TD2_CORRIGE

January 26, 2019

```
In [1]: from scipy import stats
        #from statsmodels.graphics.tsaplots import plot_acf
        import random


        # =======================================
        #               CONSTANTES
        # =======================================
        RANGE_EXO_3 = 10
        SEED_EXO_3  = 1337

        # =======================================
        #                FUNCTIONS
        # =======================================

        # =======================================
        #                EXERCICE 1
        # =======================================
        def exo1(X0, a, c, m, size, affiche=False):
                """Methode Congruentielle Lineaire
                """

                # ====== VARIABLES ====== #
                X       = [X0]
                Ri      = -1
                periode = -1
                trouve  = False

                # ====== ALGORITHME ====== #
                for i in range(1, size):
                        X.append(0.0) # C'est la valeur Xi
                        X[i] = (a*X[i-1] + c) % m
                        Ri = X[i] / m

                        if( (X[i] == X0) and not(trouve) ):
                                periode = i
                                trouve  = True
                        #END_IF
                if(affiche):
                        print(Ri)
                #END_IF
        #END_FOR

        # ====== OUTPUT ====== #
        print("Periode(%d, %d, %d, %d) = %d" % (X[0], a, c, m, periode))
        return X
#END_DEF


# =======================================
#               EXERCICE 2
# =======================================
def exo2(m1, affiche, *generateurs):
        """Methode de Congruences lineaires combinees
        """

        # ====== VARIABLES ====== #
        X       = [0.0]
        nb_gen  = len(generateurs)
        nb_Xi   = len(generateurs[0])
        Xi      = -1
        Ri      = -1
        periode = -1
        trouve  = False

        # ====== ALGORITHME ====== #
        for i in range(nb_Xi):
                for j in range(1, nb_gen):
                        X[i] += ( ( (-1)**(j-1) ) * generateurs[j][i] )
                #END_FOR

                X[i] %= (m1 - 1)

                if(X[i] > 0):
                        Ri = X[i] / m1
                else:
                        Ri = (m1 - 1) / m1
                #END_IF

                if( (X[i] == X[0]) and not(trouve) and i!=0 ):
                        periode = i
                        trouve  = True
                #END_IF

                if(affiche):
                        print(Ri)
                #END_IF

                X.append(0.0)
        #END_FOR_i

        # ====== OUTPUT ====== #
        print("Periode(Xk, m1 = %d) = %d" % (m1, periode))
        return 0
#END_DEF


# ========================================
#               EXERCICE 3
# ========================================

def exo3(seed=False):
        """Generation de nombres aleatoires
        """

        # ===== VARIABLES ===== #
        x1     = []
        x2     = []
        x3     = []
        result = []

        # Le seed permet "d'influencer" la RNG
        if(seed):
                random.seed(SEED_EXO_3)
        #END_IF

        print("===== Random.uniform(0,1) =====")
        for i in range(RANGE_EXO_3):
                x1.append(random.uniform(0,1))
                print(x1[i])
        #END_FOR

        print("\n====== stats.uniform.rvs(RANGE) =====")
        x2 = stats.uniform.rvs(size=RANGE_EXO_3)
        for nombre in x2:
                print(nombre)
        #END_FOR

        print("\n===== random.randrange() =====")
        for i in range(RANGE_EXO_3):
                x3.append(random.randrange(1,101,1))
                print(x3[i])
        #END_FOR
        print("\n===== random.randint() =====")
        for i in range(RANGE_EXO_3):
                print(random.randint(1, 100))
        #END_FOR

        print("\n ===== Test de KS =====")
        result = stats.kstest(x1, 'uniform')
        print(result)
        result = stats.kstest(x2, 'uniform')
        print(result)
        result = stats.kstest(x3, 'uniform')
        print(result)

        print("\n ===== Test de Chi Carre =====")
        result = stats.chisquare(x1)
        print(result)
        result = stats.chisquare(x2)
        print(result)
        result = stats.chisquare(x3)
        print(result)

        print("\n ===== Test de chi2_contingency =====")
        result = stats.chi2_contingency(x1)
        print(result)
        result = stats.chi2_contingency(x2)
        print(result)
        result = stats.chi2_contingency(x3)
        print(result)

#END_DEF


# =======================================
#                  MAIN
# =======================================

if(__name__ == '__main__'):
        print("====== EXERCICE 1 ======")
        exo1(99, 798, 394, 5967, 543)
        exo1(1, 3, 0, 7, 543)

        print("\n====== EXERCICE 2 ======")
        X1 = exo1(2, 3, 0, 13, 100)
        X2 = exo1(5, 5, 0, 11, 100)
        print("")
        exo2(13, False, X1, X2)
```

```
                print("\n====== EXERCICE 3 ======")
                exo3()

                print("\n====== EXERCICE 4 ======")
        #END_IF
```

```
====== EXERCICE 1 ======
Periode(99, 798, 394, 5967) = -1
Periode(1, 3, 0, 7) = 6

====== EXERCICE 2 ======
Periode(2, 3, 0, 13) = 3
Periode(5, 5, 0, 11) = 5

Periode(Xk, m1 = 13) = 5

====== EXERCICE 3 ======
===== Random.uniform(0,1) =====
0.8870528953548551
0.12600109130128578
0.809051974290405
0.005626228623497376
0.734128422711549
0.2842283437398473
0.5809061795116536
0.9616406707645044
0.09956087922298051
0.45335105499651784

====== stats.uniform.rvs(RANGE) =====
0.33923023426532506
0.5300982049784521
0.544406343490097
0.10235890631901123
0.9949171391692703
0.6854160893076298
0.6158711867619862
0.5736543318339761
0.45117727503547667
0.2642099101095535

===== random.randrange() =====
99
36
5
14
87
22
```

```
80
6
52
39

===== random.randint() =====
61
32
85
79
2
93
71
6
75
40

 ===== Test de KS =====
KstestResult(statistic=0.1739989086987142, pvalue=0.9225827344670295)
KstestResult(statistic=0.21458391069237026, pvalue=0.6979760660281604)
KstestResult(statistic=1.0, pvalue=0.0)

 ===== Test de Chi Carre =====
Power_divergenceResult(statistic=2.2519686668862295, pvalue=0.9868278505437748)
Power_divergenceResult(statistic=1.0624072672993656, pvalue=0.9992797064814476)
Power_divergenceResult(statistic=242.54545454545453, pvalue=3.733027569300815e-47)

 ===== Test de chi2_contingency =====
(0.0, 1.0, 0, array([0.8870529 , 0.12600109, 0.80905197, 0.00562623, 0.73412842,
       0.28422834, 0.58090618, 0.96164067, 0.09956088, 0.45335105]))
(0.0, 1.0, 0, array([0.33923023, 0.5300982 , 0.54440634, 0.10235891, 0.99491714,
       0.68541609, 0.61587119, 0.57365433, 0.45117728, 0.26420991]))
(0.0, 1.0, 0, array([99., 36.,  5., 14., 87., 22., 80.,  6., 52., 39.]))

====== EXERCICE 4 ======
```