Arjun Ray

Euler's Integrator

Facility for Rare Isotope Beams, Michigan State University, East Lansing, MI 48824 USA

2025

This material is based upon work supported by the U.S. Department of Energy Office of Science under Cooperative Agreement DE-SC0000661, the State of Michigan and Michigan State University. Michigan State University operates FRIB as a DOE Office of Science National User Facility in support of the mission of the Office of Nuclear Physics.

Adaptive Euler Integration with Midpoint Error Control

Euler's method

$$y_{(n+1)} = y_n + h \cdot f(t_n, y_n)$$

Simple, fast, but only first-order accurate. Errors accumulate quickly if step size is too large

Techniques used are MidPoint Based Error Estimation, Error Control, Adaptive Step Size Control and Parallelization

Midpoint-Based Error Estimation

We compute two different estimates of the solution at the next time point. By comparing the two, we estimate the error.

1. Full Step: We take one regular Euler step using the current slope.

1. Midpoint Step: We take a half step, compute the slope there, and use it to complete the step.

.

Error Control Logic

We compare the two estimates to get the numerical error, scaled using user-defined tolerances.

If the error is below a threshold, we accept the step; otherwise, we retry with a smaller one.

This keeps errors within bounds.

Adaptive Step Size Control

The code adjusts step size based on error:

-it grows if error is small

- shrinks if too large.

A safety factor prevents drastic changes, helping balance speed and accuracy.

```
static constexpr double SAFETY  = 0.3;   // step-size safety factor
static constexpr double FAC_MIN = 0.1;   // minimum shrink factor
static constexpr double FAC_MAX = 1.0;   // maximum grow factor
```

Accuracy and Speed

- Small step size → High accuracy, low speed
- Large step size → High speed, low accuracy

Tuning Parameters

– SAFETY: Controls how conservatively step size adapts

– FAC_MIN, FAC_MAX: Clamp the shrink/grow rate

Basically, the more accurate the code is, the slower it will run.

The pre-set values are set to achieve a balance between speed and accuracy.

This method is inefficient for handling stiff problems.

Comparison-1

Using v3, and a step-size bound of 1-10,

Our original ODE gives-

iter=2 q_{prob}=73.75 $\langle q \rangle$=73.75(1.25)

****

(or 4.104 sec)

****

And Euler's method gives-

iter=2 q_{prob}=73.75 $\langle q \rangle$=73.75(1.25)

****

(or 3.973 sec)

****

-Accuracy is high when step-size is smaller.

-It is also slightly faster using this version.

-If we increased SAFETY, our program would give a less accurate answer, but will run faster.

Comparison-2

Using v.4, and a step-size bound of 1-250,

Our original ODE gives-

iter=3 q_{prob}=71.89 <q>=72.47(1.21)

****

(or 10.578 sec)

****

And Euler's method gives-

iter=3 q_{prob}=71.90 <q>=72.48(1.21)

****

(or 4.647 sec)

****

-Euler's method is much faster in this case but with some loss in accuracy.

-Decreasing FAC_MAX or SAFETY would give a more accurate answer, but with the cost of runtime(would still be significantly faster than the original method)

-Accuracy will decrease with a higher minimum step.