Numerical Solutions of First Order Differential Equations

Aleksandar Ivanov

November 18, 2020

1 Problem Statement

Solve the first order differential equation

$$\frac{\mathrm{d}T}{\mathrm{d}t} = -k(T - T^*), \tag{1}$$

which models simple cooling to an environment with temperature T^*, using a variety of numerical methods for solving first order differential equations (Euler method, Runge-Kutta method, Adams-Bashforth-Moulton method...). How small a step h is necessary? Choose a method (and step size) for calculating the solution for different values of k.

2 Equation Setup

As always, when dealing with physical problems using numerical methods, it's useful to nondimensionalize the problem. This helps us in three major ways. One is that we don't have to deal with units, which we don't store numerically, another reason is that our equation is simplified to its most essential dynamical components, and finally we generically have better numerical stability since our quantities stay closer to $\mathcal{O}(1)$ than they would otherwise.

For our equation we introduce

$$x = \frac{T - T^*}{T^*}, \qquad \tau = kt, \tag{2}$$

which cast out equation into the simple form

$$\dot{x} = -x, \tag{3}$$

where we have used a dot to denote differentiation with respect to τ.

Our equation obviously has the exact solution

$$x(t) = x_0 \exp(-\tau), \tag{4}$$

or in terms of T and t,

$$T(t) = T^* + (T_0 - T^*) \exp(-kt). \tag{5}$$

We will be comparing to this solution whenever we need to check the accuracy of our numerically calculated solutions.

3 Methods

The numerical methods we will be using will be:

1. The Euler method,
2. The Heun (modified Euler) method,
3. The Runge-Kutta method of order 4,
4. The Runge-Kutta-Fehlberg 4(5) method, and
5. The Adams-Bashforth-Moulton method of order 4.

All of these are explicit methods. One and three are all examples of Runge-Kutta type methods of different orders, two and five are examples of methods of the type predictor-corrector while 4 is an example of a method with self-correcting step size.

The predictor-corrector method initially needs the derivative at the first four points to get started, and we provide this by using the 4th order Runge-Kutta method.

4 Accuracy tests

To test the accuracy for the constant step size methods, we numerically calculate the solution of our equation and check its average deviation from the exact solution, for a variety of step sizes. We can do this in two ways; One is two fix the number of points and change the interval size while the

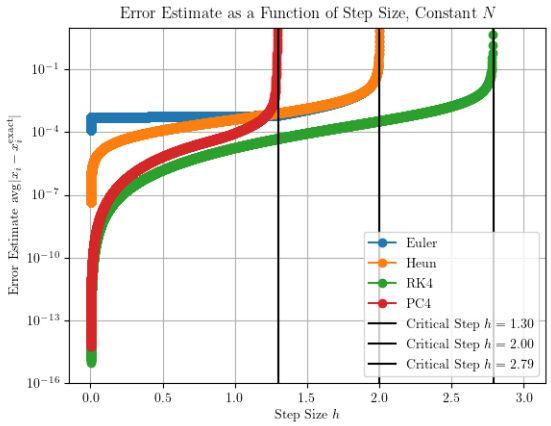

Figure 1: Error estimate of the different constant-step methods as a function of the step size h, keeping the number of points N constant.

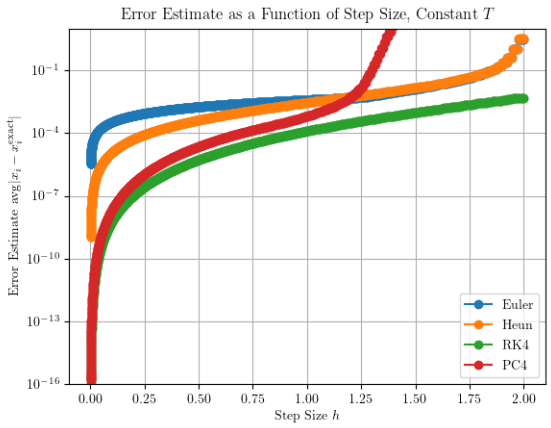

Figure 2: Error estimate of the different constant-step methods as a function of the step size h, keeping the interval size T constant.

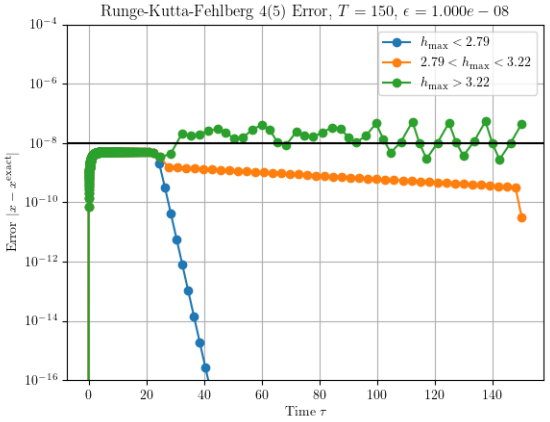

Figure 3: Error of the Runge-Kutta-Fehlberg 4(5) variable-step method as a function of time τ for the different stability regimes. The minimum step used was $h_{\min} = 10^{-6}$

other is to fix the interval size and change the number of points.

These two tests are shown in figs. 1 and 2, respectively, using the initial condition $x_0 = 1$. They show an overall similar behavior for all tested methods. As expected, the simple Euler method is the least accurate, followed by the second order Heun method and with Runge-Kutta of 4th order being the most accurate of the bunch [3]. The predictor-corrector method of 4th order was somewhat less accurate than RK4, but either way its main selling point is minimization of function calls, so it's done good considering.

We can also already see stability considerations sneaking in as extremely fast growth of our error, especially well delineated in the constant N fig. 1.

On the lower end of step sizes, on the other hand, we are, as always, limited by floats' double precision, but this is hard to pin only on the algorithms, since all of our comparisons and even the 'exact' solution also suffer from the same error.

The error is, of course, also dependent on the initial condition, since that is the value that sets the scale of our function. Tests with different initial conditions in the range of a couple of orders of magnitude around $\mathcal{O}(1)$ gave exactly the same behavior as the one shown in figs. 1 and 2, with the only difference being that the curves get shifted upwards vertically, since we're plotting an absolute

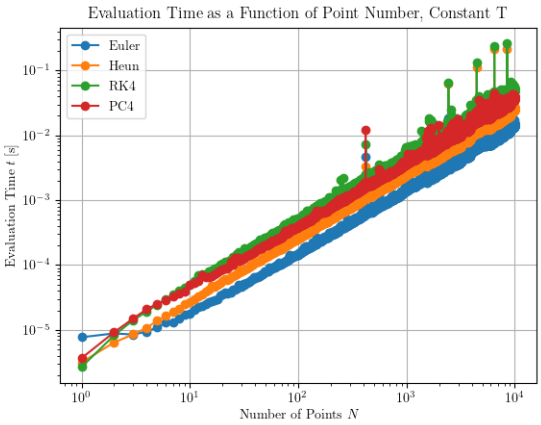

Figure 4: Evaluation time of the different constant-step methods as a function of the number of points N, keeping the interval size T constant.

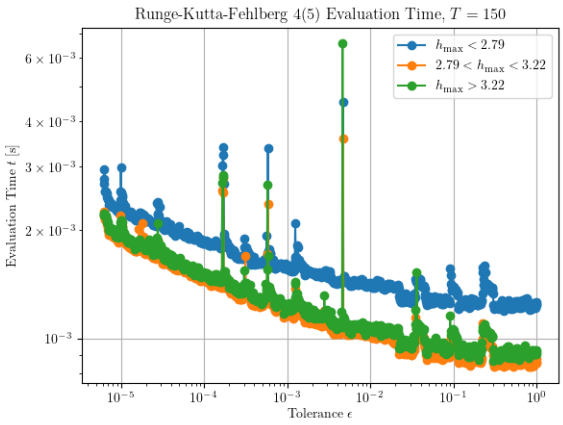

Figure 5: Evaluation time for the Runge-Kutta-Fehlberg 4(5) method as a function of the tolerance ϵ, for the different stability regimes.

error in a logarithmic scale.

For the variable-step method, we have fig. 3; it shows the error from the exact solution for a tolerance of $\epsilon = 10^{-8}$ and a time interval $T = 150$ as a function of the time τ. We have three regions depending on stability, since RK4 becomes unstable for $h > 2.79$ and RK5 becomes unstable for $h > 3.22$ [2]. We see that when the algorithm is stable it has no problem keeping under the tolerance. This is generically the case as long as the tolerance doesn't get too close to the floating point double precision. Surprisingly though, even in the cases when it should naively be unstable it still manages to keep its error near the tolerance.

When scaling the solution back to get the original, our error in T will of course be $\Delta T = T^* \Delta x$.

5 Speed Tests

The speed of evaluation for the constant-step methods depends mainly on the number of points for which we do the calculation; it's basically independent of interval size. Figure 4 shows this dependence. We see that the evaluation time scales similarly for all the methods, which is expected since at their core they all have the same array manipulations, the only difference being in the amount that they use. This gives them all an $\mathcal{O}(N)$ time complexity, as we can see from the figure. We also see that the Euler method, being the one with the least amount of calculations, is the fastest. The predictor-corrector 4th order method is also almost always faster than the Runge-Kutta 4th order method.

For the variable-step method it's sensible to see how the evaluation time depends on the tolerance ϵ. Exactly this is shown in fig. 5, which as expected says that the evaluation time is longer the smaller the tolerance is, and it's also longer for smaller maximal steps. We don't see any major change in evaluation time between the three stability regimes.

6 Stability

As we already saw in fig. 1, our algorithm are not stable for arbitrary step size. Here by stability we mean that we don't get arbitrarily large deviations between two solutions with initial conditions that are close to each other. Equivalently, it means that we don't get large deviations from the exact solution.

For the Euler and Runge-Kutta methods we can calculate the stable intervals theoretically, and we get $h \in (0, 2]$ for the Euler method, $h \in (0, \approx 2.79]$ for RK4 method, and $h \in (0, \approx 3.22]$ for the RK5 method [2]. This is exactly confirmed by fig. 1 for the RK4 and Euler cases. Figure 3 agrees with the

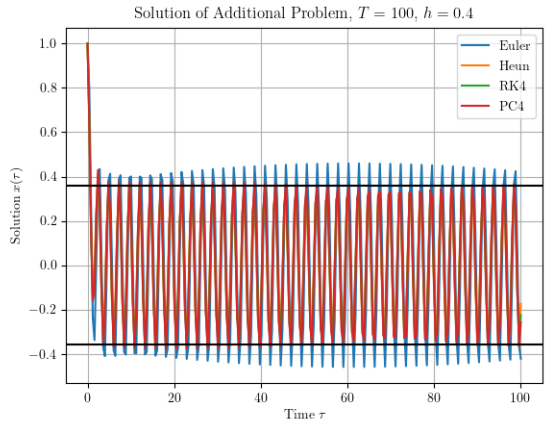

Figure 6: Solution using the constant-step methods and a relatively large step size $h = 0.4$ of eq. (8)

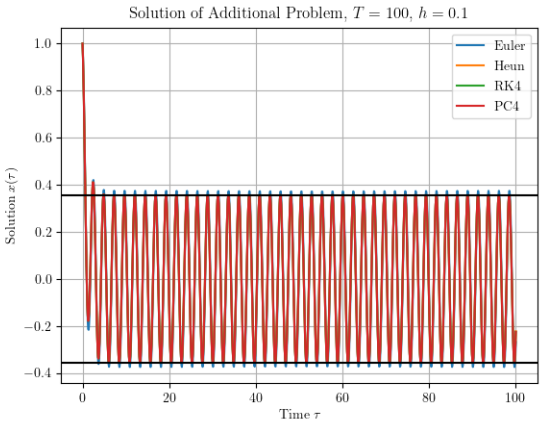

Figure 7: Solution using the constant-step methods and a relatively small step size $h = 0.1$ of eq. (8)

cutoff for RK5 being 3.22 since it displays different behavior above that, but it isn't an outright confirmation. It makes sense that the Heun method's stability cutoff is the same as the Euler method's because it uses the Euler method as its predictor.

The stability range of the Adams-Bashforth-Moulton predictor-corrector 4th order method is harder to calculate, but the result of fig. 1 agree with the theoretical value in [1].

7 Discussion

In trying to solve our cooling model, we have observed some of the benefits and drawbacks of particular algorithms for numerically solving differential equations. Of the methods we've covered, the Euler method is characterized by its simplicity, the 4th order Runge-Kutta by its stability and the Runge-Kutta-Fehlberg 4(5) by its accuracy.

We have also used the mathematical trick of nondimensionalization to compress our parameter space and bring the equations into a form with only their basic dynamical components. Had we not done this, the only difference would have been in the scale of the solution and a change in what parameter characterizes stability, namely the dimensionless kh_t, where h_t is the step size in the same units as t.

8 Additional Problem Statement

The temperature can also change because of the Sun's shining, which we model as

$$\frac{\mathrm{d}T}{\mathrm{d}t} = -k(T - T^*) + A\sin(\omega t + \delta), \qquad (6)$$

where k =0.1 /h, $\omega = 2\pi/(24\,\mathrm{h})$ and $\delta \approx 2.618$ is a phase offset. Find the solution to this equation. What kind of method would you use if you wanted to calculate the maximal temperatures and their occurrence times particularly well?

9 Additional Setup

We proceed again with nondimensionalization. This time we need to define the variables

$$x = (T - T^*)\frac{k}{A}, \qquad \tau = kt, \qquad \tilde{\omega} = \frac{\omega}{k}, \qquad (7)$$

which make out equation

$$\dot{x} = -x + \sin(\tilde{\omega}t + \delta). \qquad (8)$$

Here we have $\tilde{\omega} \approx 2.618$ by using the given data. We see that we have no free parameters in our equation, since we have fixed $\tilde{\omega}$ and δ; The scale of the

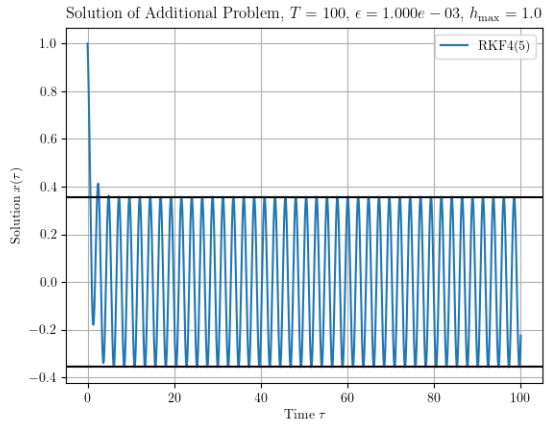

Figure 8: Solution of eq. (8) using the variable-step method with step size $h \in [10^{-3}, 10^{0}]$ and tolerance $\epsilon = 10^{-3}$.

problem is again determined only by the initial condition, and only for the decaying solution at that.

We can calculate the solution theoretically too, but the only part that will be important to us is the amplitude of the particular solution. This is because we want to see which method is the best for finding the maxima correctly. The particular solution's amplitude is given by

$$A' = \frac{1}{\sqrt{1+\tilde{\omega}^2}}. \tag{9}$$

Plotting the solutions of the different methods on the interval $\tau \in [0, 100]$ we get figs. 6 and 8. The former shows the solutions for a larger step size, while the latter shows them for a smaller step size. We see that the best methods to use are, predictably, the Adams-Bashforth-Moulton 4th order and the Runge-Kutta 4th order methods, since they don't need as large a step size to see the oscillations in the solution.

The variable-step method RKF4(5) also gives a very good answer in terms of accuracy, but it is much slower than the previous two because it needs to constantly change its step size, which means that it throws out a lot of the calculations it does.

References

[1] Robert R. Brown, James D. Riley, and Morris M. Bennett. "Stability Properties of Adams-Moulton Type Methods". In: *Mathematics of Computation* 19.89 (1965), pp. 90–96. ISSN: 00255718, 10886842. URL: `http://www.jstor.org/stable/2004101`.

[2] Jason Frank. *Stability of Runge-Kutta Methods*. [Last accessed: 18.11.2020]. 2008. URL: `https://webspace.science.uu.nl/~frank011/Classes/numwisk/ch10`.

[3] Brian D. Storey. *Numerical Methods for Differential Equations*. [Last accessed: 18.11.2020]. 2003. URL: `http://faculty.olin.edu/bstorey/Notes/DiffEq.pdf`.