BIICHLE Dorian
PIASENTIN Béryl

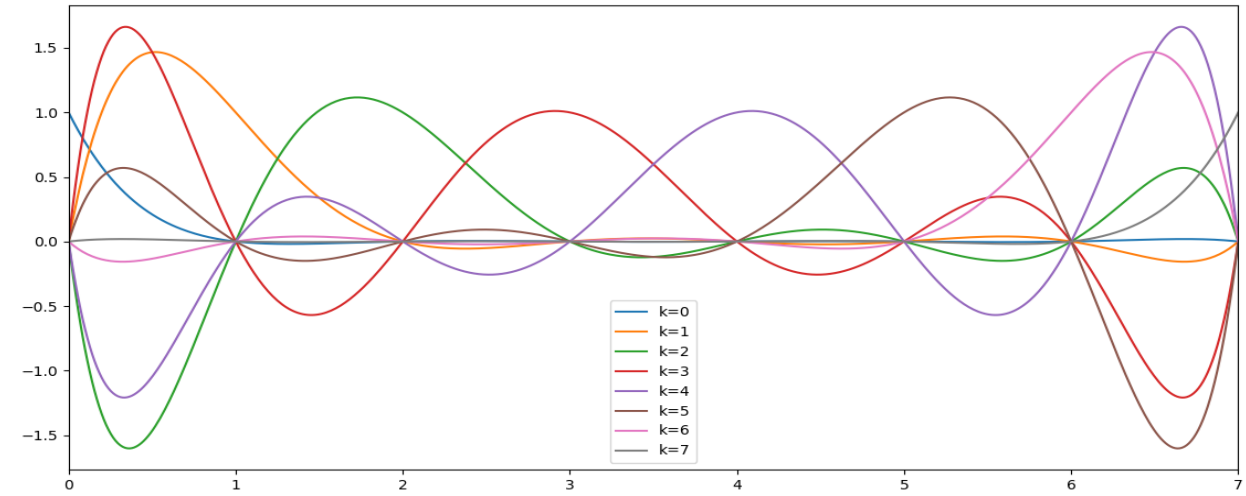
TP2 : interpolation de Lagrange – base de Lagrange

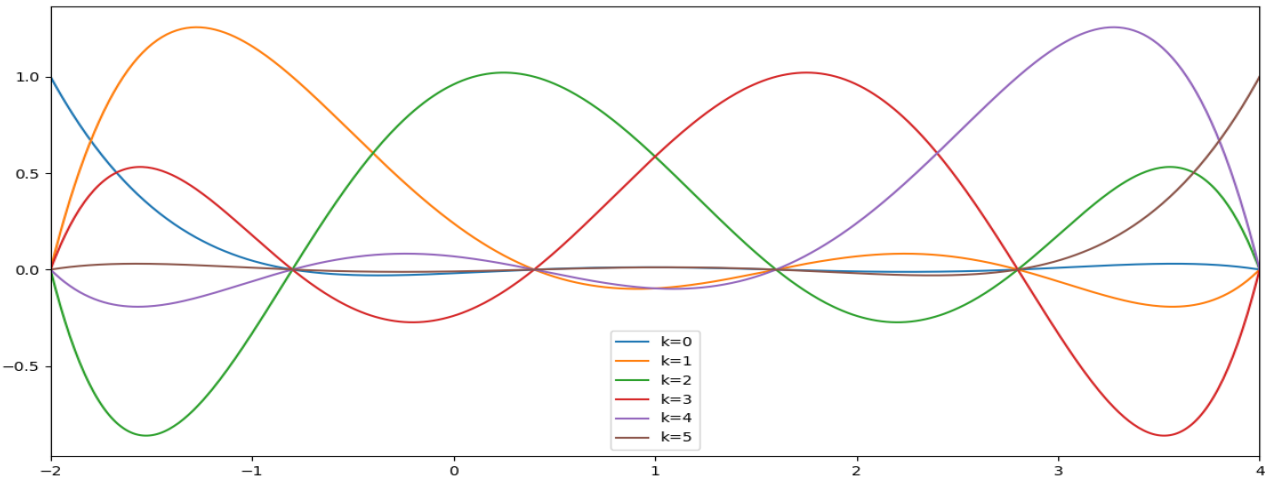
Exercice 1

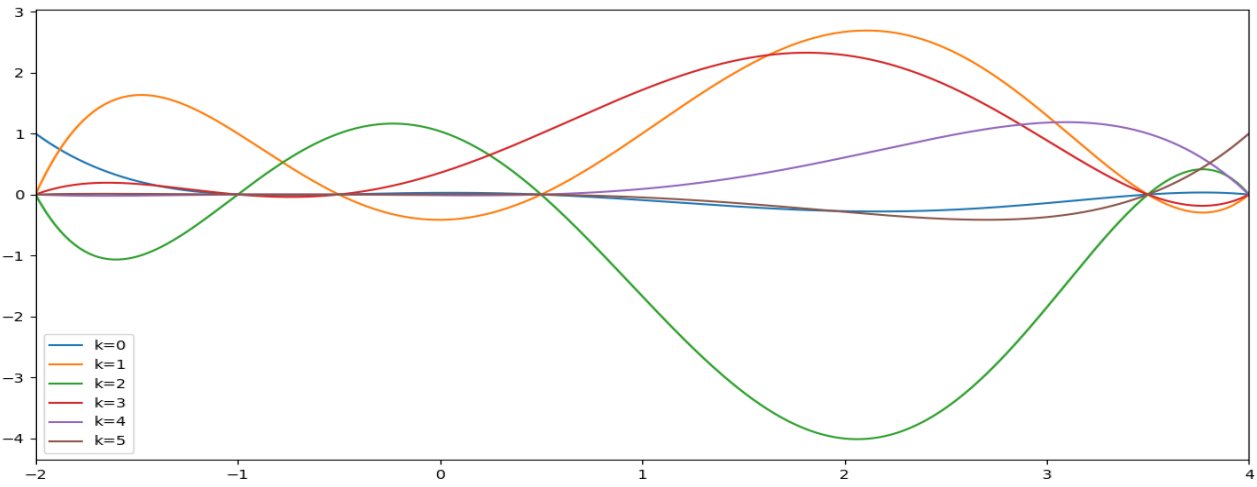
```
#####
# 1 #
#####
import matplotlib.pyplot as plt
import numpy as np
plt.close('all')

def Lk(x,xi,k):
    # x vecteur de 500 valeurs
    # xi vecteur des points d'interpolation
    # k-ième polynôme de Lagrange

    n = np.size(xi)
    y = np.ones(np.size(x))
    for i in range(n):
        if (i!=k):
            y *= (x - xi[i]) / (xi[k] - xi[i])
    return y
```

```
def ex1():
    plt.figure()
    x = np.linspace(0,7,500)
    xi = np.linspace(0,7,8)

    for j in range(8):
        y = Lk(x,xi,j)
        nom = "k="+str(j)
        plt.plot(x,y,label = nom)

    plt.xlim(x[0], x[-1])
    plt.legend()

    plt.figure()
    x = np.linspace(-2,4,500)
    xa = np.linspace(-2,4,6)

    for j in range(6):
        y = Lk(x,xa,j)
        nom = "k="+str(j)
        plt.plot(x,y,label = nom)

    plt.xlim(x[0], x[-1])
    plt.legend()

    plt.figure()
    xb = [-2,-1, -0.5, 0.5,3.5,4] # on garde le même x

    for j in range(6):
        y = Lk(x,xb,j)
        nom = "k="+str(j)
        plt.plot(x,y,label = nom)

    plt.xlim(x[0], x[-1])
    plt.legend()
```

Graphiques obtenus en appelant la fonction ex1 :



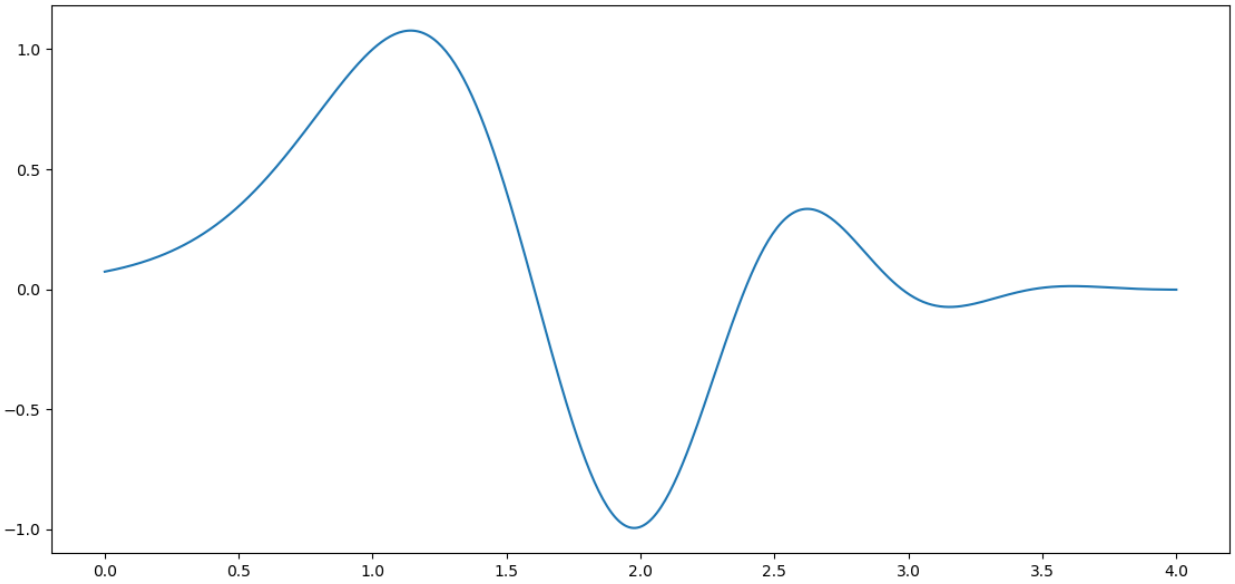




Le premier graphique montre les 8 premiers polynômes de Lagrange, le second les 6 premiers, et le dernier montre les 6 premiers également mais avec des points qui ne sont pas espacés régulièrement.

Exercice 2

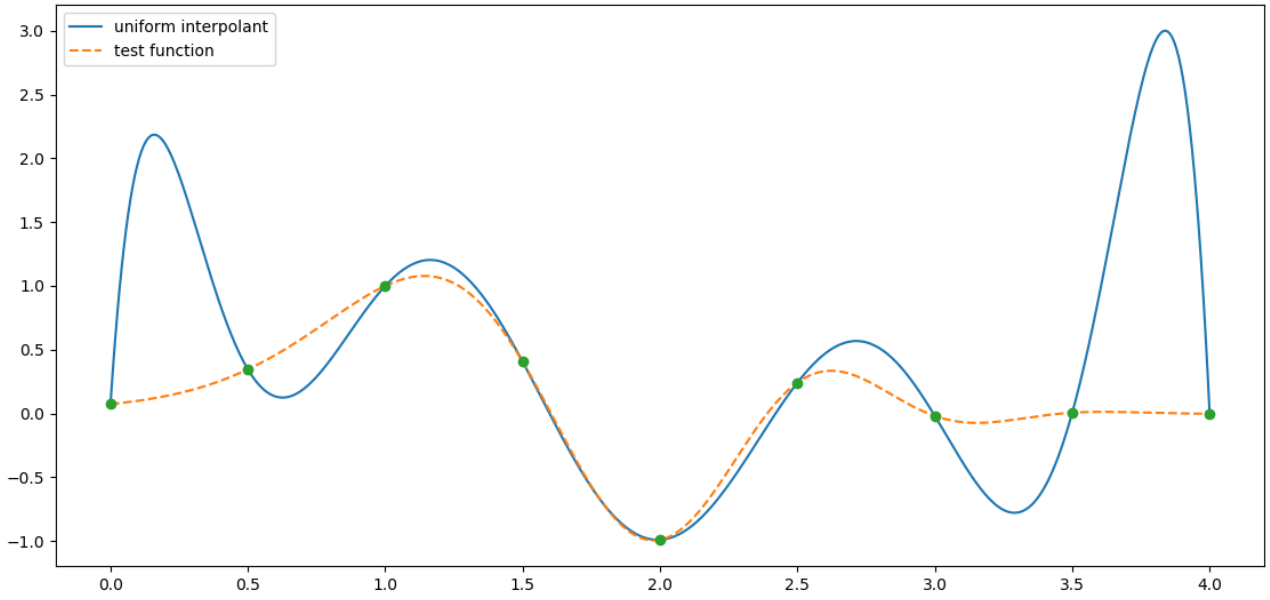
```
#####
# 2 #
#####

def f(x):
    return np.cos(1-x**2)*np.exp(-x**2+3*x-2)

def plotf():
    x = np.linspace(0,4,500)
    plt.figure()
    plt.plot(x,f(x),label = "test function f")

def ex2() :
    plotf()
```



Ce graphique obtenu avec la fonction ex2 est la représentation de f.

Exercice 3

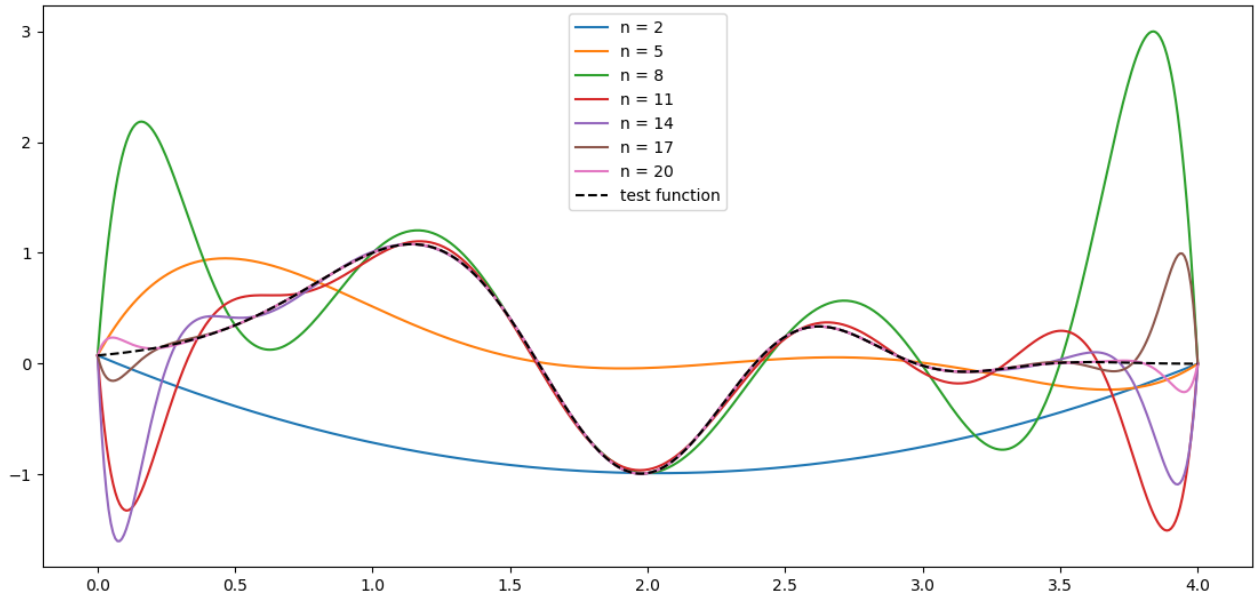
```
#####
# 3 #
#####

def interpolation(f,n,a,b):
    # f est la fonction que l'on veut interpoler
    # n est le degré du polynôme d'interpolation que l'on souhaite
    # [a,b] est l'intervalle sur lequel on veut interpoler f
    px = 0
    x = np.linspace(a,b,500)
    xi = np.linspace(a,b,n+1)
    fxi = f(xi)
    for k in range(n+1):
        px += fxi[k] * Lk(x,xi,k)

    return x,px,xi,fxi

def ex3():
    #Interpolation de f par un polynôme de Lagrange de degré 8
    x,px,xi,fxi = interpolation(f,8,0,4)
    plt.figure()
    plt.plot(x, px,label="uniform interpolant")
    plt.plot(x,f(x),"--",label="test function")
    plt.plot(xi,fxi,"o")
    plt.legend()
```



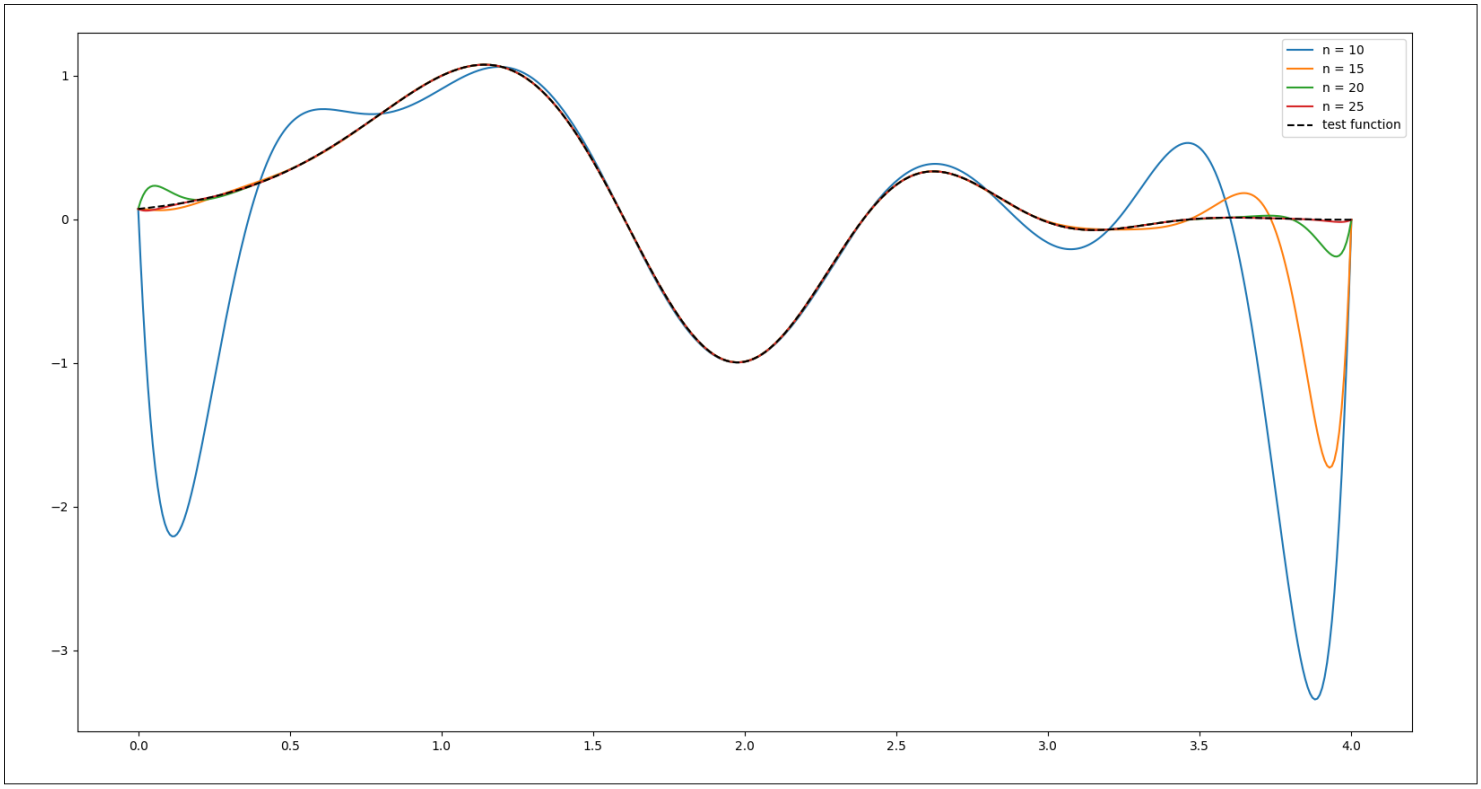
Le graphique obtenu contient la fonction f (en pointillés), le polynôme de degré 8 qui l'interpole, et les points d'interpolation.

Exercice 4

```
#####
# 4 #
#####

def ex4():
    plt.figure()
    #Affichage des polynômes de degré 2 à 20, par pas de 3,
    #interpolant la fonction f
    a = 0
    b = 4
    for n in range(2, 21, 3):
        x,px,_,_ = interpolation(f,n,a,b)
        plt.plot(x, px, label="n = {}".format(n))
    plt.plot(x,f(x),"--k",label="test function")
    plt.legend()
```

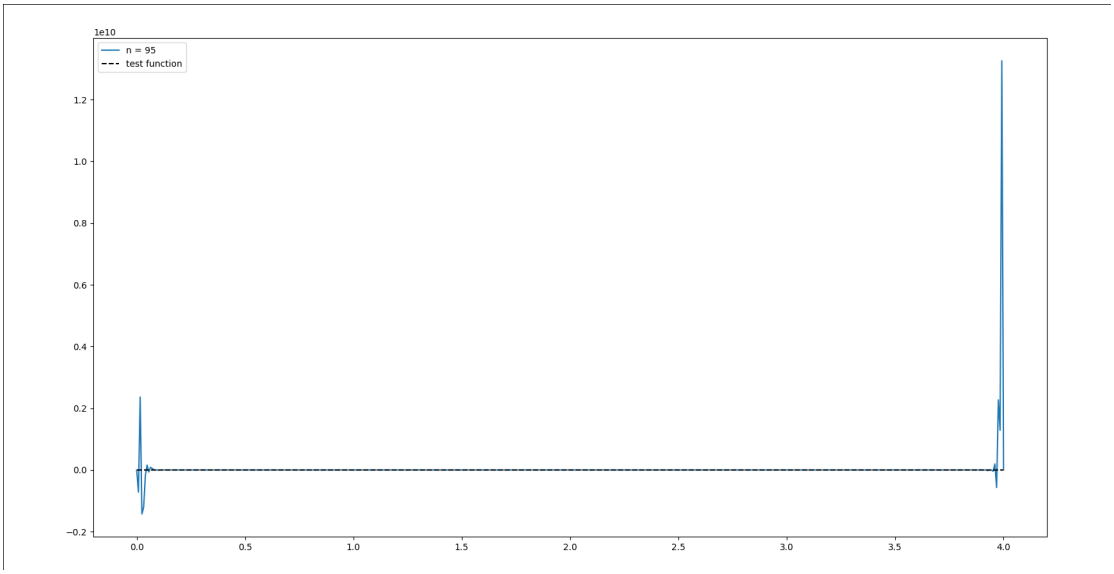


Ce graphique est obtenu en appelant la fonction ex4. Il contient l'interpolation de la fonction f par des polynômes de Lagrange du degré 2 au degré 20, par pas de 3.



Voici la même interpolation, mais avec des polynômes d'interpolation du degré 10 au degré 25 par pas de 5. Nous pouvons remarquer que l'interpolation par le polynôme de degré 10 donne une courbe avec des variations assez prononcées sur les cotés.

De plus, plus le degré augmente, plus le polynôme est proche de la fonction sur la partie centrale, mais on constate toujours de gros écarts sur les bords de l'intervalle.

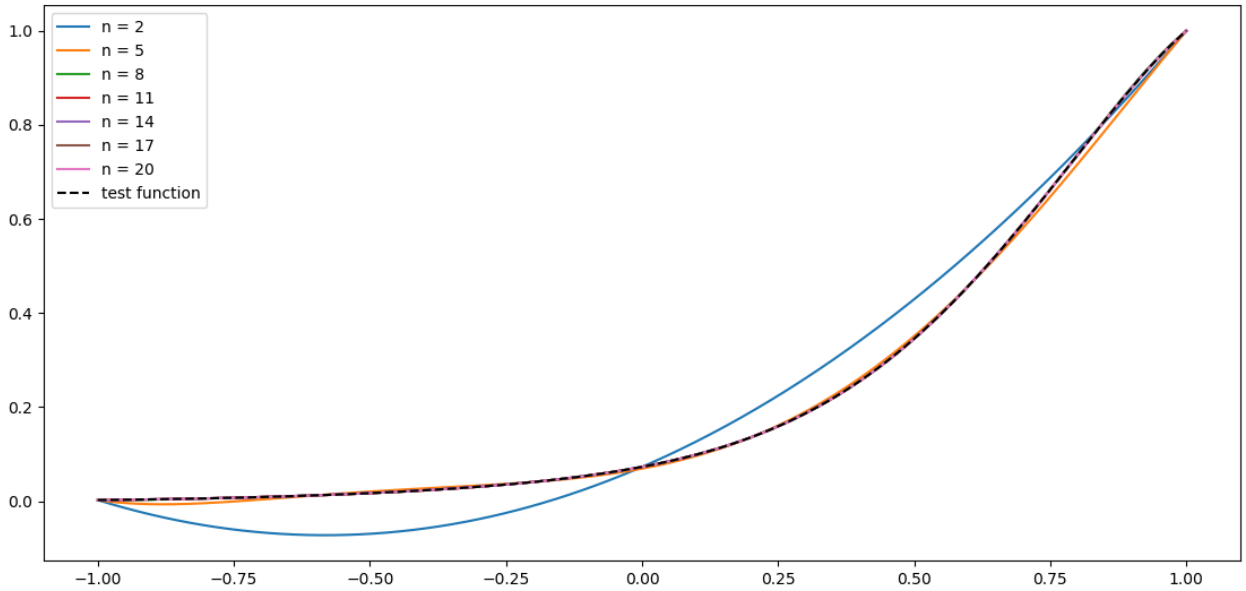


Enfin, voici toujours l'interpolation de la même fonction mais cette fois-ci par un polynôme de degré 95. L'écart du polynôme par rapport à la fonction sur les bords de l'intervalle est ici très conséquent, jusqu'à une pointe d'ordonnées de plus de 12 000 000 000 sur le coté droit !

Nous pouvons en conclure qu'augmenter le degré du polynôme d'interpolation n'a pas forcément pour effet d'avoir une meilleur approximation de la fonction sur l'intervalle entier, mais uniquement sur le centre de celui-ci (du moins avec des point d'abscisses équidistantes pour l'interpolation).

Exercice 5

```
#####
# 5 #
#####

def g(x):
    return 1/(1+25*x*x)

def ex5():
    plt.figure()
    #Affichage des polynômes de degré 2 à 20, par pas de 3,
    #interpolant la fonction g
    a = -1
    b = 1
    for n in range(2, 21, 3):
        x,px,_,_ = interpolation(f,n,a,b)
        plt.plot(x,px,label="n = {}".format(n))
    plt.plot(x,f(x),"--k",label="test function")
    plt.legend()
```



Ce graphique est obtenu en appelant la fonction ex5. Il contient l'interpolation de la fonction g avec des polynômes de Lagrange variants du degré 2 au degré 20 par pas de 3.
Nous pouvons observer que le degré 2 est le seul à vraiment se détacher de la courbe de la fonction originale.
On n'observe pas le même phénomène que pour la fonction précédente : ici, plus le degré du polynôme d'interpolation augmente, plus il se rapproche de la fonction de base sur tout l'intervalle.