Math 5604 Homework 1

Jacob Hauck

February 1, 2024

Problem 1.

Consider the IVP

$$y' = 3 + e^{-t} - y, \quad t > 0; \qquad y(0) = 1. \tag{1}$$

The code to answer the following questions is found in `problem1_calculations.m`. The output from running this script is copied into `problem1_output.txt`.

1.1) Multiplying both sides by the integrating factor e^t gives

$$y'e^t + ye^t = 3e^t + 1. \tag{2}$$

The left-hand side is $(ye^t)'$, so integrating on both sides gives

$$ye^t = 3e^t + t + C, \tag{3}$$

for some constant C, so $y(t) = 3 + (t + C)e^{-t}$. The initial condition $y(0) = 1$ implies that $C = -2$, so

$$y(t) = 3 + (t - 2)e^{-t}. \tag{4}$$

1.2) (**a**) To discretize the IVP on $[0, 2]$ using the forward Euler method, we need to have an evenly-spaced set of time samples $\{t_i\}_{i=0}^n$ defined by

$$t_i = \begin{cases} 0 & i = 0 \\ t_{i-1} + k, & i \geq 1, \end{cases}, \qquad i = 0, 1, \ldots, n. \tag{5}$$

The value k is the step size and is chosen so that $t_n = 2$; that is, $k = \frac{2}{n}$. We will attempt to find an approximation $\{y_i\}_{i=0}^n$ of the values $\{y(t_i)\}_{i=0}^n$. To find $\{y_i\}$, we create and solve a system of equations from the ODE by approximating $y'(t_i)$ by the forward difference $y'(t_i) \approx \frac{y(t_{i+1}) - y(t_i)}{k}$, where $i < n$. Since we know that $y(0) = 1$ from the initial condition, we are led to the scheme

$$\begin{cases} y_0 = 1 \\ \frac{y_{i+1} - y_i}{k} = 3 + e^{-t_i} - y_i, & 0 \leq i < n, \end{cases} \tag{6}$$

which allows to write an explicit recursive formula for y_i:

$$\begin{cases} y_0 = 1 \\ y_{i+1} = y_i + k(3 + e^{-t_i} - y_i), & 0 \leq i < n. \end{cases} \tag{7}$$

The code for this scheme can be found in `problem1_fe.m`.

(**b**) According the output from `problem1_output.txt`, the numerical value of $y(2)$ is 3.012754.

(**c**) Below (Figure 1) is the plot generated by `problem1_calculations.m`.

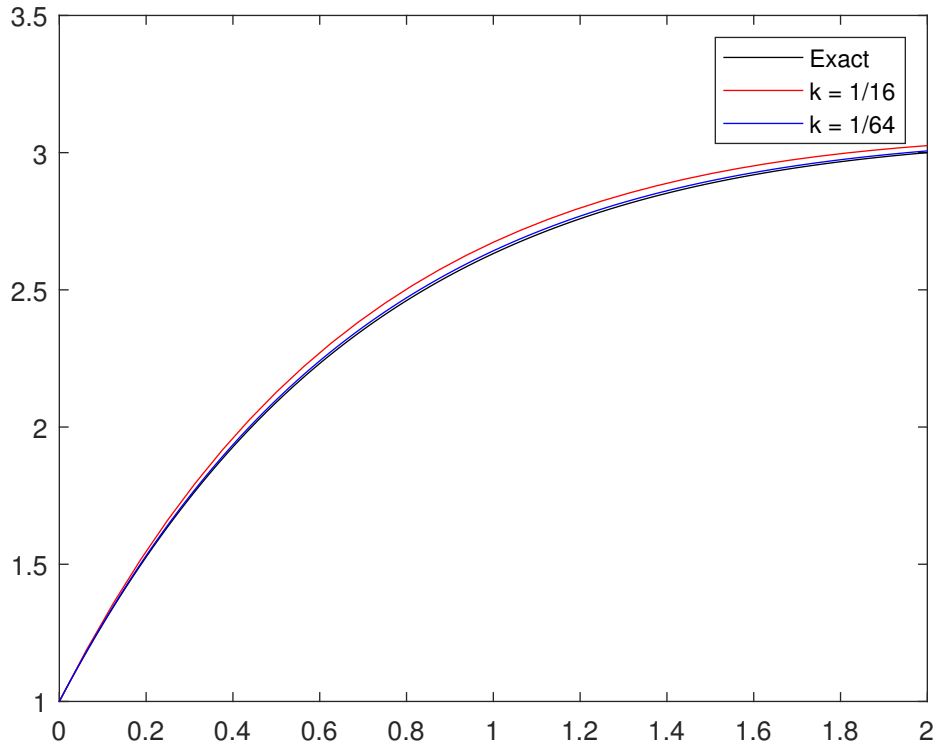

Figure 1: The exact solution of the ODE and the numerical approximations using the forward Euler method with $k = \frac{1}{16}$ and $k = \frac{1}{64}$.

1.3) (**a**) To discretize the IVP on $[0, 2]$ using the backward Euler method, we need to have an evenly-spaced set of time samples $\{t_i\}_{i=0}^n$ defined by

$$t_i = \begin{cases} 0 & i = 0 \\ t_{i-1} + k, & i \geq 1, \end{cases}, \qquad i = 0, 1, \ldots, n. \tag{8}$$

The value k is the step size and is chosen so that $t_n = 2$; that is, $k = \frac{2}{n}$. We will attempt to find an approximation $\{y_i\}_{i=0}^n$ of the values $\{y(t_i)\}_{i=0}^n$. To find $\{y_i\}$, we create and solve a system of equations from the ODE by approximating $y'(t_i)$ by the backward difference $y'(t_i) \approx \frac{y(t_i) - y(t_{i-1})}{k}$, where $i > 0$. Since we know that $y(0) = 1$ from the initial condition, we are led to the scheme

$$\begin{cases} y_0 = 1 \\ \frac{y_i - y_{i-1}}{k} = 3 + e^{-t_i} - y_i, & 0 < i \leq n, \end{cases} \tag{9}$$

which allows to write an explicit recursive formula for y_i:

$$\begin{cases} y_0 = 1 \\ y_i = \frac{y_{i-1} + k(3 + e^{-t_i})}{1 + k}, & 0 < i \leq n. \end{cases} \tag{10}$$

The code for this scheme can be found in `problem1_be.m`.

(**b**) According the output from `problem1_output.txt`, the numerical value of $y(2)$ is 2.987379.

(**c**) Below (Figure 2) is the plot generated by `problem1_calculations.m`.

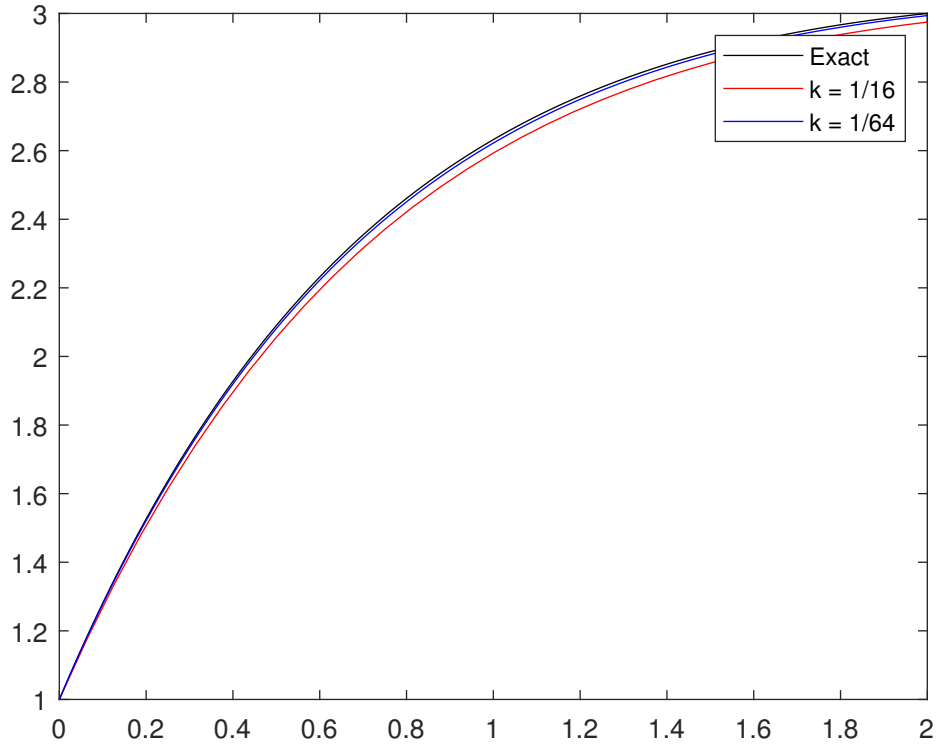

Figure 2: The exact solution of the ODE and the numerical approximations using the forward Euler method with $k = \frac{1}{16}$ and $k = \frac{1}{64}$.

1.4) Below (Table 1) are numerical errors at $t = 1$ for the forward and backward Euler methods at different step sizes. I make the following observations about the errors.

- The error is roughly proportional to h; every time h decreases by a factor 2, the error decreases roughly by a factor of 2 as well.
- The error for both the forward and backward Euler methods is roughly the same.

Time step (k)	Forward Euler error	Backward Euler error
1/4	0.105516	0.097216
1/8	0.051788	0.049683
1/16	0.025638	0.025109
1/32	0.012754	0.012621
1/64	0.006360	0.006327
1/128	0.003176	0.003168
1/256	0.001587	0.001585
1/512	0.000793	0.000793

Table 1: Numerical errors in $t = 2$; comparison of forward and backward Euler methods

Problem 2.

Consider the IVP

$$y' = \frac{3t^2 + 10t + 1}{2(y+1)}, \quad t > 0; \qquad y(0) = -2. \tag{11}$$

The code to answer the parts of this question can be found in `problem2_calculations.m`, and the output from running this script has been copied to `problem2_output.txt`.

2.1) Multiplying both sides by $2(y+1)$ gives

$$2(y+1)(y+1)' = 3t^2 + 10t + 1. \tag{12}$$

The left-hand side is $\left((y+1)^2\right)'$, so integrating on both sides gives

$$(y+1)^2 = t^3 + 5t^2 + t + C \tag{13}$$

for some constant C. The initial condition $y(0) = -2$ implies that $C = 1$. Therefore,

$$y(t) = -1 \pm \sqrt{t^3 + 5t^2 + t + 1}. \tag{14}$$

The initial condition forces us to choose a negative sign after taking the square root; thus,

$$y(t) = -1 - \sqrt{t^3 + 5t^2 + t + 1}. \tag{15}$$

2.2) To discretize the IVP on $[0, 1]$ using the backward Euler method, we need to have an evenly-spaced set of time samples $\{t_i\}_{i=0}^n$ defined by

$$t_i = \begin{cases} 0 & i = 0 \\ t_{i-1} + k, & i \geq 1, \end{cases}, \qquad i = 0, 1, \ldots, n. \tag{16}$$

The value k is the step size and is chosen so that $t_n = 1$; that is, $k = \frac{1}{n}$. We will attempt to find an approximation $\{y_i\}_{i=0}^n$ of the values $\{y(t_i)\}_{i=0}^n$. To find $\{y_i\}$, we create and solve a system of equations from the ODE by approximating $y'(t_i)$ by the backward difference $y'(t_i) \approx \frac{y(t_i) - y(t_{i-1})}{k}$, where $i > 0$. Since we know that $y(0) = 1$ from the initial condition, we are led to the scheme

$$\begin{cases} y_0 = 1 \\ \frac{y_i - y_{i-1}}{k} = \frac{3t_i^2 + 10t_i + 1}{2(y_i + 1)}, & 0 < i \leq n, \end{cases} \tag{17}$$

which allows to write an implicit recursive formula for y_i:

$$\begin{cases} y_0 = 1 \\ 2(y_i + 1)(y_i - y_{i-1}) - k(3t_i^2 + 10t_i + 1) = 0, & 0 < i \leq n. \end{cases} \tag{18}$$

We can solve the implicit equation for y_i numerically using Newton's method. Indeed, if we set

$$f_i(y) = 2(y+1)(y - y_{i-1}) - k(3t_i^2 + 10t_i + 1), \qquad 0 < i \leq n, \tag{19}$$

then finding y_i is equivalent to finding the root of f_i. Newton's method is easy to apply once we note that $f_i'(y) = 2(y - y_{i-1}) + 2(y+1)$.

The code for this scheme can be found in `problem2_be.m`, and it refers to the `newton.m` script to run Newton's method.

2.3) When $k = \frac{1}{16}$ and the numerical tolerance for Newton's method is $\varepsilon = 0.1$, the numerical value of $y(1)$ is -3.812471, according to `problem2_output.txt`.

2.4) When $k = \frac{1}{16}$ and the numerical tolerance for Newton's method is $\varepsilon = 10^{-8}$, the numerical value of $y(1)$ is -3.772976, according to `problem2_output.txt`.

2.5) Below (Table 2) are the numerical errors at $t = 2$ for different step sizes and Newton's method tolerances. I make the following observations about the errors.

- In every case, the error is roughly proportional to h; every time h decreases by a factor of 2, the error also decreases by roughly a factor of 2.
- The error is not very sensitive to the value of ε. It changes by almost nothing between $\varepsilon = 10^{-3}$ and $\varepsilon = 10^{-8}$.
- The error with $\varepsilon = 0.1$ is very slightly less. In the backward Euler method, there are two sources of error: the error between the discrete scheme and the true solution, and the error between the numerical solution of the discrete scheme and the true solution of the discrete scheme. As $h \to 0$, the error between the numerical scheme and the true solution should approach 0, and as $\varepsilon \to 0$ the error between the Newton's method approximation and the true discrete scheme should approach 0, but the two errors are not strictly additive. In this case, by chance, the error in Newton's method is actually in our favor, but this should not be expected in general.

Time step (k)	$\varepsilon = 0.1$ error	$\varepsilon = 10^{-3}$ error	$\varepsilon = 10^{-8}$ error
1/4	0.414940	0.417989	0.417989
1/8	0.211888	0.215835	0.215836
1/16	0.101119	0.109862	0.109862
1/32	0.015956	0.055450	0.055451
1/64	0.007944	0.027855	0.027860
1/128	0.003964	0.013964	0.013964
1/256	0.001980	0.006991	0.006991
1/512	0.000989	0.003497	0.003498

Table 2: Numerical errors at $t = 1$