Numerical Methods for Differential Equations
Homework #1

Lorenzo Schiavone

December 7, 2024

Contents

1 Exercise 1

We implemented the 2-step Simpson's method

$$y_{n+2} = y_n + \frac{h}{3}(f_n + 4f_{n+1} + f_{n+2})$$

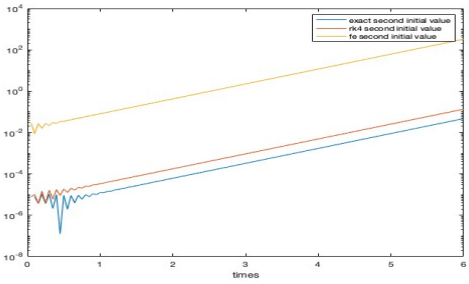
in MATLAB as follows.

```
function y_est = simpson_method(fun, h, T, y_0, y_1)
    %% no handle of different last h
    N = floor(T/h)+1; y_est = zeros(1,N); y_est(1) = y_0; y_est(2)=y_1;
    for n = 3:N
        y_est(n) = fzero(@(y) -y + y_est(n-2) + h/3 * (fun((n-2)*h, y_est(
            n-2)) + 4 * fun(h*(n-1), y_est(n-1)) + fun(h*n, y)), y_est(n-1)
            );
    end
end
```

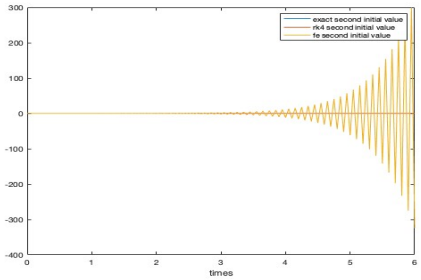
Then, after computing the second initial value y_1, the function *simpson_method* is called to solve the test equation $y' = -5y$, $y(0) = 1$ with $h = 0.05$ and $T = 6$ by

```
fun = @(t,y) -5 *y;
h = .06; T = 6; y_0 =1; y_1 = ...;
y_est = simpson_method(fun, h, T, y_0, y_1);
```

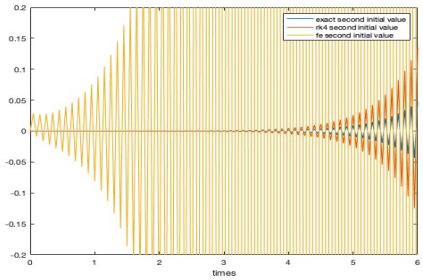
We compute the second initial value as the exact solution, by a 4th order Runge-Kutta method and with the Forward Euler Method. Figures 1(a), 1(c), 1(b) illustrate the error as the difference between the exact



(a)



(b)



(c)

Figure 1: Error with the exact solution for 2-step Simpson estimates with different second initial values. At the top a semilogarithmic plot of the absolute error, at the bottom right a zoomed plot for $-0.2 \leq y \leq 0.2$.

and the approximate solution. The semilogarithmic plot in Figure 1(a) shows that in all the three cases

the error increases exponentially in time with the same behavior, since the Simpson's method is nowhere absolutely stable. The initial error in the second initial value represents the starting point of the exponential growth but it is not propagated in time because the Simpson's method is 0-stable, as the roots of the first characteristic polynomial are -1, 1 which are allowed since they are simple.

The oscillatory and increasing nature of the error, due to the negative root of the first characteristic polynomial, can be observed in Figure 1(b). Since the error scale in the run with the second initial value computed using the Forward Euler Method is significantly higher than in the other case, we also provide a zoomed-in frame to highlight the oscillations in the error for the other case as well.

2 Exercise 2

We solve the equation

$$y'(t) = -10y^2(t), \quad y(0) = 1$$

with the 4-stage Runge-Kutta method and $h = 2^{-k}$, $T = 2$, $k = 5, \ldots, 10$. We implement the method as follow.

```
function y_est = runge_kutta4(fun, h, T, y_0)
    N = floor(T/h)+1;
    y_est = zeros(length(y_0),N);
    y_est(:,1) = y_0;
    for n = 2:N
        t = n*(h-1);
        k1 = fun(t, y_est(:,n-1));
        k2 = fun(t + h/2, y_est(:,n-1) +1/2 * h * k1);
        k3 = fun(t + h/2, y_est(:,n-1) +1/2 * h * k2);
        k4 = fun(t +h, y_est(:,n-1) + h*k3);
        y_est(:,n) = y_est(:,n-1) + 1/6 * h *(k1+ 2*k2 +2*k3 +k4);
    end
end
```

The error for decreasing step size is presented in Figure 2 and summarized in Table 1. As the step size decreases, so does the error. Moreover, the slope of the line in the log-log plot is approximately -3.92 consistent with the method's fourth-order accuracy $O(h^{-4})$.

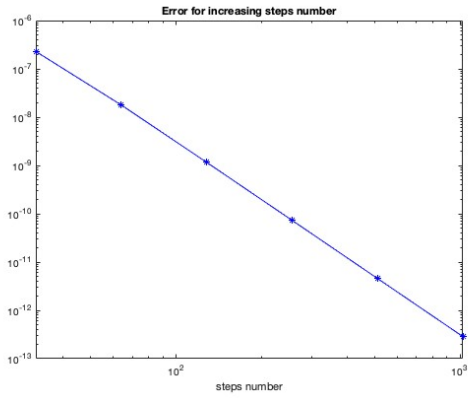


Figure 2: RK4 Error Reduction with smaller step size.

n° of steps	error
32	$2.29 \cdot 10^{-7}$
64	$1.78 \cdot 10^{-8}$
128	$1.16 \cdot 10^{-9}$
256	$7.31 \cdot 10^{-11}$
512	$4.58 \cdot 10^{-12}$
1024	$2.86 \cdot 10^{-13}$

Table 1: Final error with increasing number of steps.

3 Exercise 3

i) (Obtain the BDF2 formula) To obtain the BDF2 formula we need to interpolate (t_n, y_n), (t_{n+1}, y_{n+1}) and (t_{n+2}, y_{n+2}) with a degree two polynomial by using the Lagrange Polynomials, where $t_n = nh$ for some step size $h > 0$. The interpolating Lagrange polynomial is

$$L(t) = \sum_{j=0}^{2} y_{n+j} l_j(t), \quad \text{where } l_j(t) = \prod_{0 \le m \le 2, m \ne j} \frac{t - t_m}{t_j - t_m}.$$

Therefore, we can compute

$$\begin{aligned}
l_0(t) &= \frac{t-t_1}{t_0-t_1}\frac{t-t_2}{t_0-t_2} = \frac{1}{2h^2}(t-t_1)(t-t_2),\\
l_1(t) &= \frac{t-t_0}{t_1-t_0}\frac{t-t_2}{t_1-t_2} = -\frac{1}{h^2}(t-t_0)(t-t_2),\\
l_2(t) &= \frac{t-t_0}{t_2-t_0}\frac{t-t_1}{t_2-t_1} = \frac{1}{2h^2}(t-t_0)(t-t_1).
\end{aligned}$$

Then, we approximate $f(t_{n+2}, y_{n+2})$ with $L'(t)|_{t=t_2}$. Since

$$\begin{aligned}
l_0'(t)|_{t=t_2} &= \frac{1}{2h^2}[(t-t_1)+(t-t_2)]|_{t=t_2} = \frac{1}{2h^2}(t_2-t_1) = \frac{1}{2h^2}h = \frac{1}{2h},\\
l_1'(t)|_{t=t_2} &= -\frac{1}{h^2}[(t-t_0)+(t-t_2)]|_{t=t_2} = -\frac{1}{h^2}(t_2-t_0) = -\frac{1}{h^2}2h = -\frac{2}{h},\\
l_2'(t)|_{t=t_2} &= \frac{1}{2h^2}[(t-t_0)+(t-t_1)]|_{t=t_2} = \frac{1}{2h^2}[(t_2-t_0)+(t_2-t_1)] = \frac{1}{2h^2}3h = \frac{3}{2h},
\end{aligned}$$

wo obtain

$$f(t_{n+2}, y_{n+2}) \approx L'(t)|_{t=t_{n+2}} = \frac{1}{2h}y_n - \frac{2}{h}y_{n+1} + \frac{3}{2h}y_{n+2} = \frac{1}{2h}(y_n - 4y_{n+1} + 3y_{n+2}).$$

The 2-step Backward Differentiation Formula, after rearranging it, is

$$y_{n+2} = \frac{4}{3}y_{n+1} - \frac{1}{3}y_n + \frac{2}{3}hf(t_{n+2}, y_{n+2}).$$

ii) (Local truncation error) We want to compute the local truncation error at each step while approximating the derivative $y'(t_{n+2}) = f(t_{n+2}, y(t_{n+2}))$, i.e.,

$$\begin{aligned}
\tau_{n+2} &= f(t_{n+2}, y(t_{n+2})) - L'(t_{n+2}).\\
&= f(t_{n+2}, y(t_{n+2})) - \frac{1}{2h}(y_n - 4y_{n+1} + 3y_{n+2}).
\end{aligned}$$

Taylor expansion up to order 3 of $y(t)$ gives

$$\begin{aligned}
y(t_n) = y(t_{n+2} - 2h) &= y_{n+2} - 2hf_{n+2} + \frac{(2h)^2}{2}y''_{n+2} - \frac{(2h)^3}{6}y'''(\eta)\\
y(t_{n+1}) = y(t_{n+2} - h) &= y_{n+2} - hf_{n+2} + \frac{h^2}{2}y''_{n+2} - \frac{h^3}{6}y'''(\xi)
\end{aligned}$$

for some $\eta \in [t_n, t_{n+2}]$ and $\xi \in [t_n, t_{n+1}]$. Thus

$$\begin{aligned}
\tau_{n+2} &= f_{n+2} - \frac{1}{2h}(y_n - 4y_{n+1} + 3y_{n+2})\\
&= f_{n+2} - \frac{1}{2h}\left(y_{n+2} - 2hf_{n+2} + \frac{4h^2}{2}y''_{n+2} - \frac{8h^3}{6}y'''(\eta) - 4(y_{n+2} - hf_{n+2} + \frac{h^2}{2}y''_{n+2} - \frac{h^3}{6}y'''(\xi)) + 3y_{n+2}\right)\\
&= f_{n+2} - \frac{1}{2h}\left(2hf_{n+2} - \frac{h^3}{6}(8y'''(\eta) - 4y'''(\xi))\right) = \frac{h^2}{12}(8y'''(\eta) - 4y'''(\xi)) = \frac{h^2}{3}(2y'''(\eta) - y'''(\xi)).
\end{aligned}$$

If we assume $y'''(\xi) \approx y'''(\eta)$ then the truncation error is

$$\tau_{n+2} = \frac{h^2}{3}(2y'''(\eta) - y'''(\xi)) \approx \frac{h^2}{3}|y'''(\eta)| \leq \frac{h^2}{3}\|y'''\|_\infty.$$

iii) (Absolute stability for $\overline{h} < 0$) We now want to determine that BDF2 is absolutely stable in the real interval $\overline{h} \in (-\infty, 0)$. The method applied to the test equation

$$y' = \lambda y, \quad y(0) = 1$$

for $\lambda < 0$, gives

$$y_{n+2} - \frac{4}{3}y_{n+1} + \frac{1}{3}y_n = \frac{2}{3}h\lambda y_{n+2}$$

and the recurrence relation

$$(1 - \frac{2}{3}h\lambda)y_{n+2} = \frac{1}{3}(4y_{n+1} - y_n).$$

By multiplying both members by 3 and seeking a solution of the form of $y_n = z^n$, we obtain:

$$p(z) := z^2(3 - 2\overline{h}) - 4z + 1 = 0,$$

where $\overline{h} = h\lambda$ and we have divided by z^n as it cannot be zero, otherwise the solution would not satisfy the initial condition. For absolutely stability we need that the roots of $p(z)$,

$$z_\pm = \frac{2 \pm \sqrt{4 + 2\overline{h} - 3}}{3 - 2\overline{h}} = \frac{2 \pm \sqrt{1 + 2\overline{h}}}{3 - 2\overline{h}},$$

are in the unit circle and if they are in the border they have multiplicity one. There is a double solution only if $\overline{h} = -1/2$. In that case $|z| = 2/|3 + 1| = 1/2 < 1$ that implies absolute stability.

For $1 + 2\overline{h} < 0$, i.e., $\overline{h} < -1/2$, the roots are complex conjugate and their norm squared is

$$z_+ \cdot z_- = \frac{4 - (1 + 2\overline{h})}{(3 - 2\overline{h})^2} = \frac{1}{|3 - 2\overline{h}|}$$

and is less than one if and only if $|3 - 2\overline{h}| > 1$ that is always true for $\overline{h} < 1/2$, as we have that $3 - 2\overline{h} > 3 - 1 = 2$.

For $-1/2 < \overline{h} < 0$, the roots are real. Also in this case, the denominator is always positive as $-\overline{h} > 0$ implies $3 - 2\overline{h} > 3$. Then $|z_\pm| < 1$ holds if and only if

$$-(3 - 2\overline{h}) < 2 \pm \sqrt{1 + 2\overline{h}} < 3 - 2\overline{h}$$

that can be rewritten as

$$-5 + 2\overline{h} < \pm\sqrt{1 + 2\overline{h}} < 1 - 2\overline{h}.$$

We have one set of inequality for z_+ and one for z_-. For z_+, we have that $\sqrt{1 + 2\overline{h}} > -5 + 2\overline{h}$ because $-5 + 2\overline{h} < 0$ for $h < 2/5$ that is the case as $\overline{h} < 0$. For the other inequality we can square both member because $3 - 2\overline{h} > 0$ as this holds when $\overline{h} < 3/2$, and we obtain $1 + 2\overline{h} < 1 - 4\overline{h} + 4\overline{h}^2$ that is $4\overline{h}^2 - 6\overline{h} > 0$, i.e., as $\overline{h} < 0$, $\overline{h} < 3/2$ that is always true for $\overline{h} < 0$.

For z_-, the inequalities are $2\overline{h} - 1 < \sqrt{1 + 2\overline{h}} < 5 - 2\overline{h}$. Again, $2\overline{h} - 1 < 0$ when $\overline{h} < 1/2$ and therefore this inequality is always verified for $h < 0$. For the right hand side we can square both member as $5 - 2\overline{h} > 0$ when $\overline{h} < 5/2$, and we obtain $1 + 2\overline{h} < 25 - 20\overline{h} + 4\overline{h}^2$ that is $4\overline{h}^2 - 22\overline{h} + 24 > 0$ or $2\overline{h}^2 - 11\overline{h} + 12 > 0$. It represents a parabola with positive concavity intersecting the x-axis in $\frac{11 \pm 5}{4}$, i.e., for $\overline{h} = 3/2$ and $\overline{h} = 4$. Therefore $2\overline{h}^2 - 11\overline{h} + 12 > 0$ for $\overline{h} > 4$ and $\overline{h} < 3/2$ that includes $-1/2 < \overline{h} < 0$.

4 Exercise 4

We want to solve the linear system of ODEs

$$\mathbf{y}'(t) = -A\mathbf{y}$$
$$\mathbf{y}(0) = \mathbf{y_0} = (1, 1, \dots, 1)^T$$

for $t \in (0, T]$, $T = 0.1$ and $A = \texttt{delsq(numgrid('S'), 60)} * (60-1)^2$ the 5-points Finite Difference discretization of the Laplacian. The *exact* solution to this problem at the final time is

$$\exp(-TA)\mathbf{y_0}.$$

We want to determine the interval of absolute stability of the 4 stage Runge-Kutta method for this problem: the method applied to the problem yields

$$\mathbf{y}_{n+1} = (I + hB + \frac{1}{2}h^2B^2 + \frac{1}{6}h^3B^3 + \frac{1}{24}h^4B^4)\mathbf{y}_n,$$

for $B = -A$. As B is symmetric, it is orthogonally diagonalizable, i.e., there exists an orthogonal matrix V such that $B = V\Lambda V'$ and $\Lambda = \text{diag}(\lambda_1, \dots, \lambda_N)$ contains in the diagonal the eigenvalues of B. With the change of variable $z = V'y$, the system is decoupled

$$\begin{cases} z_{n+1}^{(1)} = (1 + h\lambda_1 + \frac{1}{2}h^2\lambda_1^2 + \frac{1}{6}h^3\lambda_1^3 + \frac{1}{24}h^4\lambda_1^4)z_n^{(1)} \\ \dots \\ z_{n+1}^{(N)} = (1 + h\lambda_N + \frac{1}{2}h^2\lambda_N^2 + \frac{1}{6}h^3\lambda_N^3 + \frac{1}{24}h^4\lambda_N^4)z_n^{(N)}. \end{cases}$$

Therefore, the method is absolutely stable if and only if every equation is stable, i.e., if and only if

$$|(1 + h\lambda_i + \frac{1}{2}h^2\lambda_i^2 + \frac{1}{6}h^3\lambda_i^3 + \frac{1}{24}h^4\lambda_i^4)| \leq 1$$

for every $i = 1, \dots, N$. We compute the highest norm of real $\overline{h}$ such that $|(1 + \overline{h} + \frac{1}{2}\overline{h}^2 + \frac{1}{6}\overline{h}^3 + \frac{1}{24}\overline{h}^4)| \leq 1$

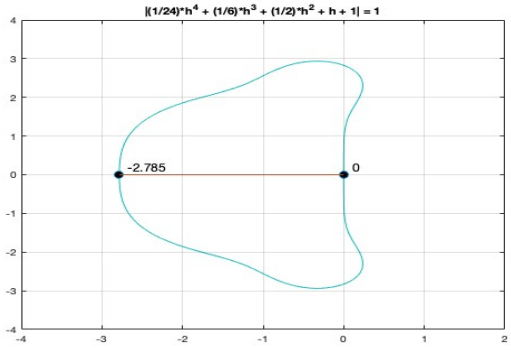


Figure 3: Region of absolutely stability of Runge Kutta 4 method.

and it is equal to 2.78. Therefore, we need

$$|h| < \frac{2.78}{|\lambda_i|}$$

for all $i = 1, \dots, N$ and in particular $|h| < 2.78 \cdot \max|\lambda_i|^{-1}$. The MATLAB function `eigs(A, 1, 'lm')` provides $\max|\lambda_i| = 27828$ so that the interval of absolute stability is $0 < h < 1.001 \cdot 10^{-4}$.

To solve numerically the problem, we make use of the `ode45` MATLAB function, the Crank Nicolson method (CN) and the Backward Differentiation Formula (BDF3) with step size $h \in \{10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}\}$.

Method	**n° of steps**	**CPU time (in sec)**	**infinity norm final error**
ODE45	3353	0.729	$2.16 \cdot 10^{-5}$
CN	10	0.032	$4.58 \cdot 10^{-1}$
CN	100	0.139	$1.40 \cdot 10^{-5}$
CN	1000	1.460	$1.40 \cdot 10^{-7}$
CN	10000	8.897	$1.40 \cdot 10^{-9}$
BDF3	10	0.159	$7.96 \cdot 10^{-4}$
BDF3	100	0.198	$7.06 \cdot 10^{-7}$
BDF3	1000	1.326	$7.16 \cdot 10^{-10}$
BDF3	10000	9.602	$4.61 \cdot 10^{-13}$

Table 2: number of steps, CPU time and error between the exact and the approximate solution at the final time $T = 0.1$ for the different runs.

For each run, we record the number of steps employed by the method, the CPU time, and the infinity norm of the difference between the exact and the approximate solution at the final time $T = 0.1$. The results are reported in Table 2.

For efficiency, every linear system is solved using MATLAB's `pcg` function with a tolerance `tol` $= h^{-3}$, and preconditioned by the incomplete Cholesky factor with drop tolerance `droptol` $= 10^{-2}$.

The MATLAB `ode45` function takes a large number of steps and achieves a small infinity norm final error in a short computational time because each iteration of the method is computationally cheap and the step sizes are adaptive.

For the Crank-Nicolson method the final error decreases by a factor of 10^2 whenever the step size h is reduced by a factor of 10, in accordance with the theoretical 2 accuracy order of Crank-Nicolson, except for $h = 0.01$ which is not representative of the asymptotic behavior.

In the case of the BDF3 method, the first two initial values are computed using a four stage Runge-Kutta method with a step size of $\min(h/2, 10^{-4})$, which lies within the stability interval for the RK4. The final error decreases by a factor of 10^3 whenever the step size h is reduced by a factor of 10, consistent with the theoretical 3 accuracy order of BDF3.

Given this setup, the most efficient method to achieve a given accuracy is BDF3 because it has the best trade-off between CPUtime and accuracy.

5 Exercise 5

Figure 4 shows the numerical solution of the Lokta-Volterra prey-predator model by the 4 stage Runge-Kutta method with parameters $\alpha = 0.2$, $\beta = 0.01$, $\gamma = 0.004$, $\delta = 0.07$, step size $h = 10^{-3}$, final time $T = 300$ and initial condition $x(0) = 19$, $y(0) = 22$.

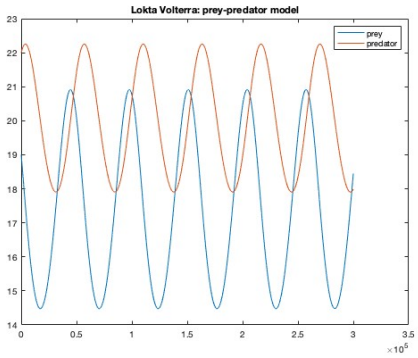


Figure 4: Prey Predator trajectory in time.

The plot shows the typical oscillatory pattern: when the prey population decreases, food becomes scarce for predators provoking a reduction in their numbers. As predator number falls, preys have more chances to grow causing a rise in their population. This causes a rebound in predator numbers, continuing the cycle with a time-shifted relation between the two groups.

6 Exercise 6

We want to compare fixed and adaptive time steps while solving the problem

$$y'(t) = -10y^2(t)$$
$$y(0) = 1$$

with the Crank Nicolson method and final time $T = 100$. Starting with $h = 10^{-3}$, we use the following adaptive step procedure with $\text{tol} = 10^{-8}$.

At each step we compute the difference, we denote *err*, between the CN estimate with time step h and the 2-stage estimate made by two consecutive CN steps with time step $h/2$. Then, the next time step will be

$$h_{\text{new}} = h\left(\frac{3}{4}\frac{\text{tol}}{\text{err}}\right)^{1/3}. \tag{1}$$

This follows from the fact that the error with a step size h is $y_{\text{exact}} - y_1^{(0)} = C_0 h^3$ and, when computing an intermediate stage, $y_{\text{exact}} - y_1^{(1)} = (C_1 + C_2)(h/2)^3$ for some constants C_1, C_2. If we assume $C_0 = C_1 = C_2$, then

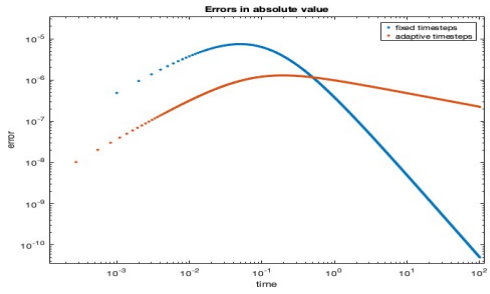
$$\text{err} = y_1^{(0)} - y_1^{(1)} = h^3(C - \frac{2}{8}C) = \frac{3}{4}Ch^3$$

gives an estimate $C \approx \frac{4}{3h^3}\text{err}$. Thus, with a step size of qh the error is

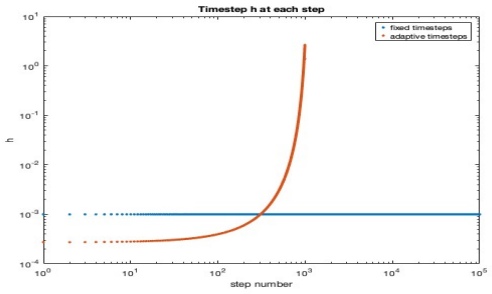
$$y_{\text{exact}} - y_1^{(q)} = C(qh)^3 \approx \frac{4}{3h^3}\text{err}(qh)^3 = \frac{4}{3}q^3\text{err}$$

that is a desired tolerance if $q = (\frac{3}{4}\frac{\text{tol}}{\text{err}})^{1/3}$ that is the ratio h_{new}/h in Equation 1.

The absolute error at each step is depicted in Figure 5(a) and the step size at different time steps in Figure 5(b).



(a) LogLogPlot of Absolute Value Error.



(b) Step size at different steps.

Even though at the beginning the adaptive method chooses smaller step size because the function is steep, when the function becomes flatter the adaptive method uses larger and larger step size, up to $h = 2.64$, and reaches the final time in less than 10^3 steps without sacrificing accuracy, much less than the 10^5 steps needed with fixed step size.

Therefore, using adaptive step size is more efficient than using fixed step size: indeed, the most costly operation is the call to the MATLAB function `fzero` and with fixed time step we call it 10^5 times while with the adaptive time step only $4 \cdot 10^3$, three for the estimate of err and one for the actual estimate of y with h_{new}, that is much less than 10^5. The CPUtime in fact are in this case 0.12s for adaptive steps compared to 0.76s with fixed steps.