In [102] :
```
%matplotlib inline
import matplotlib as mpl
import matplotlib.pyplot as plt
```

In [103] :
```
import matplotlib.colors as colors
import matplotlib.animation as animation
import matplotlib.ticker as mtick
from IPython.display import set_matplotlib_formats
set_matplotlib_formats('png', 'pdf')
plt.rcParams['animation.ffmpeg_path'] = r'C :\ffmpeg\bin\ffmpeg.exe'
```

In [104] :
```
import numpy as np
import scipy.special as spec
import scipy.integrate as inte
import sympy as sp
sp.init_printing()
```

1 Position du problème

Étant donné un fil électrique très long parcouru par un courant électrique i, on cherche à déterminer le champ magnétique créé par phénomène d'induction dans l'espace autour. On peut le détecter en approchant une boussole, de la limaille de fer ou d'autres aimants permanents du fil, voire approcher un autre fil électrique lui aussi parcouru par un courant.

2 Construction du champ magnétique

Les cellules suivantes servent à construire la fonction B correspondant à l'intensité du champ magnétique $\mathbf{B}$ à une distance r du fil, à l'instant t.

In [128] :
```
r, omega = sp.symbols('r omega', positive = True)
t, phi = sp.symbols('t phi', real=True)

mu = sp.symbols("mu0", positive = True) # Perméabilité magnétique du vide
c = sp.symbols("c", positive=True)      # Célérité de la lumière dans le vide
k = omega/c  # Nombre d'onde

# Composante du potentiel vecteur associée à la pulsation omega
A_component = -mu/(2*sp.pi)*(sp.bessely(0,k*r) + sp.I*sp.besselj(0,k*r) ) \
    *sp.exp(sp.I*(omega*t-phi))

# Composante du champ associée à la pulsation omega
B_component = -A_component.diff(r).simplify()
B_comp_real = sp.re(B_component)

# Création du champ (symbolique et fonction) en sommant les composantes
def create_Bfield(puls,phas):
    # valeurs numériques à substituer
    spectr = zip(puls,phas)
```

```
    c0 = 3e8
    mu0_v = 4e-7*np.pi

    B_field = sum([B_comp_real.subs({omega:om, phi:ph, c:c0, mu:mu0_v}) for (om,ph) 
    B_function = sp.lambdify((r, t), B_field,
        modules=['numpy',{"besselj":spec.jn, "bessely":spec.yn}])

    return B_field, B_function
```

In [127] :
```
B_comp_real
```

Out[127] :

$$\frac{\mu_0 \omega J_1\left(\frac{\omega r}{c}\right)}{2\pi c} \sin\left(\omega t - \phi\right) - \frac{\mu_0 \omega Y_1\left(\frac{\omega r}{c}\right)}{2\pi c} \cos\left(\omega t - \phi\right)$$

In [125] :
```
def graphe_B(times, ani = False):
    '''
    Construit les graphes du champ magnétique B aux temps donnés dans la liste
    "times"
    Si le drapeau 'ani' est True, alors entrer en mode "animation"
    '''
    radii = np.linspace(rmin, rmax, 1000)

    fig = plt.figure(1, figsize=(8,5), dpi=100)
    ax = plt.axes()
    ax.yaxis.set_major_formatter(mtick.FormatStrFormatter('%.2e'))

    def legende(ti):
        out = r'$t= {:.3e}$'.format(ti)
        out = out + r"$\ \mathrm{s}$"
        return out

    if not(ani):
        if hasattr(times, '__iter__'):
            for ti in times:
                champ = B_function(radii, ti)
                ax.plot(radii, champ, label=legende(ti))
        else:
            champ = B_function(radii, ti)
            ax.plot(radii, champ, label=legende(ti))
        ax.legend()
    else:
        line, = ax.plot([], [], lw=2)
        time_text = ax.text(0.02, 0.95, '',
                            transform=ax.transAxes)


        t0, t1= times
```

```
        interval = t1 - t0
        animtime = 15
        fps = 30
        dt = interval/animtime # secondes vidéo par seconde réelle
        framenum = int(np.ceil(fps*animtime))

        ymax = B_function(radii,t0).max()

        ax.set_xlim(0,rmax)
        ax.set_ylim((-ymax/2,ymax))

        def init():
            line.set_data([],[])
            time_text.set_text('')
            return line, time_text

        def animate(i):
            ti = dt*i+t0
            legende_temps = legende(ti)
            champ = B_function(radii, ti)
            line.set_data(radii, champ)
            time_text.set_text(legende_temps)
            return line, time_text

    ax.grid(True)
    ax.set_xlabel("Distance $r$ (m)")
    ax.set_ylabel("Valeur du champ (T)")
    ax.set_title(r'Champ magnétique ' + r'$\mathbf{B}$' \
                 + ' créé par un courant variable')

    fig.tight_layout()

    if ani:
        anima = animation.FuncAnimation(fig, animate, init_func=init,
                    frames = framenum, interval = interval, blit = True)
        mywriter = animation.FFMpegWriter(fps=fps, bitrate=1000)
        anima.save('champ_mag_anim.mp4', writer=mywriter)
    else:
        fig.savefig('profil_champmag.pdf')
        fig.savefig('profil_champmag.png')
        return fig, ax
```

```
In [107] : def build_field(t, colmap='bone'):
    """
    Portrait du champ magnétique à l'instant t
    """
    wind = rmax
```

```python
def field_func(x,y):
    r = np.sqrt(x*x+y*y)
    Btheta = B_function(r, t)
    direct = np.array([-y/r, x/r])
    return Btheta*direct

Y, X = np.ogrid[-wind:wind:1000j, -wind:wind:1000j]
BX, BY = field_func(X, Y)
intensity = np.sqrt(BX**2+BY**2)


fig, ax = plt.subplots(1, 1, figsize=(8,8))

heat = ax.imshow(intensity,
                 cmap=colmap,
                 norm=colors.LogNorm(),
                 extent=[-wind, wind, -wind, wind],
                 alpha=0.6)
cbar = fig.colorbar(heat,
            label='Intensité du champ (T)')

strm = ax.streamplot(X,Y, BX, BY,
    arrowstyle='->',
    color='w',
    linewidth=0.8,
    arrowsize=2,
    density=1.4,
    )

ax.grid(False)
ax.set_aspect('equal')
ax.set_xlim((-wind,wind))
ax.set_ylim((-wind,wind))
title_text = r'Champ magnétique $\mathbf{B}$ à '
title_text += r"$t={:g}$".format(t)
title_text += r" $\mathrm{s}$"
ax.set_title(title_text)
fig.tight_layout()

return fig
```

3 Tracés

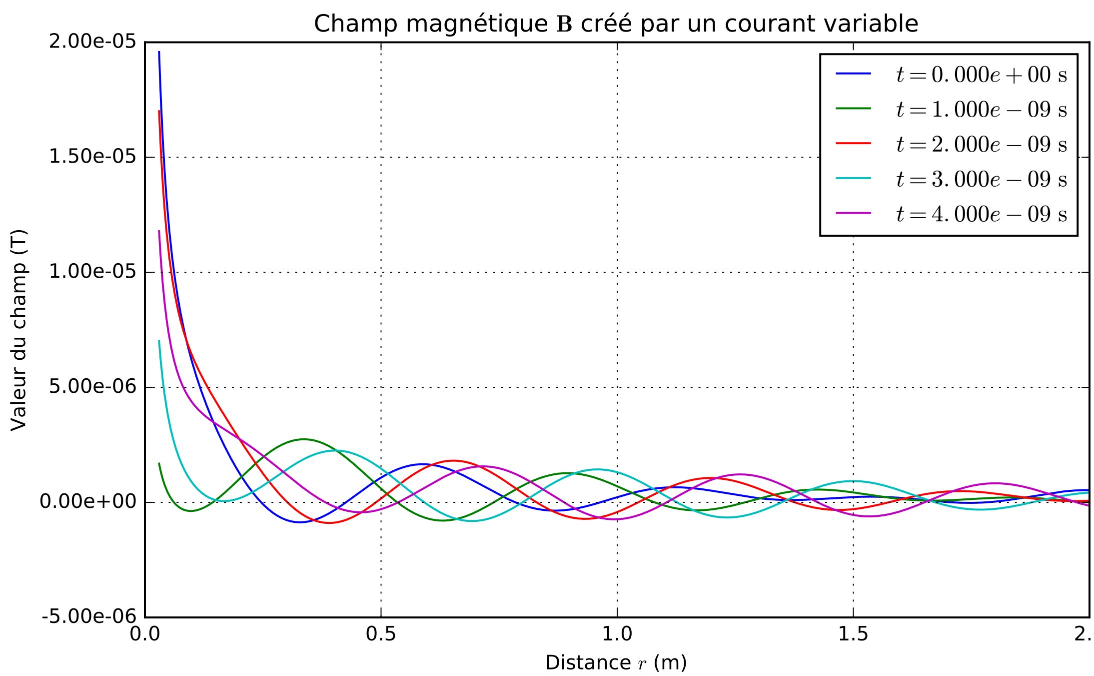
3.1 Données initiales

Entrez dans la variable `freqs` les fréquences du courant voulu, et dans `phas` les phases associées, et exécutez la cellule (`Ctrl + Entrée` sur le clavier) pour définir la fonction de champ :

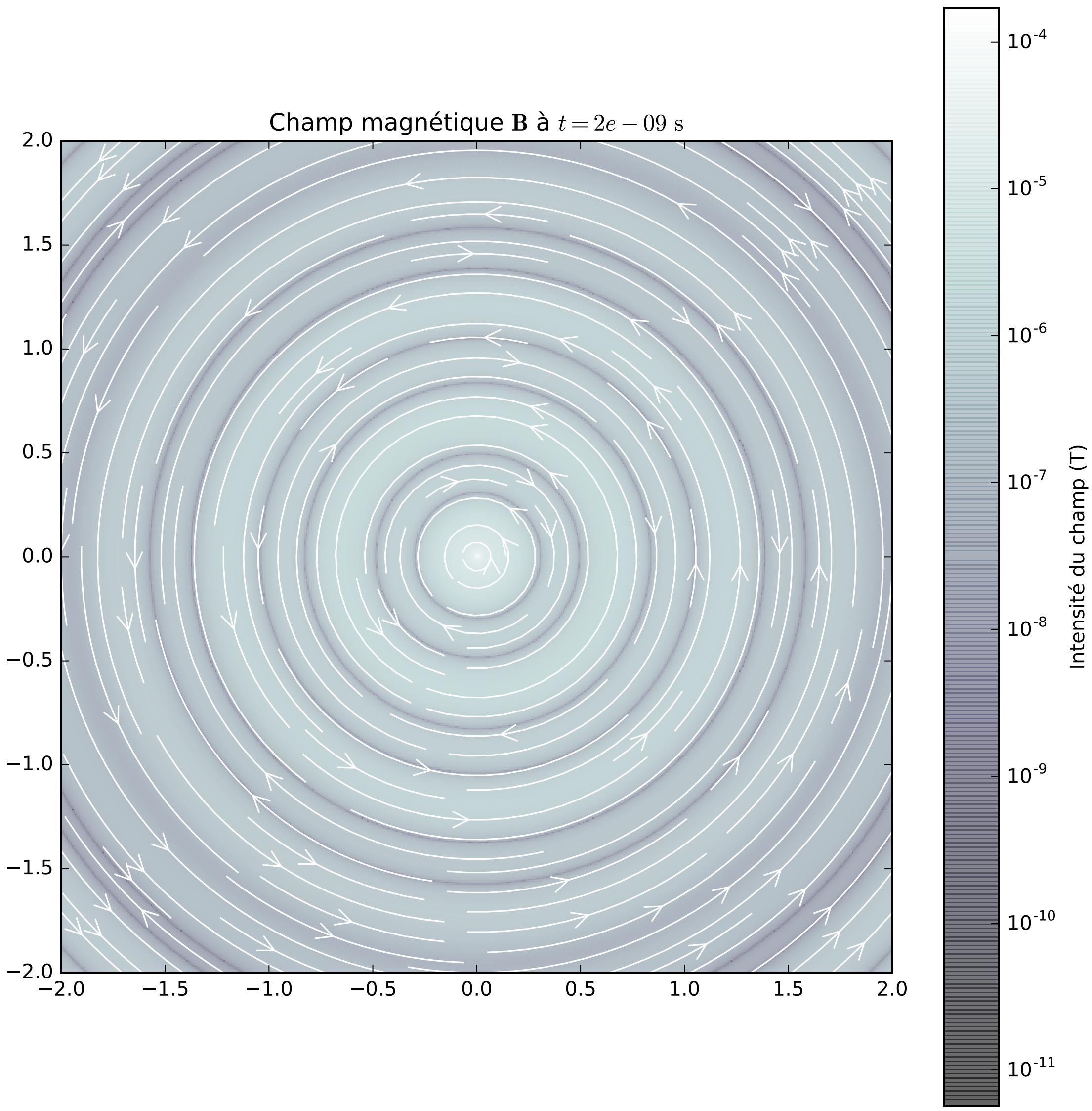
```
In [129] : freqs = [n*1e8 for n in range(5,7)] + \
               [n*1e6 for n in range(2,5)]
           puls = 2*np.pi*np.asarray(freqs) # Pulsations associées

           phases = [0,0,0.7,0.8,0.3]

           B_field, B_function = create_Bfield(puls, phases)
```

La cellule suivante définit les distances minimale et maximale pour lesquels tracer le profil du champ magnétique :

```
In [111] : rmin = 0.03
           rmax = 2

In [117] : times = [1e-9*k for k in [0, 1, 2, 3, 4]]

           graphe_B(times)

Out[117] : (<matplotlib.figure.Figure at 0x24d4ca7b390>,
            <matplotlib.axes._subplots.AxesSubplot at 0x24d4ca96ac8>)
```




```
In [116] : t = times[2]     # Temps auquel calculer le portrait du champ (pas de liste)

           field_lines = build_field(t)
           field_lines.savefig('portrait_champmag.pdf')
           field_lines.savefig('portrait_champmag.png')
```



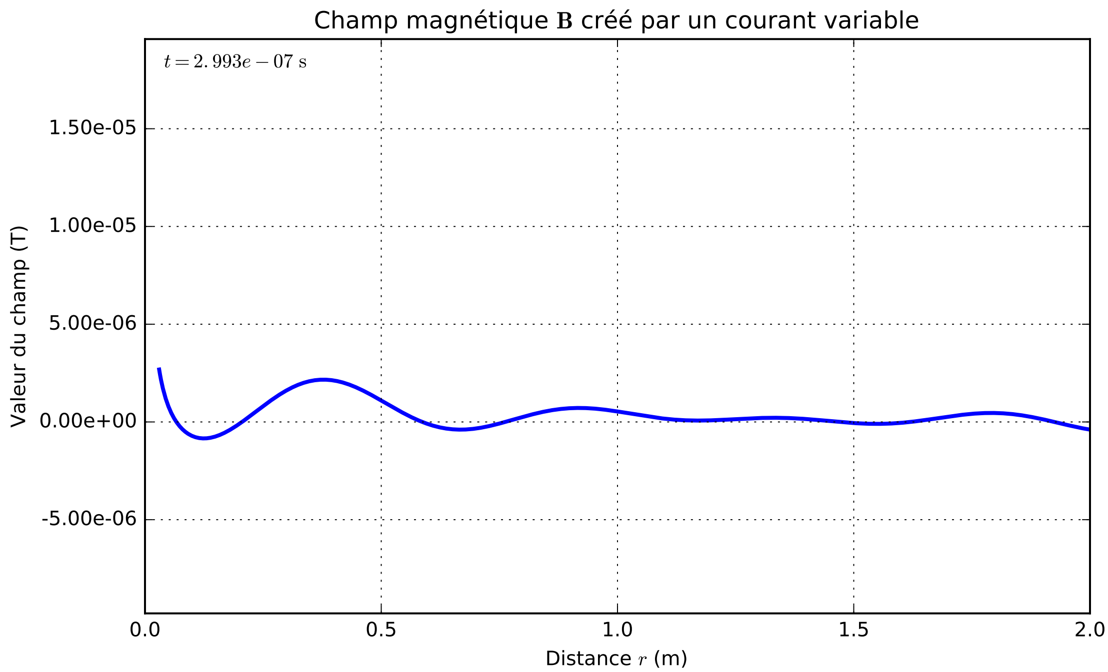
3.2 Animations

Modifiez cette cellule avec les fréquences que vous voulez utiliser pour les animations :

```
In [130] : freqs = [n*1e8 for n in range(5,7)] + \
                [n*1e7 for n in range(2,5)]
            puls = 2*np.pi*np.asarray(freqs) # Pulsations associées

            phases = [0,0,0.7,0.8,0.3]

            B_field, B_function = create_Bfield(puls, phases)
```

```
In [131] : times_an = [0, 1e-8]
```

```
graphe_B(times_an, True)
```



4 Théorie (Bac + 1,5)

Le champ magnétique $\mathbf{B}$ dérive d'un champ $\mathbf{A}$ appelé *potentiel vecteur* : $\mathbf{B} = \nabla \wedge \mathbf{A}$. Par symétrie cylindrique, on a $\mathbf{B}(\mathbf{r}, t) = B(r, t)\mathbf{e}_\theta$. Par suite $\mathbf{A}(\mathbf{r}, t) = A(r, t)\mathbf{e}_z$.

Le potentiel vecteur $\mathbf{A} = A(r, t)\mathbf{e}_z$ est solution de l'équation d'onde

$$\Delta \mathbf{A} - \frac{1}{c^2}\frac{\partial^2 \mathbf{A}}{\partial t^2} = -\mu_0 \mathbf{J}(r, t), \tag{1}$$

avec $\mathbf{J}(r, t) = \dfrac{i(t)\delta(r)}{2\pi r}\mathbf{e}_\theta$ la densité volumique de courant.

Pour un courant sinusoïdal $i(t) = I\exp(i\omega t)$, le potentiel s'écrit $A(r, t) = f(r)\exp(i\omega t)$ et l'équation aux dérivées partielles se réduit à

$$\frac{1}{r}\frac{\mathrm{d}}{\mathrm{d}r}\left(r\frac{\mathrm{d}f}{\mathrm{d}r}\right) + k^2 f(r) = -\frac{\mu_0 I \delta(r)}{2\pi r}, \tag{2}$$

avec $k = \dfrac{\omega}{c}$.

La solution générale prend la forme

$$f(r) = CJ_0(kr) + DY_0(kr)$$

où C et D dépendent de la pulsation ω du courant, et J_0, Y_0 sont les 0-ièmes fonctions de Bessel de la première et seconde espèce, solutions de

$$xy''(x) + y'(x) + xy(x) = 0.$$