Zakaria Otmane

Évacuation d’une foule

Filière MP - Option S.I.

28055

Introduction au sujet

Prévoir les mouvements de foule permet d’assurer la sécurité des personnes.

Une rénovation des stades brésiliens fut indispensable pour assurer le bon déroulement de la Coupe du monde 2014. Il fallait en effet pour les spectateurs plus de 40 minutes pour atteindre une sortie depuis leur siège, ce qui représentait un réel risque en cas d’urgence.

Aujourd’hui, pour les grands évènements, accéder à une sortie de secours ne doit pas dépasser 8 minutes.

Problématique

Quelle est la pertinence d’un modèle macroscopique dans la prédiction et l’analyse d’une évacuation urgente d’un petit parc de $100\ m^2$ en prenant en compte les contraintes de sécurité ?

Plan

- Présentation du modèle macroscopique

- Application au cône convergent

- Confrontation aux autres modèles

- Bibliographie / Annexe

Présentation du modèle macroscopique

Le modèle macroscopique est l’une des principales méthodes de description du mouvement d’une foule.

Nous considérons la foule dans son entièreté, qui est représentée par l’intermédiaire d’une densité de personnes. Le déplacement des individus est, ici, assimilé à celui d’un fluide.

Les lois de la mécanique des fluides entreront donc en action dans cette étude.

Le mouvement d'un fluide est caractérisé par deux visions :

- La description lagrangienne

Les grandeurs cinématiques du système dépendent du temps. Nous suivons le mouvement du fluide le long de sa trajectoire à travers le temps.

$$\vec{OM} = \vec{OM}(t) \quad , \quad \vec{v} = \frac{d\,\vec{OM}}{dt}(t) \quad , \quad \vec{a} = \frac{d^2\vec{OM}}{dt^2}(t)$$

- La description eulérienne

Les variables de l’espace sont décorrélées vis-à-vis du temps. Nous observons le champ des vitesses dans l’espace à un instant donné. Durant cet instant chaque point de l’espace aura un vecteur vitesse.

$$\vec{v} = \vec{v}(M,t) \quad , \quad \vec{a} = \frac{D\vec{v}}{Dt} = \frac{\partial \vec{v}}{\partial t} + (\vec{v}.\vec{\nabla})\vec{v}$$

L’accélération étant calculée avec une dérivée particulaire.

Cette description sera choisie car elle s’adapte à l’étude des conditions aux limites sur un obstacle.

Présentation du modèle macroscopique

Le mouvement de la foule est également influencé par différentes forces sociales qui modélisent les interactions entre les individus. Nous prenons en compte ces forces de contact.

La norme des forces peut devenir excessivement grande au-delà d’une densité limite et cela peut amener des phénomènes de saturation lors du déplacement de la foule.

Nous noterons la résultante des forces, $\vec{F}$.

Nous allons enfin supposer que la masse de la foule en déplacement se conserve. L’équation suivante est alors vérifiée :

$$\frac{\partial \rho}{\partial t} + \vec{\nabla}.(\rho \vec{v}) = 0$$

Le modèle satisfait ainsi les équations suivantes :

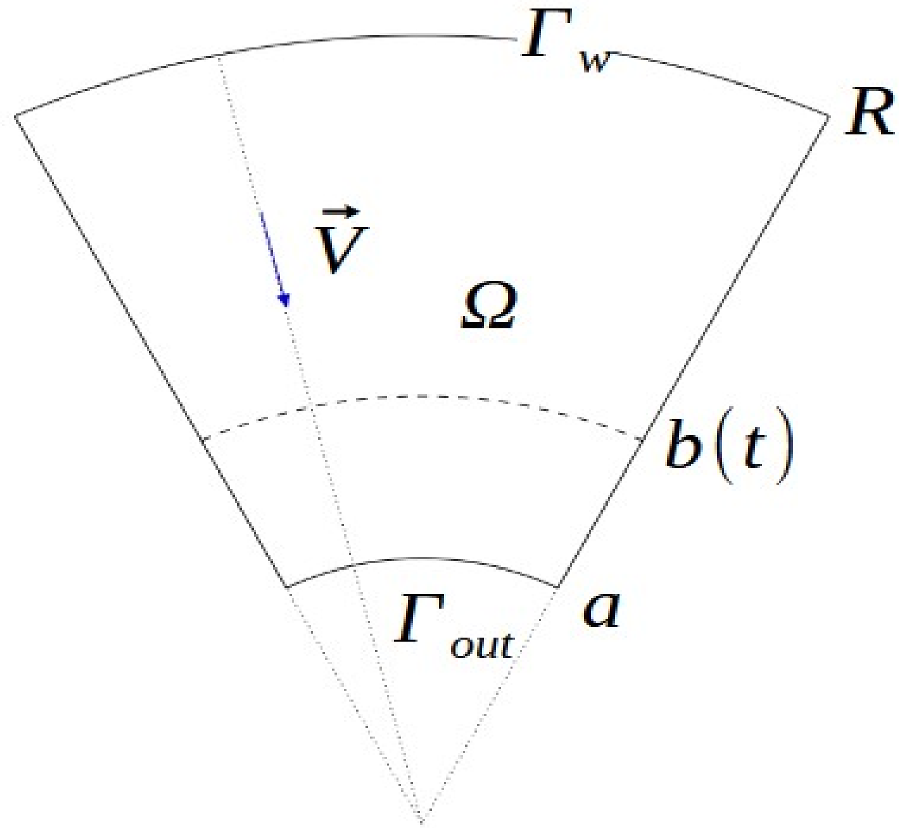
$$\frac{\partial \rho}{\partial t} + \vec{\nabla}.(\rho \vec{v}) = 0$$

$$\frac{\partial \vec{v}}{\partial t} + (\vec{v}.\vec{\nabla})\vec{v} = \vec{F}$$

Application au cône convergent

La foule, de densité ρ, se déplace dans un couloir conique Ω, de portion $[a, R]$. Les gens arrivent à l'entrée Γ_w et souhaitent aller vers l'espace de sortie Γ_{out} .

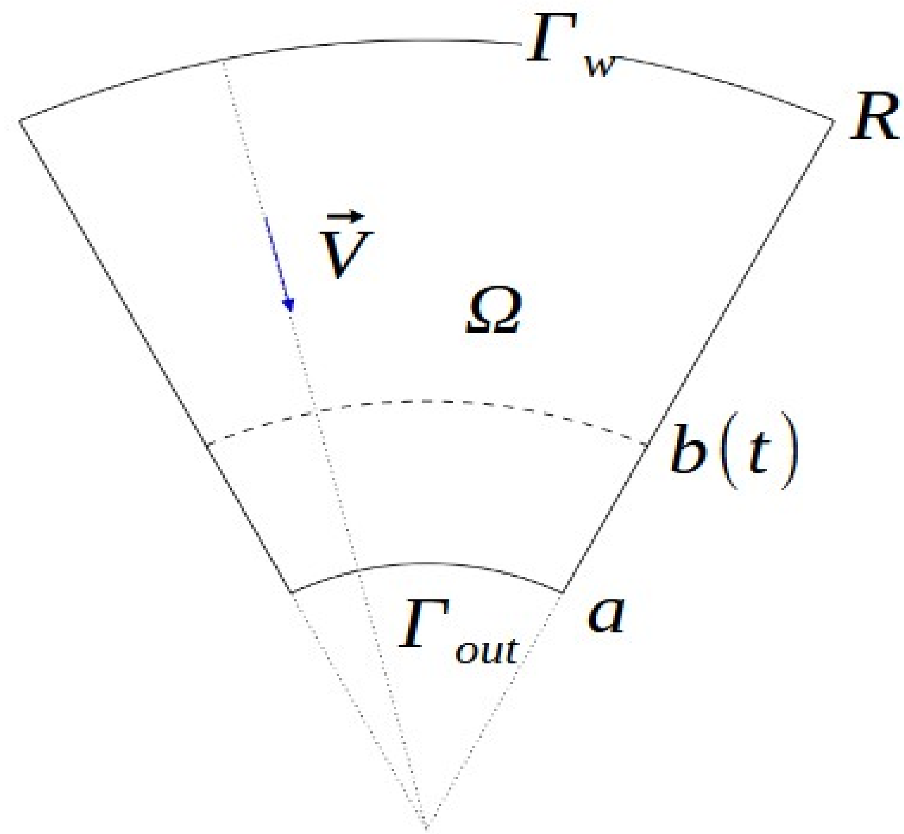
La vitesse souhaitée, correspondant à celle qu'aurait un piéton qui se déplacerait seul dans le cône jusqu'à la sortie, est notée $\vec{V}$.



Application au cône convergent

Lorsque la concentration d'individu devient trop importante, la vitesse réelle $\vec{v}$ de la foule est différente de celle souhaitée. De tels phénomènes de saturation s'opposent alors aux mouvements désirés.

L'apparition d'un tel évènement est modélisé par la grandeur $b(t)$ qualifiée d'interface de densité maximale.

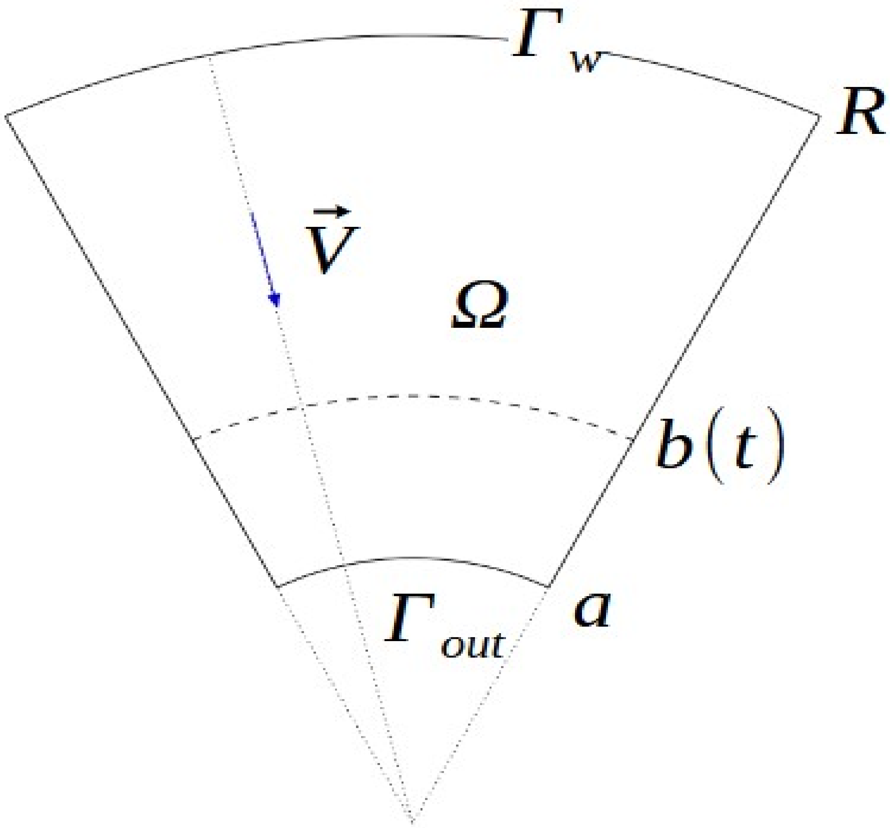


Application au cône convergent

Ainsi, l'espace compris entre la sortie et cette interface est complètement saturée. Ailleurs les gens sont libres de leurs mouvements.

La densité d'individu dans le cône ne peut donc pas excéder celle correspondant à un état de saturation. Si on note $\rho_{saturée}$ cette densité on aura donc le résultat suivant : $\rho \leqslant \rho_{saturée}$

Pour la suite, la densité à la sortie est notée ρ_{sortie} et celle à l'état initial de l'évacuation ρ_0 .

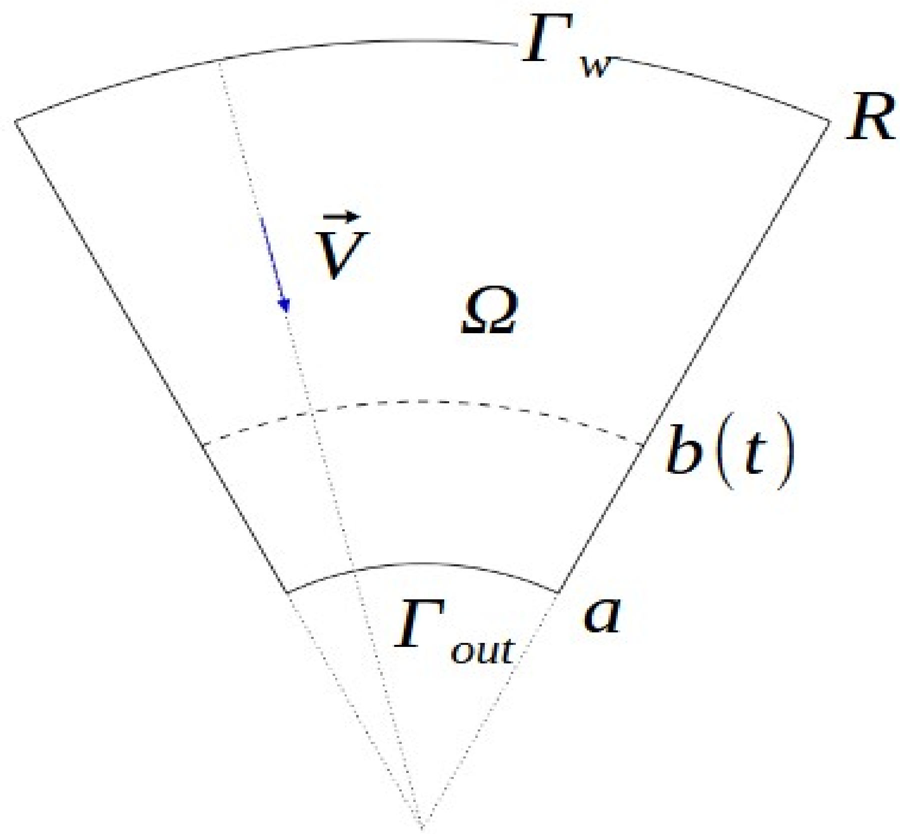


L’évacuation s’effectue donc en trois temps :

- Un début d’évacuation sans encombrement.

- L’apparition d’une zone saturée prenant de l’ampleur au fil du temps.

- Une fin d’évacuation avec un encombrement totale.



Choix des grandeurs :

- $R = 5\ m$
- $a = 1\ m$
- $V = 1{,}67\ m.s^{-1}$
- $\rho_0 = 0{,}43\ m^{-2}$
- $\rho_{saturée} = 1\ m^{-2}$
- $\rho_{sortie} = 1\ m^{-2}$

Application au cône convergent

L'objectif est maintenant de connaître $b(t)$ afin de maîtriser les phénomènes de saturation.

Un bilan de masse sur la foule permet d'aboutir à l'équation différentielle :

$$\begin{cases} b'(t) & = & f(t, b(t)) \\ b(0) & = & a \end{cases}$$

Remarque : La densité de personnes dans le cône peut être calculée à l'aide de la méthode des caractéristiques.

Application au cône convergent

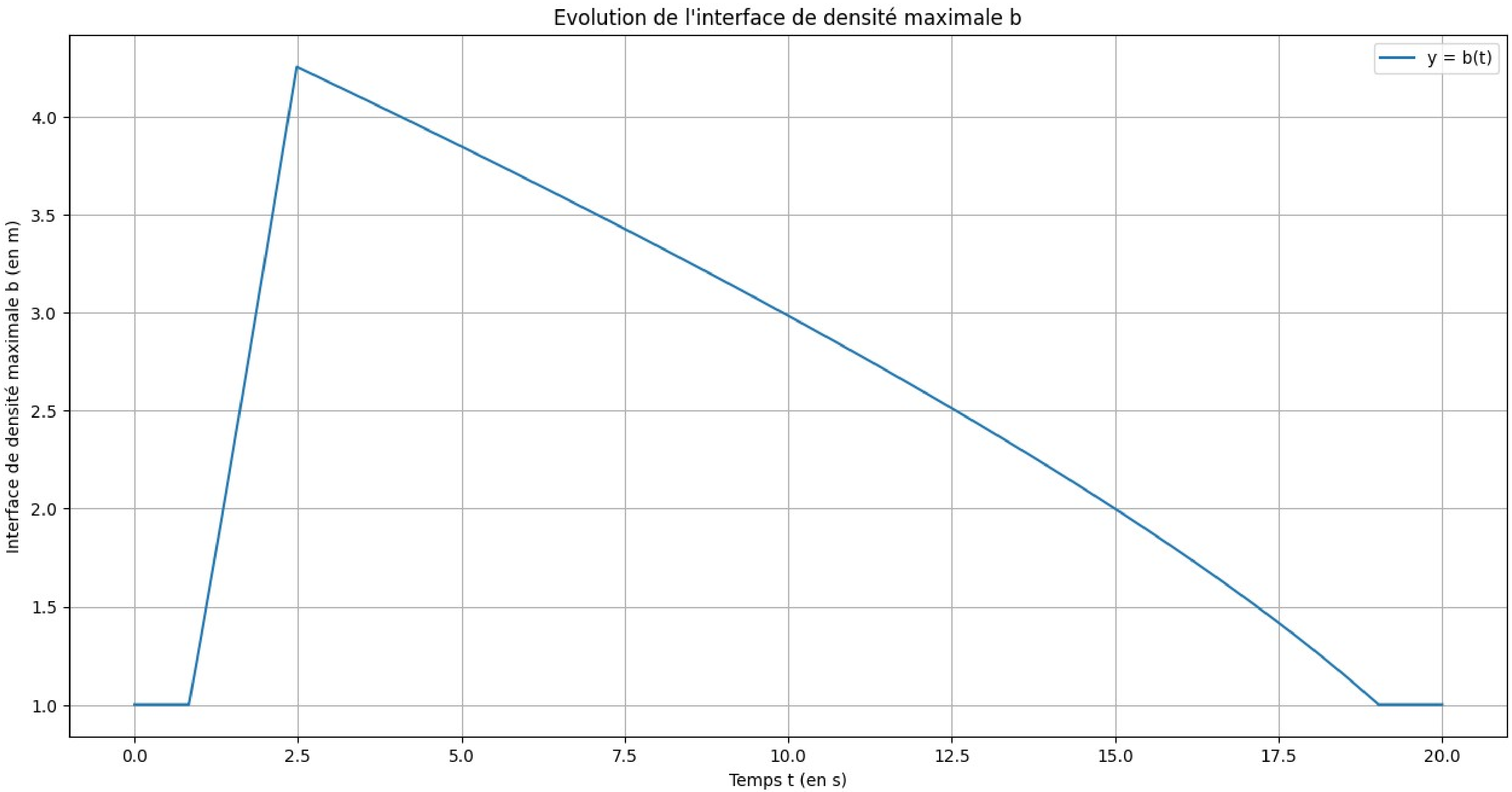
$$f(t,r) = \begin{cases} \dfrac{\left(\rho_0\left(1 + \dfrac{vt}{r}\right) - \rho_{sortie}\left(\dfrac{r - a}{r\ln\left(\frac{r}{a}\right)}\right)\right)v}{\rho_{saturée} - \rho_0\left(1 + \dfrac{vt}{r}\right)} & r \leqslant R - vt \\ -\left(\dfrac{\rho_{sortie}}{\rho_{saturée}}\right)\left(\dfrac{r - a}{r\ln\left(\frac{r}{a}\right)}\right)v & r > R - vt \end{cases}$$

Application au cône convergent

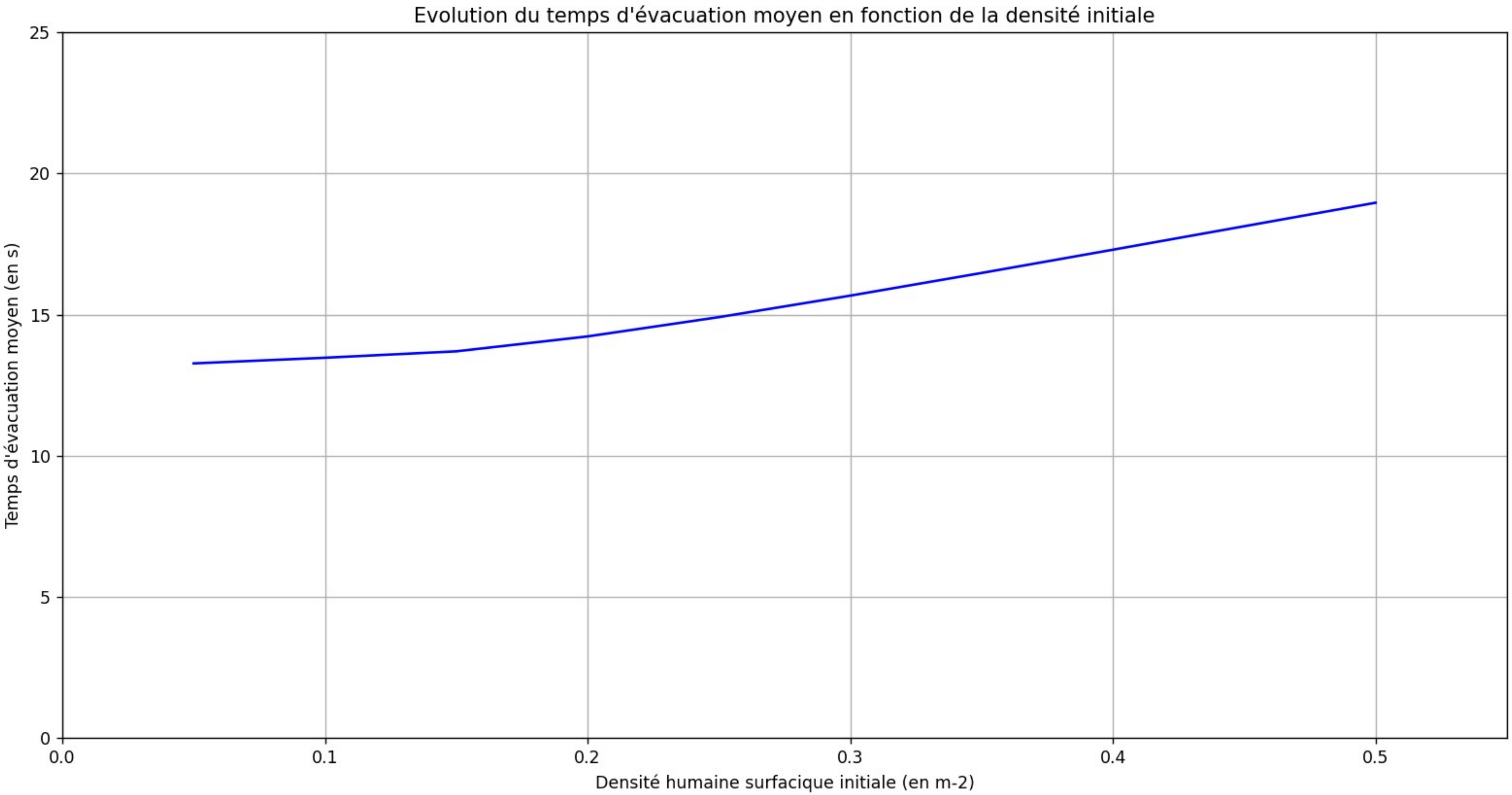
Grâce à la condition initiale, nous pouvons déterminer une solution approchée par la méthode d'Euler. Le pas de temps utilisé pour cela est :

$$dt = 0{,}005 \ s$$

Application au cône convergent



Application au cône convergent

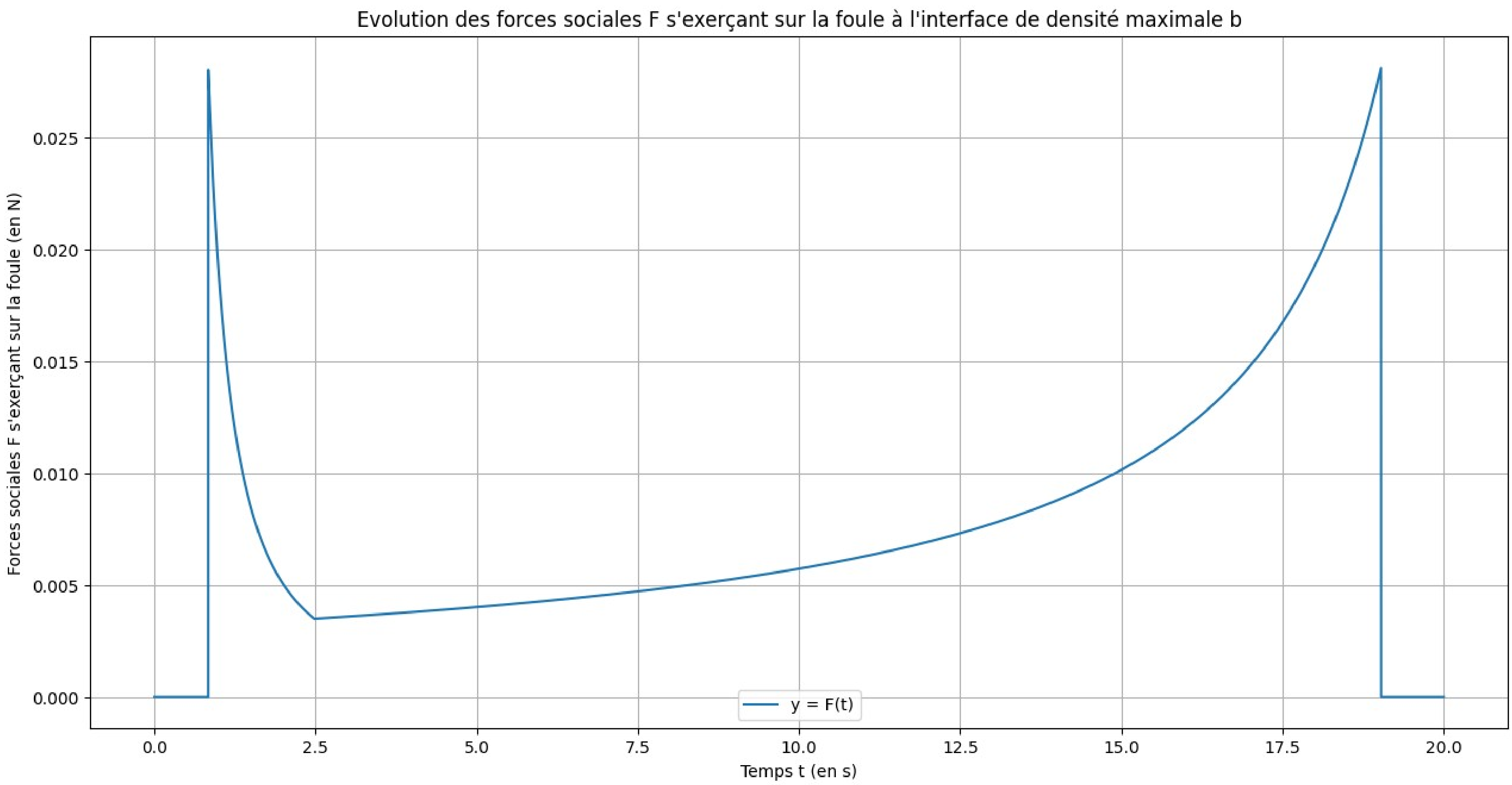


Application au cône convergent

Rappelons la relation : $\frac{\partial \vec{v}}{\partial t} + (\vec{v}.\vec{\nabla})\vec{v} = \vec{F}$

La connaissance de $b(t)$ permet de déterminer la vitesse réelle de la foule et par cela les forces présentes dans cette dernière.

Application au cône convergent



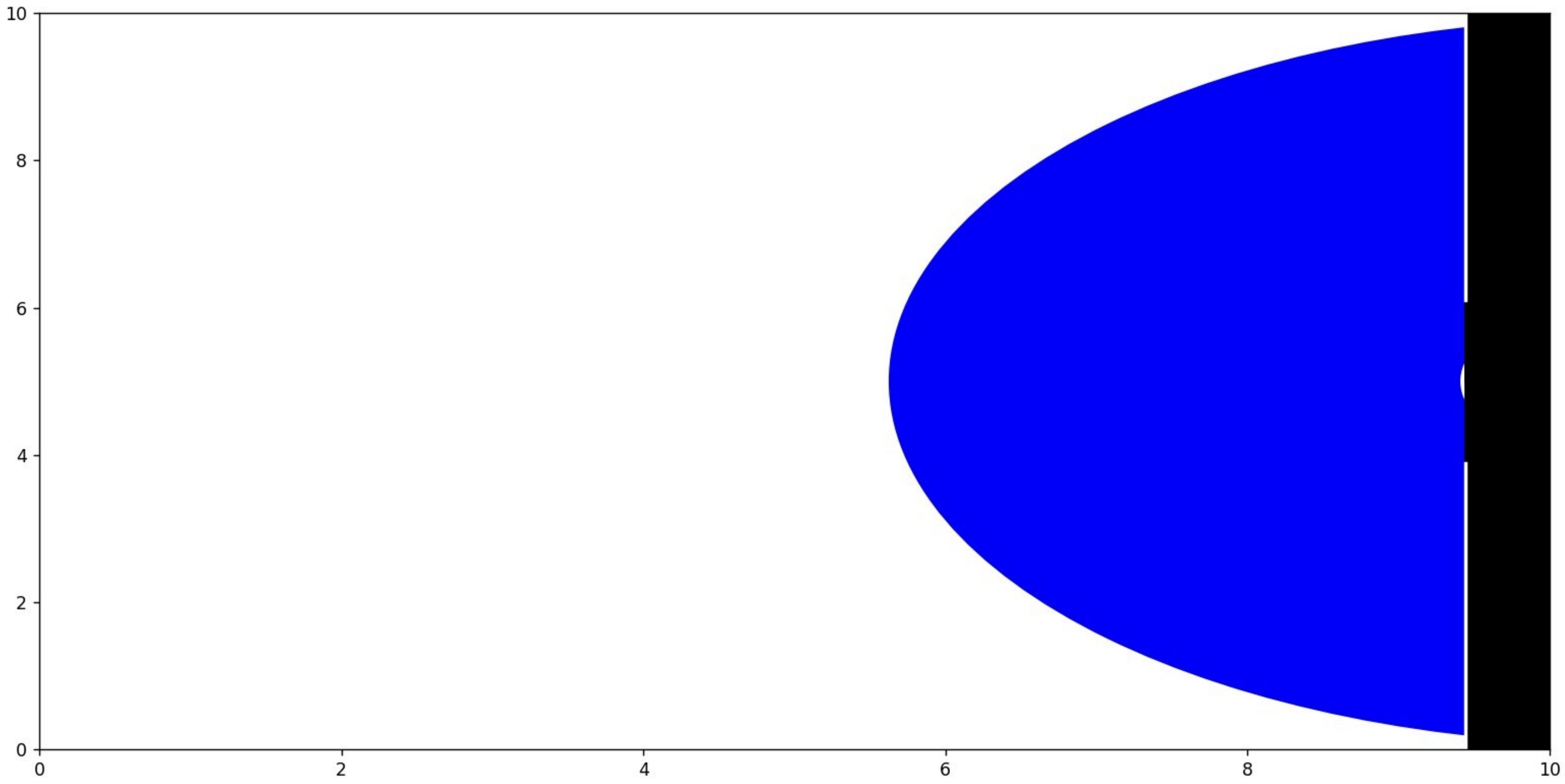
Application au cône convergent

La plupart des configurations dans lesquelles les gens se déplacent ne sont pas coniques.

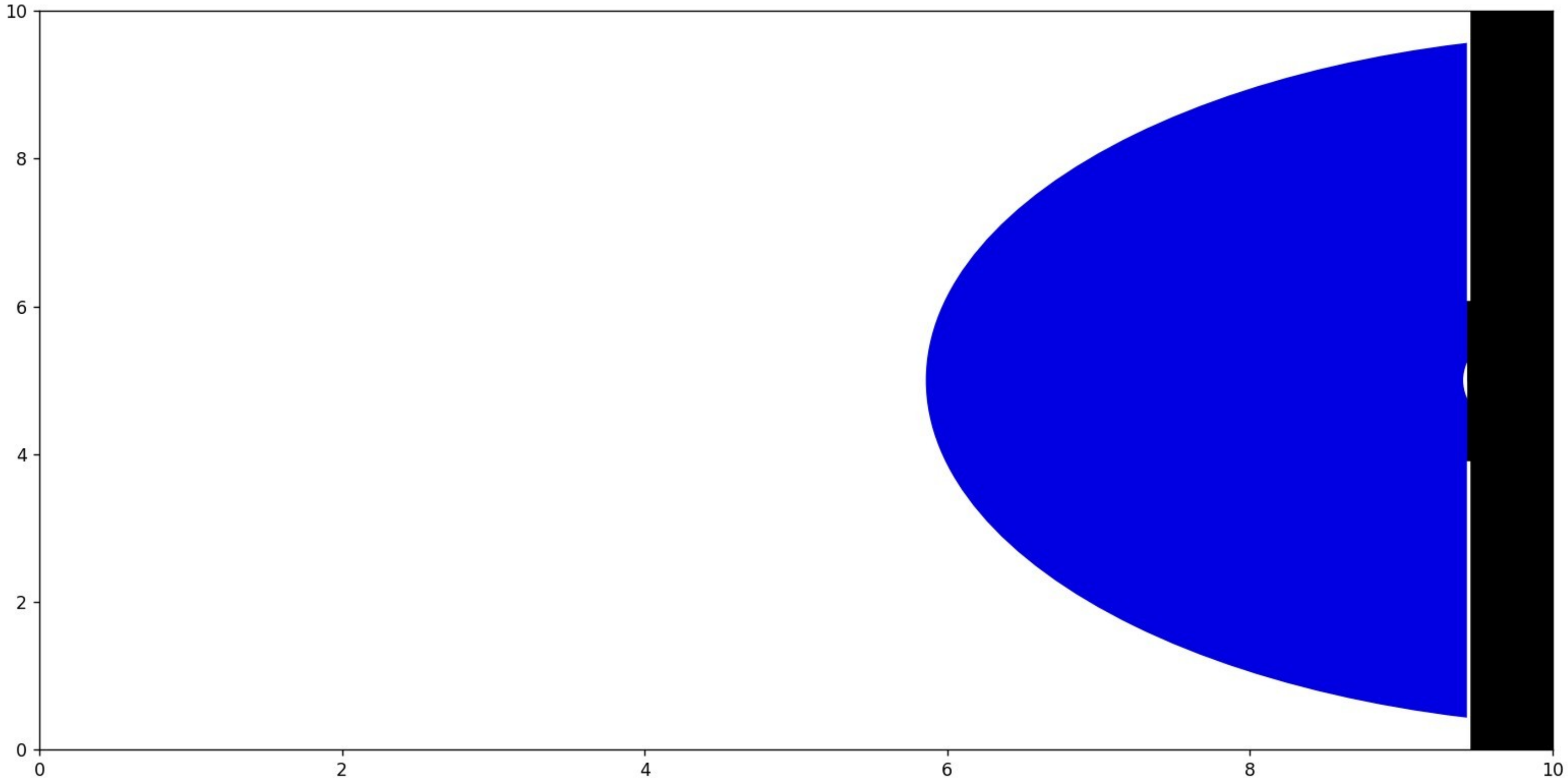
Cependant les équations qui régissent le mouvement de la foule, dans notre cône, présentent des invariances vis-à-vis des angles de ce dernier.

Ainsi les résultats montrés restent vrais même si le cône est « ouvert », autrement dit, dans un espace possédant un mur avec une sortie.

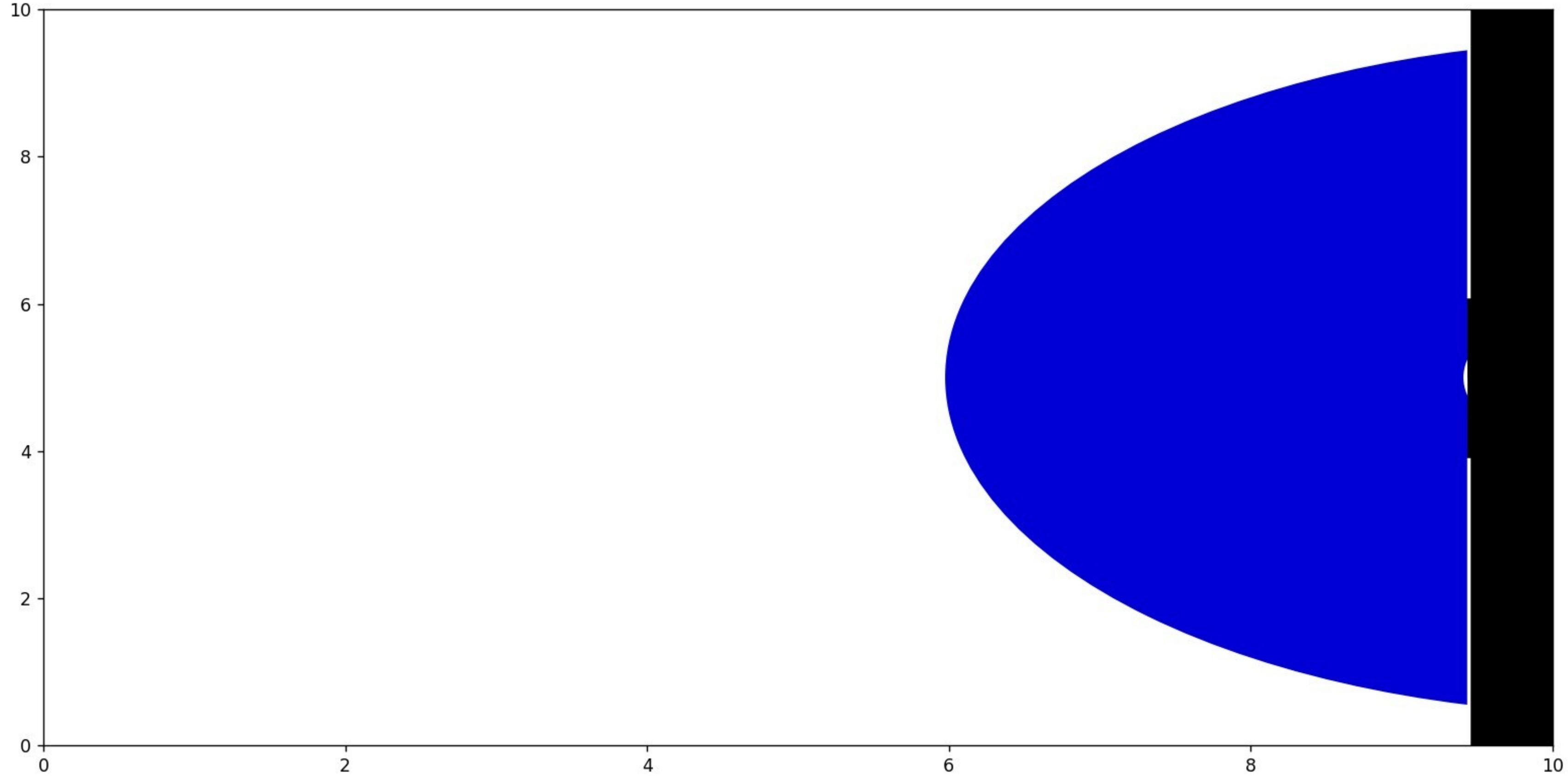
Application au cône convergent



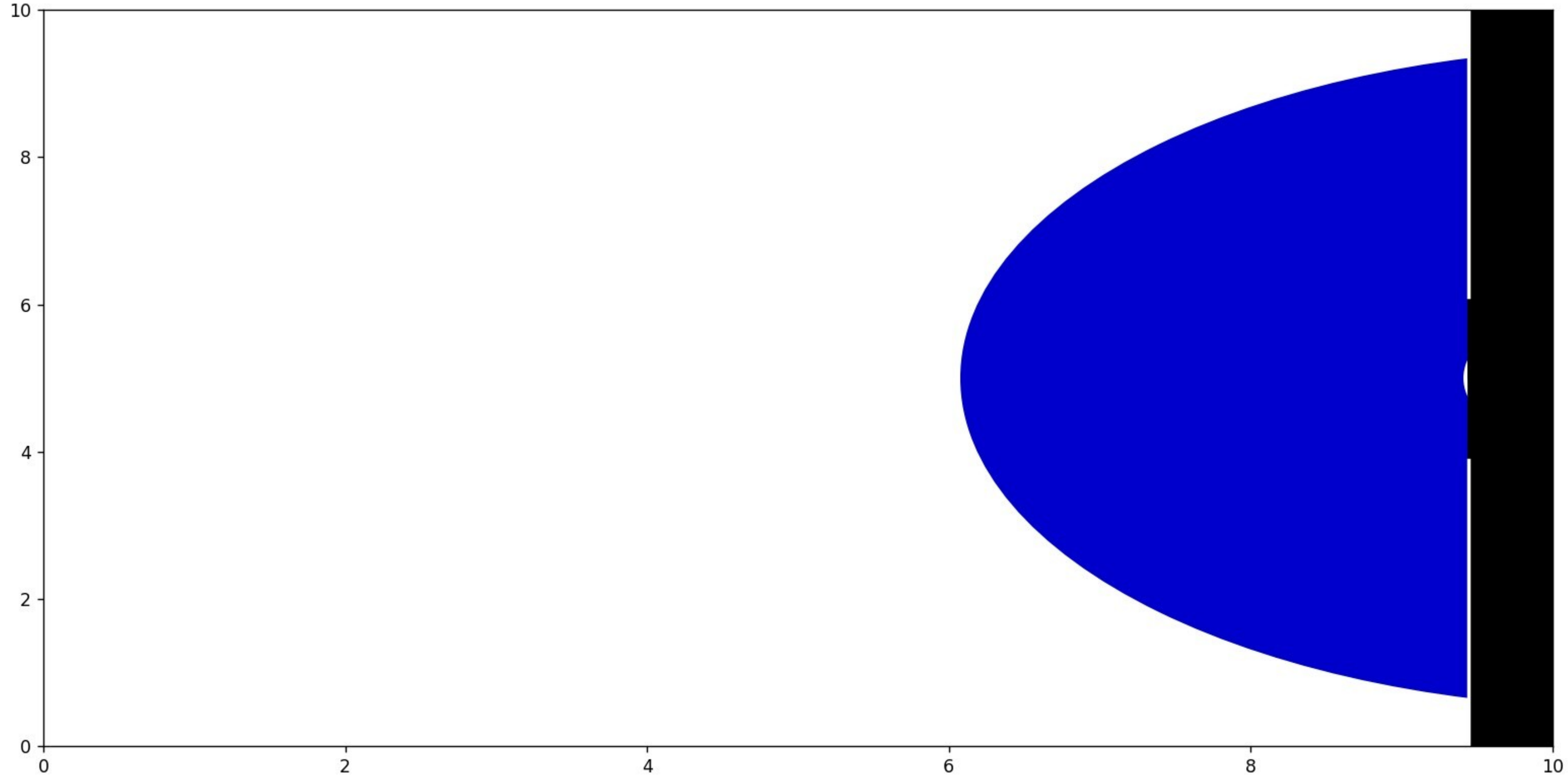
Application au cône convergent



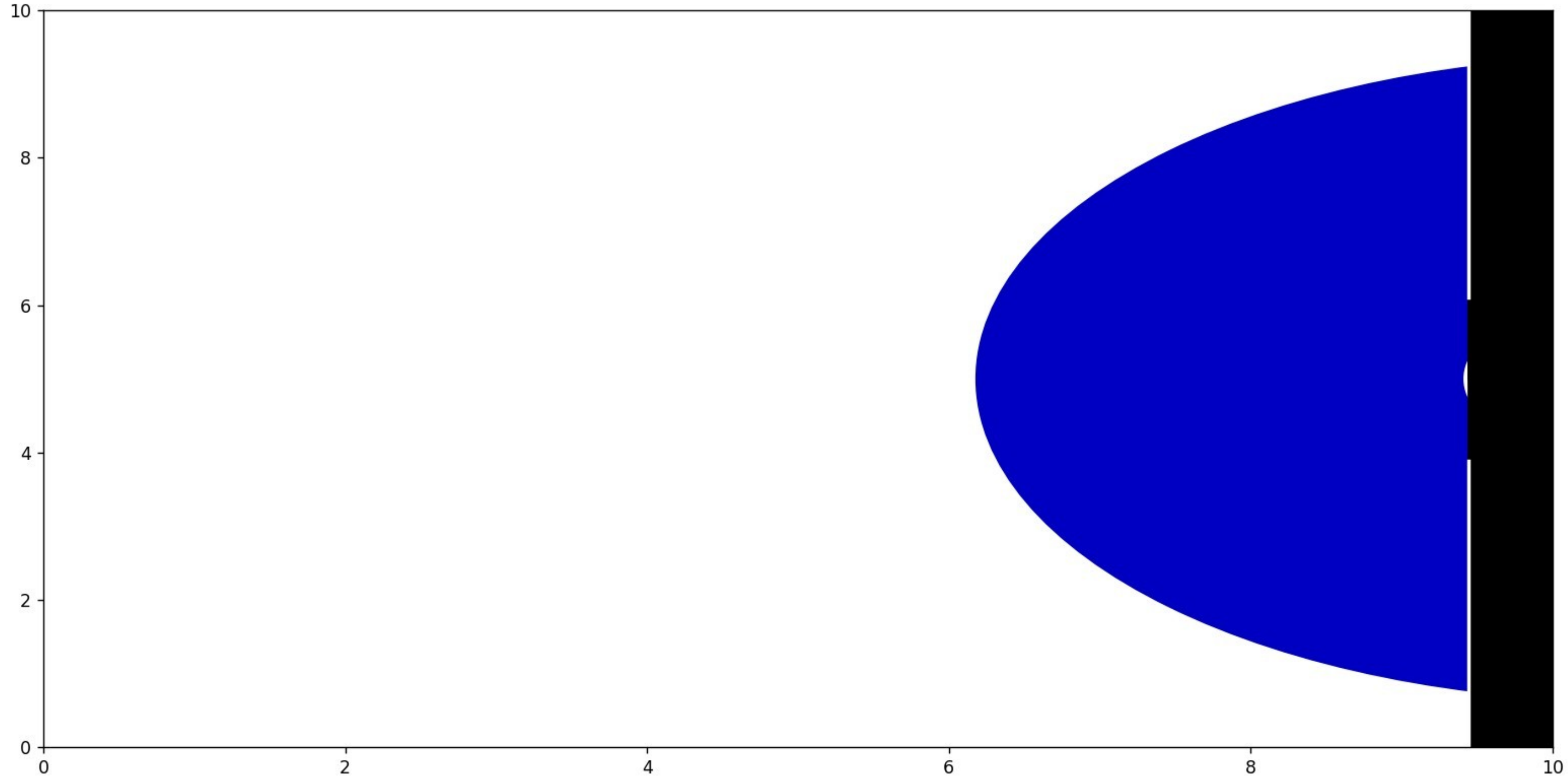
Application au cône convergent



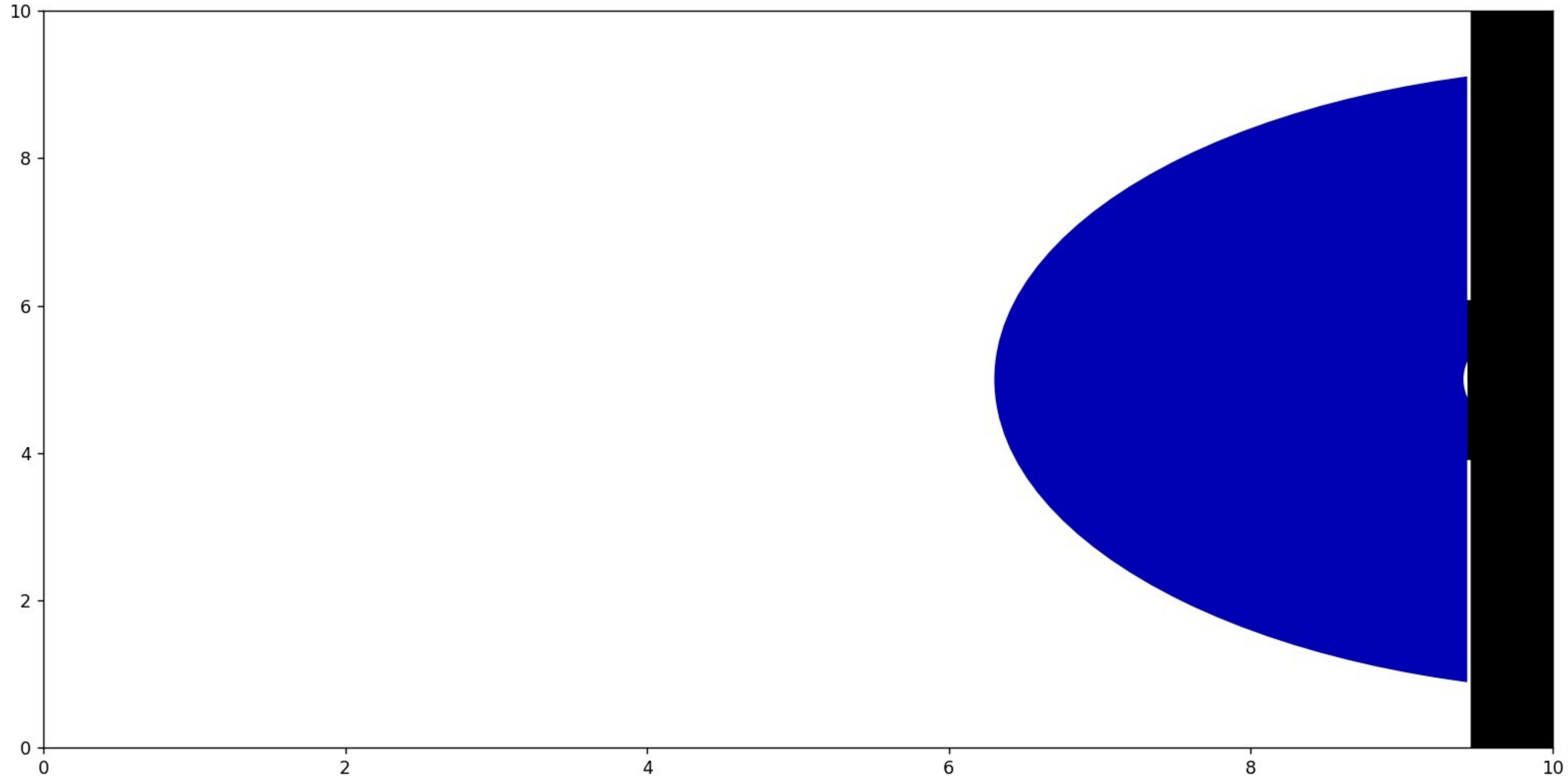
Application au cône convergent



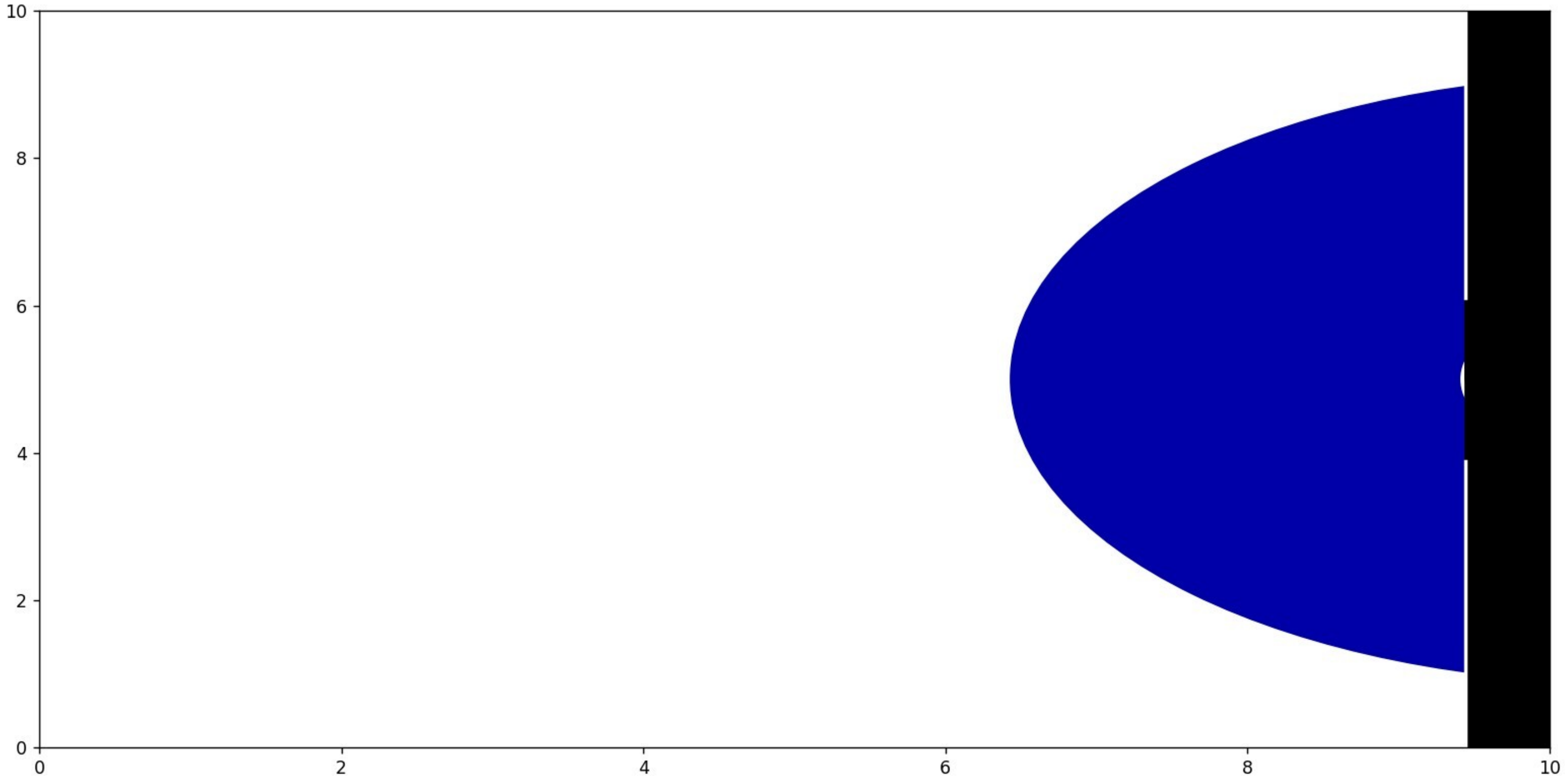
Application au cône convergent



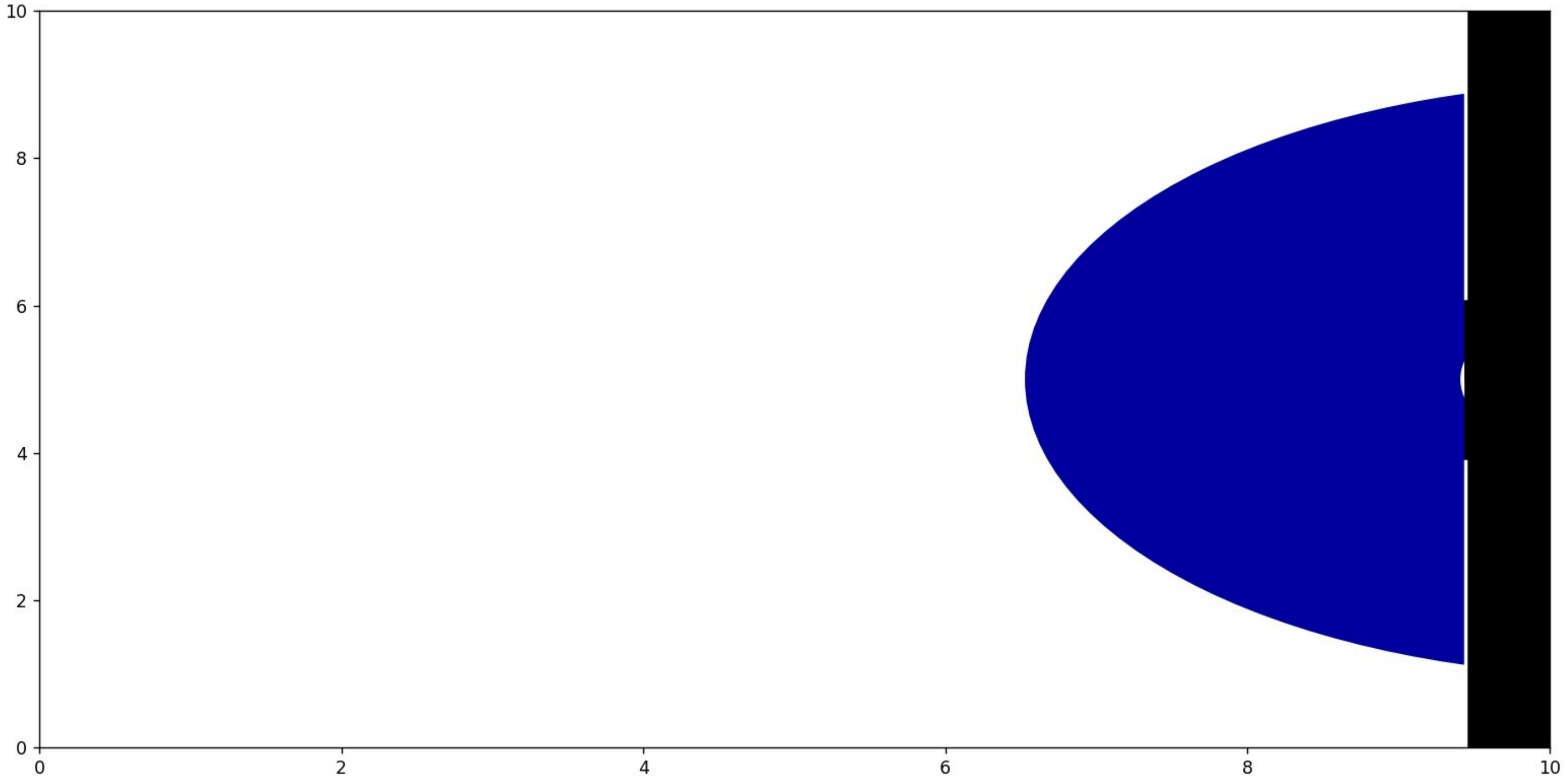
Application au cône convergent



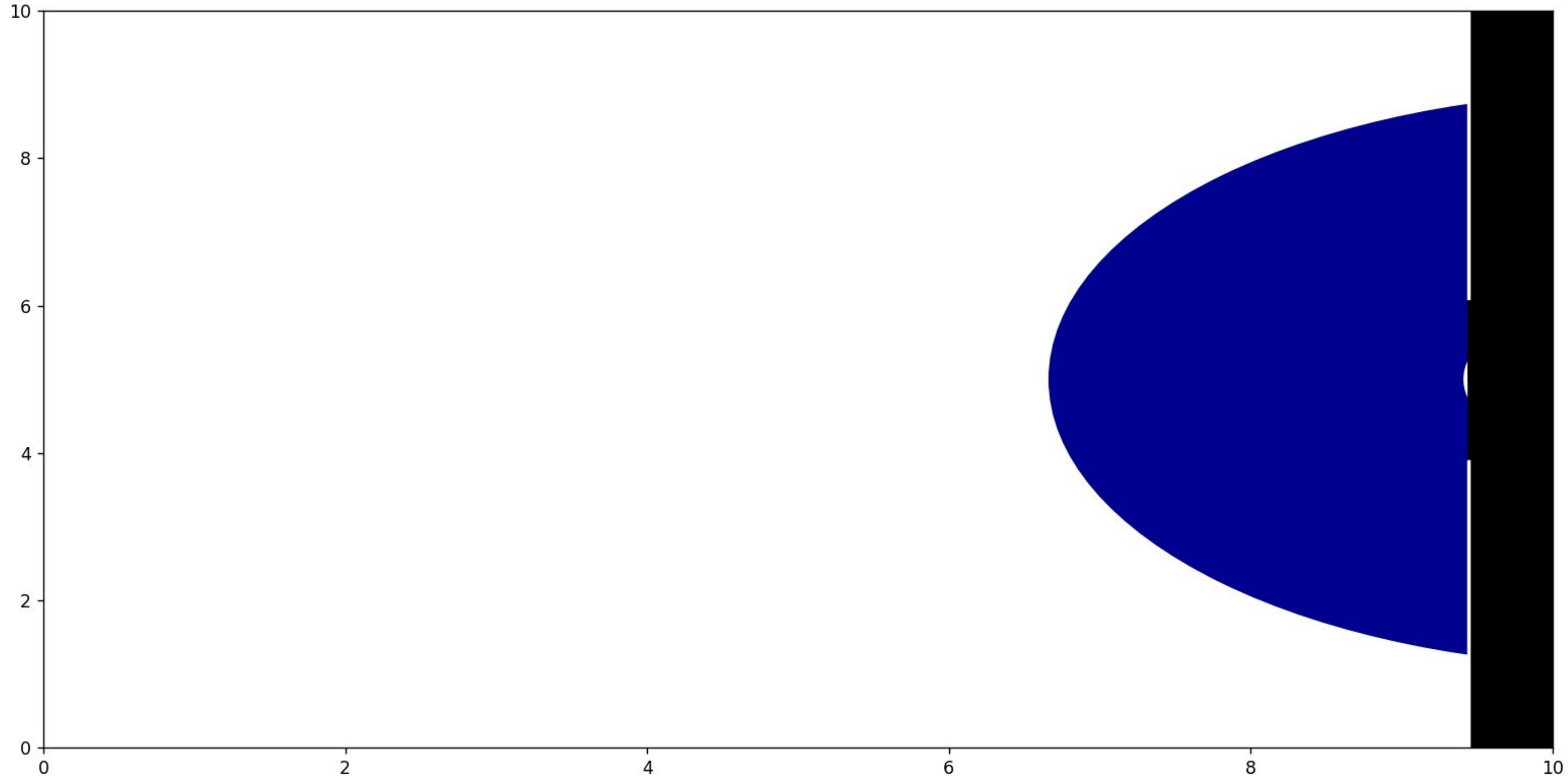
Application au cône convergent



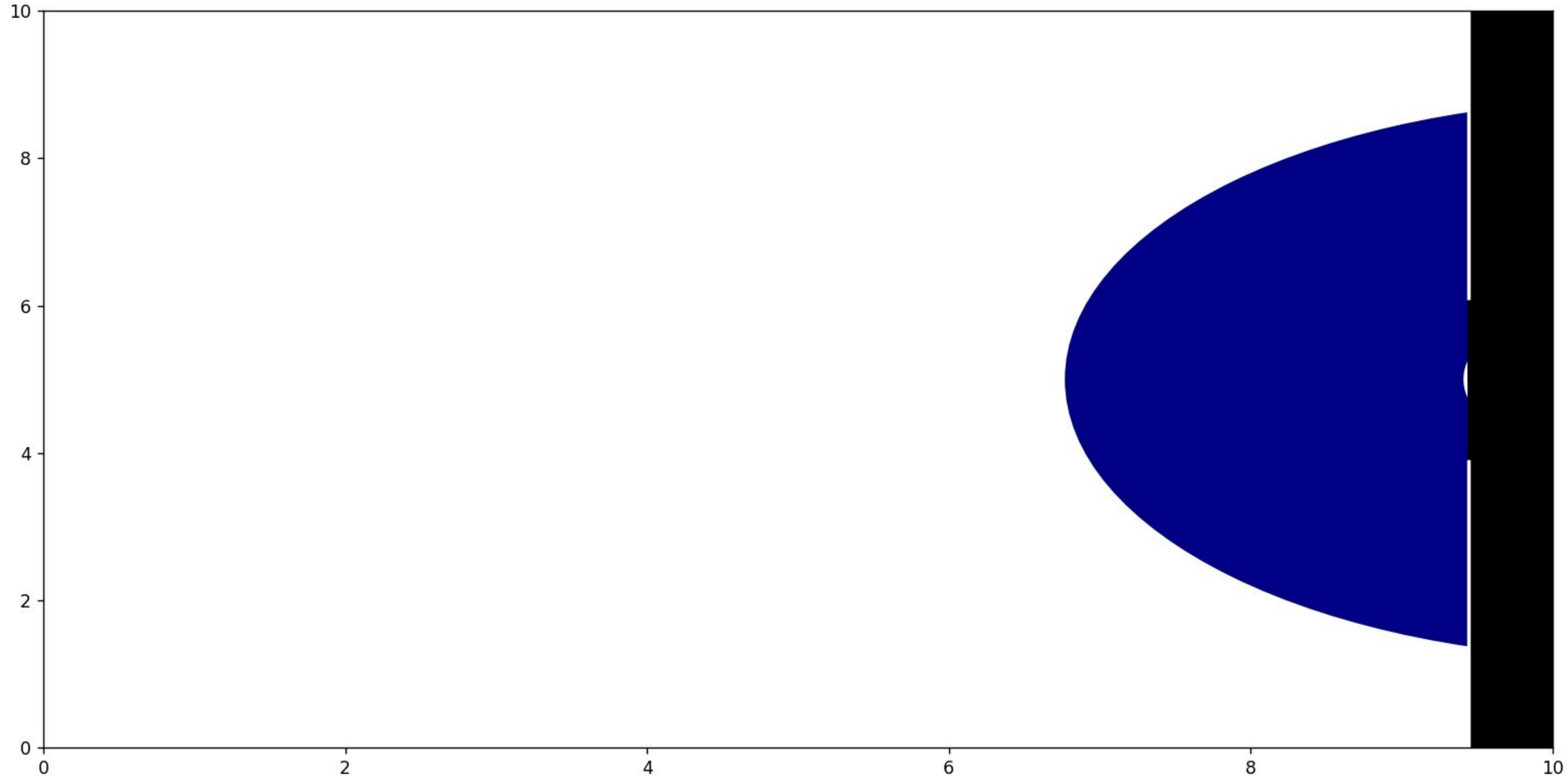
Application au cône convergent



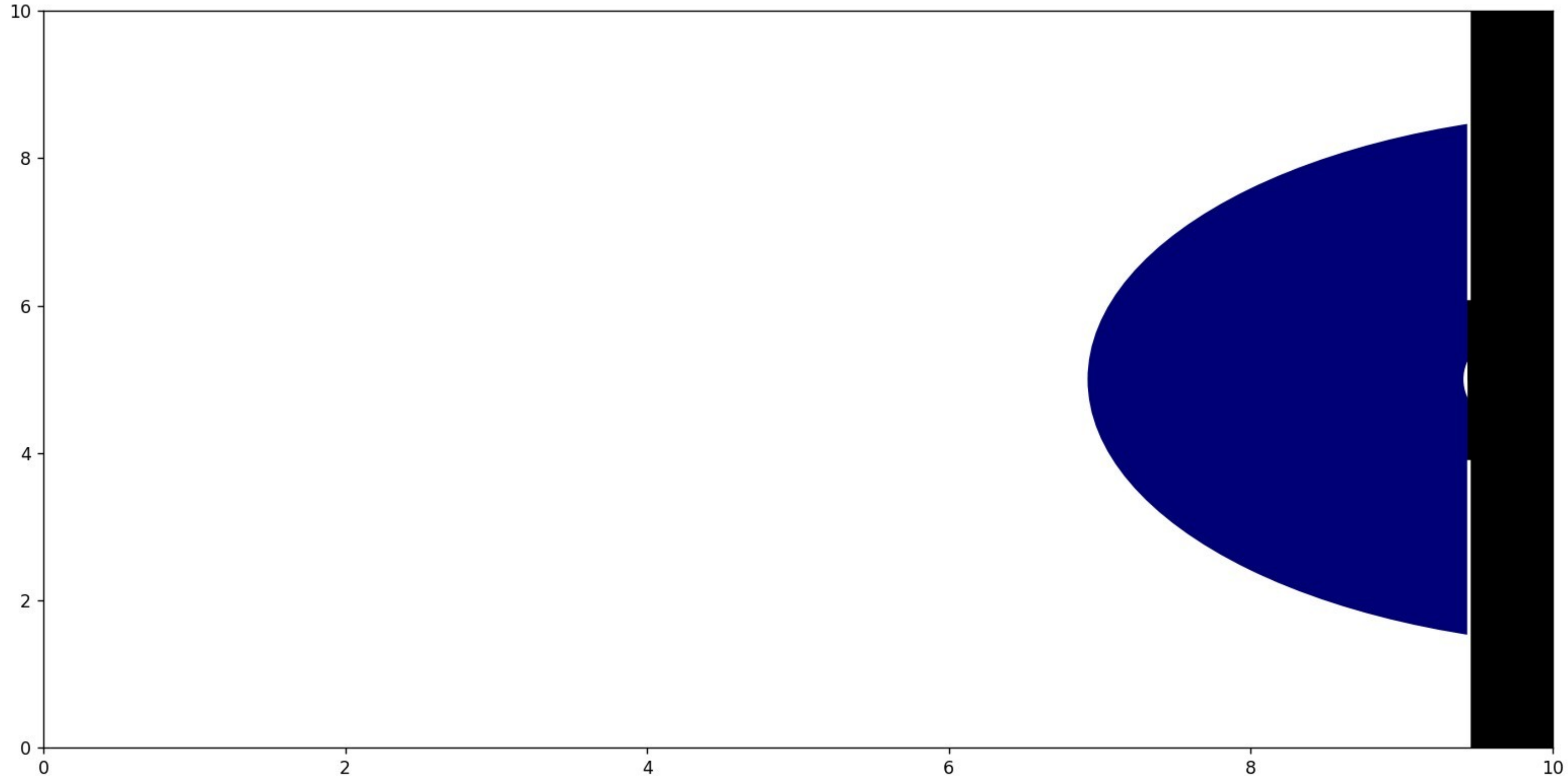
Application au cône convergent



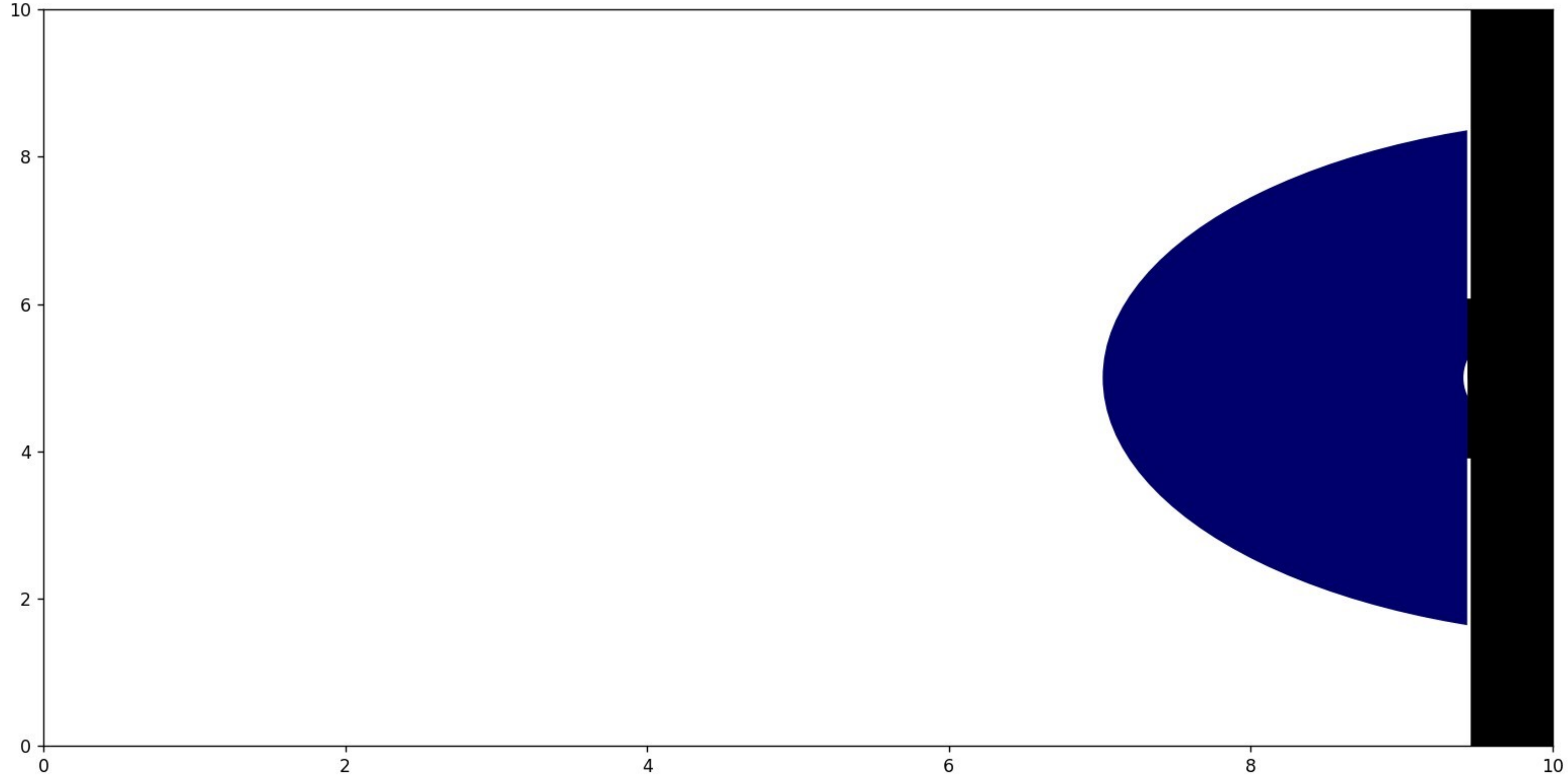
Application au cône convergent



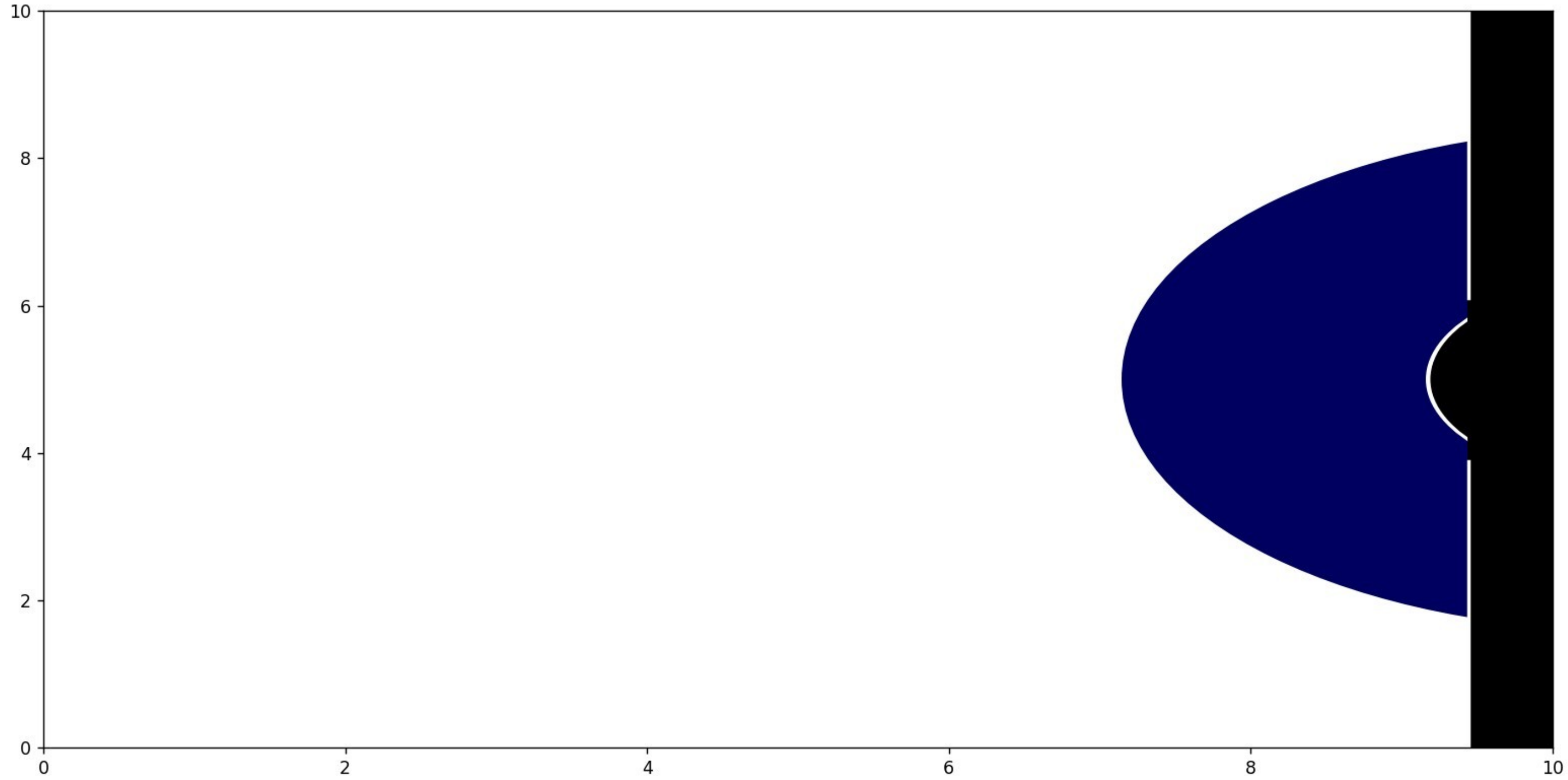
Application au cône convergent



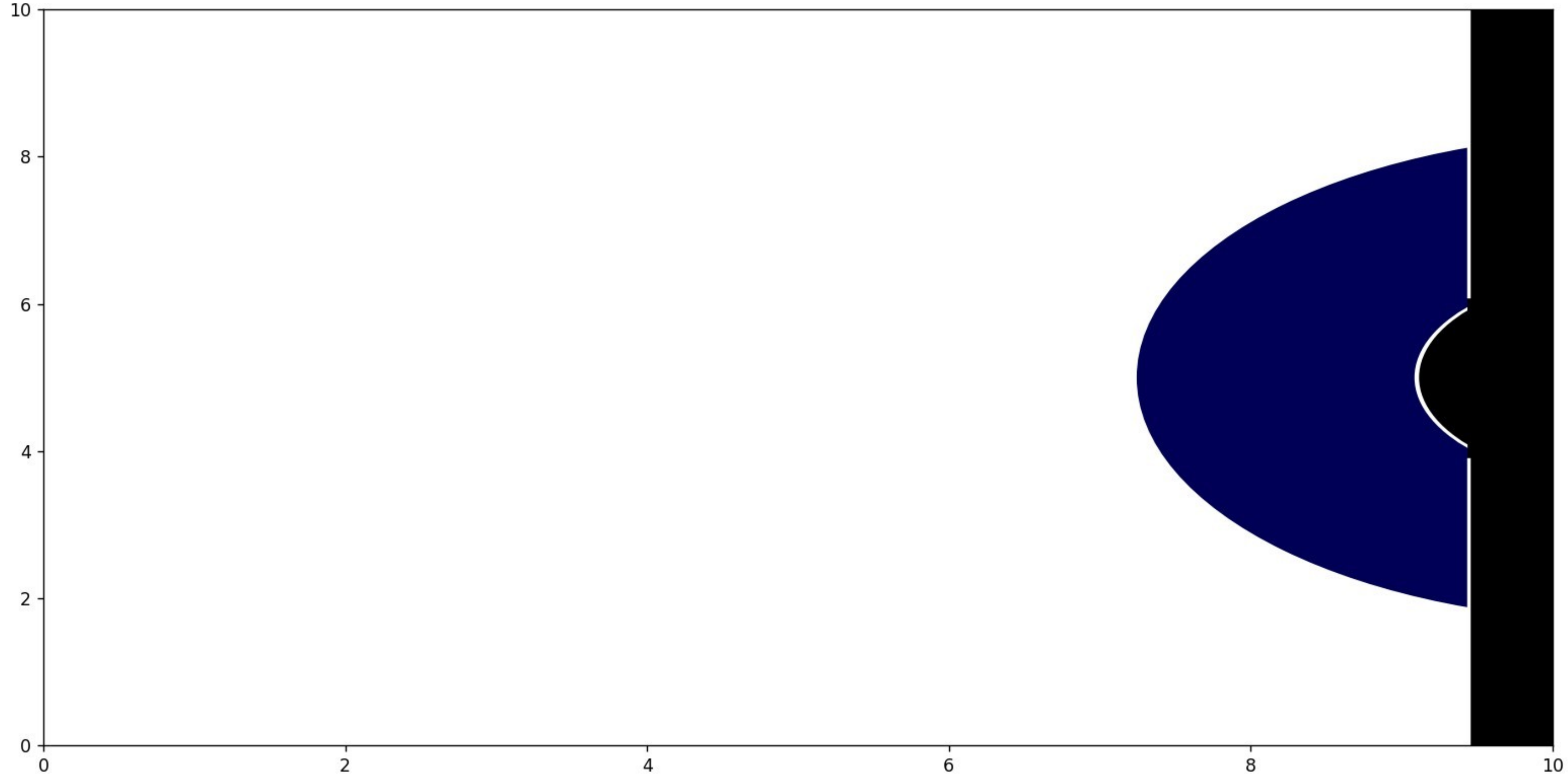
Application au cône convergent



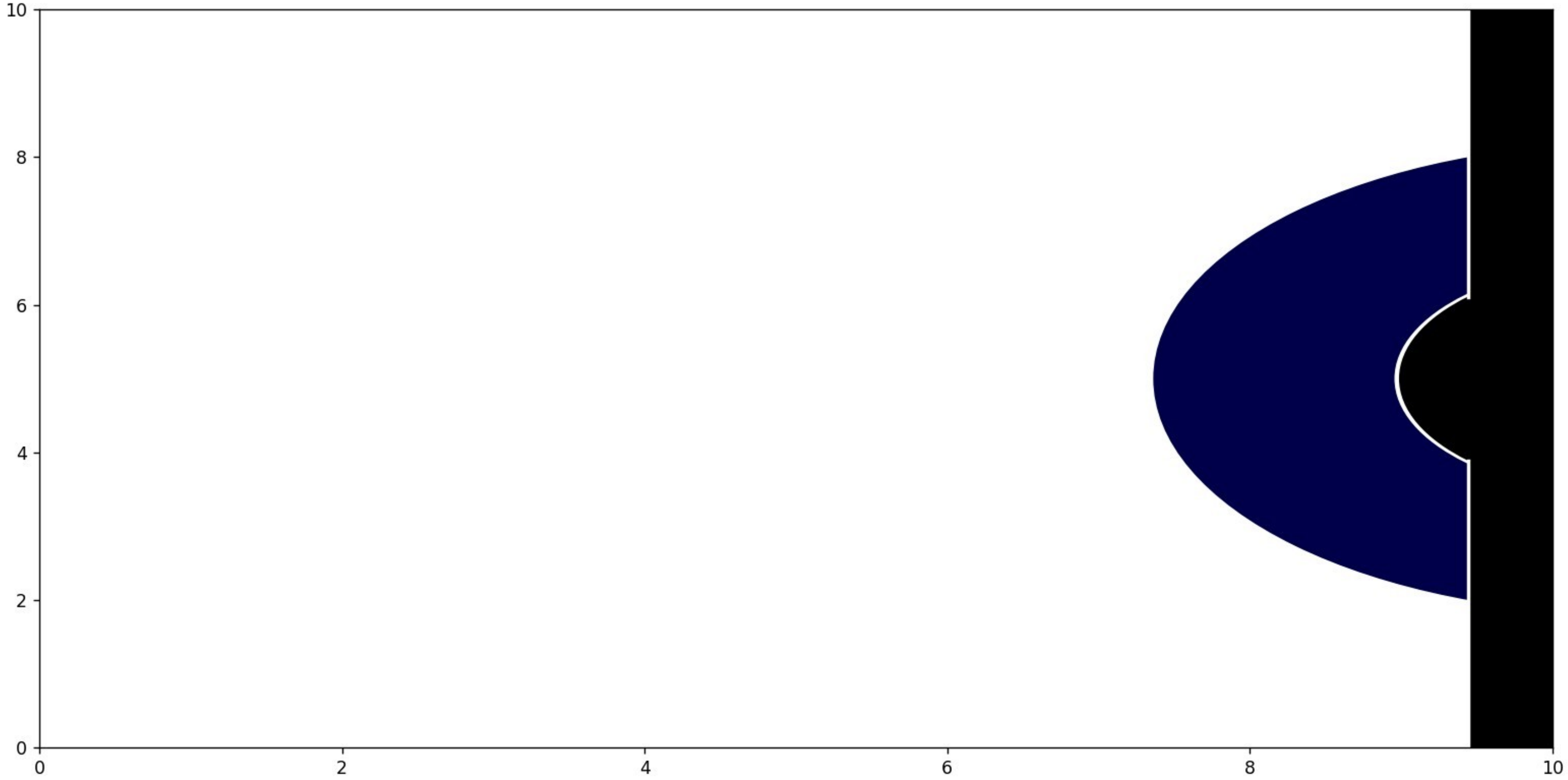
Application au cône convergent



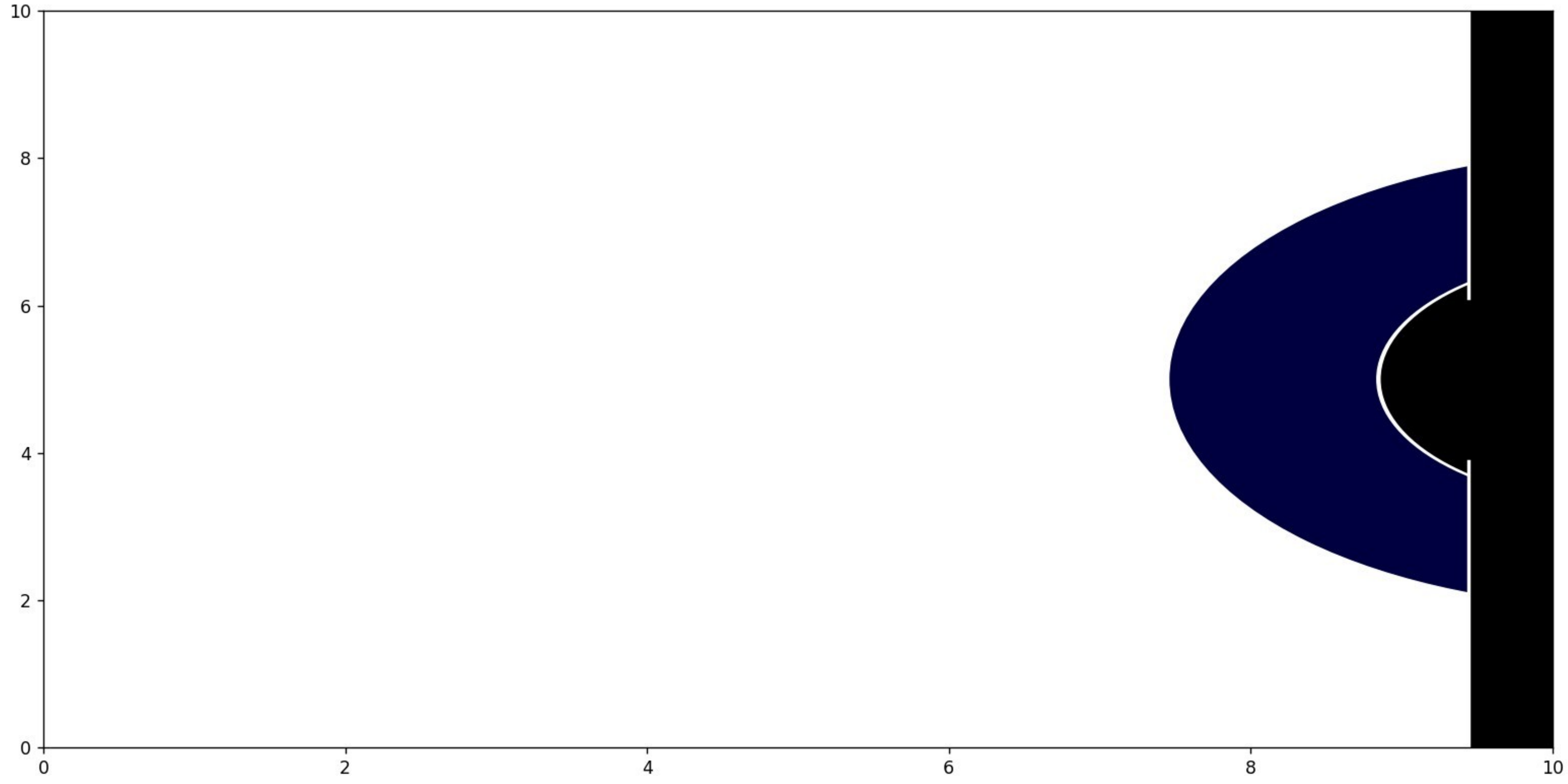
Application au cône convergent



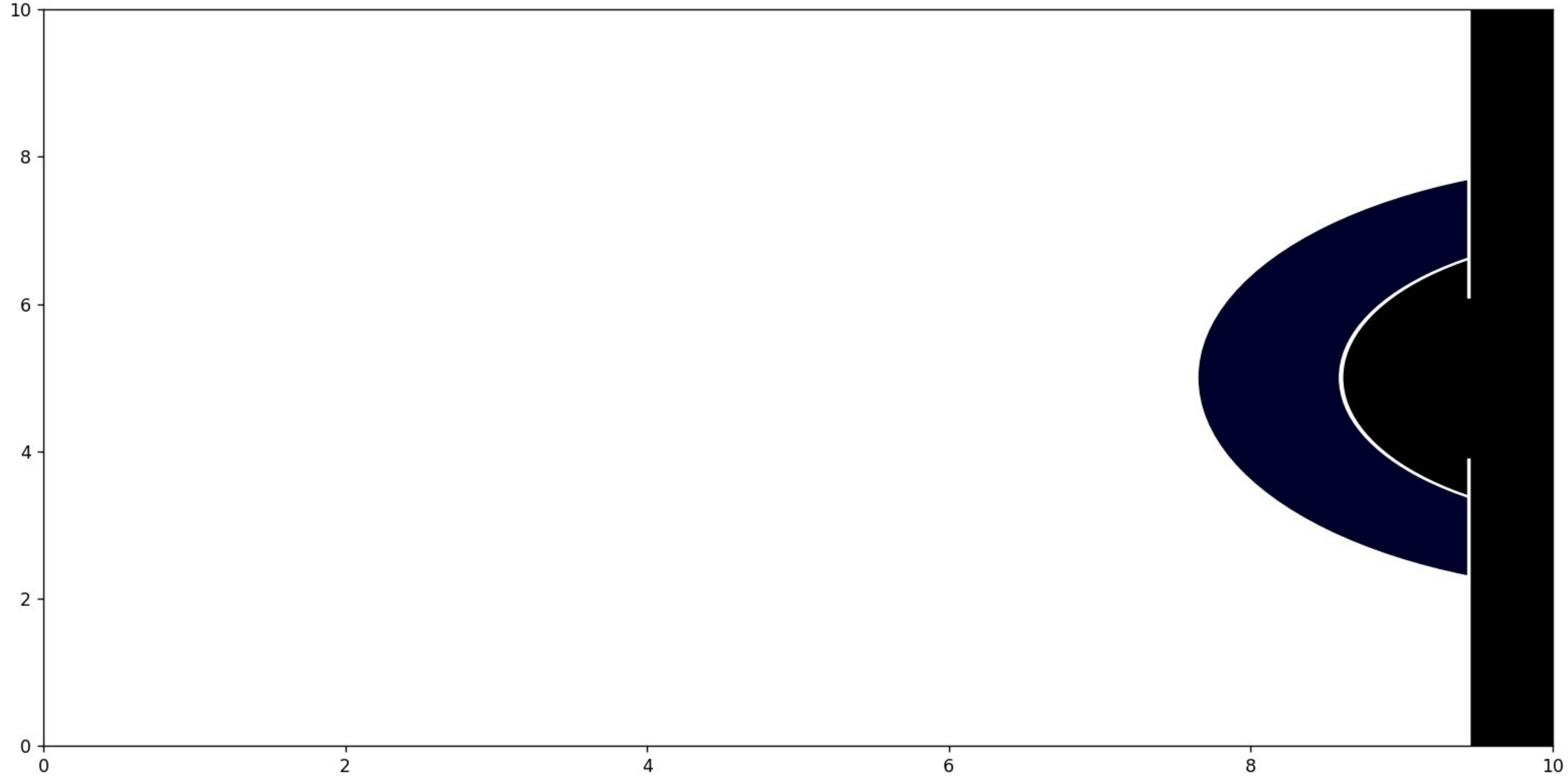
Application au cône convergent



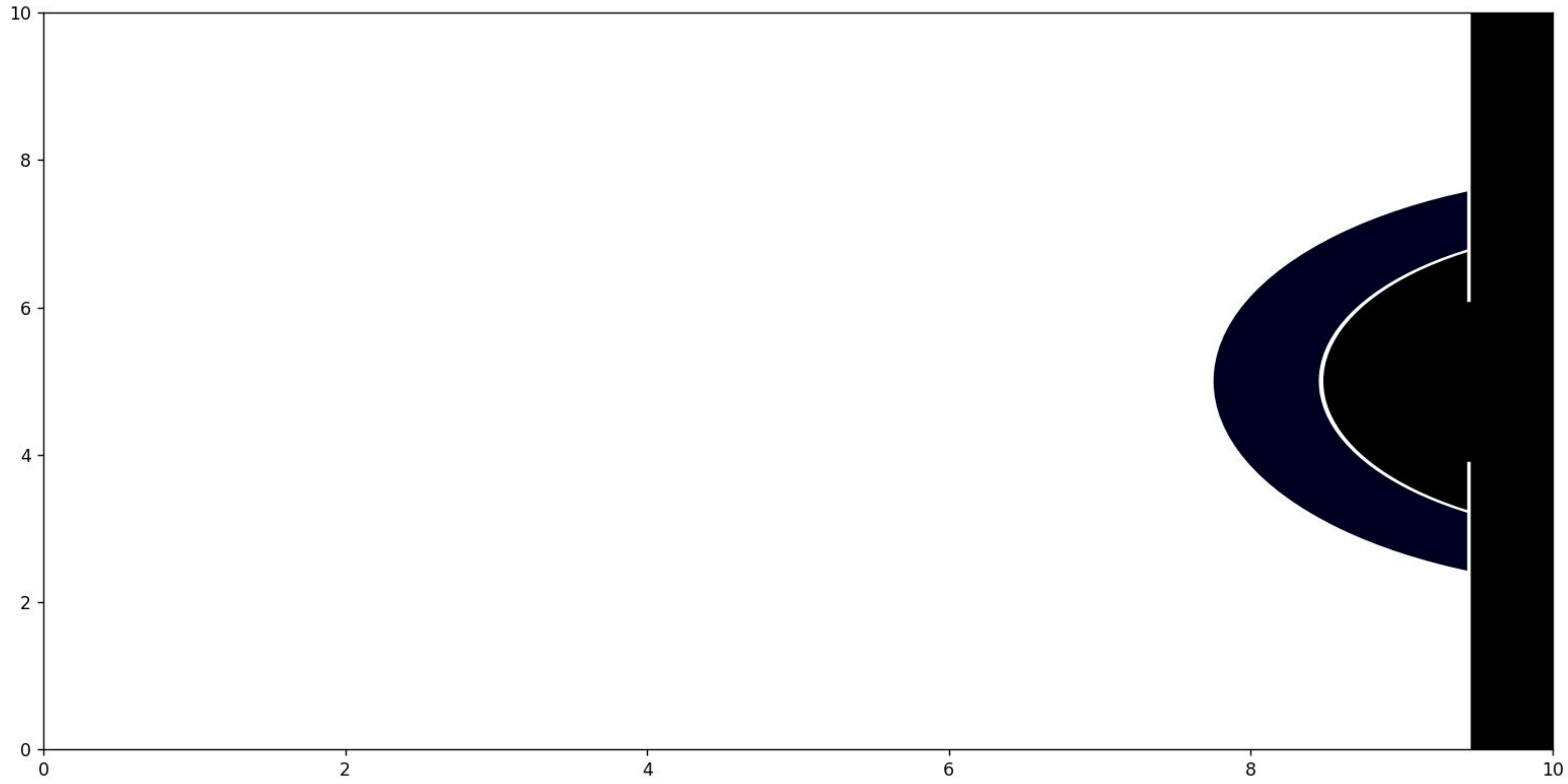
Application au cône convergent



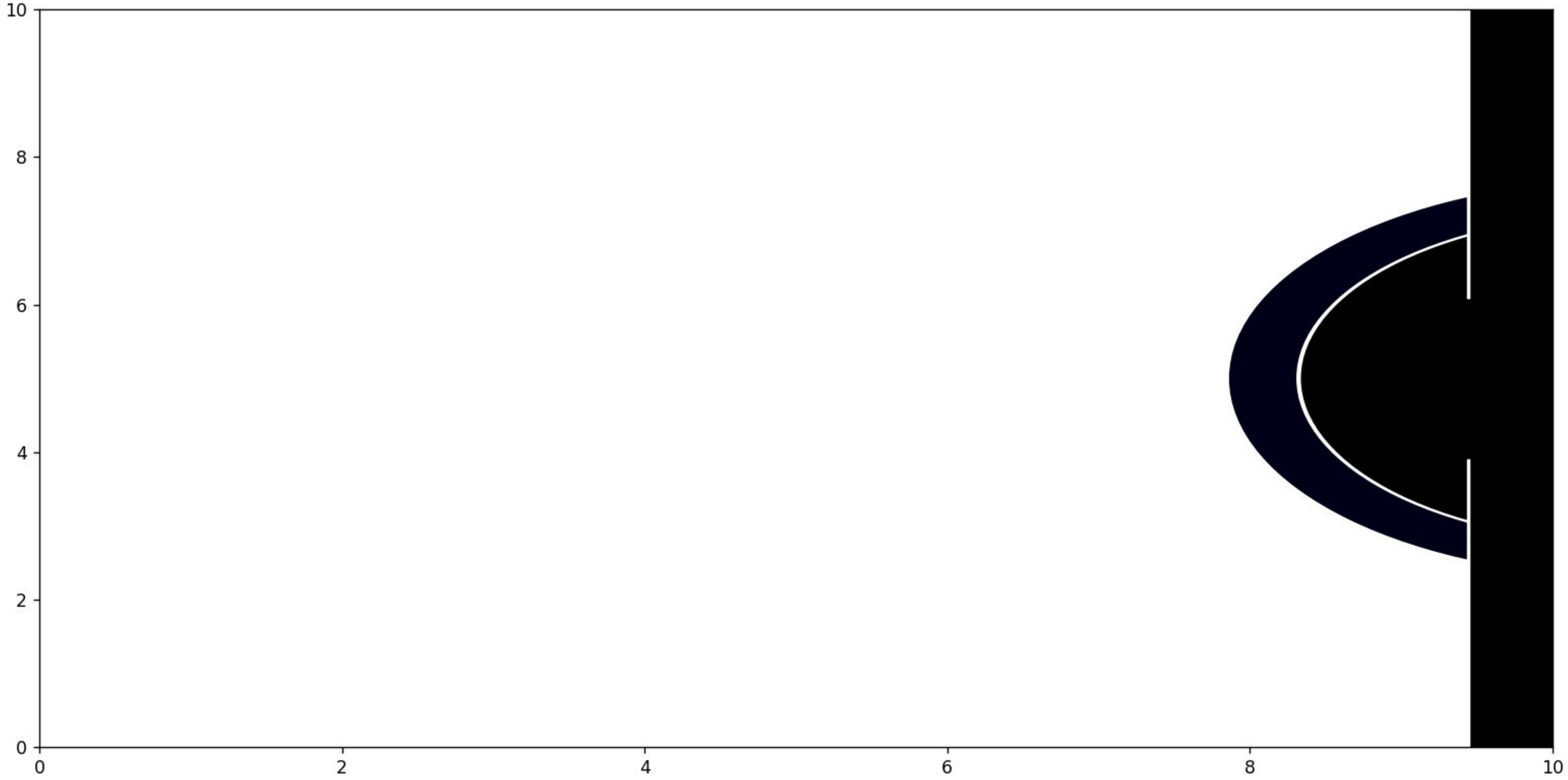
Application au cône convergent



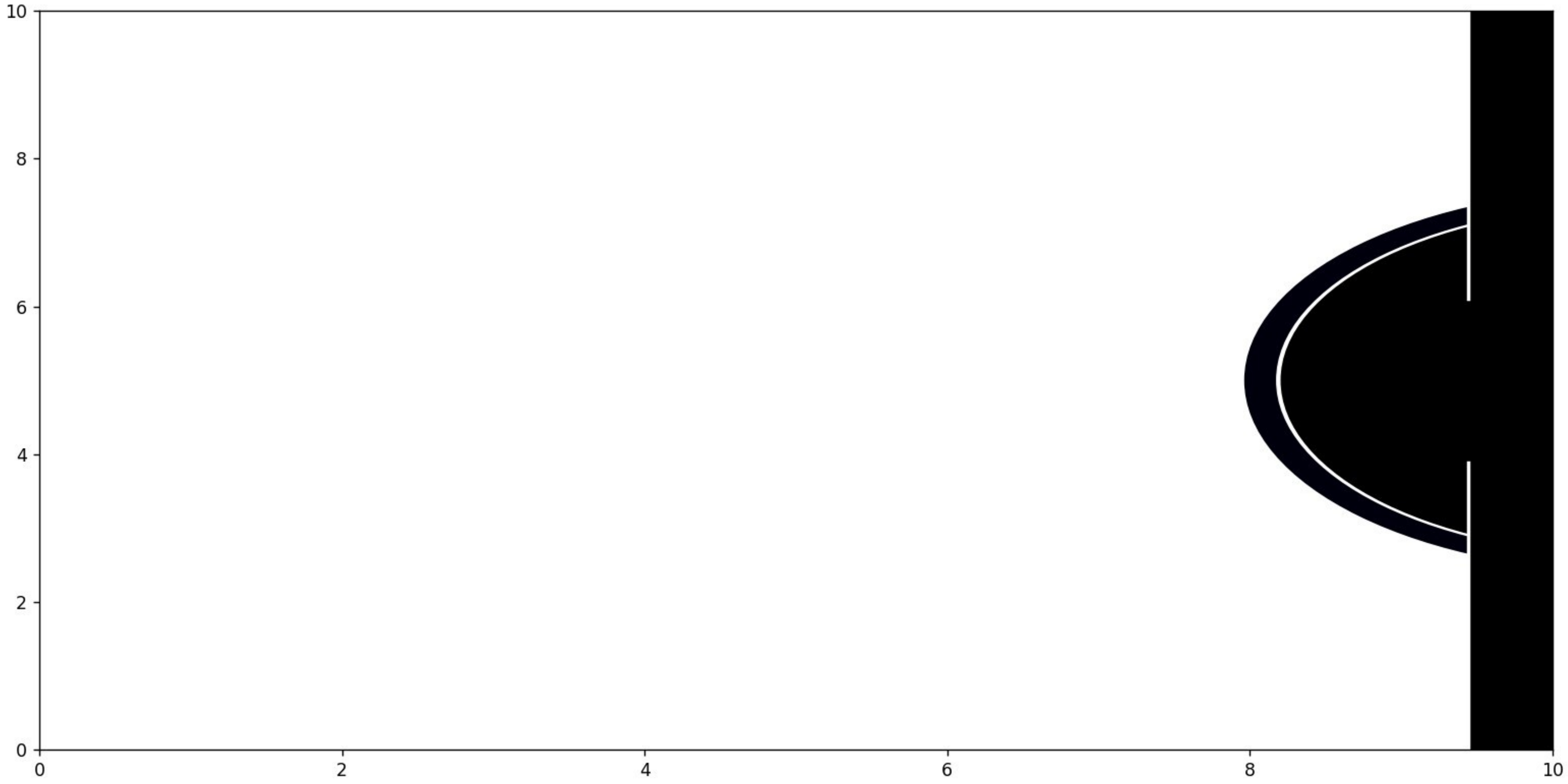
Application au cône convergent



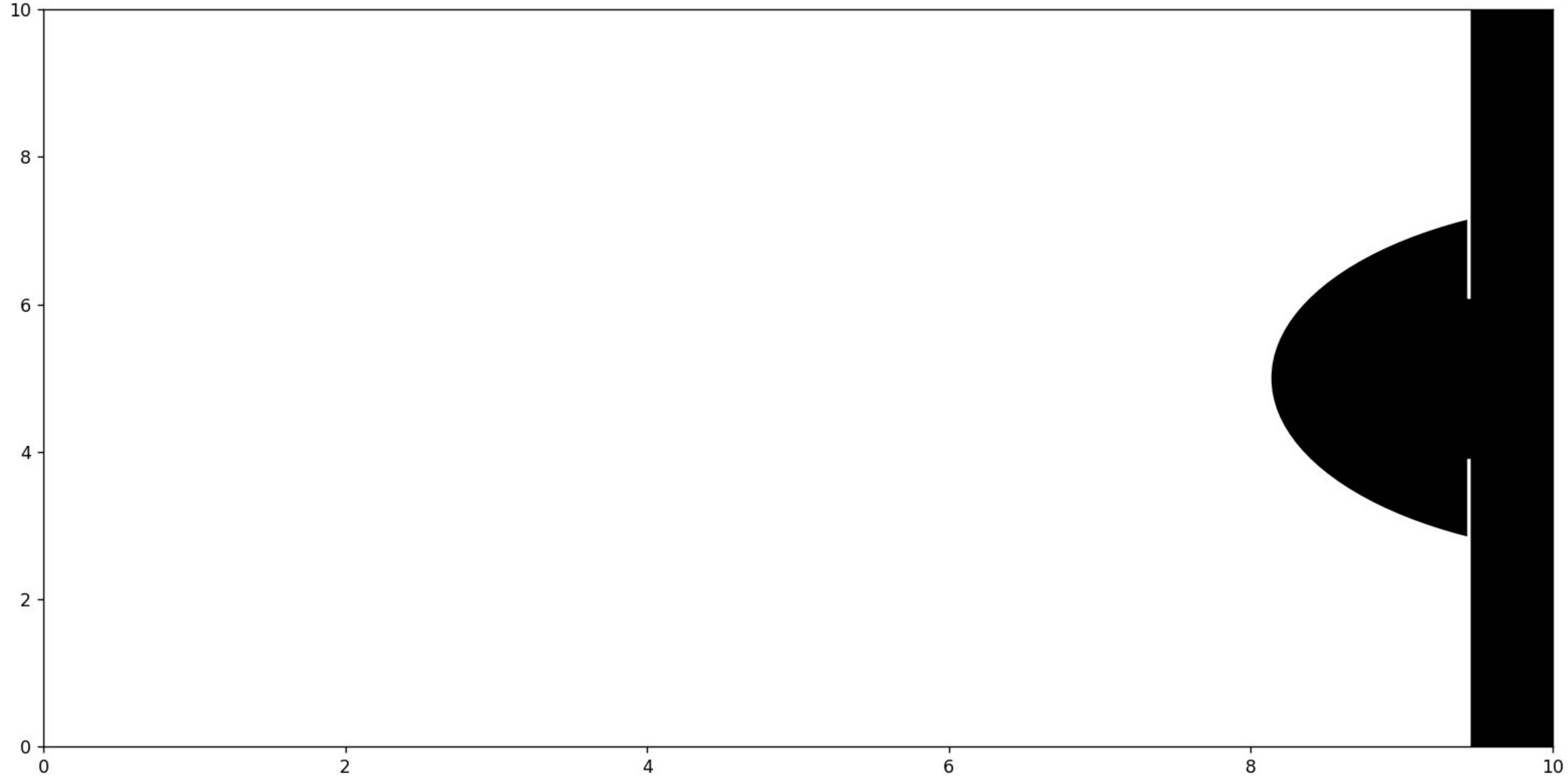
Application au cône convergent



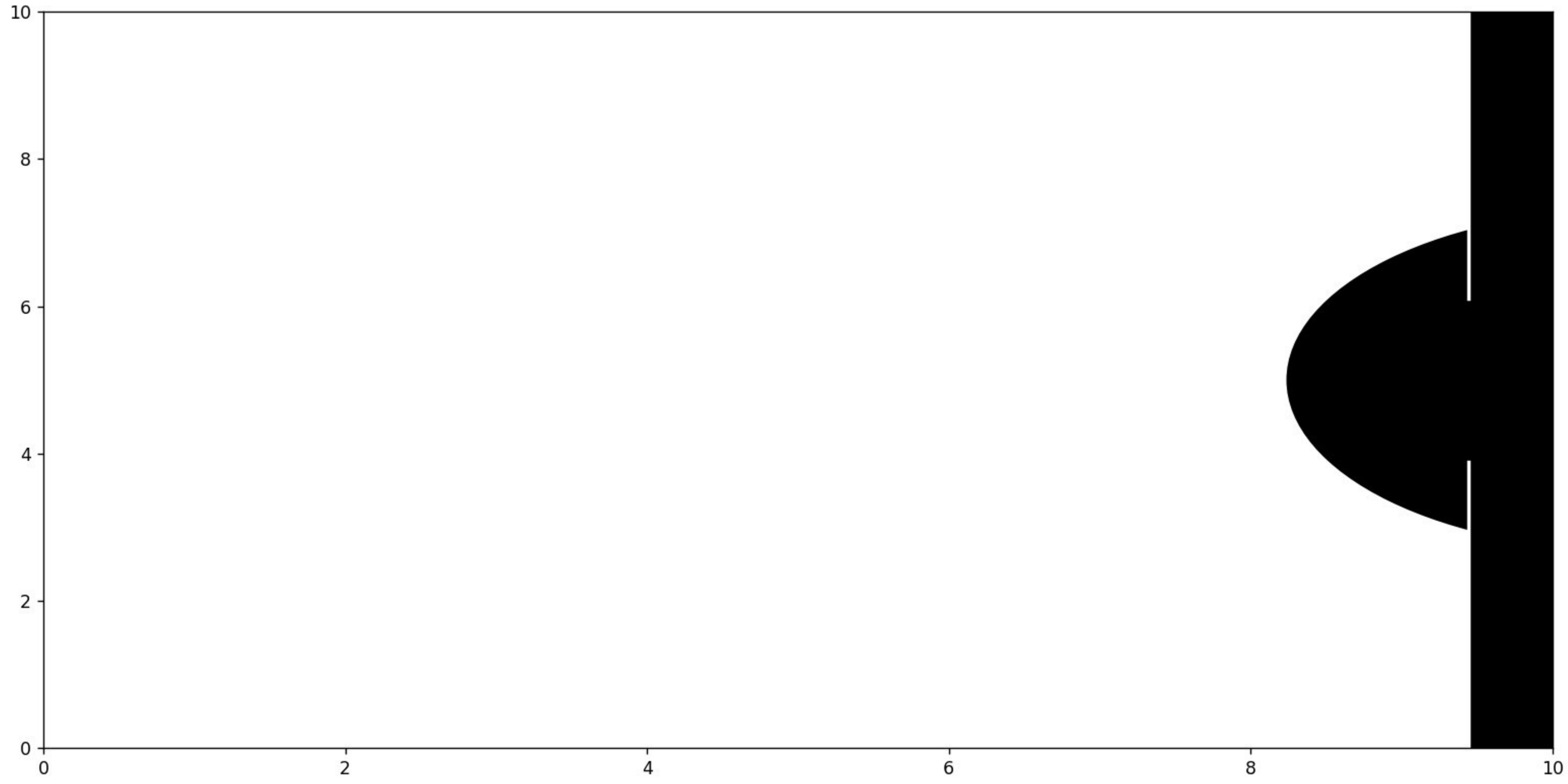
Application au cône convergent



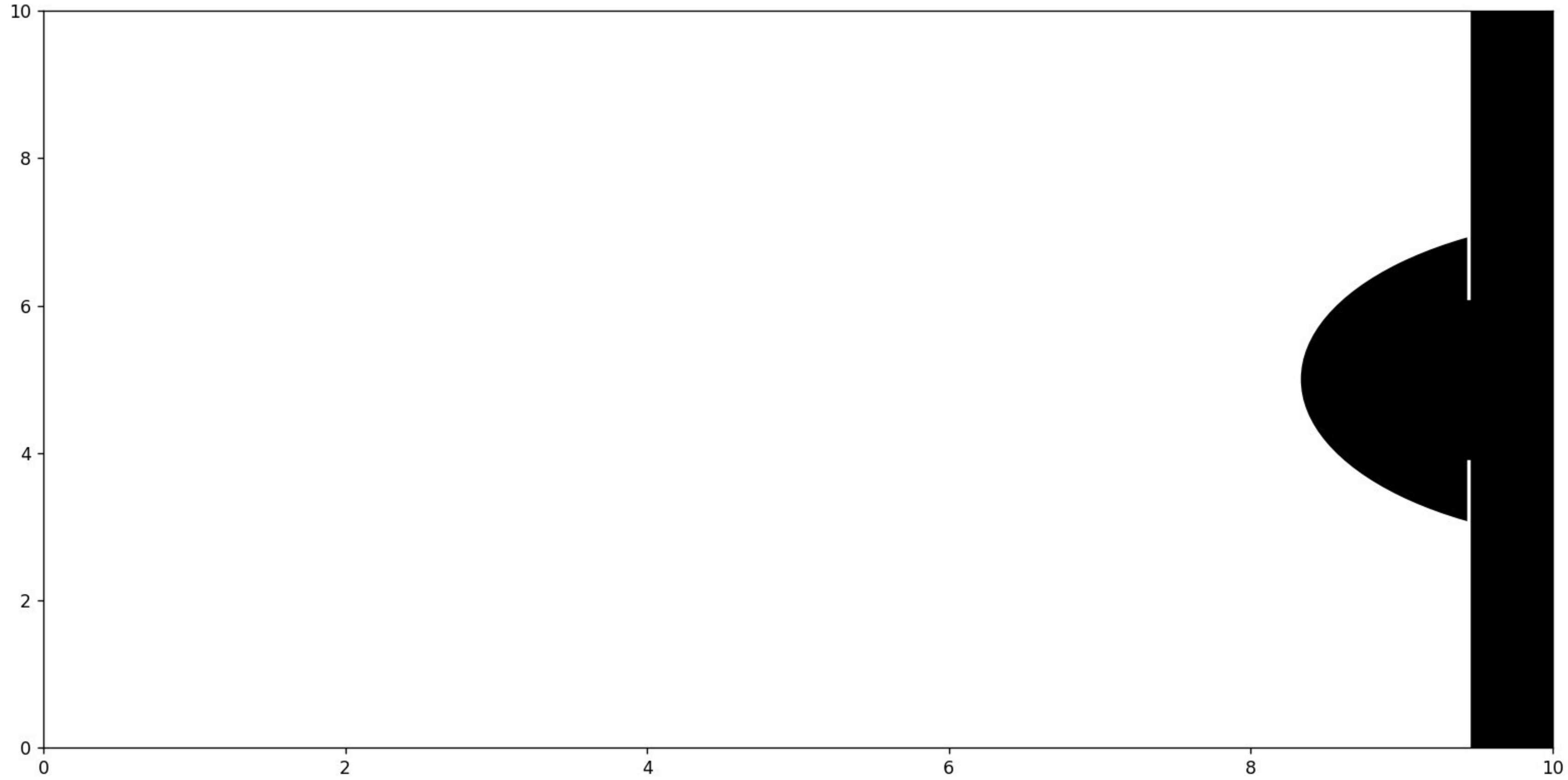
Application au cône convergent



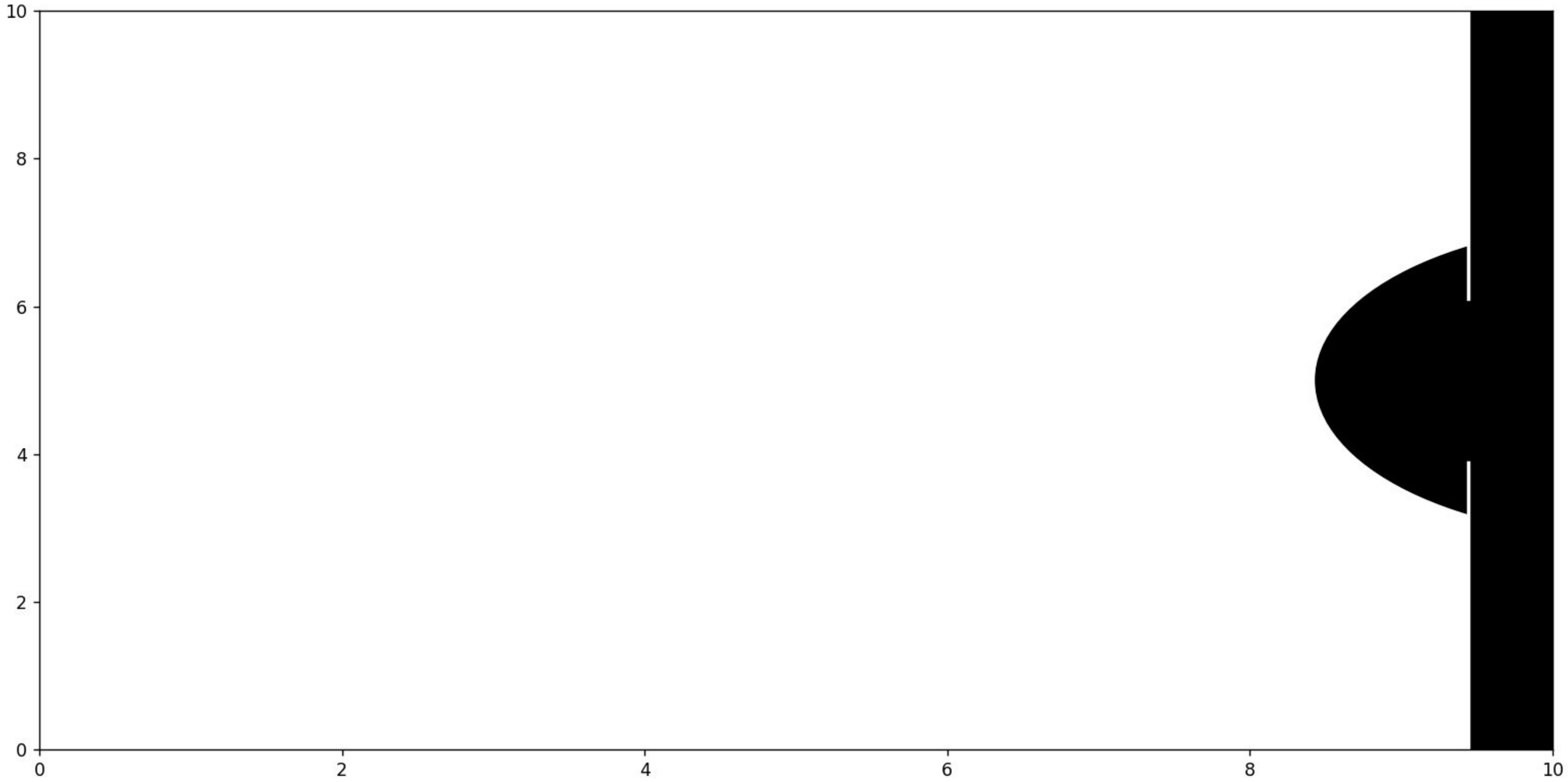
Application au cône convergent



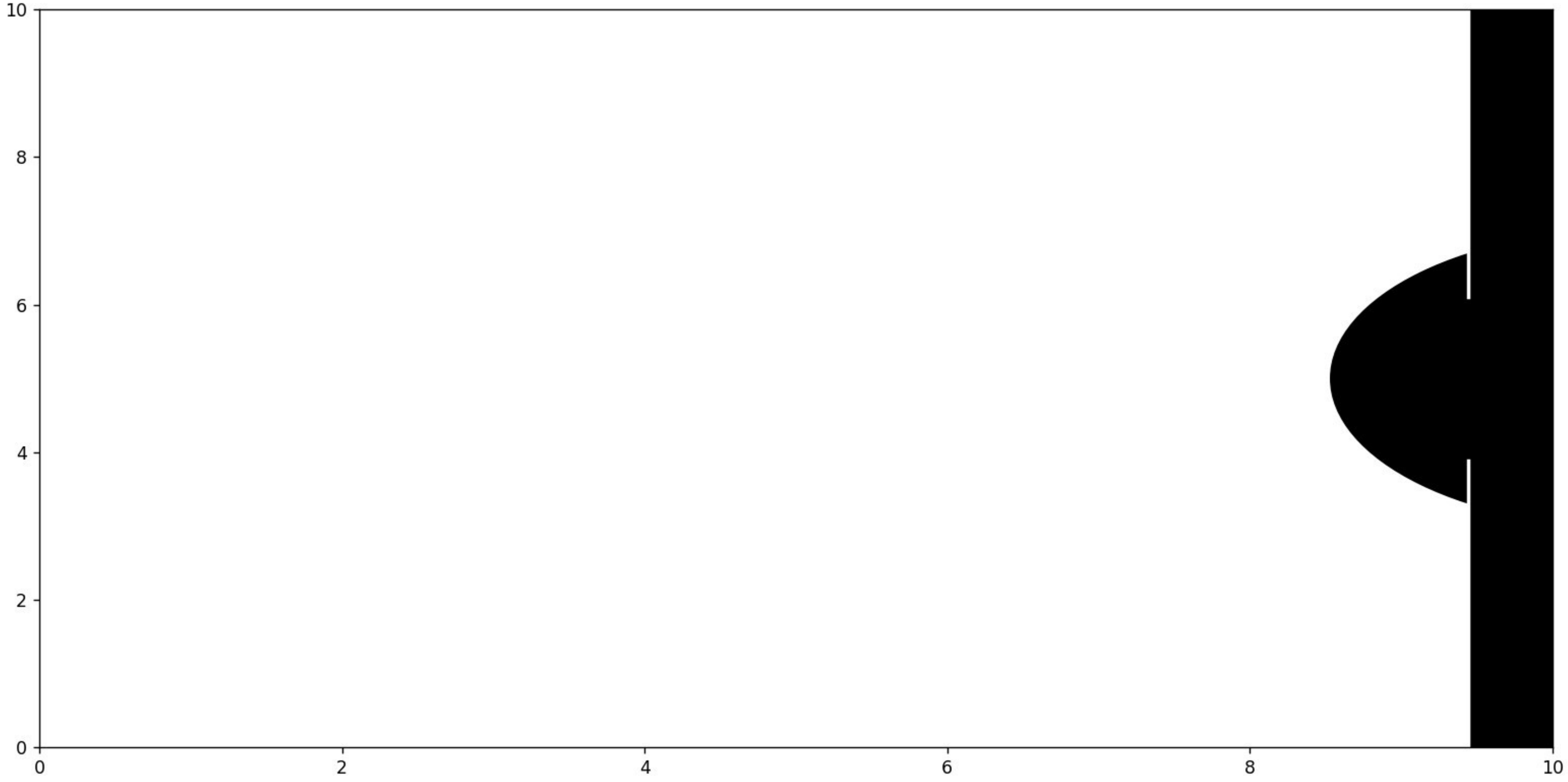
Application au cône convergent



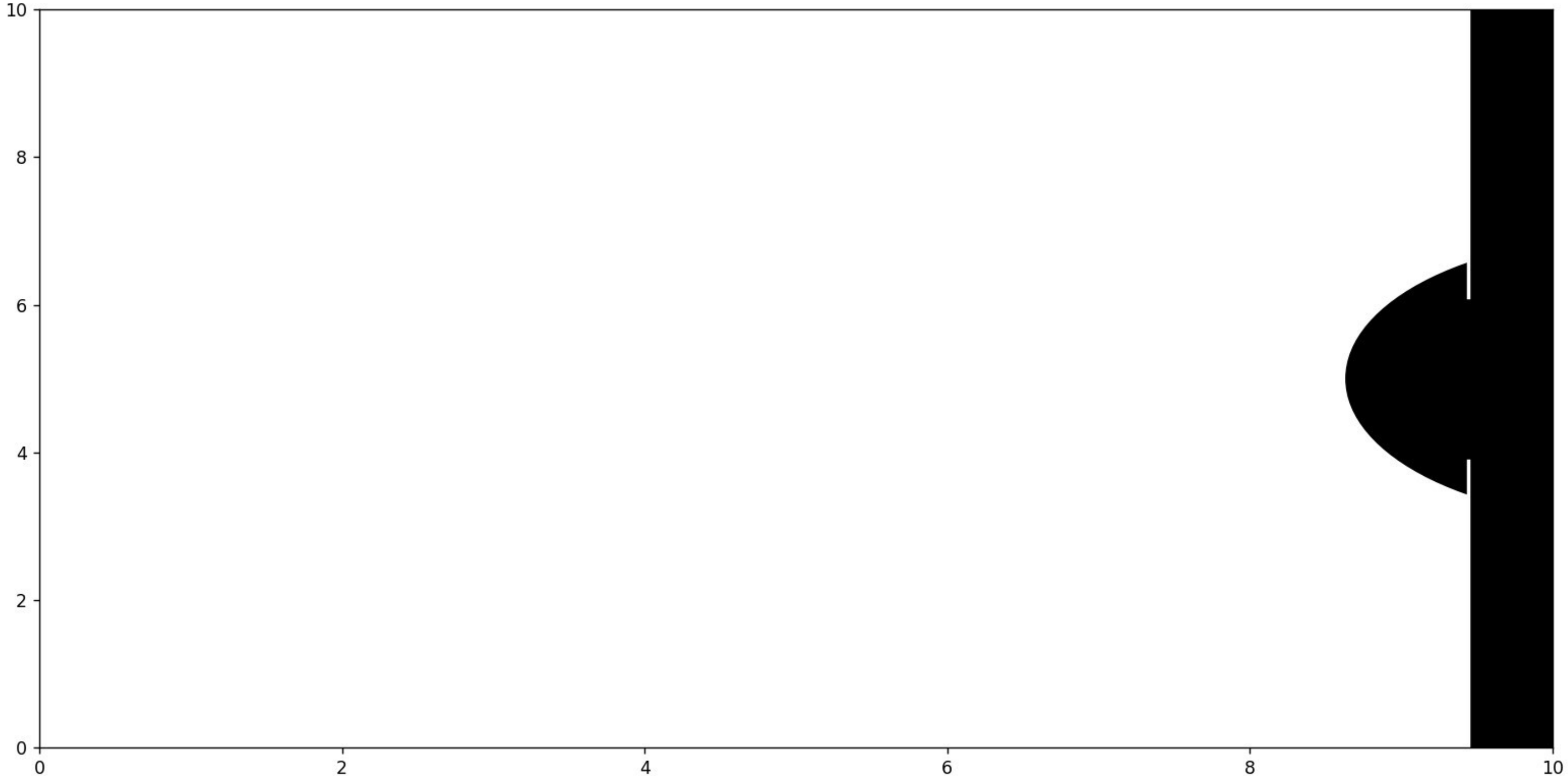
Application au cône convergent



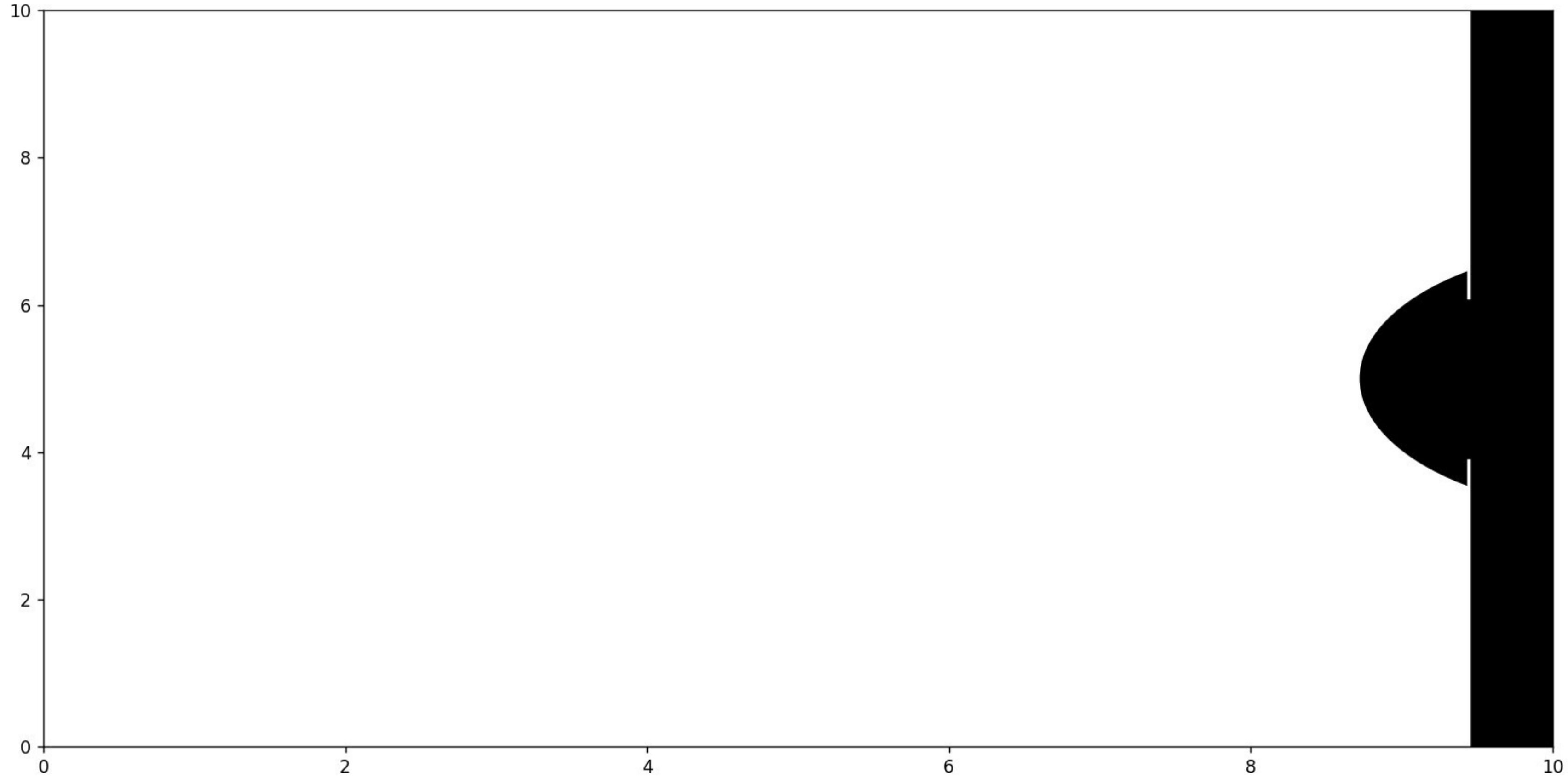
Application au cône convergent



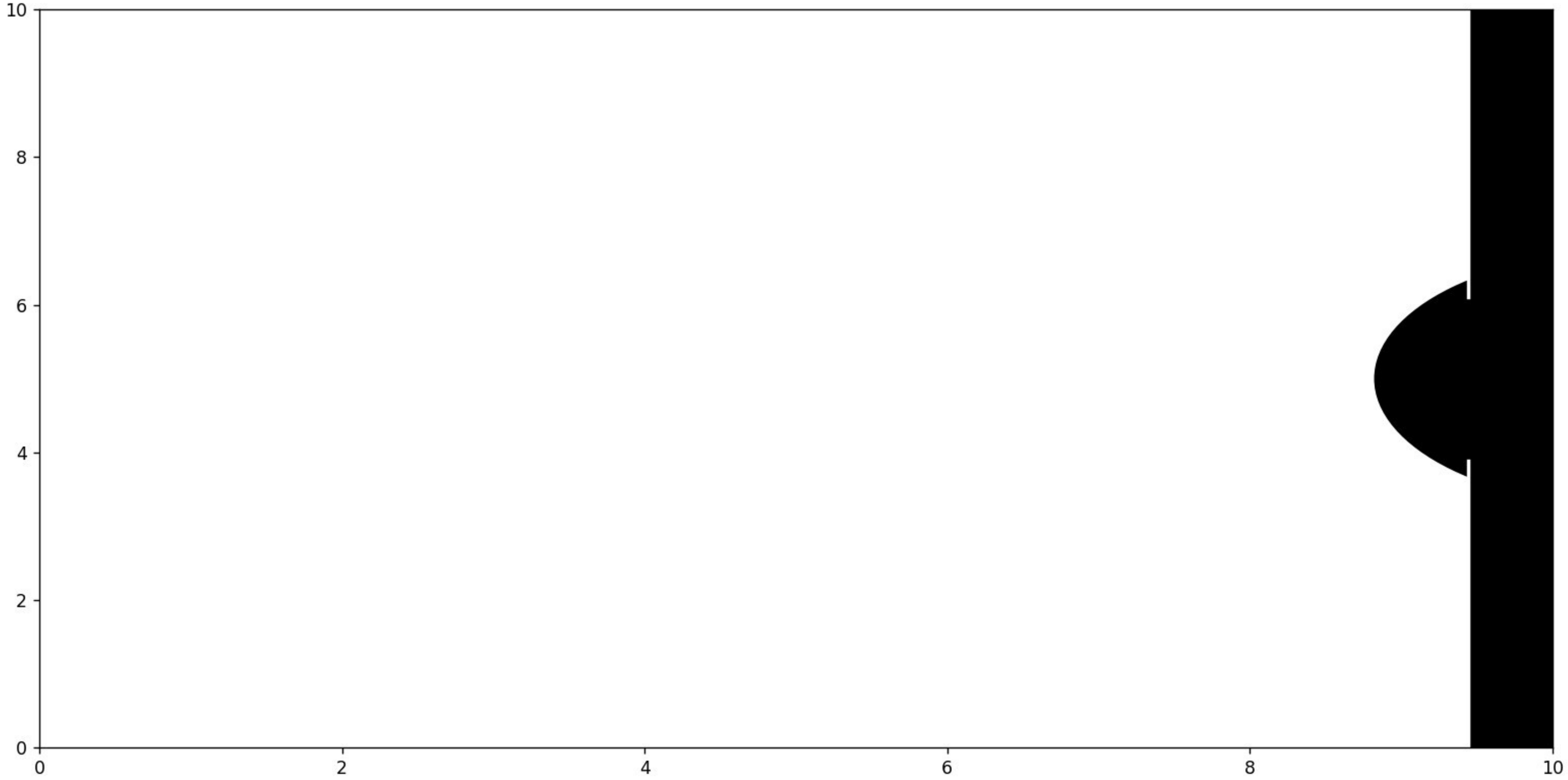
Application au cône convergent



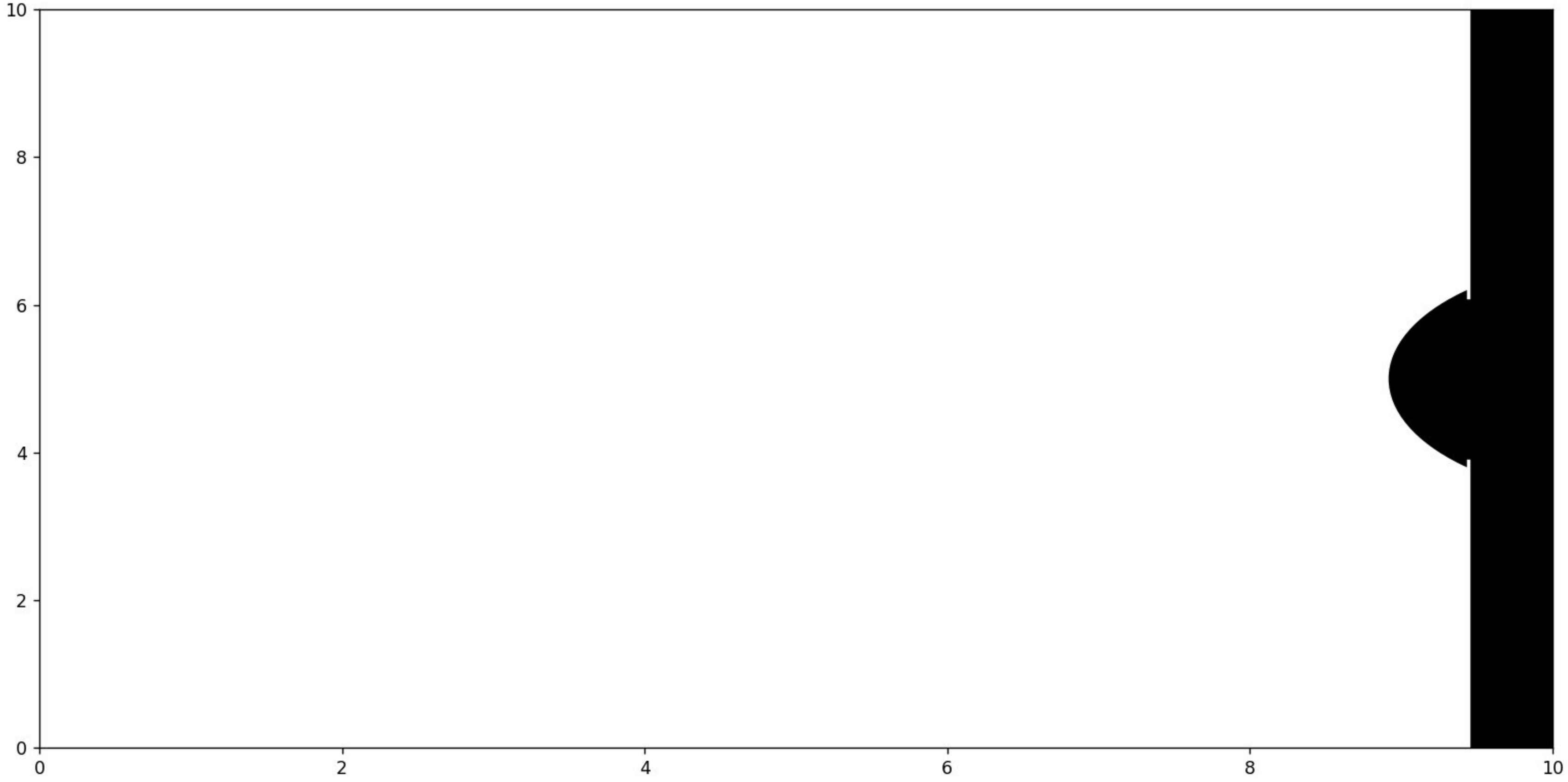
Application au cône convergent



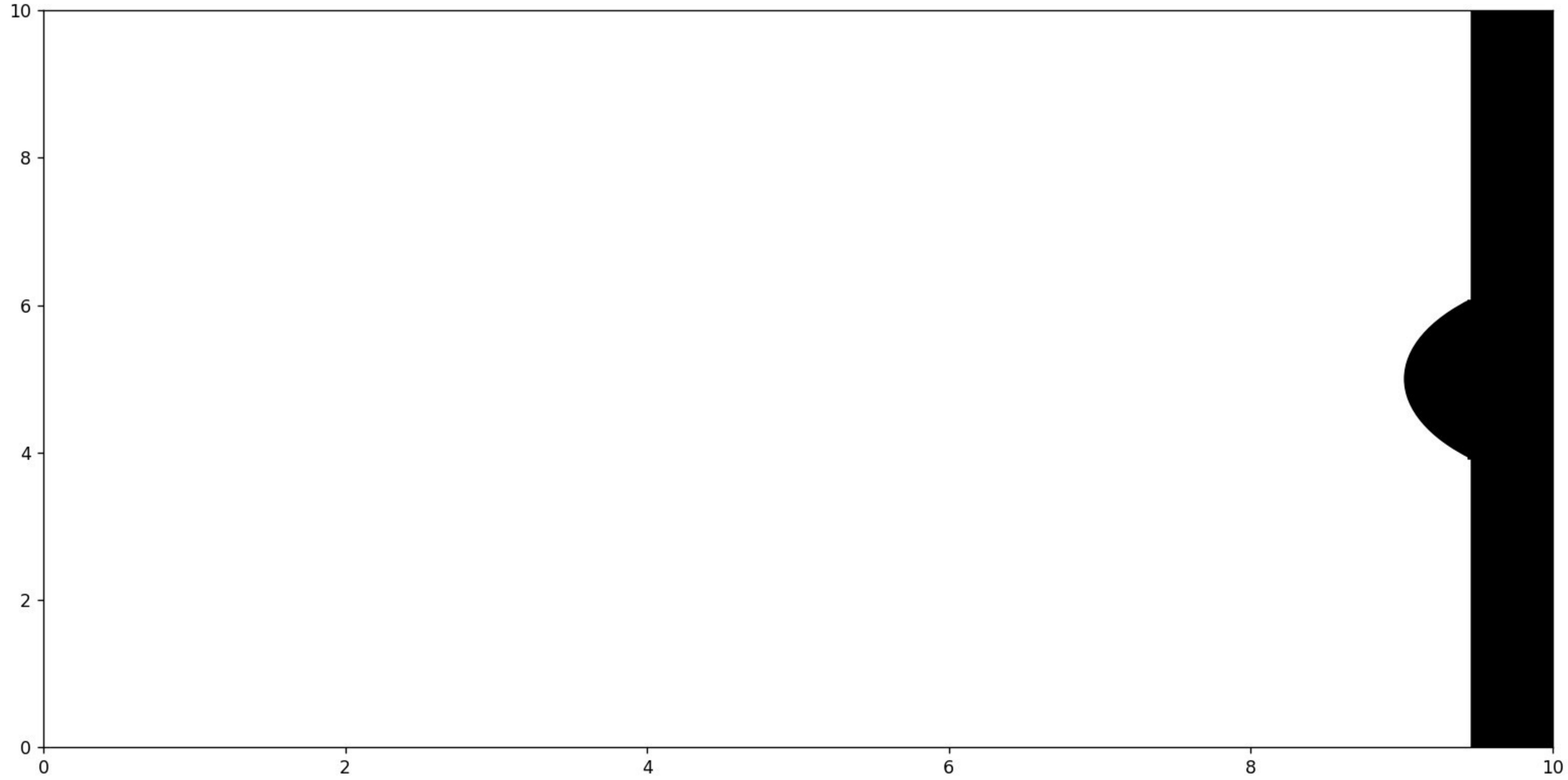
Application au cône convergent



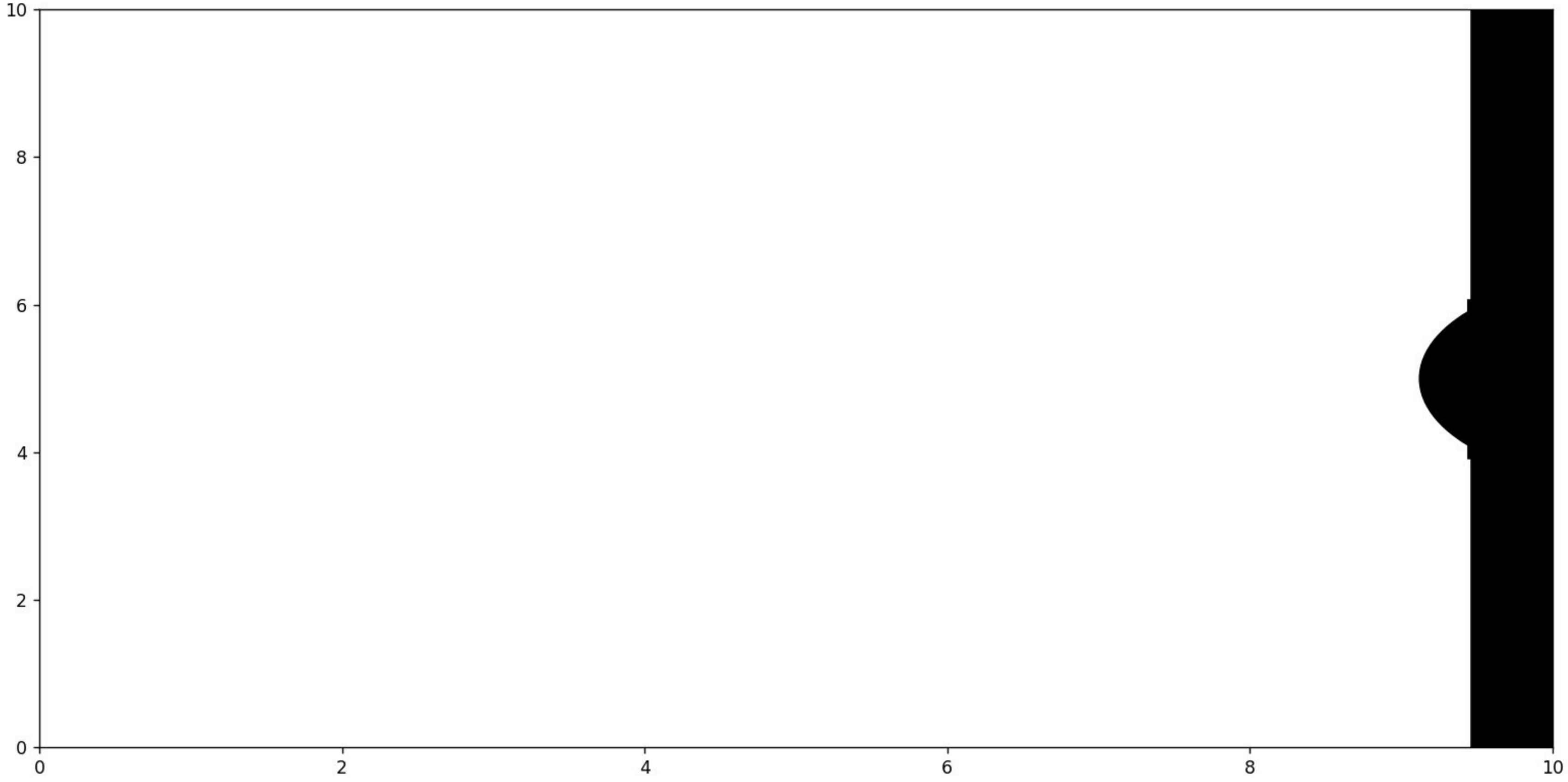
Application au cône convergent



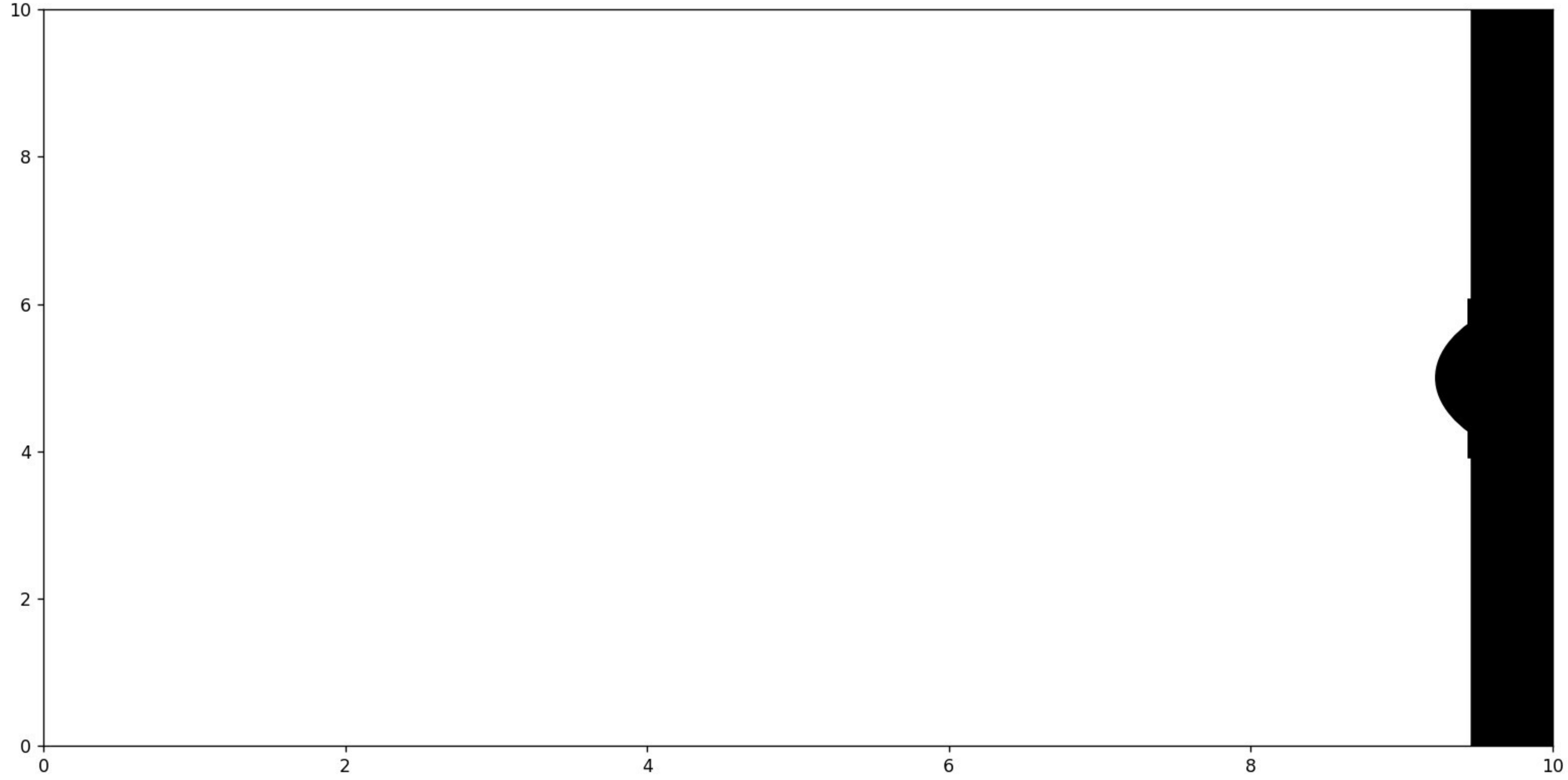
Application au cône convergent



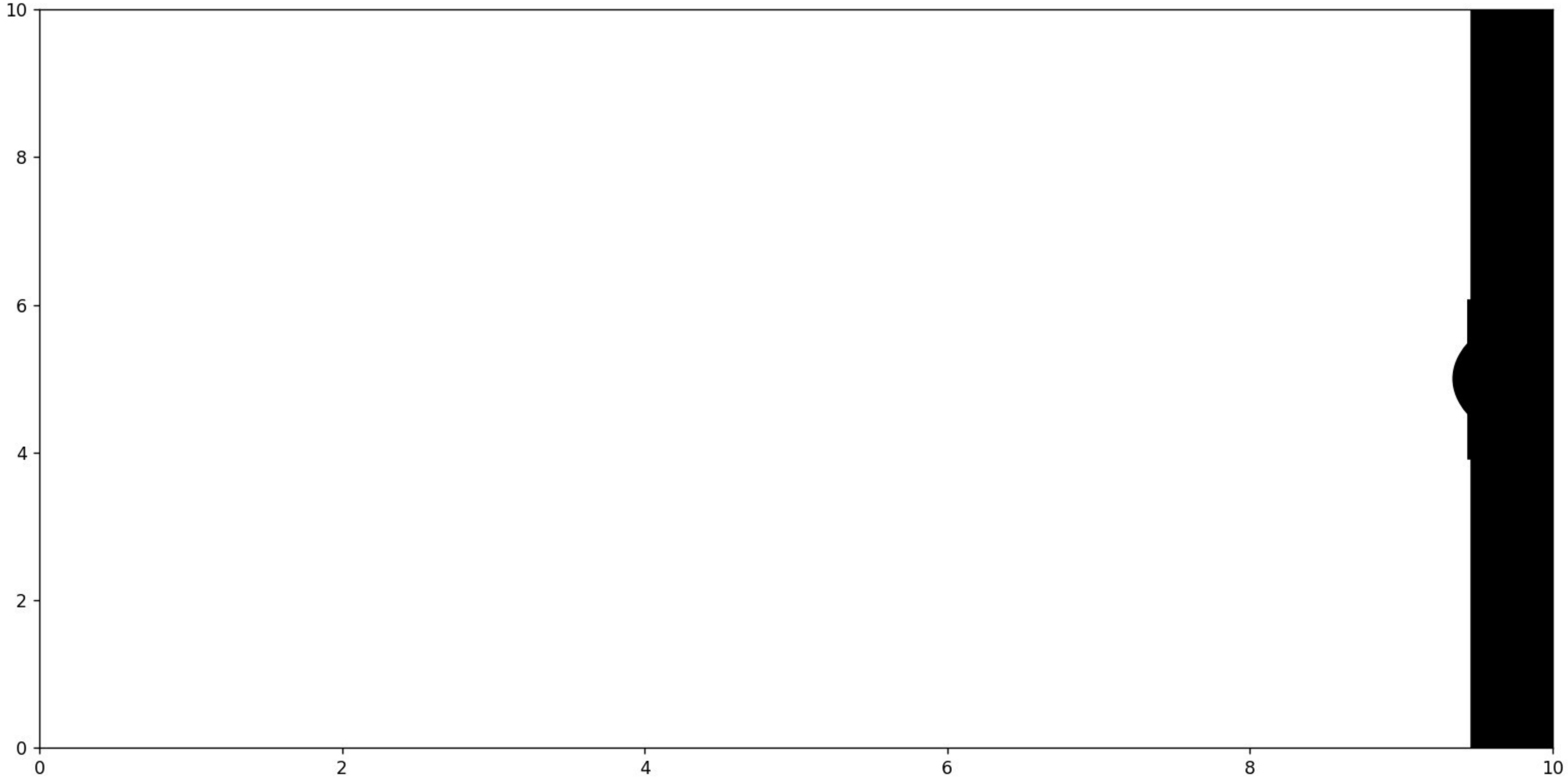
Application au cône convergent



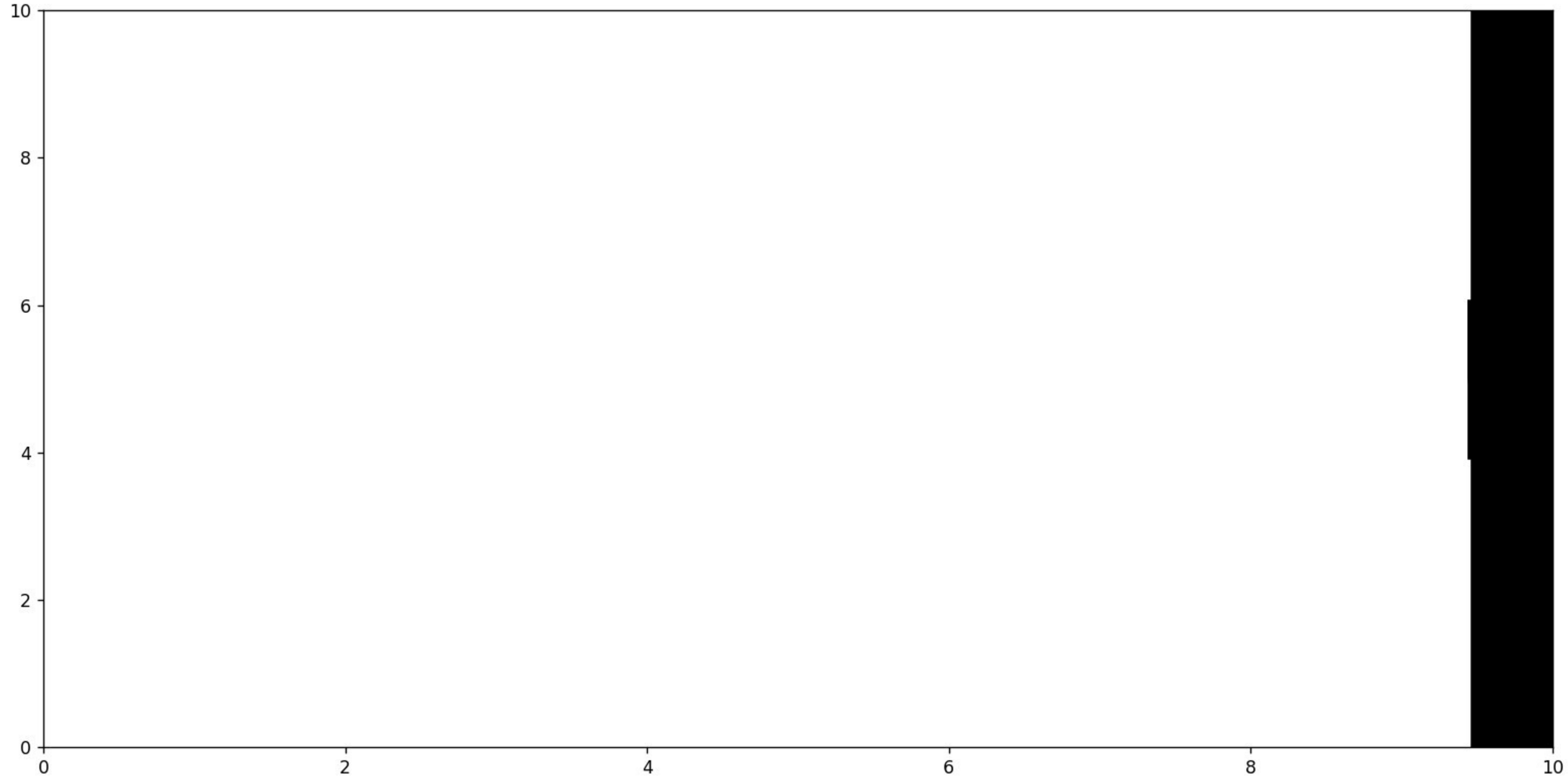
Application au cône convergent



Application au cône convergent



Application au cône convergent



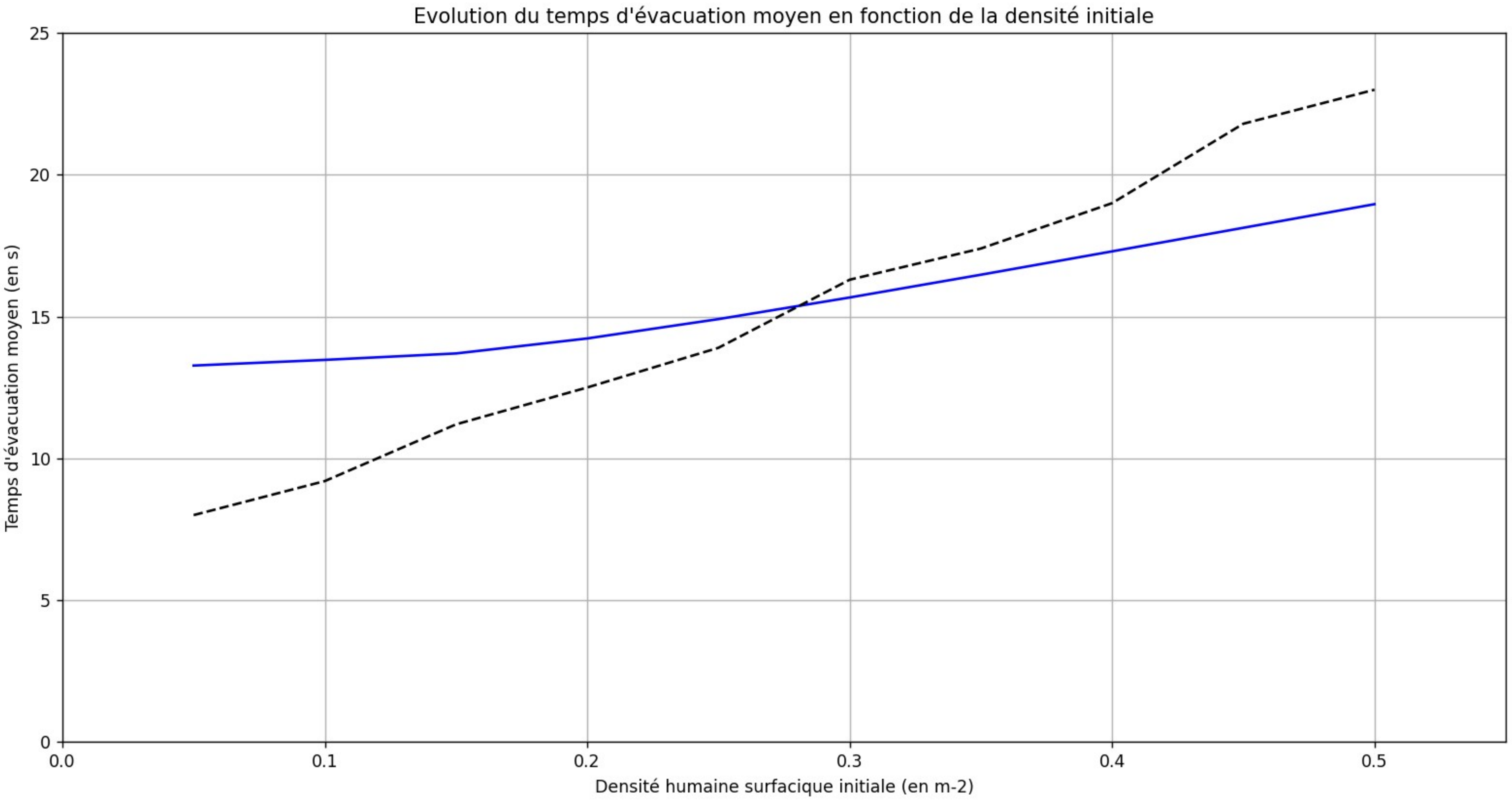
Confrontation aux autres modèles

Le modèle microscopique est une autre façon d'étudier les mouvements de foule.

Nous considérons la foule individu par individu en les assimilants à des disques qui doivent se déplacer sans chevauchement, sachant que chaque disque cherche à atteindre sa vitesse souhaitée.

L'étude est menée en utilisant des méthodes algorithmiques de « catching - up ».

Confrontation aux autres modèles



Confrontation aux autres modèles

Il est possible d'évacuer un parc de $100 \;\; m^2$ en un temps proche de $20 \;\; s$ pour une densité initiale d'individus de $0{,}43 \;\; m^{-2}$ correspondant à une présence initiale de 100 personnes du point de vue microscopique où chaque individu est un disque de rayon $r = 37 \;\; cm$ (largeur moyenne d'épaule).

Confrontation aux autres modèles

Le modèle macroscopique permet d’étudier directement le comportement d’une foule sans se soucier du nombre de personnes qui la constitue.

La comparaison des résultats trouvés avec le modèle microscopique montre une certaine précision du modèle macroscopique à travers l’étude menée.

Même si, certains phénomènes, comme les blocages, ne sont pas visibles car la mouvance et la dynamique de la foule sont simplistes par rapport à la réalité.

Le modèle est en fait plus proche de la réalité pour des évacuations massives dans des grandes zones.

Bibliographie

Roudneff-Chupin A., Modélisation de Mouvements de Foules, (Paris), **2011**

Pierron T., Équations différentielles et phénomènes de transport (Rennes), **2012**

Annexe - Programme 1 (Courbes représentatives)

```
import matplotlib.pyplot as plt
from math import *

#Calcul de la fonction f
def f(t,r,a,R,v,rho_0,rho_sortie,rho_saturée) :
  if r <= R - v*t :
    if r == a :
      return (v*(rho_0*(1 + v*t/a)))/(rho_saturée - rho_0*(1 + v*t/a))
    else :
      return v*(rho_0*(1 + v*t/r) - (rho_sortie*(r-a))/(r*log(r/a)))/(rho_saturée - rho_0*(1 + v*t/r))
  else :
    if r == a :
      return 0
    else :
      return -v*(rho_sortie/rho_saturée)*((r-a)/(r*log(r/a)))

#Application de la méthode d'Euler pour trouver b
def Euler(dt,a,R,v,rho_0,rho_sortie,rho_saturée,n) :
  t0 = (a/v)*((rho_saturée/rho_0) - 1)
  t = 0
  b = a
  T = [t]
  B = [b]
  for i in range(n+1) :
    t += dt
    T.append(t)
    if b < a :
      b = a
    b += f(t,b,a,R,v,rho_0,rho_sortie,rho_saturée)*dt
    if t <= t0 :
      B.append(a)
    else :
      B.append(b)
  return T,B

#Calcul du temps d'évacuation
def temps_evacuation(dt,a,R,v,rho_0,rho_sortie,rho_saturée,n) :
  T,B = Euler(dt,a,R,v,rho_0,rho_sortie,rho_saturée,n)
  t0 = (a/v)*((rho_saturée/rho_0) - 1) #instant à partir duquel la zone saturée se propage
  if B == [a for i in range(len(B))] :
    return (R - a)/v
  else :
    i = 0
    while T[i] < t0 :
      i += 1
    j = i
    while B[j] != a :
      j += 1
    return T[j]

#Calcul de la dérivée de b par rapport au temps
def dervivee_b(t,r,a,R,v,rho_0,rho_sortie,rho_saturée) :
```

Annexe - Programme 1 (Courbes représentatives)

```
    return f(t,r,a,R,v,rho_0,rho_sortie,rho_saturée)

#Calcul des vitesses des phases constituants le mouvement de la foule
def u(r,b,a,v) :
    if  a < r <= b :
        return (v*(b - a))/(r*log(b/a))
    else :
        return v

#Calcul de la dérivée partielle par rapport au temps des vitesses des phases constituants le mouvement de la foule
def dérivee_u_selon_t(t,r,a,R,b,v,rho_0,rho_sortie,rho_saturée) :
    if  a < r <= b :
        return v*(dervivee_b(t,r,a,R,v,rho_0,rho_sortie,rho_saturée)*log(b/a) - (b - a)*(dervivee_b(t,r,a,R,v,rho_0,rho_sortie,rho_saturée)/b))/
(r*(log(b/a)**2))
    else :
        return 0

#Calcul des forces sociales
def F(t,r,a,R,b,v,rho_0,rho_saturée,rho_sortie) :
    return abs(dérivee_u_selon_t(t,r,a,b,R,v,rho_0,rho_sortie,rho_saturée))


#Courbes
T,B = Euler(0.005,1,5,1.67,0.43,1,1,4000)
plt.plot(T,B, label ="y = b(t)")
plt.title("Evolution de l'interface de densité maximale b")
plt.xlabel("Temps t (en s)")
plt.ylabel("Interface de densité maximale b (en m)")
plt.grid()
plt.legend()
plt.show()

X = [0.1*i for i in range(1,6)]
Y = [temps_evacuation(0.005,1,5,1.67,rho_0,1,1,4000) for rho_0 in X]
plt.plot(X,Y, label = "y = f(rho_0)")
plt.title("Evolution du temps d'évacuation en fonction de la densité initiale rho_0")
plt.xlabel("Densité initiale rho_0 (en m-2)")
plt.ylabel("Temps d'évacuation T (en s)")
plt.grid()
plt.legend()
plt.show()
plt.show()

T,B = Euler(0.005,1,5,1.67,0.43,1,1,4000)
Y = [F(T[i],B[i],1,5,B[i],1.67,0.43,1,1) for i in range(len(T))]
plt.plot(T,Y,label = "y = F(t)")
plt.xlabel("Temps t (en s)")
plt.ylabel("Forces sociales F s'exerçant sur la foule (en N)")
plt.title("Evolution des forces sociales F s'exerçant sur la foule à l'interface de densité maximale b")
plt.grid()
plt.legend()
plt.show()
```

Annexe - Programme 2 (Animation)

```python
from math import *
import matplotlib.pyplot as plt
import numpy as np
from matplotlib.patches import Rectangle

#Fonction f
def f(t,r):
    if r <= R - v*t :
        if r == a :
            return (v*(rho_0*(1 + v*t/a)))/(rho_saturée - rho_0*(1 + v*t/a))
        else :
            return v*(rho_0*(1 + v*t/r) - (rho_sortie*(r-a))/(r*log(r/a)))/(rho_saturée - rho_0*(1 + v*t/r))
    else :
        if r == a :
            return 0
        else :
            return -v*(rho_sortie/rho_saturée)*((r-a)/(r*log(r/a)))

#Calcul du temps d'évacuation de la foule
def temps_evacuation(dt,n) :
  T,B = Euler(dt,n)
  t0 = (a/v)*((rho_saturée/rho_0) - 1) #instant à partir duquel la zone saturée apparaît
  i = 0
  while T[i] < t0 :
    i += 1
  j = i
  while B[j] != a :
    j += 1
  return T[j]

#Calcul l'instant où la phase saturée et non saturée se croisent
def instant_chevauchement(dt,n) :
  T,B = Euler(dt,n)
  i = 0
  while R - v*T[i] > B[i] : #condition de vérification du croisement
    i += 1
  return T[i]

#Méthode d'Euler
def Euler(dt,n) :
  t0 = (a/v)*((rho_saturée/rho_0) - 1)
  t = 0
  b = a
  T = [t]
  B = [b]
  for i in range(n+1) :
    t += dt
    T.append(t)
    if b < a :
      b = a
    b += f(t,b)*dt
    if t <= t0 :
```

Annexe - Programme 2 (Animation)

```
        B.append(a)
      else :
        B.append(b)
    return T,B

#Conditions initiales
L = 10 #taille de la pièce
r = 0.37 #rayon des individus (pour le modèle microscopique)
murs_x = L-1.5*r # emplacement des murs
sortie_x, sortie_y = murs_x , L/2
sortie_largeur = 3*2*r
y_point_de_conv_cone = sortie_y
x_point_de_conv_cone = sortie_x
R = L/2
a = sortie_largeur/2
b = a
rayon_cercle_gros = R
rayon_cercle_petit = b
rho_0 = 0.43 #
rho_saturée = 1
rho_sortie = 1
v = 1.67
n_iter = 6500

#Temporalité
t0 = (a/v)*((rho_saturée/rho_0) - 1)
t = 0
dt = 0.005


#Initialisation de la simulation
fig, ax = plt.subplots()
ax.set_xlim(0, L)
ax.set_ylim(0, L)
cercle_gros = plt.Circle((sortie_x + a, sortie_y), R, color='c')
cercle_petit = plt.Circle((sortie_x + a, sortie_y), b, color='k')
plt.plot([murs_x, murs_x], [sortie_y + sortie_largeur/2, L], 'w-', linewidth=2)
plt.plot([murs_x, murs_x], [0, sortie_y - sortie_largeur/2], 'w-', linewidth=2)
ax.add_artist(cercle_gros)
ax.add_artist(cercle_petit)

#Affichage des rectangles
rect1 = Rectangle((sortie_x, sortie_y),10,10,color='k')
rect2 = Rectangle((sortie_x, 0),10,10,color='k')
ax.add_patch(rect1)
ax.add_patch(rect2)

#Exécution de la simulation
TD = instant_chevauchement(dt,n_iter)  #instant TD
T,B = Euler(dt, n_iter)
j=0
```

Annexe - Programme 2 (Animation)

```python
while t <= temps_evacuation(dt,n_iter) :
    j+=1
    t += dt
    b = B[j]


    #Met à jour la couleur du cercle
    if t < TD:
        normalized_time = (t - dt) / TD
    else:
        normalized_time = 1.0
    if t < TD:
        circle_color = (0, 0, 1 - normalized_time)   #Bleu clair à noir
    else:
        circle_color = 'k'  #Noir
    cercle_gros.set_color(circle_color)

    cercle_gros.set_radius(R-v*t)
    cercle_petit.set_radius(b)

    plt.pause(0.01)

plt.show()
```