Particle Advection: Euler Method vs RK4 Method

Method Definitions

Both the Euler and the 4th order Runge-Kutta (RK4) methods can be used to perform particle advection over a 2D vector field [1].

For a given position ($p_i \in P$), time ($t_i \in T$), step size (h), and vector field value ($v(T \wedge P)$), the Euler method is defined as follows:

$$p_{i+1} = p_i + h * v(t_i, p_i)$$

The RK4 method is defined as:

$$\begin{aligned} p_{i+1} &= p_i + (1/6) * h * (k_1 + 2k_2 + 2k_3 + k_4) \\ k_1 &= v(t_i, p_i) \\ k_2 &= v(t_i + h/2, p_i + h/2 * k_1) \\ k_3 &= v(t_i + h/2, p_i + h/2 * k_2) \\ k_4 &= v(t_i + h, p_i + h * k_3 \end{aligned}$$

We can see that the RK4 method requires significantly more vector field calculations and arithmetic operations per position update. However, the Euler method requires a significantly smaller step size to attain the same degree of accuracy as RK4.

Accuracy Calculations

For the Euler method and a step size of h, the error can be bounded as follows:

$$\epsilon_{Euler} = \mathcal{O}(h),\ h < 1$$

For RK4, we can bound the error as:

$$\epsilon_{RK4} = \mathcal{O}(h^4),\ h < 1$$

Because $h < 1$, we can see that for the same step size, RK4 has a significantly smaller error bound ($\forall 0 < h < 1, h^4 < h$) Therefore, to attain the same degree of accuracy in both methods, RK4 can use a smaller step size.

For example, if we use a step size of h_{Euler} in the Euler method, then to obtain the same degree of accuracy in RK4, we can use the following:

$$h_{RK4} = (h_{Euler})^{0.25}$$

See the table below for a comparison of step sizes for different error bounds. We can see that as the error bound gets smaller, the step size for RK4 becomes much larger than the step size for Euler.

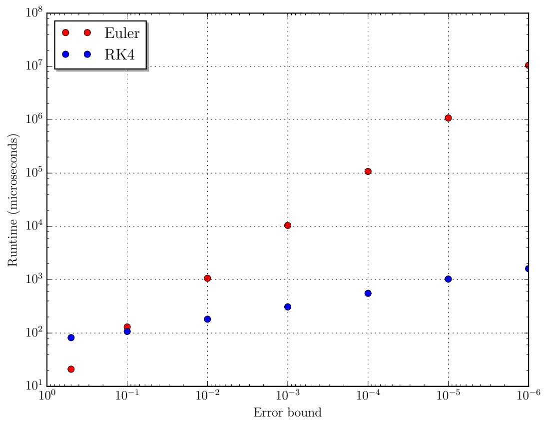
Error ϵ	h_{Euler}	h_{RK4}
0.01	0.01	0.316
$1e^{-4}$	$1e^{-4}$	0.1
$1e^{-8}$	$1e^{-8}$	0.01
$1e^{-16}$	$1e^{-16}$	$1e^{-4}$

Runtime Calculations

From this point, we can speculate that for a specified error bound, the RK4 method will have a lower runtime than the Euler due to the Euler method's small step size and resulting high number of steps required. However, we did note that each step in RK4 requires significantly more arithmetic and memory operations. The cost of these operations are dependant on the hardware used and the software implementation.

To confirm our speculations, we evaluate the run-times of the RK4 and Euler methods for different error bounds. For each error bound, the appropriate step size is used for the respective method. We execute each version using a single-threaded C++ implementation and an Intel i5-5257U CPU. The results of this evaluation are show in the figure below:

[1] The Euler method can be generalized as a 1st order Runge-Kutta method (RK1).



We can see that for large error bounds ($\epsilon > 0.1$), the Euler method actually outperforms the RK4 method. However for smaller error bounds ($\epsilon < 0.1$), RK4 significantly outperforms the Euler method. Note that the y-axis uses a logarithmic scale.

Conclusions

We conclude that the RK4 method is most appropriate any time we require a high degree of accuracy. Although the RK4 method performs more calculations per step, the smaller number of required steps far outweighs these extra calculations.