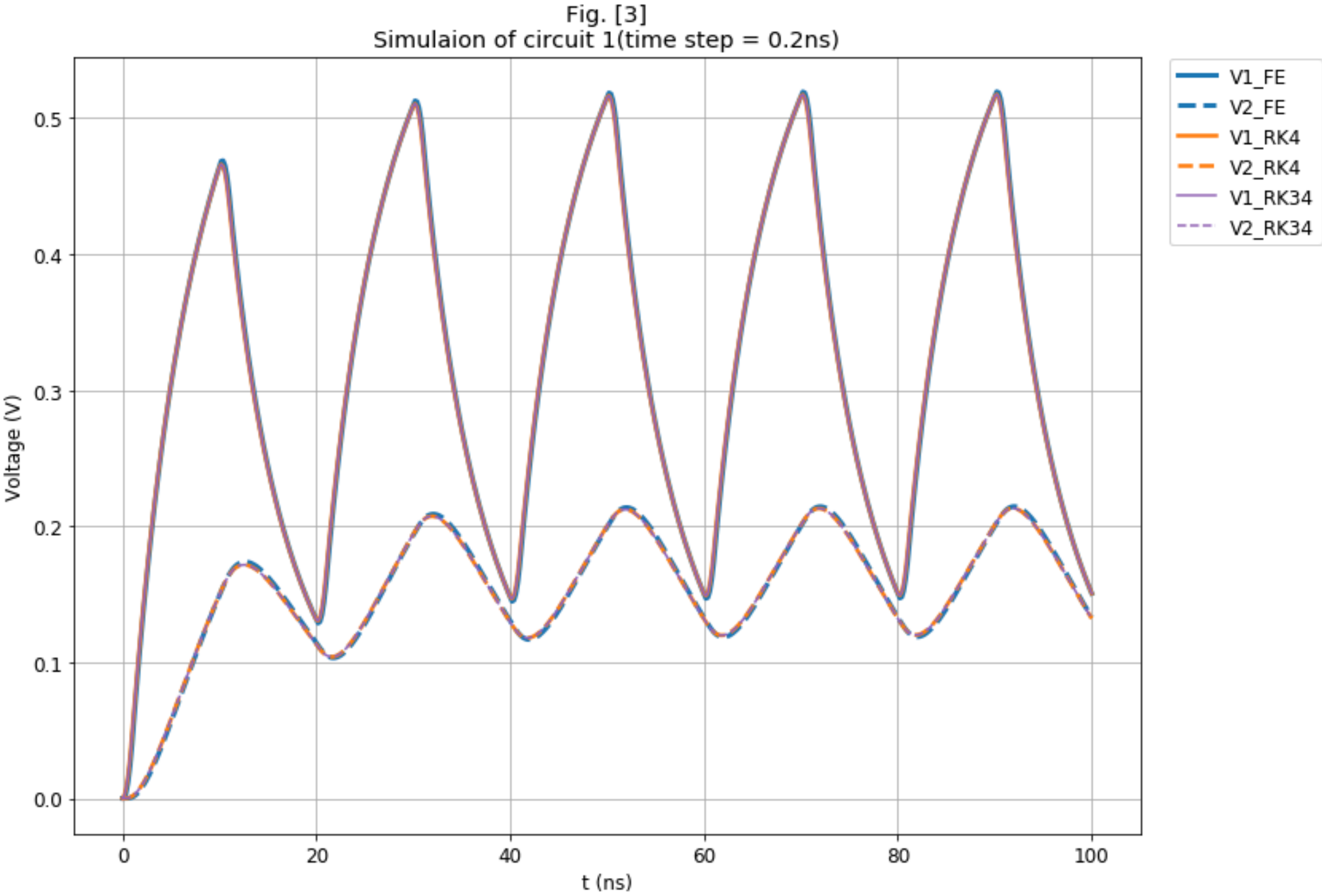
$$\frac{d}{dt}\begin{pmatrix} V_1 \\ V_2 \end{pmatrix} = \begin{pmatrix} -\left(\frac{1}{C_1R_1}+\frac{1}{C_1R_2}\right) & \frac{1}{C_1R_2} \\ \frac{1}{C_2R_2} & -\left(\frac{1}{C_2R_2}+\frac{1}{C_2R_3}\right) \end{pmatrix}\begin{pmatrix} V_1 \\ V_2 \end{pmatrix} + \begin{pmatrix} \frac{i(t)}{C_1} \\ 0 \end{pmatrix}$$

The initial $(V_1, V_2) = (0, 0)$. For RK34, set the absolute error's tolerance as $e_A = 1e-4$, and the relative error's tolerance $e_R = 1e-7$. **When $\boldsymbol{timestep = 1ns}$ ($\boldsymbol{100ns\ in\ total}$):**



<u>Observation:</u> It could be seen that when $timestep = 1ns$, (V_1, V_2) curves generated by RK34 (With time adaption) and RK4 almost overlap with each other. Forward Euler estimates next step based on the rate of current points, therefore when time-step is big, the curve (the blue ones) would seem to be dragged behind of the other two higher-order accurate methods (RK34 and RK4).
Also, it could be seen that the simulated (V_1, V_2) has a period of 20ns, equivalent to that of the input $i_{in}(t)$, validating that the ODE functions are correctly reacting to $i_{in}(t)$.

When $\boldsymbol{timestep = 0.2ns\ (100ns\ in\ total)}$:



<u>Observation:</u> It could be seen that when time step is finer, Forward Euler's next step estimation is more accurate (based on the fact that RK4 and RK34 are higher-order accurate methods, thus generate more accurate results) than when $timestep = 1ns$. The curves of three methods are smoother when time-step is smaller.

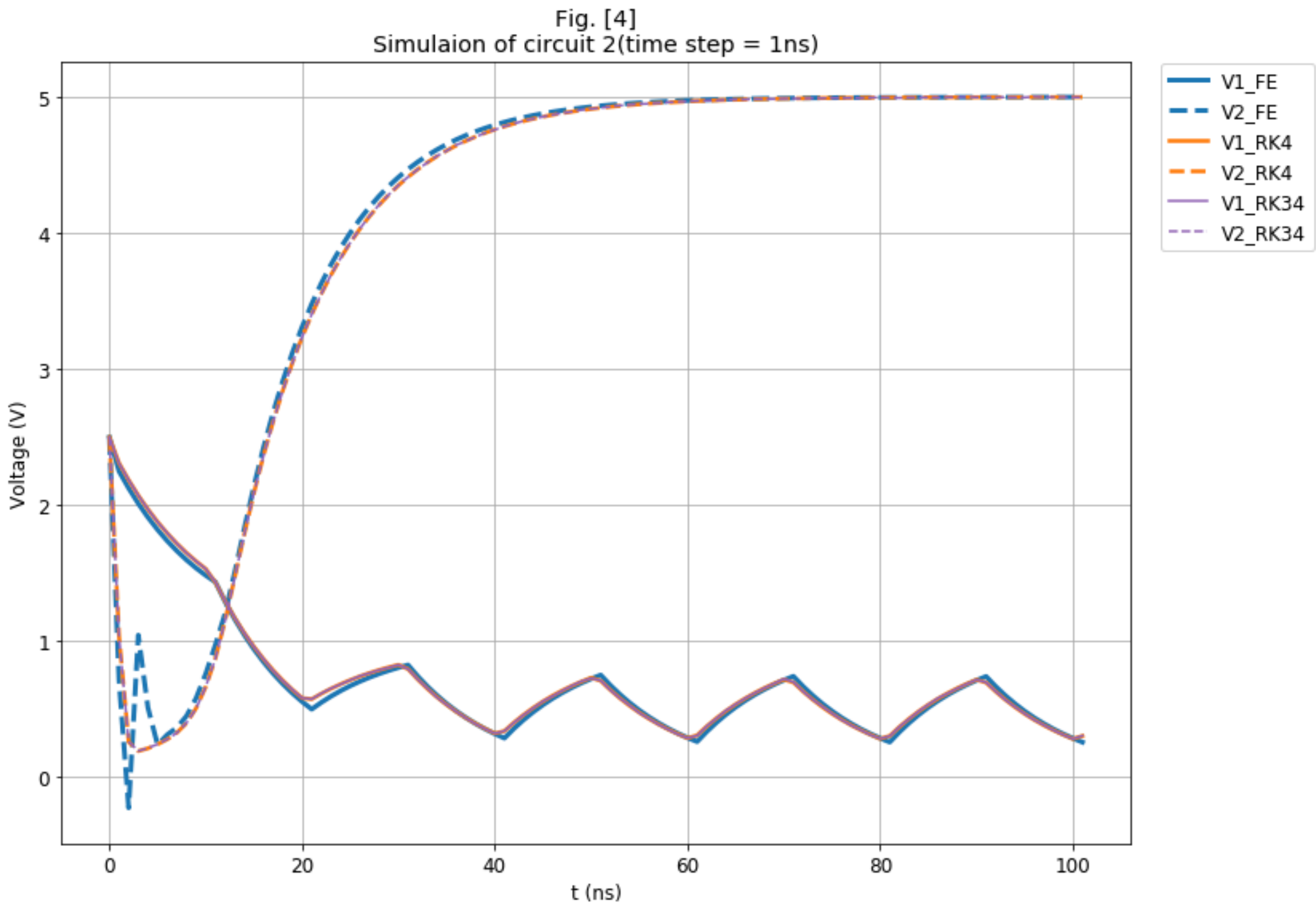
- ## Task 5: Simulation for Amplifier Circuit's

The voltages at node 1 and node 2 of Amplifier circuit in handout's Fig. 3, (V_1, V_2) could be expressed as:

$$\frac{dV_1}{dt} = f_1(V_1, V_2) = -\frac{1}{R_G C_1} V_1 + \frac{i_{in}(t)}{C_1}$$

$$\frac{dV_2}{dt} = f_2(V_1, V_2) = -\frac{1}{R_L C_2} V_2 - \frac{I_{D,EKV}(V_1, V_2)}{C_2} + \frac{V_{DD}}{R_L C_2}$$

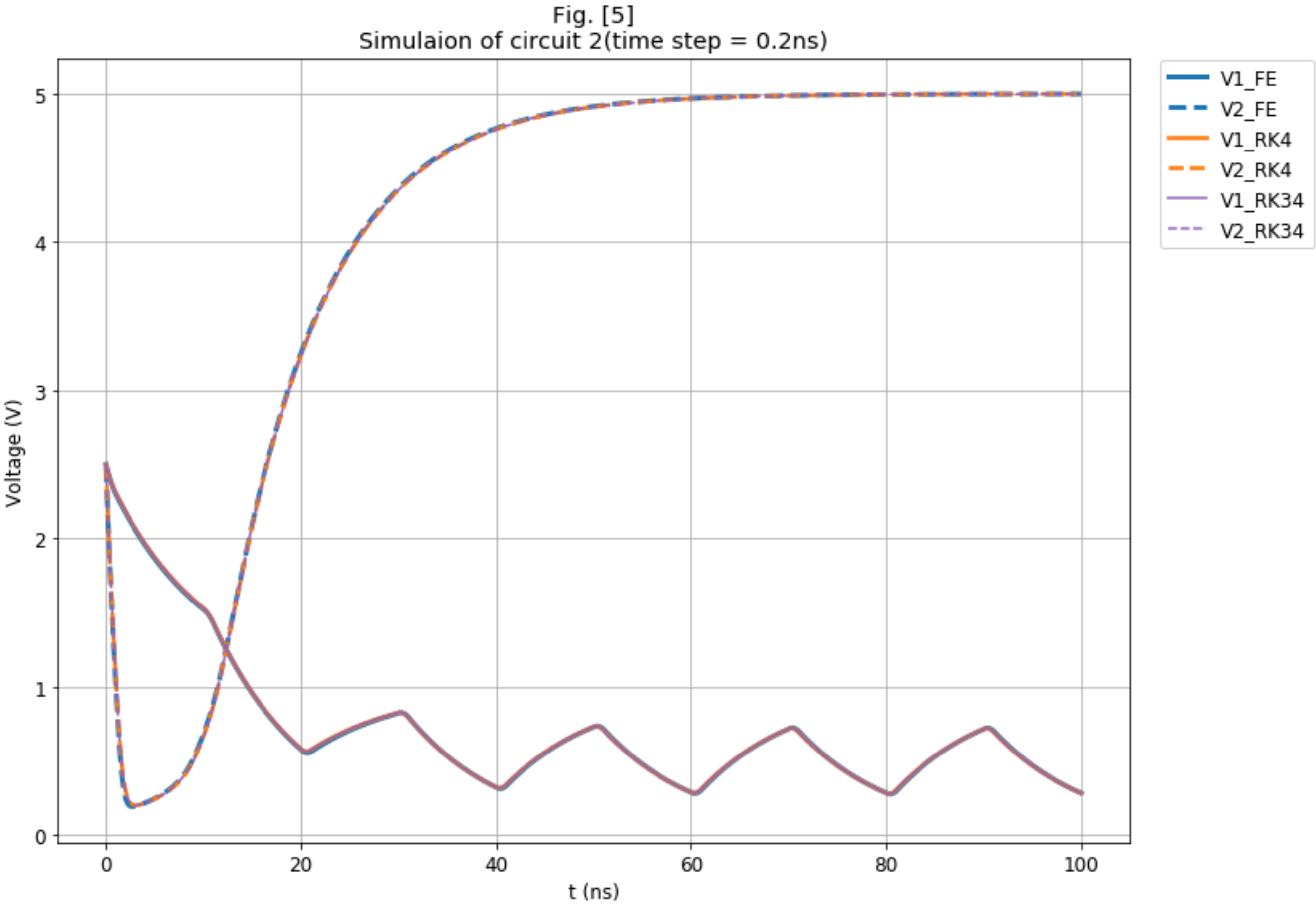
The initial $(V_1, V_2) = (0, 0)$. For RK34, set the absolute error's tolerance as $e_A = 1e-4$, and the relative error's tolerance $e_R = 1e-7$. **When** $\boldsymbol{timestep = 1ns\ (100ns\ in\ total)}$**:**



<u>Observation:</u> Still, it could be seen when $timestep = 1ns$, (V_1, V_2) curves generated by RK34 (With time adaption) and RK4 almost overlap. At around $t = 1ns$, the Forward Euler (blue curve) oscillates greatly, which could be due to when the gradient of ground-truth changed from negative to positive, Forward Euler method is unable to immediately reflect this change, and would only react to the gradient flip at next step when the ODE function directs it to change accordingly. Therefore when the gradient of ground-truth (simulated with RK34 or RK4) flips, Forward Euler would overshoot at these gradient-flipping points.

Also, it could be seen that after 20ns, when the Amplifier circuit starts to stabilize, ΔV_1 is indeed amplified to ΔV_2, and V_2 is strictly bounded between 0V-5V, satisfying the Amplifier circuit's physical law.

When $\boldsymbol{timestep = 0.2ns\ (100ns\ in\ total)}$**:**



Observation: It could be seen when $timestep = 0.2ns$, three methods' curves almost overlap. The finer time step enables Forward Euler to react sooner to the gradient flips, therefore the oscillation at around $t = 1ns$ disappears, and its curve approaches higher-order methods' curves, and is more accurate than when $timestep = 1ns$ (based on the fact that RK34, and RK4 are higher-order accurate than Forward Euler).

Also, the curves of three methods are smoother when time-step is smaller.