1)

I chose the Lorentz Equation:

2)

Chose the Adam’s Bashforth four step(4th order)(1) for the predictor and the Adam’s Mouton Four step (5th order)(2) as the corrector, with Euler forward (1st order)(3), Leap frog (2nd order)(4) and Adam’s Bashforth 3 step (3rd order)(5) methods to find the required initial conditions

Using the Runge-Kutta 4 method(6) for the Runge-Kutta method

3)

Order of convergence for the Adam’s Bashforth four step

For

Represented by

For

For

For

For

For

For

Fails at the fifth order polynomial, the method is order four as asserted earlier

Now the stability test

Set

And set

Use the Anzats

Which implies that the method is strongly stable.

As the method has an order and has solutions to the characteristic polynomial such that , the method is convergent by the Dahlquist theorem.

4)

For

Represented by

For

For

For

For

For

For

Thus the method is fifth order

Now the stability test

Set

And set

Use the Anzats

Which implies that the method is strongly stable.

By the Dahlquist theorem, the method is convergent, as expected.

5)

The Runge-Kutta 4 method is fourth order accurate.

6)

See attached

7)

See attached

8)

For the Runge-Kutta 4 method each iteration step required one evaluation of at the current time step. The Adams Bashforth 4th Order/Adams Moulton 5th Order predictor corrector method required eight evaluations of spread between the two algorithms, with evaluations at the current through three previous time steps (i.e. ) and the next time step back through the three previous time steps(i.e. ) respectively, which meant storage of four time steps, plus determination of the next time step.

9)

The two methods resulted in very similar results with interesting differences. The Runge-Kutta method was simple to implement, required minimal memory to evaluate and a 4th order accuracy, while the 5th order Predictor Corrector Multi step method required far more code, initialization of the first four functional values through other, explicit multistep methods, more memory to run (storing the previous time steps functional evaluations) but gave a similar result at small enough spacing (10000 nodes). However, it is interesting to note that the predictor corrector multistep method, due to being of a higher order accuracy, was able to compute some values for the Lorenz attractor with 1000 nodes, while the Runge-Kutta method became divergent at 1000 nodes, with both running to completion in approximately the same amount of time.

While the higher order predictor corrector displayed some positive characteristics that the Runge-Kutta 4 method did not, the ease of coding and use of the Runge-Kutta 4 method makes it a better choice for most situations, at least initially.

Tables

|  |  |  |
| --- | --- | --- |
|  | | |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
| -3 | 1 | 0 |

|  |  |  |
| --- | --- | --- |
|  | | |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

|  |  |  |
| --- | --- | --- |
|  | | |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

|  |  |  |
| --- | --- | --- |
|  | | |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

Tables continued

|  |  |  |
| --- | --- | --- |
|  | | |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

|  |  |  |
| --- | --- | --- |
|  | | |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

|  |  |  |
| --- | --- | --- |
|  | | |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |