Systems of 1st-order initial-value problems

Consider the following system of two IVPs.

$$
\begin{aligned}
y^{(1)}(t) &= -3y(t) + 2y(t)z(t) \\
y(0) &= 7 \\
z^{(1)}(t) &= -4tz(t) - y(t)z(t) \\
z(0) &= 5
\end{aligned}
$$

1. How would we write this using vector notation?

Answer: Let $w_1 = y$ and $w_2 = z$ so

$$
\mathbf{w}^{(1)}(t) = \begin{pmatrix} -3w_1(t) + 2w_1(t)w_2(t) \\ -4tw_2(t) - w_1(t)w_2(t) \end{pmatrix}
$$

$$
\mathbf{w}(0) = \begin{pmatrix} 7 \\ 5 \end{pmatrix}
$$

2. How would you author a function in Matlab to return the right-hand side of the ordinary differential equation (ODE)?

Answer:

```
f = @( t, w )( [-3*w(1) + 2*w(1)*w(2); -4*t*w(2) - w(1)*w(2)] );
```

3. Approximate the solutions to $y(0.1)$ and $z(0.1)$ using four steps of Euler's method.

Answer:

7	8.225	9.30453125	10.012634670349121	10.62776506630648
5	4.125	3.266484375	2.490324304504394	1.841199689432058

4. How could you code this in Matlab?

Answer:

```
w = [7 5]'
h = 0.025;
for t = 0 : h : 3*h
    w = w + h*f( t, w )
end
```

5. Approximate the solutions to $y(0.1)$ and $z(0.1)$ using two steps of Heun's method.

7	9.051875	10.04340244217662
5	3.3409375	2.069134298980051

4. How could you code this in Matlab?

Answer:

```
w = [7 5]'
h = 0.05;
for t = 0 : h : h
    s0 = f( t, w );
    s1 = f( t + h, w + h*s0 );
    w = w + h*(s0 + s1)/2
end
```

5. Approximate the solutions to $y(0.1)$ and $z(0.1)$ using one step of the 4th-order Runge-Kutta method.

7	10.019281775179129
5	2.007191283218939

6. How could you code this in Matlab?

Answer:

```
w = [7 5]'
h = 0.1;
s0 = f( 0, w );
s1 = f( 0 + h/2, w + h/2*s0 );
s2 = f( 0 + h/2, w + h/2*s1 );
s3 = f( 0 + h,   w + h  *s2 );
w = w + h*(s0 + 2*s1 + 2*s2 + s3)/6
```

7. Approximate the solution to $y(10)$ using 100 steps of the 4th-order Runge-Kutta method.

$w_1(10) = 1.931668252068150 \times 10^{-12}$

$w_2(10) = 1.013593401979833 \times 10^{-14}$

8. Of course, if you attempted the previous question, you did not do so by hand, so how did you do it in Matlab?

```
h = 0.1;
w = [7 5]';
for t = 0:h:(10 - h)
    s0 = f( t,       w );
    s1 = f( t + h/2, w + h/2*s0 );
    s2 = f( t + h/2, w + h/2*s1 );
    s3 = f( t + h,   w + h  *s2 );
    w = w + h*(s0 + 2*s1 + 2*s2 + s3)/6;
end
w
```

Consider the following system of six IVPs.

$$
\begin{aligned}
u_1^{(1)}(t) &= u_2(t) \\
u_1(0) &= 4 \\
u_2^{(1)}(t) &= -2u_1(t) \\
u_2(0) &= 5 \\
v_1^{(1)}(t) &= v_2(t) + u_1(t) \\
v_1(0) &= 6 \\
v_2^{(1)}(t) &= -3v_1(t) \\
v_2(0) &= 7 \\
x_1^{(1)}(t) &= x_2(t) + v_1(t) \\
x_1(0) &= 8 \\
x_2^{(1)}(t) &= -4x_1(t) \\
x_2(0) &= 9
\end{aligned}
$$

1. How would we write this using vector notation?

Answer: Let $w_1 = u_1, \ldots, w_6 = x_2$, so

$$
\mathbf{w}^{(1)}(t) = \begin{pmatrix} w_2(t) \\ -2w_1(t) \\ w_4(t) + w_1(t) \\ -3w_3(t) \\ w_6(t) + w_3(t) \\ -4w_5(t) \end{pmatrix}
$$

$$
\mathbf{w}(0) = \begin{pmatrix} 4 \\ 5 \\ 6 \\ 7 \\ 8 \\ 9 \end{pmatrix}
$$

2. How would you author a function in Matlab to return the right-hand side of the ordinary differential equation (ODE)?

Answer:

```
f = @( t, w )( [w(2); -2*w(1); w(4)+w(1); -3*w(3); w(6)+w(3); -4*w(5)] );
```

3. Approximate the solution to $y(10)$ using 10 steps of the 4th-order Runge-Kutta method.

$$\begin{array}{r} 2.712451933457805 \\ -3.027000268712214 \\ -5.921694454527967 \\ -8.370218217959568 \\ -2.366557725190852 \\ 18.46386671488615 \end{array}$$

4. Of course, if you attempted the previous question, you did not do so by hand, so how did you do it in Matlab?

```
h = 1.0;
w = [4 5 6 7 8 9]';
for t = 0:h:(10 - h)
    s0 = f( t,       w );
    s1 = f( t + h/2, w + h/2*s0 );
    s2 = f( t + h/2, w + h/2*s1 );
    s3 = f( t + h,   w + h  *s2 );
    w = w + h*(s0 + 2*s1 + 2*s2 + s3)/6;
end
w
```

Acknowledgement: Ethan Romero for pointing out the incorrect numbers in Heun's method and a missing 't' in the first system of ordinary differential equations.