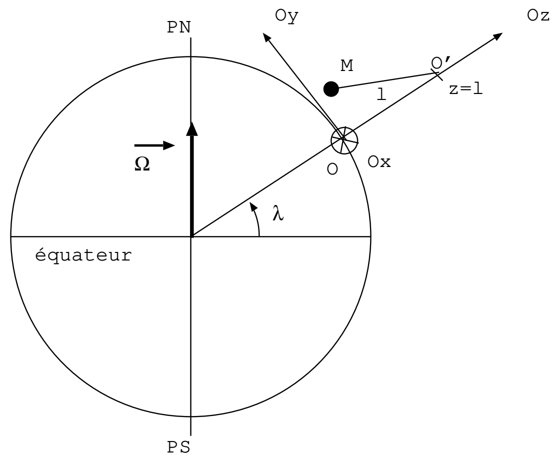
CCB modélisation

Le sujet comprend deux problèmes indépendants à traiter dans l'ordre de votre choix. Vous trouverez en fin de sujet ne annexe pour l'aide syntaxique à python. Il est demandé de numéroter les pages au format i/N où N est le nombre total de pages et i le numéro de la page.

Tout résultat doit être justifié et il est demandé de prendre soin de la présentation et de la rédaction.

I. **Pendule de Foucault**

Du point de vue historique, le pendule de Foucault est un pendule simple de longueur l, de masse m et écarté initialement de la position verticale d'une distance a. On note O' le point d'attache du pendule au plafond situé à la côté $z = l$ par rapport au plan horizontal Oxy. Oz désigne la verticale ascendante, Ox est tangent à un parallèle dirigé vers l'est et Oy est tangent à un méridien dirigé vers le nord. On note λ la latitude du lieu. Donnée: $g = 9,8\ m.s^{-2}$.



On écarte le pendule d'une distance a très petite par rapport à sa longueur. Cela a pour conséquence que le mouvement de la masse est quasiment plan, on fait donc l'hypothèse que le vecteur position de M s'écrit: $\overrightarrow{OM} = x(t)\overrightarrow{e_x} + y(t)\overrightarrow{e_y}$.

1. On note $\overrightarrow{\Omega_T}$ le vecteur associé à la rotation propre de la Terre autour de l'axe des pôles. Donner la valeur numérique de Ω_T et exprimer $\overrightarrow{\Omega_T}$ en fonction des données et des vecteurs de base $\overrightarrow{e_y}$ et $\overrightarrow{e_z}$.

2. Le référentiel terrestre n'est pas galiléen. Faire un bilan des forces exercées sur le pendule et montrer que la force d'inertie de Coriolis s'écrit $\overrightarrow{F_{ic}} = 2m\omega(\dot{y}\overrightarrow{e_x} - \dot{x}\overrightarrow{e_y}) + 2m\omega'\dot{x}\overrightarrow{e_z}$. Exprimer ω et ω' en fonction de Ω_T et λ. Pourquoi la force d'inertie d'entraînement n'intervient-elle pas explicitement dans le bilan des forces?

On admet que la tension du fil $\overrightarrow{T_f}$ avec les approximations faites peut s'écrire: $\overrightarrow{T_f} = -\dfrac{T_f}{l}(x\overrightarrow{e_x} + y\overrightarrow{e_y}) + T_f\overrightarrow{e_z}$.

3. Sur Oz, on néglige la force d'inertie de Coriolis par rapport au poids, en déduire l'expression de T_f.

4. Montrer que $x(t)$ et $y(t)$ vérifie un système d'équations couplées de la forme: $\ddot{x} = -\omega_0^2 x + 2\omega\dot{y}$ et $\ddot{y} = -\omega_0^2 y - 2\omega\dot{x}$.

Exprimer ω_0 et ω en fonction des données.

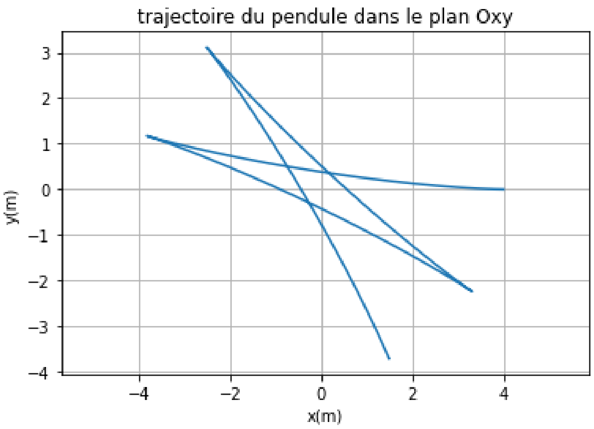
5. On résout ce système d'équations différentielles avec la fonction odeint du module scipy.integrate. L'utilisation de la fonction odeint nécessite d'écrire le système différentiel sous la forme $\dot{X} = F(X, t)$. On pose $X = [x, y, \dot{x}, \dot{y}]$ et $\dot{X} = F(X, t) = [\dot{x}, \dot{y}, \ddot{x}, \ddot{y}]$.

On donne le code permettant de résoudre les équations du mouvement et de tracer la trajectoire du pendule.

```python
import matplotlib.pyplot as plt
import numpy as np
from scipy.integrate import odeint

#Données numériques
m,l,g,OmegaT=30,67,9.8,2*np.pi/24/3600
x0,y0,v0x,v0y=4,0,0,0
omega0=(g/l)**0.5
T0=2*np.pi/omega0
omega=OmegaT*500

#Résolution des équations différentielles
t=np.linspace(0,2*T0,10000)
def eq(X,t):
    return  instruction 1
sol=odeint(eq,(x0,y0,v0x,v0y),t)
x=sol[:, instruction 2 ]
y=sol[:, instruction 3 ]

#Tracé de la trajectoire soit y en fonction de x
plt.plot( instruction 4 )
plt.title( instruction 5 )
plt.axis('equal')
plt.xlabel( instruction 6 )
plt.ylabel( instruction 7 )
plt.grid()
plt.show()
```

5.a. Exprimer les termes de $F(X, t)$ en fonction de $X[0]$, $X[1]$, $X[2]$, $X[3]$. En déduire les instructions 1, 2, et 3 (à écrire sur votre copie).

L'exécution du code donne la courbe suivante:

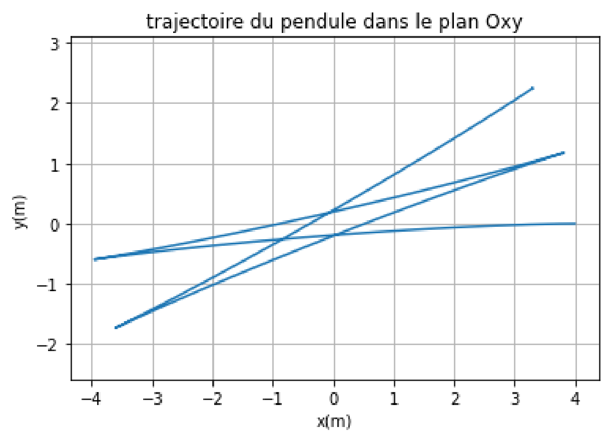


5.b. Déduire de la courbe les instructions 4, 5, 6 et 7 (à écrire sur votre copie).

5.c. On observe que le plan d'oscillations du pendule tourne au cours du temps. Déduire des conditions initiales, la position du pendule à $t = 0$ sur la courbe et donné les coordonnées du pendule à la fin de la simulation. En déduire le sens de rotation du plan d'oscillations au cours du temps. Evaluer l'angle α dont a tourné le plan d'oscillations du pendule. Donner α en radian.

5.d. Calculer à l'aide des valeurs numériques données dans le code, la période propre T_0 du pendule et la durée d'acquisition T de la courbe. Calculer la pulsation ω. En déduire la valeur numérique de $-\omega T$ et la comparer à α. Conclure.

5.e. On fait une simulation en prenant $\omega = -OmegaT * 250$ (ligne 10). On obtient la trajectoire suivante pour le pendule. De quel angle α' a tourné le plan d'oscillations? Dans quel sens a tourné le plan d'oscillations du pendule? Vérifier la cohérence avec la valeur de ω choisie pour cette simulation.



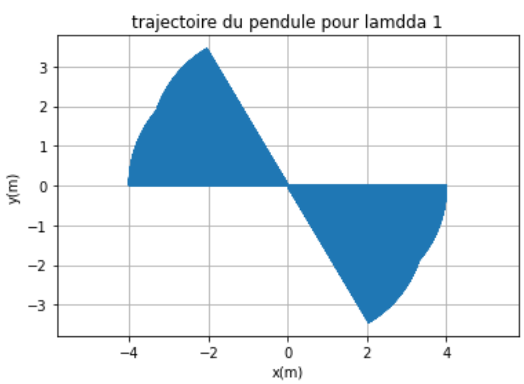
6. On réalise maintenant les simulations de la trajectoire du pendule pour des valeurs réelles de ω à savoir $\omega = \Omega_T \sin\lambda$ (relation établie question 2).

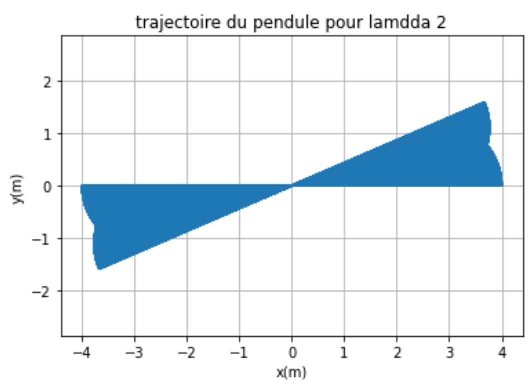
On modifie deux lignes du code précédent de la façon suivante:

ligne 1O omega=OmegaT*np.sin(lamba)

ligne 13 t=np.linpspace(0,1000*T0,10000)

On obtient deux courbes pour deux valeurs λ_1 et λ_2 de la latitude du pendule.





Déduire de ces courbes les valeurs numériques de λ_1 et λ_2. Préciser dans chaque cas l'hémisphère où a lieu l'expérience.

7. Calculer la période du plan d'oscillations d'un pendule de Foucault installé au pôle nord. Dans quel sens tourne la plan d'oscillations? Même questions pour un pendule au pôle sud.

8. Décrire le mouvement du pendule de Foucault installé à l'équateur.

II. **Etude de la circulation routière**

On étudie la circulation sur une route à une seule voie confondue avec l'axe Ox. Les voitures sont identiques et ont pour longueur L_0.

On adopte un modèle continu pour décrire la circulation routière, cela revient donc à regarder l'évolution du trafic sur des distances grandes devant la longueur L_0 d'une voiture.

On utilise les notations suivantes:

$v(x,t)$ désigne la vitesse moyenne des véhicules à la position x à l'instant t. On considère que les véhicules se déplacent selon l'axe Ox dans le sens des x croissants.

$c(x,t)$ (en véhicules par mètre) désigne la concentration de véhicules par unité de longueur de route à l'instant t et à la position x. Sur une longueur L_0, il y a, au plus, un seul véhicule, soit $c(x,t) \leq c_{max} = \dfrac{1}{L_0}$.

$q(x,t)$ (en véhicules par seconde) désigne le débit de véhicules, c'est-à-dire le nombre de véhicules par unité de temps traversant la section de la route située à la position x.

1. Donner la relation entre $q(x,t)$, $c(x,t)$ et $v(x,t)$.

2. On considère le système élémentaire constitué de la portion d'autoroute comprise entre x et $x+dx$, on suppose qu'il n'y a ni perte, ni création de véhicules.

Exprimer $N(t)$ et $N(t+dt)$, le nombres de véhicules présents dans le système aux instants t et $t+dt$.

Exprimer δN_e et δN_s, les nombres de véhicules qui entrent et qui sortent du système entre les instants t et $t+dt$.

Déduire de la conservation du nombre de véhicules, l'équation aux dérivées partielles: $\dfrac{\partial c}{\partial t} = -\dfrac{\partial q}{\partial x}$ $(*)$.

Pour comprendre comment évoluent la concentration, la vitesse moyenne ou le débit de véhicules au cours du temps sur la route, il convient donc de résoudre cette équation aux dérivées partielles à partir de la situation initiale.

Afin de résoudre numériquement l'équation $(*)$, nous avons besoin de discrétiser les variables de temps et d'espace. On choisit les paramètres suivants :

- pas d'espace (en mètres) : dx
- pas de temps (en secondes) : dt
- longueur de la portion de route étudiée : $Lr = N_x dx$
- durée de simulation : $Temps = N_t dt$

On introduit un tableau C des concentrations. Les termes de ce tableau sont de la forme $C[i,j] = C(x_i, t_j) = C(idx, jdt)]$, terme sur la ligne i et la colonne j.

Le tableau C comprend $N_x + 1$ lignes $(i = 0, 1, ..., N_x)$ et $N_t + 1$ colonnes $(j = 0, 1, ...N_t)$.

discrétisation en temps (indice j)

sens des temps croissants →

ligne i-1 xi-1=(i-1)dx	Ci-1,j-1	Ci-1,j	Ci-1,j+1
ligne i xi=idx	Ci,j-1	Ci,j	Ci,j+1
ligne i+1 xi+1=(i+1)dx	Ci+1,j-1	Ci+1,j	Ci+1,j+1
	colonne j-1 tj-1=(j-1)dt	colonne j tj=jdt	colonne j+1 tj+1=(j+1)dt

↓ discrétisation en espace (indice i) sens des x croissants

L'équation $(*)$ possède deux inconnues. Il faut donc ajouter une deuxième équation pour pouvoir la résoudre.

3. Dans le modèle de Greenshield, que l'on utilise dans cette étude, la vitesse $v(x,t)$ est une fonction affine de la concentration $c(x,t)$.

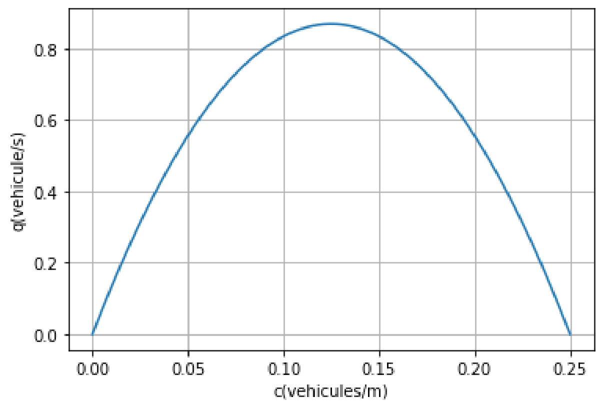
- lorsque la concentration en véhicules tend vers 0, les conducteurs peuvent rouler à la vitesse ' maximale autorisée notée v_{max}

- lorsque les véhicules sont pare-choc contre pare-choc, la concentration est égale à c_{max}: les véhicules sont à l'arrêt.

Déduire de ces informations, l'expression de $v(t,x))$ en fonction de $c(t,x)$, c_{max} et v_{max}.

4. En déduire l'expression de $q(t,x)$ en fonction de $c(t,x)$. On donne le code suivant et le résultat de son exécution:

```python
import matplotlib.pyplot as plt
import numpy as np

cmax,L0,vmax= instruction 1

def q(c):
    return -vmax/cmax*c**2+vmax*c

c=np.linspace(0,cmax,1000)
plt.plot(c,q(c))
plt.grid()
plt.xlabel('c(vehicules/m)')
plt.ylabel('q(vehicule/s)')
plt.show()
```

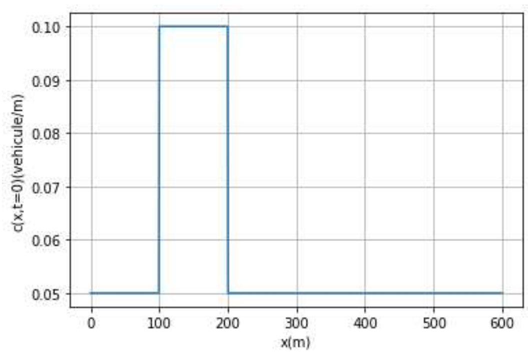


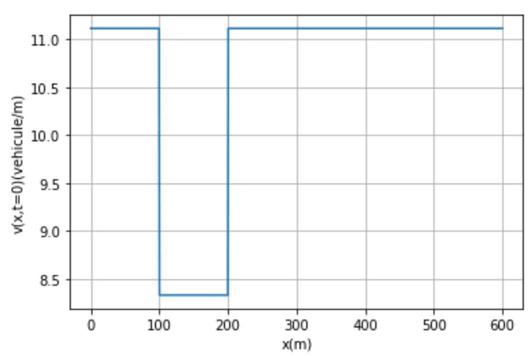
Ecrire sur votre copie l'instruction 1 en exploitant la courbe obtenue.

Commenter l'allure de la courbe.

5. On étudie une situation particulière où la concentration de voitures et la vitesse des voitures sur la route à l'instant initial est donnée par le code et les graphes suivants:

```python
Temps,Lr=60,600
Nt,Nx=1000,1000
dx=Lr/Nx #pas d'esoace
dt=Temps/Nt #pas de temps
x1,x2= instruction 2
C=np.zeros((Nx+1,Nt+1))
for i in range(0,Nx+1):
    if i<x1/dx or i>x2/dx:
        C[i,0]=  instruction 3
    else:
        C[i,0]=  instruction 4

x=np.linspace(0,Lr,Nx+1)
plt.plot(x,C[:,0])
plt.grid()
plt.xlabel('x(m)')
plt.ylabel('c(x,t=0)(vehicule/m)')
plt.show()

plt.plot(x,   instruction 5   )
plt.grid()
plt.xlabel('x(m)')
plt.ylabel('v(x,t=0)(vehicule/m)')
plt.show()
```





Déduire de la lecture du code et de la courbe concentration à l'instant initial, les instructions 2, 3 et 4 (à écrire sur votre copie).

Ecrire sur votre copie l'instruction 5.

Commenter les courbes donnant la concentration de voitures et la vitesse des véhicules à l'instant $t=0$ sur la portion de route étudiée et en déduire la portion de route où le débit de voitures est maximale.

On cherche à étudier l'évolution de la circulation au cours du temps pour les conditions initiales données précédemment en résolvant l'équation $(*)$ (question 2). La méthode de résolution est la méthode d'Euler.

6. Exprimer $\dfrac{\partial c}{\partial t}(x,t)$ en fonction de $c(x, t+dt)$, $c(x,t)$ et dt. En déduire l'expression de $\dfrac{\partial c}{\partial t}(x_i, t_j)$ en fonction de $C[i, j+1]$, $C[i,j]$ et dt.

7. On donne l'expression $\dfrac{\partial q}{\partial x}(x,t) = \dfrac{q(x+dx,t) - q(x-dx,t)}{2dx}$.

On associe au débit de voitures $q(x,t)$ un tableau Q tel que $q(x_i, t_j) = Q[i,j]$. Exprimer $\dfrac{\partial q}{\partial x}(x_i, t_j)$ en fonction de $Q[i+1, j]$, $Q[i-1, j]$ et dx.

On complète le code:

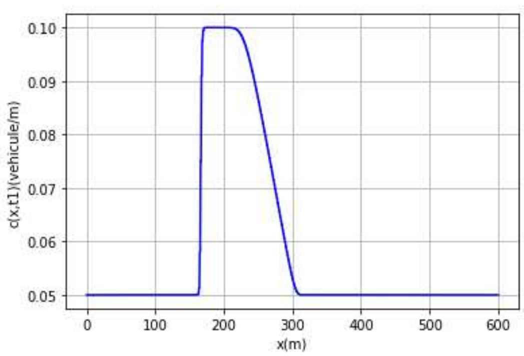
```
for j in range(0,Nt):
    Q= instruction 6
    for i in range(0,Nx+1):
        C[i,j+1]=   instruction 7
```

```
plt.plot(x,C[:, N ],'b')
plt.grid()
plt.xlabel('x(m)')
plt.grid()
plt.ylabel('c(x,t1)(vehicule/m)')
plt.grid()
plt.show()
```

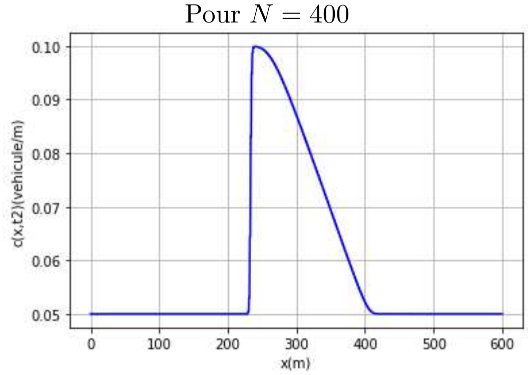
8. Ecrire sur votre copie les instructions 6 et 7.

9. L'exécution du code donne les courbes de concentrations de voitures en fonction de x pour différentes valeurs de N.

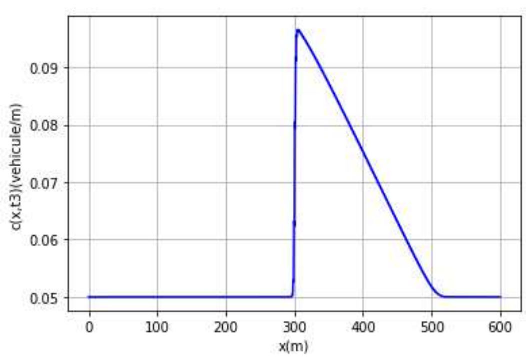
Pour $N = 200$



Pour $N = 400$



Pour $N = 600$



Déduire du code la valeur numérique de dt et en déduire les instants t_1, t_2 et t_3 correspondants à ces trois courbes.

Déduire des courbes, la vitesse des voitures à l'avant de la zone de trafic dense. Commenter cette valeur en vous appuyant sur la courbe $v(x, t=0)$ (question 5).

Comment évolue la zone de forte densité de voitures au cours du temps?

III. **Aide syntaxique sous python**

Aide pour les graphes:

np.linspace(a,b,n) : permet de générer un tableau 1D de n flottants équirépartis dans l'intervalle $[a, b]$

plt.plot(x,y) permet de tracer un graphique de n points dont les abscisses sont contenus dans le tableau 1D x et les ordonnées dans le tableau 1D y, x et y ayant la même taille n

plt.title('titre') permet d'afficher le titre d'un graphique

plt.xlabel('nom') permet de légender l'axe des abscisses

plt.ylabel('nom') permet de légender l'axe des ordonnées

plt.grid() permet d'afficher une grille sur le graphe

plt.show() permet l'affichage d'un graphe

Aide pour les tableaux 2D:

np.zeros((n,)) crée un tableau 2D contenant n lignes et p colonnes ne contenant que des zéros. Les lignes sont numérotées de 0 à $n-1$ et les colonnes sont numérotées de 0 à $p-1$.

T[0,:] désigne la première ligne du tableau T

T[:,0] désigne la première colonne du tableau T

T[i,:] désigne la ligne i du tableau T

T[:,j] désigne la colonne j du tableau T

T[i,j] désigne le terme sur la ligne i et sur la colonne j