**Numerical Analysis Assignment 3**

**Date:** May 9, 2018

Ruochen Mao  Hanbo Shao

In this assignment, we are given the ordinary differential equation $y' = \cos(2t) + \sin(3t)$, with initial condition $y(0) = 1$. We approximate the true solution of this differential equation with Euler's method, Heun2 and RK4 and compare how well these three methods approximate. The actual solution is $y(t) = \frac{1}{2}\sin(2t) - \frac{1}{3}\cos(3t) + \frac{4}{3}$ and is plotted below.

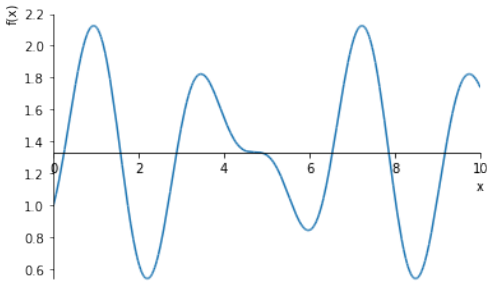


Figure 1: Plot of actual solution

| ti | yi | Euler error | Heun2 error | RK4 error |
|---|---|---|---|---|
| 00.00 | 1.0000000000 | 0.0000000000 | 0.0000000000 | 0.0000000000 |
| 00.50 | 1.7304897585 | 0.0143766108 | 0.0009316785 | 0.0000000691 |
| 01.00 | 2.1179795456 | 0.0302525217 | 0.0016231491 | 0.0000001325 |
| 01.50 | 1.4741586038 | 0.0733722242 | 0.0008158411 | 0.0000000734 |
| 02.00 | 0.6348753235 | 0.0486169620 | 0.0002904840 | 0.0000000108 |
| 02.50 | 0.7383260901 | 0.0055504418 | 0.0000088878 | 0.0000000217 |
| 03.00 | 1.4973356715 | 0.0103856813 | 0.0010784620 | 0.0000001072 |
| 03.50 | 1.8203389420 | 0.0269484910 | 0.0011963466 | 0.0000000979 |
| 04.00 | 1.5467278037 | 0.0415418946 | 0.0005099292 | 0.0000000263 |
| 04.50 | 1.3410856882 | 0.0272586316 | 0.0004250142 | 0.0000000309 |
| 05.00 | 1.3145520822 | 0.0288460879 | 0.0008735043 | 0.0000000937 |
| 05.50 | 1.0674705826 | 0.0420363258 | 0.0006476653 | 0.0000000825 |
| 06.00 | 0.8449413049 | 0.0226895598 | 0.0000112276 | 0.0000000106 |
| 06.50 | 1.2781451951 | 0.0131274286 | 0.0003027623 | 0.0000000193 |
| 07.00 | 2.0112134313 | 0.0007153370 | 0.0013805156 | 0.0000001080 |
| 07.50 | 1.9495788001 | 0.0547293075 | 0.0014422531 | 0.0000001211 |
| 08.00 | 1.0479886726 | 0.0713409026 | 0.0002400434 | 0.0000000288 |
| 08.50 | 0.5415295500 | 0.0232615800 | 0.0003589554 | 0.0000000128 |
| 09.00 | 1.0552193129 | 0.0159122422 | 0.0004949262 | 0.0000000627 |
| 09.50 | 1.7331536967 | 0.0045791999 | 0.0012970756 | 0.0000001184 |
| 10.00 | 1.7383888087 | 0.0385894906 | 0.0009092484 | 0.0000000654 |

Table 1: Comparison of Errors

We now apply three methods to approximate the true solution and results are summarized in the table above. We set the step size $h = 0.05$, i.e. 200 intervals, and then run three methods over the range $[0, 10]$. This table is a part of the full table where we chose certain error $t$ values, $t_i$. In addition, $y_i$ stands for the actual $y$ value at $t = t_i$. From the table, we see that at each $t = t_i$, RK4 method has the least error and Euler's method has the most error. This pattern appears throughout the full table if we compare errors of three methods. We can also observe this fact from the plots in the appendix where RK4 method fits the best and Euler's method fits the worst.

It is not hard to notice that this function is really "bendy". Euler's method, Heun2 and RK4 all belong to the family of Runge-Kutta methods, and they are all able to bend. As we decrease the value of $h$ to 0.02, we can see that these three methods fit the exact plot reasonably well. Also, as we decrease $h$ up to 0.00001, it appears that the three plots fit the exact plot perfectly. The advantage of RK4 method is its accuracy as RK4 method has an error of order $O(h^4)$. However, with more complicated algebraic manipulations of slopes, RK4 method's computation cost is larger. Vice versa, Euler's method is not as accurate, with an error of order $O(h)$, but it has less computation cost. Heun2 method's accuracy and computation cost is in between RK4 and Euler's method.

From above discussion, we know that RK4 method behaves better than the other two methods. We would like to see how much better RK4 method behaves when compared to the other two methods. In order to do this, we calculated the total error of the three RK methods. Dividing total error accordingly, we will be able to discover how many times better is one method than the other method. By plugging in different $h$ values, we find when $h = 0.00001$, the total error of Euler's method, Heun2 and RK4 are $7.896\times10^{-4}$, $3.636\times10^{-7}$, $1.266\times10^{-14}$, respectively. Strikingly, in terms of total error, RK4 method is $6.239 \times 10^{10}$ times better than the Euler's method and $2.872 \times 10^{7}$ times better than Heun2 method. In a nutshell, RK4 method brings a significant improvement (See table in appendix for more information). As we keep decreasing the $h$ value, nevertheless, we finally reach the computing limit when the total error of RK4 is less than $1.0 \times 10^{-13}$. This occurs when $h = 0.00002$. We can no longer reduce the total error of RK4 method after this point. However, we can still decrease $h$ value to find a better approximation given by Heun2 method.

**Extra Credit:**

We modified the RK4 method so that we can change the coefficients before slopes $K1$, $K2$, $K3$ and $K4$.

To start with, we compute the total error of the approximated list generated by RK4 method (i.e. the coefficients for slopes are 1, 2, 2, 1) and we call it minimum error. We set the tolerance to be 0.0001. If the resulting total error is less than the sum of minimum error and tolerance, we print it out. By looping all 4 of the coefficients of slopes from 1 to 5, we obtain the following table (coeff($K1$) stands for coefficient of $K1$).

Table 2: Different coeffcients for RK4 (tolerance = 0.00001)

| coeff(K1) | coeff(K2) | coeff(K3) | coeff(K4) | error |
|---|---|---|---|---|
| 1 | 1 | 3 | 1 | 0.00010148188579284234 |
| 1 | 2 | 2 | 1 | 0.00010148188576841743 |
| 1 | 3 | 1 | 1 | 0.00010148188578262829 |
| 2 | 3 | 5 | 2 | 0.00010148188576841743 |
| 2 | 4 | 4 | 2 | 0.00010148188576841743 |
| 2 | 5 | 3 | 2 | 0.00010148188576841743 |

We notice when the coefficients are 1, 2, 2, 1 and its multiples, we obtain the least error. Although coefficients 2, 5, 3, 2 and 2, 3, 5, 2 also behave nicely (close to the total error induced by 1, 2, 2, 1), they are more "complicated" to compute.

This similar pattern continues as we increase the range of numbers that coefficients of $K1, K2, K3, K4$ can loop over. (Details can be seen in table 2 of the appendix) Thus, we may conclude that 1, 2, 2, 1 is the best combination of coefficients for $K1, K2, K3, K4$ in order to minimize the total error.

# Appendix

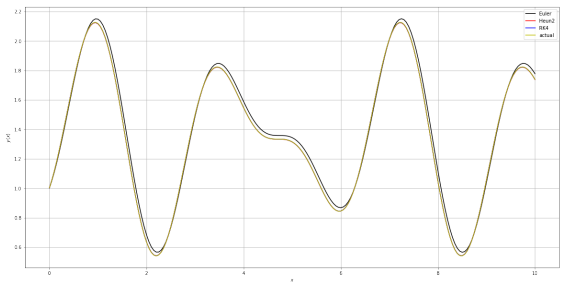

Figure 1: Plot of Approximations with $h = 0.05$

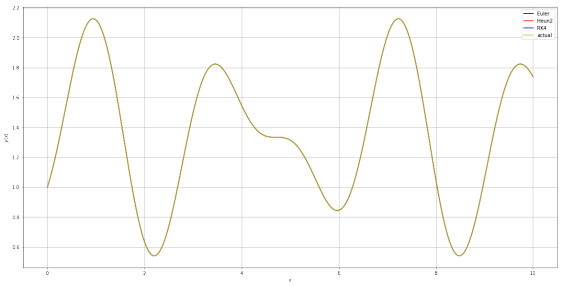

Figure 2: Plot of Approximations with $h = 0.00001$

With smaller step size, three approximation methods fit the exact solution much better. The following three plots visually demonstrate the fact that Euler's method is the least accurate and RK4 method is the most accurate. For these plots, $h = 0.25$.

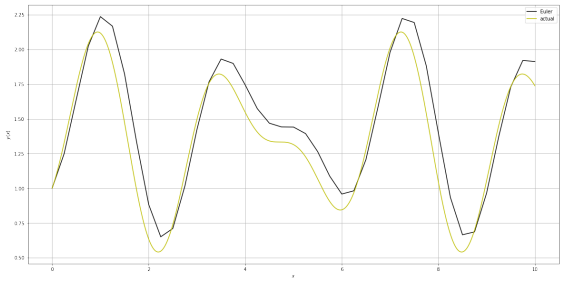

Figure 3: Euler's method

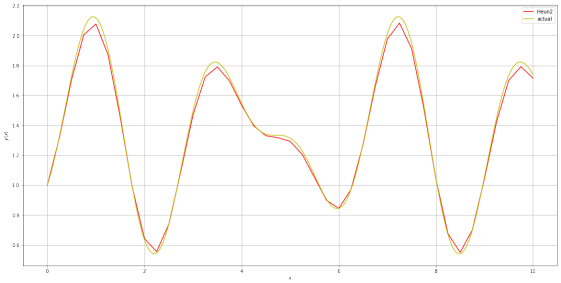

Figure 4: Heun2 Method

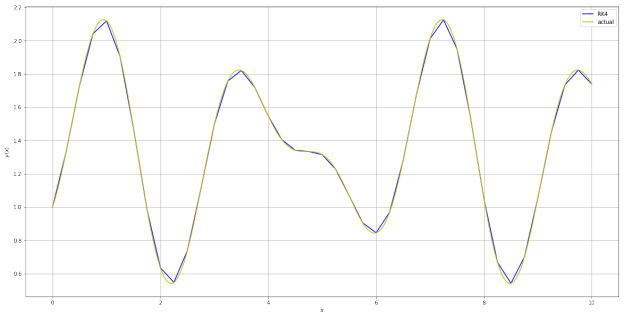

Figure 5: RK4 Method

The table below shows how much better is RK4 method when compared to Heun2 and Euler's method as well as the improvement of Heun2 over Euler's method. To create this

table, we first calculate the total error and divide them accordingly. In the table, Euler/RK4 means the number we obtain by dividing the total error of Euler's method by RK4 method and other terms are interpreted similarly.

| n | Euler/Heun2 | Euler/RK4 | Heun2/RK4 |
|---|---|---|---|
| 200 | $4.2 \times 10^1$ | $5.9 \times 10^5$ | $1.4 \times 10^4$ |
| 1000 | $2.2 \times 10^2$ | $7.5 \times 10^7$ | $3.5 \times 10^5$ |
| 2000 | $4.3 \times 10^2$ | $6.0 \times 10^8$ | $1.4 \times 10^6$ |
| 5000 | $1.1 \times 10^3$ | $9.9 \times 10^9$ | $9.1 \times 10^6$ |
| 10000 | $2.2 \times 10^3$ | $2.2 \times 10^{10}$ | $2.9 \times 10^7$ |
| 50000 | $1.1 \times 10^4$ | $8.3 \times 10^9$ | $7.6 \times 10^5$ |

Table 1: Comparison of Improvements

The table below shows the total error of different combinations of coefficients of RK4 method.

Table 2: Different coeffcients for RK4 (tolerance = 0.00001)

| coeff(K1) | coeff(K2) | coeff(K3) | coeff(K4) | error |
|---|---|---|---|---|
| 1 | 1 | 3 | 1 | 0.00010148188579284234 |
| 1 | 2 | 2 | 1 | 0.00010148188576841743 |
| 1 | 3 | 1 | 1 | 0.00010148188578262829 |
| 2 | 1 | 7 | 2 | 0.00010148188579284234 |
| 2 | 2 | 6 | 2 | 0.00010148188579284234 |
| 2 | 3 | 5 | 2 | 0.00010148188576841743 |
| 2 | 4 | 4 | 2 | 0.00010148188576841743 |
| 2 | 5 | 3 | 2 | 0.00010148188576841743 |
| 2 | 6 | 2 | 2 | 0.00010148188578262829 |
| 2 | 7 | 1 | 2 | 0.00010148188576863948 |
| 3 | 3 | 9 | 3 | 0.00010148188574043981 |
| 3 | 4 | 8 | 3 | 0.00010148188577352446 |
| 3 | 5 | 7 | 3 | 0.00010148188574909955 |
| 3 | 6 | 6 | 3 | 0.00010148188578129602 |
| 3 | 7 | 5 | 3 | 0.00010148188572400851 |
| 3 | 8 | 4 | 3 | 0.00010148188574066186 |
| 3 | 9 | 3 | 3 | 0.00010148188572400851 |
| 4 | 7 | 9 | 4 | 0.00010148188579284234 |
| 4 | 8 | 8 | 4 | 0.00010148188576841743 |
| 4 | 9 | 7 | 4 | 0.00010148188579284234 |