Solving Linear Equations with Simpy

Summary

This notebook shows how to solve linear equations corresponding to material balances on chemical processes using the {ython symbolic algebra library Sympy. The example is adapted with permission from learnCheme.com, a project at the University of Colorado funded by the National Science Foundation and the Shell Corporation.

Problem Statement

In [1]:

```
from IPython.display import YouTubeVideo
YouTubeVideo("KrrZB5LvXF4",560,315,start=0,end=144,rel=0)
```

Out[1]:

Before going further, be sure you can solve a system of two (or three) linear equations in two (or three) unknowns with paper and pencil. Most of you will have seen these problems before in your math classes.

Solving linear equations using Sympy

Assuming you have mastered the solution of linear equations with paper and pencil, let's see how to find solutions using the python symbolic algebra library Sympy.

Sympy is an example of a Python 'library'. The first step in using the library is to import it into the current workspace. It is customary to import into the workspace with the namespace `sym` to avoid name clashes with variables and functions.

In [ ]:

```
import sympy as sym
```

The system of equations to be solved is given by

$$\begin{aligned} n_1 + n_1 &= 100 \\ 0.7n_1 + 0.2n_2 &= 30 \end{aligned}$$

The next step is to introduces names for the unknown variables appearing in our problem. The Sympy function `sym.var()` constructs symbolic variables given a list of variable names.

In [3]:

```
sym.var(['n1','n2'])
```

Out[3]:

```
[n1, n2]
```

The newly constructed symbolic variable are used to create symbolic equations. The sympy function `sym.Eq()` accepts two arguments, each a symbolic expression expressing the left and right hand sides of an equation. For this problem there are two equations to be solved simultaneously, so we construct both and store them in a python list.

In [4]:

```
eqns = [
    sym.Eq(n1 + n2, 100),
    sym.Eq(0.7*n1 + 0.2*n2, 30)
]
print(eqns)
```

```
[Eq(n1 + n2, 100), Eq(0.7*n1 + 0.2*n2, 30)]
```

The last step is solve the equations using `sym.solve()`.

In [5]:

```
soln = sym.solve(eqns)
print(soln)
```

```
{n1: 20.0000000000000, n2: 80.0000000000000}
```

Putting these steps together, we have a three-step procedure for solving systems of linear equations using Sympy.

In [6]:

```
# import sympy
import sympy as sym

# Step 1. Create symbolic variables.
sym.var(['n1','n2'])

# Step 2. Create a list of equations using the symbolic variables
eqns = [
    sym.Eq(n1 + n2, 100),
    sym.Eq(0.7*n1 + 0.2*n2, 30)
]

# Step 3. Solve and display solution
soln = sym.solve(eqns)
print(soln)
```

```
{n1: 20.0000000000000, n2: 80.0000000000000}
```

Exercise

In the cell below, prepare an IPython solution to the second problem described in the screencast involving three linear equations. The problem description starts at the 2:22 mark in the screencast. You can use the example above as a template for your solution.

In [7]:

```
from IPython.display import YouTubeVideo
YouTubeVideo("KrrZB5LvXF4",560,315,start=144,end=166,rel=0)
```

Out[7]:

In [ ]:

```

```