Le service au volley

In [1]:
```
import numpy as np
import matplotlib.pyplot as plt
from scipy.integrate import odeint
#variables globales utilisées
g=9.8# acceleration de la pesannteur en m/s²
l=9 # largeur du terrain de volley en m
L1=9# profondeur du terrain pour chaque camp
L2=2*L1# longueur totale du terrain de volley
Ldiag=np.sqrt(l**2+L2**2)# distance maximale envisageable pour faire un service en diagonale du terrain (en m)
H=2.43# hauteur du filet de volley
m=0.27#masse du ballon de volley en kg
D=0.21#diametre du ballon de volley en m
rho=1.2#masse volumique de l'air en kg/m^3
S=np.pi*D**2/4# surface apparente du ballon en m²
Cd=0.5# coeficient de
beta=Cd*S*rho# coefficient de frottement quadratique
```

Hypothèses

- On suppose que la trajectoire est plane dans le plan (x,y)
- On néglige la rotation du ballon et son extension spatiale (pas de possibilité de jouer avec la bande du filet, ou de l'écrasement du ballon sur la ligne !)
- Dans le cas de la prise en compte des frottements en considère une force $$\overrightarrow f=-C_dS\rho v \overrightarrow{v}$$ (formule de Stokes) avec ρ la masse volumique de l'air S la surface apparente du ballon et C_d le coefficient de traînée

1) Modélisation sans frottement

In [2]:
```
def equasansfrot(X,t):
    """X comprend les variables de positions x et y et les composantes du vecteurs vitesses vx et vy"""
    return np.array([X[2], X[3],0,-g])
```

In [3]:
```
alpha1=0# valeur de l'angle du tir (tir horizontal si égal à 0)
N=5# nombre de valeurs de hauteur
v0=100/3.6# vitesse initiale du ballon en m/s
hmin=H# hauteur du filet
hmax=3.5# hauteur maximale raisonablement atteinte par un volleyeur
h0=np.linspace(hmin,hmax,N)# variation de la valeur de h0 pour valider une valeur
```

In [4]:
```
X0=np.zeros((N,4))# génération du vecteur conditions initiales x(0)=0, y(0)=h0, vx=v0cos(alpha),vy0=v0sin(alpha)
X0[:,1]=h0
X0[:,2]=v0*np.cos(alpha1)
X0[:,3]=v0*np.sin(alpha1)
```

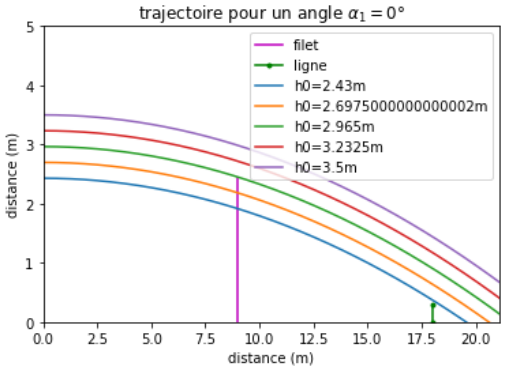
In [5]:
```
X0
```

Out[5]:
```
array([[ 0.        ,  2.43      , 27.77777778,  0.        ],
       [ 0.        ,  2.6975    , 27.77777778,  0.        ],
       [ 0.        ,  2.965     , 27.77777778,  0.        ],
       [ 0.        ,  3.2325    , 27.77777778,  0.        ],
       [ 0.        ,  3.5       , 27.77777778,  0.        ]])
```

In [6]:
```
Np=200
tf=1.5# temps final de la modélisatio
t=np.linspace(0,tf,Np)
```

In [7]:
```
sol={}#dictionnaire vide des solutions
for i in range(N):
    sol[i]=odeint(equasansfrot,X0[i],t)
```

In [8]:
```
plt.figure()
plt.plot([L1,L1],[0,H],'-m',label='filet')
plt.plot([L2,L2],[0,0.3],'.-g',label='ligne')
for i in range(N):
    plt.plot(sol[i][:,0],sol[i][:,1],label='h0='+str(h0[i])+'m')
#plt.axis('equal')
plt.xlabel('distance (m)')
plt.ylabel('distance (m)')
plt.title(r'trajectoire pour un angle $\alpha_1=0°$')
plt.ylim([0,5])
plt.xlim([0,Ldiag+1])
plt.legend(loc=0)
plt.show()
```



On constate qu'en négligeant les frottements il n'est pas envisageable d'obtenir un service qui entre dans le terrain, même en tirant en diagonal d'un coin en visant le coin opposé. On pourrait essayer de tirer avec un angle négatif d'une hauteur maximale égale à 3,50 m

In [9]:
```
h0=3.5# hauteur maximale du tir
N=5# nombre de valeurs d'angles
v0=100/3.6# vitesse initiale du ballon en m/s
alphamin=0# hauteur du filet
alphamax=-4# hauteur maximale raisonablement atteinte par un volleyeur
alphadeg=np.linspace(alphamin,alphamax,N)
alpha1=np.pi/180*np.linspace(alphamin,alphamax,N)# variation de la valeur de v0 pour valider une valeur
```

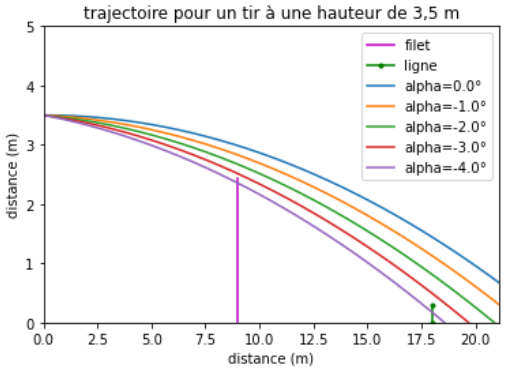
In [10]:
```
X0a=np.zeros((N,4))# génération du vecteur conditions initiales x(0)=0, y(0)=h0, vx=v0cos(alpha),vy0=v0sin(alpha)
X0a[:,1]=h0
X0a[:,2]=v0*np.cos(alpha1)
X0a[:,3]=v0*np.sin(alpha1)
```

In [11]:
```
X0a
```

Out[11]:
```
array([[ 0.        ,  3.5       , 27.77777778,  0.        ],
       [ 0.        ,  3.5       , 27.77354709, -0.48478907],
       [ 0.        ,  3.5       , 27.76085631, -0.96943046],
       [ 0.        ,  3.5       , 27.7397093 , -1.45377656],
       [ 0.        ,  3.5       , 27.71011251, -1.93767983]])
```

In [12]:
```
sola={}#dictionnaire vide des solutions pour des angles variables
for i in range(N):
    sola[i]=odeint(equasansfrot,X0a[i],t)
```

In [13]:
```
plt.figure()
plt.plot([L1,L1],[0,H],'-m',label='filet')
plt.plot([L2,L2],[0,0.3],'.-g',label='ligne')
for i in range(N):
    plt.plot(sola[i][:,0],sola[i][:,1],label='alpha='+str(alphadeg[i])+'°')
#plt.axis('equal')
plt.xlabel('distance (m)')
plt.ylabel('distance (m)')
plt.title(r'trajectoire pour un tir à une hauteur de 3,5 m')
plt.ylim([0,5])
plt.xlim([0,Ldiag+1])
plt.legend(loc=0)
plt.show()
```



Encore une fois le fait de négliger les frottements ne semble pas adapté et compatible aux vidéos, même si avec un angle voisin de -3°, un tir en diagonal, et une hateur de lancé de 3,5m, il semble qu'il y ait une solution

2) Modélisation avec frottement

In [14]:
```
def equaavecfrot(X,t):
    """X comprend les variables de positions x et y et les composantes du vecteurs vitesses vx et vy"""
    return np.array([X[2], X[3],-beta/m*X[2]*(X[2]**2+X[3]**2)**0.5,-g-beta/m*X[3]*(X[2]**2+X[3]**2)**0.5])
```

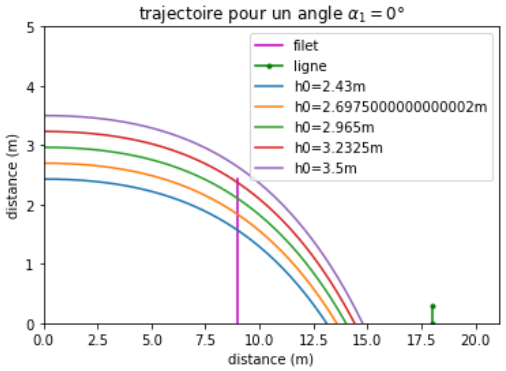
In [15]:
```
alpha1=0# valeur de l'angle du tir (tir horizontal si égal à 0)
N=5# nombre de valeurs de hauteur
v0=100/3.6# vitesse initiale du ballon en m/s
hmin=H# hauteur du filet
hmax=3.5# hauteur maximale raisonablement atteinte par un volleyeur
h0=np.linspace(hmin,hmax,N)# variation de la valeur de h0 pour valider une valeur
```

In [16]:
```
X0frot=np.zeros((N,4))# génération du vecteur conditions initiales x(0)=0, y(0)=h0, vx=v0cos(alpha),vy0=v0sin(alpha
X0frot[:,1]=h0
X0frot[:,2]=v0*np.cos(alpha1)
X0frot[:,3]=v0*np.sin(alpha1)
```

In [17]:
```
Np=200
tf=1.5# temps final de la modélisation en seconde
t=np.linspace(0,tf,Np)
```

In [18]:
```
sol2={}#dictionnaire vide des solutions
for i in range(N):
    sol2[i]=odeint(equaavecfrot,X0frot[i],t)
```

In [19]:
```
plt.figure()
plt.plot([L1,L1],[0,H],'-m',label='filet')
plt.plot([L2,L2],[0,0.3],'.-g',label='ligne')
for i in range(N):
    plt.plot(sol2[i][:,0],sol2[i][:,1],label='h0='+str(h0[i])+'m')
#plt.axis('equal')
plt.xlabel('distance (m)')
plt.ylabel('distance (m)')
plt.title(r'trajectoire pour un angle $\alpha_1=0°$')
plt.ylim([0,5])
plt.xlim([0,Ldiag+1])
plt.legend(loc=0)
plt.show()
```



On constate cette fois un fort impact de la force de frottement de l'air et dans cette simulation seul se service frappé à 3,5m franchit le filet. On va devoir considérer des services avec des angles positifs avec une hauteur raisonable (pour un joueur de volley professionnel !) de lancé de 2,5 m!

In [20]:
```
h0=2.5# hauteur maximale du tir
N=5# nombre de valeurs d'angles
v0=100/3.6# vitesse initiale du ballon en m/s
alphamin=0# hauteur du filet
alphamax=25# hauteur maximale raisonablement atteinte par un volleyeur
alphadeg=np.linspace(alphamin,alphamax,N)
alpha1=np.pi/180*np.linspace(alphamin,alphamax,N)# variation de la valeur de l'angle pour valider une valeur
```

In [21]:
```
X0frotta=np.zeros((N,4))# génération du vecteur conditions initiales x(0)=0, y(0)=h0, vx=v0cos(alpha),vy0=v0sin(alp
X0frotta[:,1]=h0
X0frotta[:,2]=v0*np.cos(alpha1)
X0frotta[:,3]=v0*np.sin(alpha1)
```

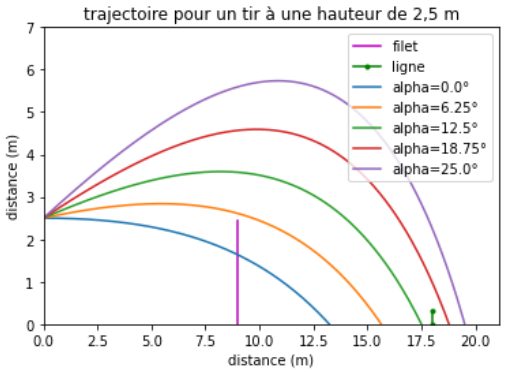
In [22]:
```
X0frotta
```

Out[22]:
```
array([[ 0.        ,  2.5       , 27.77777778,  0.        ],
       [ 0.        ,  2.5       , 27.61267606,  3.02407986],
       [ 0.        ,  2.5       , 27.11933353,  6.0122115 ],
       [ 0.        ,  2.5       , 26.30361471,  8.92887404],
       [ 0.        ,  2.5       , 25.17521631, 11.73939616]])
```

In [23]:
```
Np=300
tf=2# temps final de la modélisatio
t=np.linspace(0,tf,Np)
```

In [24]:
```
sol2frott={}#dictionnaire vide des solutions
for i in range(N):
    sol2frott[i]=odeint(equaavecfrot,X0frotta[i],t)
```

In [25]:
```
plt.figure()
plt.plot([L1,L1],[0,H],'-m',label='filet')
plt.plot([L2,L2],[0,0.3],'.-g',label='ligne')
for i in range(N):
    plt.plot(sol2frott[i][:,0],sol2frott[i][:,1],label='alpha='+str(alphadeg[i])+'°')
#plt.axis('equal')
plt.xlabel('distance (m)')
plt.ylabel('distance (m)')
plt.title(r'trajectoire pour un tir à une hauteur de 2,5 m')
plt.ylim([0,7])
plt.xlim([0,Ldiag+1])
plt.legend(loc=0)
plt.show()
```



Bilan

Les frottements ont une grande influence sur la trajectoire du ballon de volley lors d'un service :

- Avec la vitesse de 100km/h imposée par l'énoncé, il est compliqué de trouver des paramètres de lancé (h0, alpha) compatibles à la réalisation de services réussis alors que c'est une vitesse atteignable pour des volleyeurs professionnels.
- Lorsqu'on considère les frottements, on retrouve des trajectoires habituelles dans des plages larges de paramètres de lancé 2,0m<h_0<3,5 m et 0°<α<30°.
- La trajectoire n'est pas plane si on prend en compte la rotation du ballon et les forces de frottements de l'air qui agissent sur la rotation du ballon. De plus on avoir l'impresion que le ballon "flotte". Tout ceci pourrait être étudié avec le cours de mécanique des fluides de seconde année mais Une première approche serait de considérer l'effet Magnus pour affiner notre modèle...