Pen and paper

1. Consider the following approximation to the initial value problem $u' = F(t,u),\ u(t_0) = u_0$:

$$\frac{u_{k+1} - u_k}{\Delta t} = c_1 F(t_k, u_k) + c_2 F(t_{k+1}, u_{k+1}), \tag{1}$$

   (a) Let $\tilde{u}_{k+1}$ be the value of u at timestep t_{k+1} updated by (1) with the exact value $u(t_k)$. In other words, $\tilde{u}_{k+1}$ satisfies

$$\frac{\tilde{u}_{k+1} - u(t_k)}{\Delta t} = c_1 F(t_k, u(t_k)) + c_2 F(t_{k+1}, \tilde{u}_{k+1}). \tag{2}$$

   Show first $\tilde{u}_{k+1} - u(t_k)$ is at least of first order and then

$$\tilde{u}_{k+1} - u(t_k) = (c_1 + c_2) F \Delta t + c_2 \left[F_t + (c_1 + c_2) F F_u\right] \Delta t^2 + O(\Delta t^3), \tag{3}$$

   where $F = F(t_k, u(t_k))$, $F_t = F_t(t_k, u(t_k))$, and $F_u = F_u(t_k, u(t_k))$.

   (b) Show that

$$u(t_{k+1}) - u(t_k) = F\Delta t + \frac{1}{2}(F_t + F F_u)\Delta t^2 + O(\Delta t^3). \tag{4}$$

   (c) Argue that $c_1 + c_2 = 1$ must be satisfied for a valid (i.e. consistent) numerical scheme.

   (d) Determine the values of c_1 and c_2 that give a second-order scheme.

2. Consider the explicit trapezoid scheme

$$u_{k+1} = u_k + \frac{\Delta t}{2}\left[F\left(t_k, u_k\right) + F\left(t_k + \Delta t, u_k + \Delta t F(t_k, u_k)\right)\right]. \tag{5}$$

   Show that the local truncation error is $O(\Delta t^3)$ and thus the scheme is second-order.

3. Consider the following initial value problem:

$$u' = \frac{t^3}{u}, \quad u(0) = 1. \tag{6}$$

   (a) Derive the exact solution.

   (b) Write down the forward Euler scheme.

   (c) Write down the backward Euler scheme. Express u_{k+1} explicitly in terms of t_k, u_k, and Δt, i.e., $u_{k+1} = G(t_k, u_k, \Delta t)$.

   (d) By showing that

$$G(t_k, u_k, \Delta t) = u_k + \frac{t_k^3}{u_k}\Delta t + O(\Delta t^2), \tag{7}$$

   argue that the backward Euler scheme to (6) is a first-order scheme.

MATLAB

1. (a) Write a MATLAB function

```
function [tvals,uvals] = Euler(F,t0,T,u0,n)
```

that computes the solution $u(t)$ $(t_0 \le t \le T)$ to the initial value problem

$$u'(t) = F(t, u(t)), \quad u(t_0) = u_0. \tag{8}$$

using the Euler method. Note that `tvals` contains points $t_k = t_0 + k\Delta t$ $(k = 0, 1, \cdots, n)$ with $\Delta t = (T - t_0)/n$ and `uvals` contains the values of the approximate solution at these points.

(b) Validate your code by using the following initial value problem:

$$u' = \left(1 - \frac{4}{3}t\right)u, \quad u(0) = 1. \tag{9}$$

In other words, (i) plot the approximate solutions $u(t)$ $(0 \le t \le 3)$ for $n = 10$ and 20 with the exact solution $u(t) = \exp(t - \frac{2}{3}t^2)$ and (ii) draw a log-log plot of the error at the final point $t = T$ versus the number of subintervals n. Observe how the errors decrease.

2. Consider the following initial value problem for $u(t)$ $(0 \le t \le 2)$:

$$\frac{du}{dt} = \frac{2u - 18t}{1 + t}, \quad u(0) = 4 \tag{10}$$

(a) Using your MATLAB function `Euler` with $n = 100$, 200, 500, and 1000, compute the approximate values of $u(2)$.

(b) Using these values, estimate (i) the true value of $u(2)$ and (ii) the smallest value of n that would give the error at $t = 2$ smaller than 10^{-2}. Hint: Use the following relation:

$$u^{(n)}(T) = u^{(\infty)}(T) + \frac{C}{n}, \tag{11}$$

where $u^{(n)}(T)$ denote the approximate solution at $t = T$ obtained from n subintervals, whereas $u^{(\infty)}(T)$ is the exact solution at $t = T$.

3. (a) Write a MATLAB function for the explicit midpoint scheme

```
function [tvals,uvals] = explicit_midpoint(F,t0,T,u0,n)
```

(b) Validate your code by using the initial value problem (8).

(c) Perform a *fair* comparison between the Euler and explicit midpoint methods.