

```
from scipy.integrate import dblquad
from Zernike import *
from Représentation_plan import *

def ps(f,g,domaine_rho,domaine_phi):
    return dblquad(lambda rho, phi : 1/(math.pi)*f(rho,phi)*g(rho,phi)*rho,
domaine_phi[0], domaine_phi[1], lambda rho : domaine_rho[0],
lambda rho : domaine_rho[1], epsabs=1e-5, epsrel=1e-5)[0]

plan_z1=calc_z_centre(t,200,5,5)
lx=[i*0.5 for i in range (m1)]
ly=[j*0.5 for j in range (n1)]

for i in range (6):

    z1=interpole2v(lx,ly,plan_z1)
    projection_zernike = []
    for n in range(i+1):
        pr = []
        for m in range(-n,n+1,2):
            Zernike_xy(0,0,m,n) #Calcul du polynômes Z(n,m)
            prod = ps(Zernike,plan1_polaire_decale,[0,1],[0,2*math.pi])/ps(Zernike,
Zernike,[0,1],[0,2*math.pi]) #Porjection du front d'onde sur Z(n,m)
            pr.append(prod)
        projection_zernike.append(pr)

def resultat(x,y):
    res = 0
    for n in range(i+1):
        k = 0
        for m in range(-n,n+1,2):
            Zernike_xy.n, Zernike_xy.m = n,m
            res += (projection_zernike[n][k])*Zernike_xy(x,y)
            k += 1
    return res

ax = Axes3D(plt.figure())
resultat = np.vectorize(resultat)
R = np.arange(0,1,0.05)
Phi = np.arange(0,2*math.pi+0.1,0.05)
R,P = np.meshgrid(R, Phi)
X , Y = R*np.cos(P) , R*np.sin(P)
Z = resultat(X,Y)
ax.plot_surface(Y, X, Z, cmap=cm.coolwarm)
plt.xlabel('x', color = 'red')
plt.ylabel('y', color = 'red')
plt.show()
```