3.7. Show that the truncation error for backward Euler is $O(\Delta t)$.

$$\tau(t) = \left| \dot{u}(t) - F(u(t)) \right|$$

$$= \left| \frac{u(t) - u(t-\Delta t)}{\Delta t} - \dot{u}(t) \right|$$

Note By Taylor's Expansion

$$u(t-\Delta t) = u(t) - \Delta t\, \dot{u}(t) + \frac{(\Delta t)^2}{2} \ddot{u}(t) + O(\Delta t^3)$$

Substituting back in

$$\tau(t) = \left| \frac{\cancel{u(t)} - \cancel{u(t)} + \Delta t\, \cancel{\dot{u}}(t) - \frac{(\Delta t)^2}{2} \ddot{u}(t) + O(\Delta t^3)}{\Delta t} - \cancel{\dot{u}(t)} \right|$$

$$= \left| O((\Delta t)^2) - \frac{(\Delta t)}{2} \ddot{u}(t) \right|$$

$$= O(\Delta t)$$

Thus $\tau(t) = O(\Delta t)$ $\square$

3.8. Show that the midpoint method is second order.

$$\dot{u}(t) = \frac{u(t+\Delta t) - u(t-\Delta t)}{\Delta t}$$

$$F(u(t)) = 2f(u(t))$$

$$\tau(t) = \left| \frac{u(t+\Delta t) - u(t-\Delta t)}{\Delta t} - 2\dot{u}(t) \right|$$

By Taylor's Expansion

$$u(t+\Delta t) = u(t) + \Delta t\, \dot{u}(t) + \frac{(\Delta t)^2}{2} \ddot{u}(t) + O((\Delta t)^3)$$

$$u(t-\Delta t) = u(t) - \Delta t\, \dot{u}(t) + \frac{(\Delta t)^2}{2} \ddot{u}(t) + O((\Delta t)^3)$$

Substituting back in

$$\tau(t) = \left| \frac{\cancel{u(t)} + \Delta t \cancel{\dot{u}} + \cancel{\frac{(\Delta t)^2}{2}} \ddot{u}(t) + O((\Delta t)^3) - \cancel{u(t)} + \Delta t \cancel{\dot{u}}(t) - \cancel{\frac{(\Delta t)^2}{2}} \ddot{u}(t) + O((\Delta t)^3)}{\Delta t} - 2\cancel{\dot{u}(t)} \right|$$

$$= O((\Delta t)^2)$$

Thus midpoint method has second order

3.9. Derive the 3-step Adams–Bashforth method, and show that it is really third order.

$r=3$

$$u(t+3\Delta t)=u(t+2\Delta t)+\Delta t\left[b_0 f(u(t))+b_1 f(u(t+\Delta t))+b_2 f(u(t+2\Delta t))\right]$$

$$\tau(t)=\frac{u(t+3\Delta t)-u(t+2\Delta t)}{\Delta t}-b_0 f(u(t))-b_1 f(u(t+\Delta t))-b_2 f(u(t+2\Delta t))$$

By Taylors expansion

$$u(t+3\Delta t)=u(t)+3\Delta t\,\dot u(t)+\frac{(3\Delta t)^2}{2}\ddot u+\frac{(3\Delta t)^3}{6}\dddot u(t)+O(\Delta t^4)$$

$$u(t+2\Delta t)=u(t)+2\Delta t\,\dot u(t)+\frac{(2\Delta t)^2}{2}\ddot u(t)+\frac{(2\Delta t)^3}{6}\dddot u(t)+O(\Delta t^4)$$

$$\dot u(t+\Delta t)=\dot u(t)+\Delta t\,\ddot u(t)+\frac{(\Delta t)^2}{2}\dddot u(t)+O(\Delta t^3)$$

$$\dot u(t+2\Delta t)=\dot u(t)+2\Delta t\,\ddot u(t)+\frac{(2\Delta t)^2}{2}\dddot u(t)+O(\Delta t^3)$$

$$\tau(t)=\dot u(t)+\frac{5\Delta t}{2}\ddot u(t)+\frac{19(\Delta t)^2}{6}\dddot u(t)-b_0\dot u(t)-b_1\dot u(t+\Delta t)-b_2\dot u(t+2\Delta t)+O(\Delta t^3)$$

$$=(1-b_0-b_1-b_2)\dot u(t)+\Delta t\left[\frac{5}{2}-b_1-2b_2\right]\ddot u(t)$$

$$+(\Delta t)^2\left[\frac{19}{6}-\frac{1}{2}b_1-2b_2\right]\dddot u(t)+O(\Delta t^3)$$

$1=b_0+b_1+b_2$

$\frac{5}{2}=b_1+2b_2$

$\frac{19}{6}=\frac{1}{2}b_1+2b_2$

$b_2=\frac{23}{12}$ $\quad b_0=\frac{5}{12}$

$b_1=-\frac{8}{6}$

$$\Rightarrow \quad \tau(t)=O((\Delta t)^3)$$

$$\Rightarrow V_{i+3}=V_{i+2}+(\Delta t)\left[\frac{5}{12}f(V_i)-\frac{8}{6}f(V_{i+1})+\frac{23}{12}f(V_{i+2})\right]$$

3.10. Derive the 2-step Adams–Moulton method, and show that it is third order.

$r = 2$

$$u(t+2\Delta t) = u(t+\Delta t) + \Delta t\left[b_0 f(u(t)) + b_1 f(u(t+\Delta t)) + b_2 f(u(t+2\Delta t))\right]$$

$$\tau = \frac{u(t+2\Delta t) - u(t+\Delta t)}{\Delta t} - b_0 f(u(t)) - b_1 f(u(t+\Delta t)) - b_2 f(u(t+2\Delta t))$$

$$= \dot{u}(t) + \frac{3(\Delta t)}{2}\ddot{u}(t) + \frac{7(\Delta t)^2}{6}\dddot{u}(t) - b_0\dot{u}(t) - b_1\dot{u}(t+\Delta t) - b_2\dot{u}(t+2\Delta t) + O(\Delta t^3)$$

$$= (1 - b_0 - b_1 - b_2)\dot{u}(t) + \Delta t\left(\frac{3}{2} - b_1 - 2b_2\right)\ddot{u}(t) + (\Delta t)^2\left(\frac{7}{6} - \frac{1}{2}b_1 - 2b_2\right)\dddot{u}(t) + O(\Delta t^3)$$

$1 = b_0 + b_1 + b_2$ $\qquad b_0 = -\frac{1}{12}$

$\frac{3}{2} = 0 + b_1 + 2b_2$ $\qquad b_2 = \frac{5}{12}$

$\frac{7}{6} = 0 + \frac{1}{2}b_1 + 2b_2$ $\qquad b_1 = \frac{2}{3}$

$$\Rightarrow \tau(t) = O(\Delta t)^3$$

$$V_{i+2} = V_{i+1} + (\Delta t)\left[-\frac{1}{12}f(V_i) + \frac{2}{3}f(V_{i+1}) + \frac{5}{12}f(V_{i+2})\right]$$

In [ ]:
```
import numpy as np
import matplotlib.pyplot as plt
```

In [ ]:
```
def adams_moulton(a, b, delta_t, T, ini_val):
    t_vals = np.linspace(0, T, int(np.floor(T / delta_t)) + 1)
    f = lambda x: a * x + b
    v = np.zeros(np.shape(t_vals))
    v[0] = ini_val
    v[1] = v[0] + delta_t * (a * v[0] + b)
    for i in range(2, len(t_vals)):
        v[i] = v[i-1] + delta_t * (-1/2 * f(v[i-2]) + 3/2 * f(v[i-1]))
    return t_vals, v
```

In [ ]:
```
a = 1
b = 2
delta_t = 0.01
ini_val = 1
T = 3
x, y = adams_moulton(a, b, delta_t, T, ini_val)
plt.subplot(121)
ts = np.linspace(0, 3, 301)
plt.plot(ts, 3*np.exp(ts)-2, label='exact')
plt.plot(x, y, label='adams_moulton')
plt.legend()
plt.title('Adams_Moulton vs Actual')
plt.xlabel('time t')
plt.ylabel('u(t)')
plt.subplot(122)
plt.plot(ts, 3*np.exp(ts)-2 - y, label='Error')
plt.title('Error')
plt.xlabel('time t')
plt.ylabel('Error')
plt.legend()
plt.tight_layout()
plt.show()
```