Systems of first-order ODEs

As we previously discussed, it is possible to consider a type of differential equations with several unknown functions \( y_1(t), y_2(t), \ldots, y_n(t) \) (where each \( y_i : \mathbb{R} \to \mathbb{R} \)) and \( n \) total given equations:

\[
\begin{align*}
y_1'(t) &= f_1(t, y_1, y_2, \ldots, y_n) \\
y_2'(t) &= f_2(t, y_1, y_2, \ldots, y_n) \\
&\quad \vdots \\
y_n'(t) &= f_n(t, y_1, y_2, \ldots, y_n)
\end{align*}
\]

The methods for approximating solutions to individual ODEs carry over naturally to systems of ODEs. For example

\[
\begin{align*}
y_1'(t) &= -2y_1 \sin(y_2) = f_1(y_1, y_2) \\
y_2'(t) &= -y_1^2 y_2 = f_2(t, y_1, y_2)
\end{align*}
\]
Let us apply the Forward Euler method. We will use the notation $y^{(n)}_i = y_i(t_n)$

Thus:

$y^{(n+1)}_1 = y^{(n)}_1 + \Delta t f_1(t_n, y^{(n)}_1, y^{(n)}_2) = y^{(n)}_1 - \Delta t y^{(n)}_1 \sin(y^{(n)}_2)$

$y^{(n+1)}_2 = y^{(n)}_2 + \Delta t f_2(t_n, y^{(n)}_1, y^{(n)}_2) = y^{(n)}_2 - \frac{y^{(n)}_1}{y^{(n)}_2} y^{(n)}_2$

Thus we proceed as follows:

\[
\begin{array}{c}
y^{(0)}_1 \\
y^{(0)}_2
\end{array}
\rightarrow
\begin{array}{c}
y^{(1)}_1 \\
y^{(1)}_2
\end{array}
\rightarrow
\begin{array}{c}
y^{(2)}_1 \\
y^{(2)}_2
\end{array}
\rightarrow
\quad \vdots
\]

Another example... let's apply **Backward Euler** to

\[
\begin{align*}
p'(t) &= -c_1 p(t) \\
q'(t) &= c_1 p(t) - c_2 q(t) \\
r'(t) &= c_2 q(t)
\end{align*}
\]

\[
\begin{pmatrix} p' \\ q' \\ r'
\end{pmatrix} =
\begin{pmatrix} -c_1 & 0 & 0 \\ c_1 & -c_2 & 0 \\ 0 & c_2 & 0
\end{pmatrix}
\begin{pmatrix} p \\ q \\ r
\end{pmatrix}
\]
Backward Euler gives:

\[
\begin{align*}
    p^{(n+1)} &= p^{(n)} + \Delta t f(t, p^{(n)}, q^{(n)}, r^{(n)}) \\
    &= p^{(n)} - \Delta t c_1 p^{(n+1)} \\
    q^{(n+1)} &= q^{(n)} + \Delta t (c_1 p^{(n+1)} - c_2 q^{(n+1)}) \\
    r^{(n+1)} &= r^{(n)} + \Delta t c_2 q^{(n+1)}
\end{align*}
\]

(1) \Rightarrow \begin{pmatrix} 1 + \Delta t c_1 \end{pmatrix} p^{(n+1)} = p^{(n)}

(2) \Rightarrow -\Delta t c_1 p^{(n+1)} + (1 + \Delta t c_2) q^{(n+1)} = q^{(n)}

(3) \Rightarrow -\Delta t c_2 q^{(n+1)} + r^{(n+1)} = r^{(n)}

\[
\begin{pmatrix}
    1 + \Delta t c_1 & 0 & 0 \\
    -\Delta t c_1 & 1 + \Delta t c_2 & 0 \\
    0 & -\Delta t c_2 & 1
\end{pmatrix}
\begin{pmatrix}
    p^{(n+1)} \\
    q^{(n+1)} \\
    r^{(n+1)}
\end{pmatrix}
= 
\begin{pmatrix}
    p^{(n)} \\
    q^{(n)} \\
    r^{(n)}
\end{pmatrix}
Ok, what about stability and accuracy, though?

→ More complicated analysis!

A. Stability of the solutions to the system of ODE's.

Here, asymptotic stability again means that if

\[ y_1, y_2, \ldots, y_n \quad \text{and} \quad \tilde{y}_1, \tilde{y}_2, \ldots, \tilde{y}_n \]

are 2 different solutions to the system, then

\[ \lim_{t \to \infty} |y_i(t) - \tilde{y}_i(t)| = 0 \quad \forall \ i = 1, 2, \ldots, n \]

Making a reliable determination requires tools we don't have in CS412... but we can do so in special cases:

* Consider the system

\[ y'_1 = f_1(t, y_1, \ldots, y_n) \]

\[ y'_2 = f_2(t, y_1, \ldots, y_n) \]

\[ \vdots \]

\[ y'_n = f_n(t, y_1, \ldots, y_n) \]

Define the Jacobian matrix as

\[ J = \begin{pmatrix}
\frac{df_1}{dy_1} & \cdots & \frac{df_1}{dy_n} \\
\frac{df_2}{dy_1} & \cdots & \frac{df_2}{dy_n} \\
\vdots & \ddots & \vdots \\
\frac{df_n}{dy_1} & \cdots & \frac{df_n}{dy_n}
\end{pmatrix} \]
• If $J$ is symmetric and negative definite (i.e. $x^T J x < 0 \ \forall x \neq 0$)
then the solutions are asymptotically stable.

• If $J$ is diagonally dominant and all diagonal entries are negative, then the solutions are asymptotically stable.

\[ \begin{align*}
p' &= -3p + q \\
q' &= p - 3q + r \\
r' &= q - 3r \end{align*} \]
\[ J = \begin{pmatrix} -3 & 1 & 0 \\
1 & -3 & 1 \\
0 & 1 & -3 \end{pmatrix} \]
diagonally dominant! negative diagonal entries! \implies \text{asymptotically stable solutions!}

Thm: If a system has asymptotically stable solutions and the method we use is unconditionally stable for individual equations, then it will be unconditionally stable for systems, too.

Note: if the method has a stability condition for single equations, it will also have such a condition on systems; however, it is not clear what this maximum allowable $\Delta t$ exactly is! \implies (can try different values and reduce $\Delta t$ if solutions are becoming unbounded.)
The accuracy exhibited in individual equations carries over to systems, e.g. 
Forward Euler \( \Rightarrow \) 1st order
Backward Euler
Trapezoidal Rule \( \Rightarrow \) 2nd order

Systems of ODEs arise naturally from the consideration of higher-order differential equations, e.g.

\[
y^{(n)}(t) = f(t, y, y', y'', y''', \ldots, y^{(n-1)}(t))
\]

E.g. \( y''(t) = -5y'(t) - Ay(t) \)

or \( y''(t) = -2t \sin(y') \) e\(^y\)

How about the stability of these solutions?

Once again, this can be a complicated determination...

However, there is the special case when the equation is "linear", e.g.

\[
y^{(n)}(t) = c_0(t)y(t) + c_1(t)y'(t) + c_2(t)y''(t) + \ldots + c_{n-1}(t)y^{(n-2)}(t)
\]
In this case we employ the following criterion:

→ Replace the k-th derivative \( y^{(k)}(t) \) with the k-th power of an auxiliary variable \( \lambda \), i.e. \( y^{(k)}(t) \rightarrow \lambda^k \)

Thus you generate a polynomial:

\[
\lambda^n = c_0(t) + c_1(t)\lambda + c_2(t)\lambda^2 + \cdots + c_{n-1}(t)\lambda^{n-1}
\]

→ If, for all \( t \geq t_0 \), this polynomial has (possibly complex) roots with negative real parts, then the ODE has asymptotically stable solutions.

\[\text{eg. } y'' = -5y' = 4y\]

\[\rightarrow \lambda^2 = -5\lambda - 4 \Rightarrow \lambda^2 + 5\lambda + 4 = 0 \quad \Rightarrow \lambda_1 = -1 < 0 \quad \lambda_2 = -4 < 0 \quad \text{Asym. Stable!}\]

\[\text{also. } y'' = -6y' - 10y\]

\[\rightarrow \lambda^2 + 6\lambda + 10 = 0 \quad \Rightarrow \lambda_1 = -3 + i \quad \text{Re}\{\lambda_1\} < 0 \quad \lambda_2 = -3 - i \quad \text{Re}\{\lambda_2\} < 0 \quad \text{As. stable!}\]
Solution method: Convert the higher-order ODE into a system of 1st order ones!

Methodology: Consider \( y^{(n)}(t) = f(t, y, y', y'', \ldots, y^{(n-1)}) \)

Define \( y_k(t) := y^{(k)}(t) \) \( k = 0, 1, \ldots, n-1 \).

Thus, the ODE becomes:

\[
\begin{align*}
y'_{n-1}(t) &= f(t, y_0, y_1, y_2, \ldots, y_{n-1}) \\
\vdots & \quad \vdots \\
y'_2 &= y' \\
y' &= y_0
\end{align*}
\]

Along with the \( n-1 \) equations:

\[
\begin{align*}
y'_k(t) &= y_{k+1}(t) & k &= 0, 1, \ldots, n-2
\end{align*}
\]

Putting everything together, as a system:

\[
\begin{align*}
y'_0 &= y_1 \\
y'_1 &= y_2 \\
\vdots & \quad \vdots \\
y'_{n-2} &= y_{n-1} \\
y'_{n-1} &= f(t, y_0, y_1, \ldots, y_{n-1})
\end{align*}
\]
e.g.
\[ y'' = -6y' - 10 \]
becomes
\[ y_0' = y_1 \]
\[ y_1' = -6y_1 - y_0 \]

or
\[ y'''' = -2t \sin (yy') e^y'' \]
becomes
\[ y_0' = y_1 \]
\[ y_1' = y_2 \]
\[ y_2' = -2t \sin (y_0y_1) e^{y_2} \]

Then we solve it with any desired method!

**Final note.** The initial conditions supplied for a higher order ODE \( y^{(m)} = f(t, y, y', \ldots, y^{(m-1)}) \) are

\[ y(t_0) = c_0 \]
\[ y'(t_0) = c_1 \]
\[ y^{(m-1)}(t_0) = c_{m-1} \]
and converted \( y_0(t_0) = c_0 \)
\[ y_1(t_0) = c_1 \]
\[ y_{m-1}(t_0) = c_{m-1} \]