Lesson 12

Colt Bradley

1 Introduction

The purpose of his lesson is to learn how to numerically solve systems of equations. The first step is to re-arrange systems of equations into matrix equations, like the example that follows:

$$2x + 7z = 26 \tag{1a}$$
$$3y + 5z = 45 \tag{1b}$$
$$x + 3y - z = 0 \tag{1c}$$
$$\begin{pmatrix} 2 & 0 & 7 \\ 0 & 3 & 5 \\ 1 & 3 & -1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} 26 \\ 45 \\ 0 \end{pmatrix} \tag{1d}$$

In this way, any system of n equations and variables can be represented as a $n \times n$ matrix equation. In python, there is a simple command that solves these systems. We'll demonstrate in both of the examples below.

2 Statics

The first example is a simple statics problem with a mass connected to a wall by a string. There are three unknowns, the x component of the force F_x, the y-component of the force F_y, and tension in the string T. We can write a system of equations for these forces as follows:

$$F_x - T\cos\theta = 0 \tag{2a}$$
$$T_y + T\sin\theta - mg = 0 \tag{2b}$$
$$(T\sin\theta)(l/2) - F_y(l/2) = 0 \tag{2c}$$

Next, we move constants to the left and identify which terms go with which variables. This will become our matrix equations.

$$\begin{pmatrix} 1 & 0 & -\cos\theta \\ 0 & 1 & \sin\theta \\ 0 & -\frac{l}{2} & \sin\theta\frac{l}{2} \end{pmatrix} \begin{pmatrix} F_x \\ F_y \\ T \end{pmatrix} = \begin{pmatrix} 0 \\ mg \\ 0 \end{pmatrix} \tag{3}$$

Writing a code in python, we solve using the given constants. We let $m = 14$ kg, $l = 1.2$ m, $g = 9.8$ m/s, and $\theta = 35°$. The answer we get for these values follows.

$$F_x = 144.8 \tag{4a}$$
$$F_y = 68.6 \tag{4b}$$
$$T = -1.3 \tag{4c}$$

3 Kirchoff's Laws

In this example, our system is solving a circuit using Kirchoff's laws. This yeilds four equations; one for each loop and one for conservation of current.

$$\mathcal{E}_1 - I_1R_1 - I_3R_3 = 0 \tag{5a}$$
$$\mathcal{E}_2 - I_2R_2 - I_3R_3 = 0 \tag{5b}$$
$$\mathcal{E}_2 - I_2R_2 + I_1R_1 - \mathcal{E}_1 = 0 \tag{5c}$$
$$I_1 + I_2 - I_3 = 0 \tag{5d}$$

We can quickly see that the conservation of current equation is essential, since looking at **??** we see that the matrix is singular. To avoid this, choose any two of the first three equations plus the current conservation equation.

$$\begin{pmatrix} 0 & R_2 & R_3 \\ R_1 & 0 & R_3 \\ R_1 & -R_2 & 0 \end{pmatrix} \tag{6}$$

We'll then solve these like we did in the previous example, moving anything that doesn't rely on our unknown currents (any $\mathcal{E}$) to the right hand side then building a matrix equation out of it.

$$\begin{pmatrix} R_1 & 0 & R_3 \\ 0 & R_2 & R_3 \\ 1 & 1 & -1 \end{pmatrix} \begin{pmatrix} I_1 \\ I_2 \\ I_3 \end{pmatrix} = \begin{pmatrix} \mathcal{E}_1 \\ \mathcal{E}_2 \\ 0 \end{pmatrix} \tag{7}$$

This equation can be solved with almost identical code to the first, and when we solve we get.

$$I_1 = .062 \tag{8a}$$
$$I_2 = .025 \tag{8b}$$
$$I_3 = .088 \tag{8c}$$

4 Code

```
#Colt Bradley
#2.25.16
#Homework 12

#import modules
import numpy as n

#define variables
m = 14
l = 1.2
g = 9.8
theta = 35

#Define the matricies
big = n.matrix([[1, 0 ,-n.cos(theta)],[0,1,n.sin(theta)],\
[0, -l/2., n.sin(theta)*l/2]])
col = n.matrix([[0],[m*g],[0]])

#use linear algebra package, print result
print n.linalg.solve(big,col)

##############################################################################
#Part 2
##############################################################################

#define variables, emf in Volts and r in ohms
emf_1 = 12
emf_2 = 9
r_1 = 100
r_2 = 130
```

```
r_3 = 65

#define matricies
res = n.matrix([[r_1,0,r_3],[0,r_2,r_3],[1,1,-1]])
emfs = n.matrix([[emf_1],[emf_2],[0]])

#use linear algebra package, print result
print n.linalg.solve(res,emfs)
```