AMATH 586 Homework 1 Solutions

1 Problem 1

Using the Taylor series representation of the matrix exponential:

1.1 Part (a)

Verify the identities

$$\frac{d}{dt}e^{tA} = Ae^{tA} = e^{tA}A$$

for an $n \times n$ matrix A.

1.1.1 Solution

$$\begin{aligned}
\frac{d}{dt}e^{tA} &= \frac{d}{dt}\left[I + tA + \frac{1}{2!}t^2A^2 + \frac{1}{3!}t^3A^3 + \dots\right] \\
&= A + tA^2 + \frac{1}{2!}t^2A^3 + \dots \\
&= A\left(I + tA + \frac{1}{2!}t^2A^2 + \dots\right) \\
&= Ae^{At}
\end{aligned}$$

If we instead factor out A on the right side, we get $e^{tA}A$. Therefore the matrices commute, and

$$\frac{d}{dt}e^{tA} = Ae^{tA} = e^{tA}A.$$

1.2 Part (b)

Verify that $u(t) = e^{tA}\eta$ is indeed the solution to the IVP

$$\begin{cases} u'(t) = Au(t), \\ u(0) = \eta. \end{cases}$$

1.2.1 Solution

Using the identity from (a), we can verify that $u'(t) = Au(t)$:

$$\frac{d}{dt}u(t) = \frac{d}{dt}e^{tA}\eta = Ae^{tA}\eta = Au(t)$$

Now we verify the initial condition:

$$u(0) = e^{0\cdot A}\eta = \left(I + 0 \cdot A + \frac{1}{2!} \cdot 0^2 \cdot A^2 + \dots\right)\eta = \eta$$

2 Problem 2

Construct a system (i.e., needs to be not scalar valued)

$$\begin{cases} u'(t) = f(u(t)), \end{cases}$$

and two choices of initial data $u_0 \neq v_0$ so that the two solutions

$$\begin{cases} u'(t) = f(u(t)), \\ u(0) = u_0, \end{cases} \qquad \begin{cases} v'(t) = f(v(t)), \\ v(0) = v_0, \end{cases}$$

satisfy

$$\|u(t) - v(t)\|_2 = \|u(0) - v(0)\|_2 e^{Lt} \tag{1}$$

where L is a Lipschitz constant for $f(u)$. Recall that we have shown that for any solution

$$\|u(t) - v(t)\|_2 \leq \|u(0) - v(0)\|_2 e^{Lt}.$$

So, you are tasked with showing that this is sharp. Then show that the equality (1) fails to hold for $u'(t) = -f(u(t))$, $v'(t) = -f(v(t))$ with the same initial conditions.

2.0.1 Solution

Consider the simple system where $u'(t) = Iu(t)$, which has solution $u(t) = e^{It}u_0$. Then we have

$$\begin{aligned} \|u(t) - v(t)\|_2 &= \left\|e^{It}u_0 - e^{It}v_0\right\|_2, \\ &= \left\|e^{It}\left(u_0 - v_0\right)\right\|_2, \\ &= \left\|\begin{bmatrix} e^t & 0 \\ 0 & e^t \end{bmatrix}\left(u_0 - v_0\right)\right\|_2, \\ &= \left\|e^t I\left(u_0 - v_0\right)\right\|_2, \\ &= \left|e^t\right| \left\|I\left(u_0 - v_0\right)\right\|_2, \\ &= e^t \left\|u_0 - v_0\right\|_2, \end{aligned}$$

so the Lipschitz constant $L = 1$. However, if we instead consider $u'(t) = -Iu(t)$, which has solution $u(t) = e^{-It}u_0$, we have

$$\|u(t) - v(t)\|_2 = e^{-t} \|u_0 - v_0\|_2 < e^t \|u_0 - v_0\|_2 \quad \text{for} \quad t > 0,$$

so the equality (1) no longer holds.

3 Problem 3

Consider the IVP

$$\begin{cases} u_1'(t) = 2u_1(t), \\ u_2'(t) = 3u_1(t) - u_2(t), \end{cases}$$

with initial condition specified at time $t = 0$. Solve this problem in two different ways:

3.1 Part (a)

Solve the first equation, which only involves u_1, and then insert this function into the second equation to obtain a nonhomogeneous linear equation for u_2. Solve this using (5.8). Check that your solution satisfies the initial conditions and the ODE.

3.1.1 Solution

The first equation with initial condition $u_1'(0) = \eta_1$ has solution $u_1(t) = \eta_1 e^{2t}$. Plugging this solution into the second equation, we get the nonhomogeneous linear equation $u_2'(t) = 3\eta_1 e^{2t} - u_2(t)$ with initial condition $u_2(0) = \eta_2$. Using (5.8) with $A(t) = -1$ and $g(t) = 3\eta_1 e^{2t}$, we get

$$\begin{aligned} u_2(t) &= \eta_2 e^{-t} + \int_0^t e^{-(t-\tau)} \left(3\eta_1 e^{2\tau}\right) d\tau \\ &= \eta_2 e^{-t} + 3\eta_1 e^{-t} \int_0^t e^{3\tau} d\tau \\ &= \eta_2 e^{-t} + \eta_1 e^{-t} \left(e^{3t} - 1\right) \\ &= \eta_1 e^{2t} + (\eta_2 - \eta_1) e^{-t} \end{aligned}$$

Finally, we check that these solutions satisfy the initial conditions:

$$\begin{aligned} u_1(0) &= \eta_1 e^{2\cdot 0} = \eta_1 \\ u_2(0) &= \eta_1 e^{2\cdot 0} + (\eta_2 - \eta_1) e^0 = \eta_2 \end{aligned}$$

3.2 Part (b)

Write the system as $u' = Au$ and compute the matrix exponential using (D.30) to obtain the solution.

3.2.1 Solution

In matrix form, our system is

$$\begin{bmatrix} u_1'(t) \\ u_2'(t) \end{bmatrix} = \begin{bmatrix} 2 & 0 \\ 3 & -1 \end{bmatrix} \begin{bmatrix} u_1(t) \\ u_2(t) \end{bmatrix},$$

where our matrix A has the eigenvector decomposition

$$\begin{bmatrix} 2 & 0 \\ 3 & -1 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} -1 & 0 \\ 0 & 2 \end{bmatrix} \begin{bmatrix} -1 & 1 \\ 1 & 0 \end{bmatrix}.$$

Therefore using (D.29) and (D.30), our solution is

$$\begin{aligned} \begin{bmatrix} u_1(t) \\ u_2(t) \end{bmatrix} &= e^{At} \begin{bmatrix} \eta_1 \\ \eta_2 \end{bmatrix} \\ &= \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} e^{-t} & 0 \\ 0 & e^{2t} \end{bmatrix} \begin{bmatrix} -1 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} \eta_1 \\ \eta_2 \end{bmatrix} \\ &= \begin{bmatrix} e^{2t} & 0 \\ -e^{-t} + e^{2t} & e^{-t} \end{bmatrix} \begin{bmatrix} \eta_1 \\ \eta_2 \end{bmatrix}, \end{aligned}$$

which matches the solution we got in (b).

4 Problem 4

Consider the IVP

$$\begin{cases} u_1'(t) = 2u_1(t), \\ u_2'(t) = 3u_1(t) + 2u_2(t), \end{cases}$$

with initial conditions specified at time $t = 0$. Solve this problem.

4.0.1 Solution

$$u_1(t) = \eta_1 e^{2t} \quad \text{and} \quad u_2(t) = \eta_2 e^{2t} + 3\eta_1 t e^{2t}$$

5 Problem 5

Consider the Lotka-Volterra system

$$\begin{cases} u_1'(t) = \alpha u_1(t) - \beta u_1(t) u_2(t), \\ u_2'(t) = \delta u_1(t) u_2(t) - \gamma u_2(t). \end{cases}$$

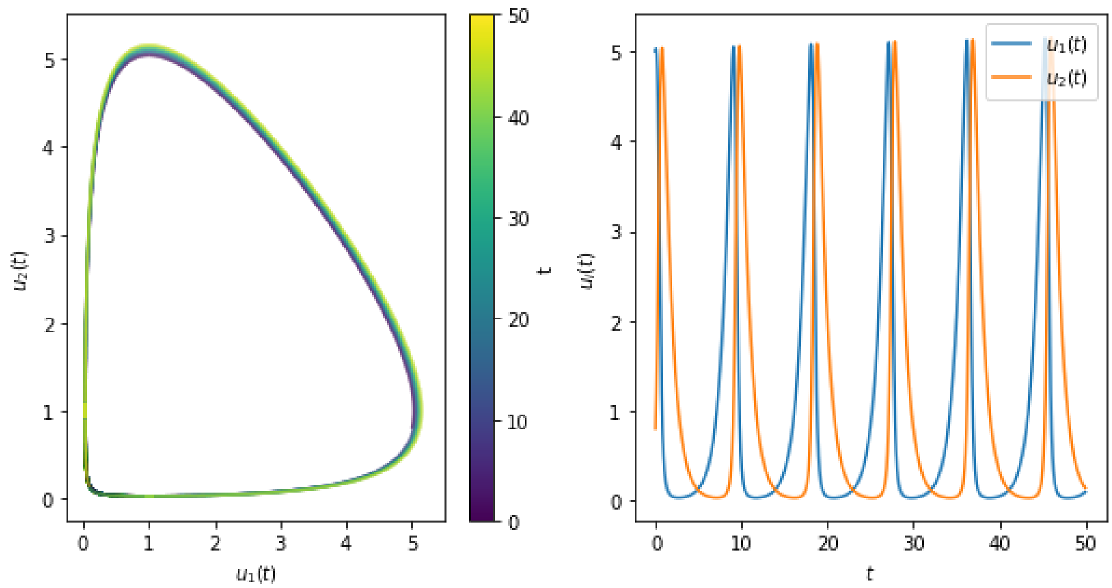
For $\alpha = \delta = \gamma = \beta = 1$ and $u_1(0) = 5$, $u_2(0) = 0.8$ use the forward Euler method to approximate the solution with $k = 0.001$ for $t = 0, 0.001, \ldots, 50$. Plot your approximate solution as a curve in the (u_1, u_2)-plane and plot your approximations of $u_1(t)$ and $u_2(t)$ on the same axes as a function of t. Repeat this with backward Euler. What do you notice about the behavior of the numerical solution? The most obvious feature is most apparent in the (u_1, u_2)-plane.

5.0.1 Solution

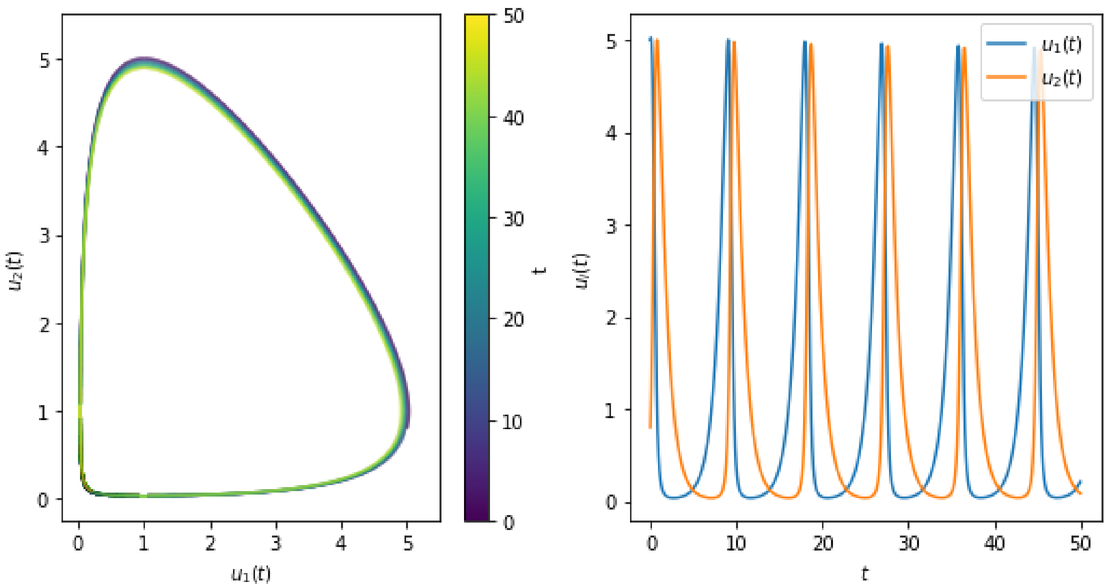
[1]:
```
from matplotlib.collections import LineCollection
import matplotlib.pyplot as plt
import numpy as np
```

[2]:
```
def plot_results(t_vals, u_vals):

    fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(10, 5))

    points = np.array(u_vals).T.reshape(-1, 1, 2)
    segments = np.concatenate([points[:-1], points[1:]], axis=1)
    norm = plt.Normalize(t_vals.min(), t_vals.max())
    lc = LineCollection(segments, cmap='viridis', norm=norm)
    lc.set_array(t_vals)
    line = ax1.add_collection(lc)
    cbar = fig.colorbar(line, ax=ax1)
    cbar.set_label('t')
    ax1.set_xlim((-0.25, 5.5))
    ax1.set_ylim((-0.25, 5.5))
    ax1.set_xlabel('$u_1(t)$')
    ax1.set_ylabel('$u_2(t)$')

    ax2.plot(t_vals, u_vals[0, :])
    ax2.plot(t_vals, u_vals[1, :])
    ax2.set_xlabel('$t$')
    ax2.set_ylabel('$u_i(t)$')
    ax2.legend(('$u_1(t)$', '$u_2(t)$'), loc='upper right')
```

[3]:
```
# Forward Euler

def f(u, alpha=1, delta=1, gamma=1, beta=1):
    u_prime = np.zeros_like(u)
    u_prime[0] = alpha*u[0] - beta*u[0]*u[1]
    u_prime[1] = delta*u[0]*u[1] - gamma*u[1]
    return u_prime

k = 0.001
t_vals = np.arange(0, 50 + k, k)
u_vals = np.zeros((2, len(t_vals)))
u_vals[:, 0] = np.array([5, 0.8])
for t in range(len(t_vals) - 1):
    u_vals[:, t+1] = u_vals[:, t] + k*f(u_vals[:, t])
u_forward = u_vals

plot_results(t_vals, u_forward)
```

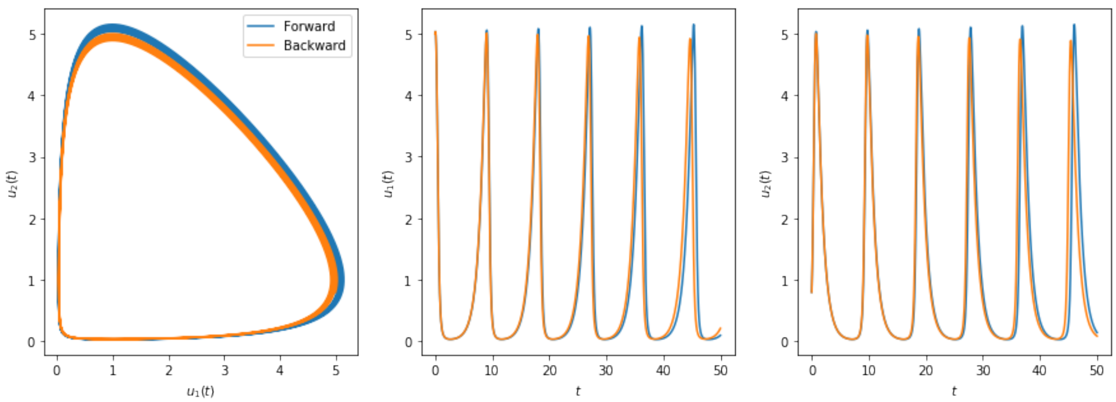


[4]:
```
# Backward Euler

def g(u, u_n, alpha=1, delta=1, gamma=1, beta=1):
    return u - u_n - k*f(u, alpha, delta, gamma, beta)

def D(u, alpha=1, delta=1, gamma=1, beta=1):
    return np.array([[1 - k*(alpha + beta*u[1]), k*beta*u[0]],
                     [-k*delta*u[1], 1 - k*(delta*u[0] - gamma)]])

k = 0.001
t_vals = np.arange(0, 50 + k, k)
u_vals = np.zeros((2, len(t_vals)))
u_vals[:,0] = np.array([5, 0.8])
max_iter = 10
for t in range(len(t_vals) - 1):
    u_vals[:, t+1] = u_vals[:, t]
    for j in range(max_iter):
        u_temp = u_vals[:, t+1]
        u_vals[:, t+1] = u_temp - np.linalg.solve(D(u_temp), g(u_temp, u_vals[:,␣
↪t]))
        if max(np.abs(u_vals[:, t+1] - u_temp)) < k/10:
            break
        if j == max_iter:
            print('Newton did not terminate.')
u_backward = u_vals

plot_results(t_vals, u_backward)
```



[5]:
```
# Compare solutions

fig, (ax1, ax2, ax3) = plt.subplots(1, 3, figsize=(15, 5))

ax1.plot(u_forward[0,:], u_forward[1, :])
ax1.plot(u_backward[0,:], u_backward[1, :])
ax1.set_xlabel('$u_1(t)$')
ax1.set_ylabel('$u_2(t)$')
ax1.legend(('Forward', 'Backward'), loc='upper right')

ax2.plot(t_vals, u_forward[0, :])
ax2.plot(t_vals, u_backward[0, :])
ax2.set_xlabel('$t$')
ax2.set_ylabel('$u_1(t)$')

ax3.plot(t_vals, u_forward[1, :])
ax3.plot(t_vals, u_backward[1, :])
ax3.set_xlabel('$t$')
ax3.set_ylabel('$u_2(t)$')
```

[5]:
```
Text(0, 0.5, '$u_2(t)$')
```



Takeaway: Solutions from forward Euler tend to grow, while solutions from backward Euler tend to decay.

6 Problem 6

Determine the coefficients β_0, β_1, β_2 for the third order, 2-step Adams-Moulton method. Do this in two different ways:

6.1 Part (a)

Using the expression for the local truncation error in Section 5.9.1.

6.1.1 Solution

From (5.45), we know that Adams methods have the form

$$U^{n+r} = U^{n+r-1} + k \sum_{j=0}^{r} \beta_j f\left(U^{n+j}\right)$$

with $\alpha_r = 1$, $\alpha_{r-1} = -1$, and $\alpha_j = 0$ for $j < r-1$. For a 2-step Adams-Moulton method, we let $r = 2$, $\alpha_2 = 1$, $\alpha_1 = -1$, and $\alpha_0 = 0$. To derive a third-order accurate method, we need the first four terms of the expression for $\tau(t_{n+2})$ to vanish, specifically:

1. $\sum_{j=0}^{2} \alpha_j = 0$, which is already satisfied by our choices of α_j,
2. $\sum_{j=0}^{2} \left(j\alpha_j - \beta_j\right) = 0$, which simplfies to $\sum_{j=0}^{2} \beta_j = 1$,
3. $\sum_{j=0}^{2} \left(\frac{1}{2}j^2\alpha_j - j\beta_j\right)$, which simplifies to $\beta_1 + 2\beta_2 = \frac{3}{2}$, and
4. $\sum_{j=0}^{2} \left(\frac{1}{6}j^3\alpha_j - \frac{1}{2}j^2\beta_j\right)$, which simplifies to $\frac{1}{2}\beta_1 + 2\beta_2 = \frac{7}{6}$.

To find the coefficients, we solve the following linear system

$$\begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 2 \\ 0 & \frac{1}{2} & 2 \end{bmatrix} \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \end{bmatrix} = \begin{bmatrix} 1 \\ \frac{3}{2} \\ \frac{7}{6} \end{bmatrix}$$

to get the solution $\beta_0 = -\frac{1}{12}$, $\beta_1 = \frac{2}{3}$, and $\beta_2 = \frac{5}{12}$. Therefore the third order, 2-step Adams-Moulton method is

$$\begin{aligned} U^{n+2} &= U^{n+1} + k\left(-\frac{1}{12}F\left(U^n\right) + \frac{2}{3}f\left(U^{n+1}\right) + \frac{5}{12}f\left(U^{n+2}\right)\right) \\ &= U^{n+1} + \frac{k}{12}\left(-f\left(U^n\right) + 8f\left(U^{n+1}\right) + 5f\left(U^{n+2}\right)\right) \end{aligned}$$

6.2 Part (b)

Using the relaxation

$$u(t_{n+2}) = u(t_{n+1}) + \int_{t_{n+1}}^{t_{n+2}} f(u(s))ds.$$

Interpolate a quadratic polynomial $p(t)$ through the three values $f(U^n)$, $f(U^{n+1})$, and $f(U^{n+2})$ and then integrate this polynomial exactly to obtain the formula. The coefficients of the polynomial will depend on the three values $f(U^{n+j})$. It's easiest to use the "Newton form" of the interpolating polynomial and consider the three times $t_n = -k$, $t_{n+1} = 0$, and $t_{n+2} = k$ so that $p(t)$ has the form

$$p(t) = A + B(t+k) + C(t+k)t$$

where A, B, and C are the appropriate divided differences based on the data. Then integrate from 0 to k. (The method has the same coefficients any time, so this is valid.)

6.2.1 Solution

Using the three times and form of $p(t)$ above, we can solve for coefficients:

$$\begin{aligned} A &= f(U^n) \\ B &= \frac{f(U^{n+1}) - f(U^n)}{k} \\ C &= \frac{f(U^{n+2}) - 2f(U^{n+1}) + f(U^n)}{2k^2} \end{aligned}$$

Plugging the polynomial with these coefficients into the integral equation above and simplifying, we have:

$$
\begin{aligned}
U^{n+2} &= U^{n+1} + \int_0^k f(U^{n+1}) + \frac{f(U^{n+2}) - f(U^n)}{2k}s + \frac{f(U^{n+2}) - 2f(U^{n+1}) + f(U^n)}{2k^2}s^2 ds \\
&= U^{n+1} + f(U^{n+1})k + \frac{f(U^{n+2}) - f(U^n)}{4}k + \frac{f(U^{n+2}) - 2f(U^{n+1}) + f(U^n)}{6}k \\
&= U^{n+1} - \frac{k}{12}f\left(U^n\right) + \frac{2k}{3}f\left(U^{n+1}\right) + \frac{5k}{12}f\left(U^{n+2}\right) \\
&= U^{n+1} + \frac{k}{12}\left(-f\left(U^n\right) + 8f\left(U^{n+1}\right) + 5f\left(U^{n+2}\right)\right)
\end{aligned}
$$