Solving equations

Tuesday, December 29, 2020 3:35 PM

```
import numpy as np

#Consider the system of LINEAR equations
#x + y + z = 6
#2y + 5z = -4
#2x + 5y - z = 27

#This is the same as
#|1 1  1| |x| = | 6|
#|0 2  5| |y| = |-4|
#|2 5 -1| |z| = |27|
#"A"     "X" = "B"

#We can solve for X=A^-1 B

A = np.array([[1,1,1],[0,2,5],[2,5,-1]])
B = np.array([[6],[-4],[27]])
B = np.array([6,-4,27])

X = np.linalg.solve(A,B)
print(X)

# %%
#What about NON-LINEAR equations

import numpy as np
from scipy.optimize import fsolve

#How are the solutions?
#x^2 + y = 5
#x^2 + y^2 = 7


#First what is graphical solution?
import matplotlib.pyplot as plt

fig = plt.figure(1,figsize=(5,5))
delta = 0.025
x, y = np.meshgrid(
   np.arange(-4, 4.1, delta),
   np.arange(-4, 4.1, delta)
)
f1 = x**2 + y-5
f2 = x**2+y**2-7

# plt.contour(x, y,x**2 + y - 5,[0])
# plt.contour(x, y,x**2 + y**2 - 7,[0])
plt.contour(x,y,f1,[0])
```

```
plt.contour(x,y,f2,[0])
plt.show()


#here is one way to return a single root
def myFunction(z):
  x = z[0]
  y = z[1]
  F = np.empty((2))
  F[0] = x**2 + y-5
  F[1] = x**2+y**2-7
  return F

zGuess = np.array([3,-2])
z = fsolve(myFunction,zGuess)
print(z)

# %%
#Using gekko
from gekko import GEKKO
m = GEKKO()
x,y = [m.Var(1) for i in range(2)]
m.Equations([x**2+y==5,\
          x**2+y**2==7])
m.solve(disp=False)
print(x.value,y.value)

# %%
import sympy as sym
sym.init_printing()
x,y = sym.symbols('x,y')
f = sym.Eq(x**2+y,5)
g = sym.Eq(x**2+y**2,7)


print(sym.solve([f,g],(x,y)))
```