Maple - Systems of Differential Equations

This section examines systems of differential equations with **Maple** providing basic line commands to solve and geometrically interpret this type of problem. More specifically, we examine the basic commands to manage the **Greenhouse/Rockbed Model** from the lecture notes given by the linear system of differential equations:

$$\begin{pmatrix} \dot{u}_1 \\ \dot{u}_2 \end{pmatrix} = \begin{pmatrix} -\frac{13}{8} & \frac{3}{4} \\ \frac{1}{4} & -\frac{1}{4} \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \end{pmatrix} + \begin{pmatrix} 14 \\ 0 \end{pmatrix}.$$

As noted earlier, **Maple** has the *dsolve* routine, which can manage to find the general or specific solution to many elementary equations found in this course. This example is no exception. We begin by entering the system of differential equations in **Maple** as follows:

```
> de1 := diff(u1(t), t) =-(13/8)·u1(t) + 3/4·u2(t) + 14;
```

$$de1 := \frac{d}{dt}\, u1(t) = -\frac{13}{8}\, u1(t) + \frac{3}{4}\, u2(t) + 14 \qquad \textbf{(1)}$$

```
> de2 := diff(u2(t), t) = (1/4)·u1(t) - 1/4·u2(t);
```

$$de2 := \frac{d}{dt}\, u2(t) = \frac{1}{4}\, u1(t) - \frac{1}{4}\, u2(t) \qquad \textbf{(2)}$$

```
> dsolve({de1, de2}, {u1(t), u2(t)});
```

$$\left\{u1(t) = e^{-\frac{7}{4}t}\, \_C2 + e^{-\frac{1}{8}t}\, \_C1 + 16,\ u2(t) = -\frac{1}{6}\, e^{-\frac{7}{4}t}\, \_C2 + 2\, e^{-\frac{1}{8}t}\, \_C1 + 16\right\} \qquad \textbf{(3)}$$

```
> desys := {de1, de2}; ic := {u1(0) = 5, u2(0) = 25};
```

$$desys := \left\{\frac{d}{dt}\, u1(t) = -\frac{13}{8}\, u1(t) + \frac{3}{4}\, u2(t) + 14,\ \frac{d}{dt}\, u2(t) = \frac{1}{4}\, u1(t) - \frac{1}{4}\, u2(t)\right\}$$

$$ic := \{u1(0) = 5, u2(0) = 25\} \qquad \textbf{(4)}$$

```
> combine(dsolve(desys union ic, {u1(t), u2(t)}));
```

$$\left\{u1(t) = -\frac{186}{13}\, e^{-\frac{7}{4}t} + \frac{43}{13}\, e^{-\frac{1}{8}t} + 16,\ u2(t) = \frac{31}{13}\, e^{-\frac{7}{4}t} + \frac{86}{13}\, e^{-\frac{1}{8}t} + 16\right\} \qquad \textbf{(5)}$$

The third command line shows the *dsolve* command with the general solution found for this model system. To obtain a particular solution, we first create the two sets consisting of *desys*, which is the set of the differential equations, and *ic*, which is the set of initial conditions. The next line shows how **Maple** can use its *combine* function to put together the general system of differential equations with a specific set of initial conditions to produce the unique solution to this problem.

Linear Analysis for System of Differential Equations

Maple has a special package *LinearAlgebra* for managing matrices. It performs many of the same operations as MatLab, but does them more algebraically rather than numerically. Below we see a set of commands for analyzing our model above.

```
> with(LinearAlgebra) :
> A := Matrix([[-13/8, 3/4], [1/4, -1/4]]); b := Vector([-14, 0]);
```

$$A := \begin{bmatrix} -\frac{13}{8} & \frac{3}{4} \\ \frac{1}{4} & -\frac{1}{4} \end{bmatrix}$$

$$b := \begin{bmatrix} -14 \\ 0 \end{bmatrix} \quad (6)$$

```
> ue := LinearSolve(A, b);
```

$$ue := \begin{bmatrix} 16 \\ 16 \end{bmatrix} \quad (7)$$

```
> Eigenvectors(A);
```

$$\begin{bmatrix} -\frac{7}{4} \\ -\frac{1}{8} \end{bmatrix}, \begin{bmatrix} -6 & \frac{1}{2} \\ 1 & 1 \end{bmatrix} \quad (8)$$

```
> CharacteristicPolynomial(A, lambda);
```

$$\lambda^2 + \frac{15}{8}\lambda + \frac{7}{32} \quad (9)$$

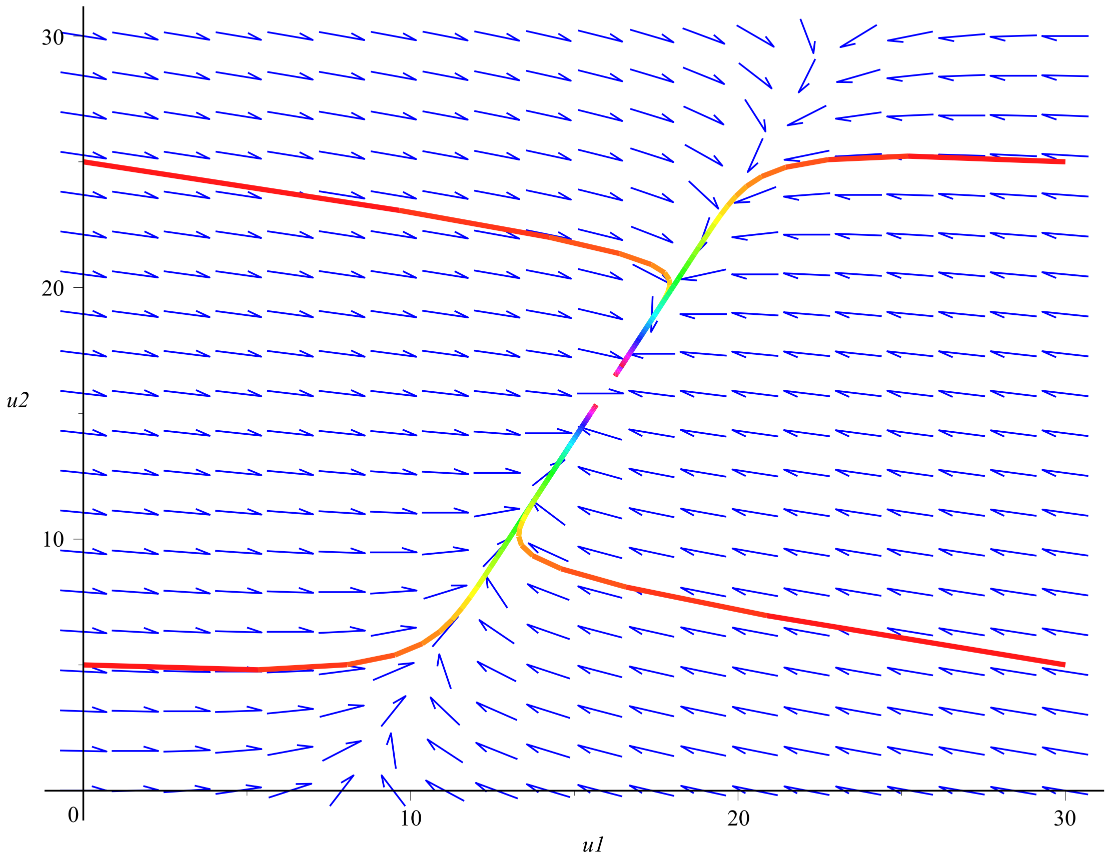
The first line invokes the **Maple** *LinearAlgebra* package. Next the matrix **A** and vector **b** are defined. The special program *LinearSolve* is able to solve the system of linear equations with coefficient matrix **A**. The result gives the equilibrium solution for our greenhouse model. The **Maple** command *Eigenvectors* applied to **A** produces both the *eigenvalues* (first vector) and the associated *eigenvectors* (following columns in the output matrix). There are many other operations, which can help you store information about a matrix, and there are many additional functions, which aid in studying Linear Algebra. The *CharacteristicPolynomial* function is shown as an example.

Phase Portraits in Maple

Maple has a special package of tools for handling the graphics of **differential equations**. One particular function called *DEplot* provides very attractive **phase portraits**. It doesn't have the simple point and click features of the MatLab *pplane8* program, but it is not very difficult to use. Below are a series of **Maple** commands to produce a nice **phase portrait** for our example.

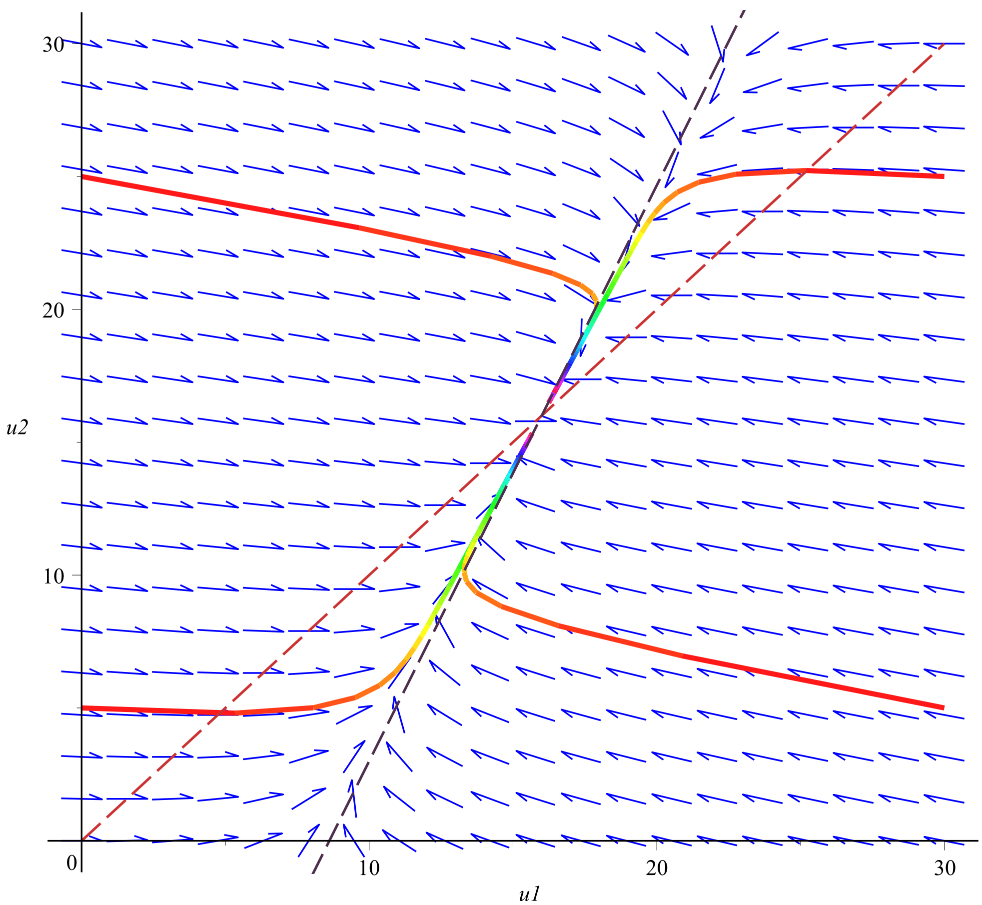
```
> with(DEtools) :
> DEplot({de1, de2}, [u1(t), u2(t)], t = 0..20, [[u1(0) = 0, u2(0) = 5], [u1(0) = 30, u2(0)
      = 5], [u1(0) = 0, u2(0) = 25], [u1(0) = 30, u2(0) = 25]], u1 = 0..30, u2 = 0..30, color
      = blue, linecolor = t);
```

The command *DEplot* has many features that can be varied. It starts with the system of differential equations that are being solved, *de1, de2*. The next element in the command is the **state variables**, which are being solved. This is followed by the time interval of interest. Next one can specify any **solution trajectories** based on the starting points. This is a set of initial conditions of interest to the viewer. Next one specifies the dimensions of the plotting area. The last two commands are simply color changes from the default for the **direction field** arrows (color or colour) and the trajectories (linecolor). The later is given the variable t to indicate the direction of flow from red to purple. Below shows the result of this command.



We can use Maple's *with(plots):* command to overlay multiple plots. In this case, we'll simply add the *nullclines* to our plot, as we have learned that nullclines are useful to locate equilibria. The commands simply store the different graphics in variables, $P1$, $P2$, and $P3$, and use the *display* command to view all elements superimposed.

```
> with(plots) :
> P1 := DEplot({de1, de2}, [u1(t), u2(t)], t = 0 ..20, [[u1(0) = 0, u2(0) = 5], [u1(0) = 30,
      u2(0) = 5], [u1(0) = 0, u2(0) = 25], [u1(0) = 30, u2(0) = 25]], u1 = 0 ..30, u2 = 0 ..30,
      color = blue, linecolor = t) :
> P2 := plot(u1, u1 = 0 ..30, u2 = 0 ..30, linestyle = dash, color = orange) :
> P3 := plot(-56/3 + 13/6 · u1, u1 = 0 ..30, u2 = 0 ..30, linestyle = dash, color = violet) :
> display(P1, P2, P3);
```



30
20
10
0
u2
10
20
30
u1