TD2

Première étape. Construire sous Python un cylindre de rayon $R = 0.2$ de paramétrisation suivante :

$$\begin{array}{rccc} f_0 : & [0,1]\times[-0.5,0.5] & \longrightarrow & \mathbb{R}^3 \\ & (x_1,x_2) & \longmapsto & (R\cos(2\pi x_1), R\sin(2\pi x_1), x_2) \end{array}$$

1. Soit $x_1 \times x_2 = [0,1]\times[-0.5,0.5]$. Utiliser **np.meshgrid(x1,x2)** pour définir la grille de valeur sur laquelle on va évaluer f_0.
2. Utiliser **ax.plot_surface(v1,v2,v3)** pour générer le cylindre.

Deuxième étape. Corruguer le cylindre.
3. Choisir une direction dans laquelle vous voulez ajouter des corrugations comme ∂_1, ∂_2, ∂_{1+2},...(si vous choisissez ∂_1 alors $u = Nx_1$ dans $\Gamma(x,u)$), et calculer $\partial_i f_0$.
3bis. Calculer $t = \frac{\partial_i f_0}{\|\partial_i f_0\|}$ et le vecteur normal n de f_0.
4. Utiliser les corrugations de Thurston.

Troisième étape. Corruguer un cône. Soit

$$\begin{array}{rccc} f_0 : & [0,1]\times[-0.5,0.5] & \longrightarrow & \mathbb{R}^3 \\ & (x_1,x_2) & \longmapsto & (x_2\cos(2\pi x_1), x_2\sin(2\pi x_1), x_2) \end{array}$$

5. Construire un cône.
6. Dans quelle direction utiliser les corrugations ?
7. Choisir un champ de vecteurs t_1 qui permettra de désingulariser le cône, ainsi qu'un champ de vecteurs n.
8. Choisir un champ de vecteur t_1 pour désingulariser localement un cône.