1 Ill-conditioned Normal Equation

...

Solution:

```
import numpy as np
import scipy.linalg as linalg
from time import time

def A_ill(m,n,delta, eps):
    A = np.random.rand(m,n)
#    aux = (np.linspace(0,100,m))[np.newaxis].T
#    A = aux**(list(range(n)))
    A = A + delta * np.eye(m,n)
    # A is spd and thus invertivle, thus all cols independent
    # now change last col to a perturbed version of col 1
    A[:,-1] = A[:,0] + eps*A[:,-1]
    # now last and first col are nearly the same but A still invertible
    # the consequence: one eigval is close to zero and therefore cond(A) >>> 0
    return A

if __name__ == "__main__":
    # params
    m, n = 9000, 100
    delta, eps = 0.001, 1e-01
    runs = 1

    # original system

    x, y = np.mgrid[-1:1:.01, -1:1:.01]


    import matplotlib.pyplot as plt


    n = 8000
    sigma=0.05
    noise = np.random.normal(0,sigma,n)
    z = np.array([-0.5] + list(noise) + [0.5])

    p = [0,6,7,8,9,15]
#    p = list(range(n))
    z_i = z[np.newaxis].T
    b = np.array([-0.5] + list(np.random.normal(0,sigma,n)) + [0.5])
    A = z_i**p
    m,n = np.shape(A)
    plt.plot(z,b,'rx')

    print("method   ||Ax-b||")


    for i in range(runs):
        print("run", i)
#        A = A_ill(m,n,delta, eps)
```

```
#         b = np.random.rand(m)
        print("cond(A) =%10.2e"%(np.linalg.cond(A.T@A)))

        Q, R = linalg.qr(A)
        x1 = linalg.solve_triangular(R[0:n, 0:n], Q[0:m, 0:n].T@b)
        print("QR      %10.2e"  %(np.linalg.norm(A@x1 - b)) )

        x2 = linalg.lstsq(A,b)[0]
        print("SVD     %10.2e"  %(np.linalg.norm(A@x2 - b)) )

        x3 = linalg.solve(A.T@A, A.T@b, assume_a = "pos")
        print("direct %10.2e"  %(np.linalg.norm(A@x3 - b)) )

        Z = np.linspace(-0.5-sigma, 0.5+sigma, 150) # other z values
        Y1 = (Z[np.newaxis].T**p).dot(x1) # evaluate model on Z
        Y2 = (Z[np.newaxis].T**p).dot(x2) # evaluate model on Z
        Y3 = (Z[np.newaxis].T**p).dot(x3) # evaluate model on Z
        plt.figure()
        plt.title("Polynomial = " + str(p))
        plt.plot(z_i, b, 'ro')
        plt.plot(Z, Y1, 'b')
        plt.plot(Z, Y2, 'r')
        plt.plot(Z, Y3, 'g')
        plt.legend(["(Z,Y)","QR","SVD", "LL"])
        plt.show()
```