Riley Payung

Jason Kinser

CDS 411

October 28th, 2020

HW Simultaneous Equations Answers

In [1]:
```
import numpy as np
```

Problem 1

Write a Python script to solve the following for x and y.

$$4x + 2y = 14$$
$$y - x = -5$$

In [2]:
```
# Code and answers
M = np.array(((4,2),(-1,1)));
M_inv = np.linalg.inv(M);
vec = np.array((14,-5));
a = M_inv.dot(vec)
print(a)
```

```
[ 4. -1.]
```

In [3]:
```
print(4*a[0] + 2*a[1])
print(a[1]-a[0])

print("Condition:",np.linalg.cond(M))
```

```
14.0
-5.0
Condition: 3.3699240762154825
```

Problem 2

Write a Python script to solve the following for x, y and z.

$$1.1y - 3x - 0.9z = 3.61$$
$$4.7z - 4.1x - 0.17y = -42.527$$
$$2.9y + 2.2z - 1.1x = 3.87$$

In [4]:
```
# Code and answers
M2 = np.array(((-3,1.1,-.9),(-4.1,-0.17,4.7),(-1.1,2.9,2.2)));
M2_inv = np.linalg.inv(M2);
vec2 = np.array((3.61,-42.527,3.87));
a2 = M2_inv.dot(vec2);
print(a2)
```

```
[ 3.2  7.1 -6. ]
```

In [7]:
```
print(1.1*a2[1]-3*a2[0]-0.9*a2[2])
print(4.7*a2[2]-4.1*a2[0]-0.17*a2[1])
print(2.9*a2[1] + 2.2*a2[2] -1.1*a2[0])

print("Condition:",np.linalg.cond(M2))
```

```
3.609999999999996
-42.527
3.8699999999999983
Condition: 2.540084175492046
```

Problem 3

Given the following equations.

$$ax + by = c$$
$$dx + ey = f$$

1. Randomly generate values for a, b, d and e that are between -1 and 1.
2. Randomly generate values for c and f.
3. Compute x and y
4. Use the given two lines of code to print out the results.

In [8]:
```
# your code
a,b,d,e = np.random.ranf(4) * 2 - 1
c,f = np.random.ranf(2) * 2 - 1


M3 = np.array(((a,b),(d,e)))
vec3 = np.array((c,f))
M3_inv = np.linalg.inv(M3);
x,y = M3_inv.dot(vec3);

print(a*x + b*y, c)
print(d*x + e*y, f)

print("Condition:",np.linalg.cond(M3))
```

```
-0.9335829384096305 -0.9335829384096306
0.1640101963414633 0.1640101963414633
Condition: 1.5124799029652138
```

Problem 4

Write a Python script to solve for x, y and z

$$4x + 6y - z = 0$$
$$-x + 2y - 2z = 2$$
$$3x + 8y - 3z = 1$$

In [9]:
```
# code and answer
M4 = np.array(((4.0000001,6,-1),(-1,2,-2),(3,8,-3)));
M4_inv = np.linalg.inv(M4);
vec4 = np.array((0,2,1));
x,y,z = M4_inv.dot(vec4);
print(x,y,z)
print("Condition Number:",np.linalg.cond(M4))
```

```
9999999.974146163 -9000000.376731548 -14000001.36380463
Condition Number: 391707870.76701885
```

Problem 5

The following situation creates a singular matrix. Can the problem be solved if we change the values on the right hand side of the equation. Explain your answer in one sentence.

$$4x + 2y - z = 0$$
$$x - 1y - 2z = 2$$
$$5x + y - 3z = 1$$

It may work. If we changed the right hand side instead of the left hand side, we would get different equations. I guess the only way to find out would be to test it, which I have done below:

In [10]:
```
M5 = (((4,2,-1),(1,-1,-2),(5,1,-3)));
vec5 = ((1,6,3));
M5_inv = np.linalg.inv(M5);
x,y,z = M5_inv.dot(vec5)
print(x,y,z)
print("Condition Number of changing the right hand side:",np.linalg.cond(M5))
```

```
1.5011998757901652e+16 -2.101679826106231e+16 1.8014398509481984e+16
Condition Number of changing the right hand side: 1.3945178717295934e+17
```

It seems like changing the vector on the right hand side does not work, since we have an extremely large number. It works just about as good as changing the value of any variable in the equations

Problem 6

Georgina has 30 coins which consists of dimes and nickels. The total amount that she has is $2.45. How many dimes and how many nickels does she have?

In [12]:
```
# your code and the number of each type of coin
M6 = np.array(((1,1),(.05,.10)));
vec6 = np.array((30,2.45));
M6_inv = np.linalg.inv(M6);
n,d = M6_inv.dot(vec6)
print(n,d);
print(.5*n+.10*d);
print(np.linalg.cond(M6))
```

```
11.0 19.0
7.4
40.22513992488611
```

Problem 7

Georgina is back again. This time she has pennies, nickels, dimes and quarters. She has 30 coins. The total amount she has is $3.92. She has one more nickel than she has pennies. The total number of dimes and nickels is 15. How many of each type of coin does she have?

In [13]:
```
# your code and the number of each type of coin
M7 = np.array(((0.01,0.05,0.10,0.25),(-1,1,0,0),(0,1,1,0),(1,1,1,1)))
vec7 = np.array((3.92,1,15,30));
M7_inv = np.linalg.inv(M7);
p,n,d,q = M7_inv.dot(vec7)
print(p,n,d,q)
print(round(1*p + 5*n + 10*d + 25*q))
print(round(1*p + 1*n + 1*d + 1*q))
print(round(n-p))
print(np.linalg.cond(M7))
```

```
4.413793103448276 5.413793103448276 9.586206896551724 10.586206896551722
392.0
30.0
1.0
16.487275589948496
```