# PROPAGATION DES FEUX DE FORET

- **PLAN:**
- 1-INTRODUCTION
- 2-EXPÉRIENCE
- 3-MODÈLISATION
- 3.1-MODÈLE DÉTERMINISTE
- 3.2-MODÈLE PROBABILISTE
- 4-RESULTATS(CONCLUSION)

# 1-INTRODUCTION

Les feux de forêt sont causés majoritairement par l'humain et ont des conséquences :

- Destruction des habitats humaine et animale
- Pollution de l'air

le but c'est de mieux comprendre la propagation d'un feu de forêt pour le combattre et protéger l'environnement

# 2-EXPERIENCE

Outils pour la modélisation d'un de forêt:
Plaque de fer carré 25 cm
Terre + feuilles d'arbres
Ventilateur

À t=0

À t=60s

À t=100s

Cette expérience nous montre  que la propagation du feu se fait de manière circulaire dans forêt

# 3-MODÉLISATION

POUR MODÉLISER LA PROPAGATION D'UN DE DE FORET ON VA UTILISER DEUX MODÈLES:

A-MODÈLE DÉTERMINISTE QUI SE BASE SUR L'ÉQUATION DE DIFFUSION DE LA CHALEUR

B-MODÈLE STOCHASTIQUE QUI SE BASE SUR LES PROBABILITÉS(PERCOLATION)

# 3.1-MODÈLE DETERMINISTE

A-équation de la chaleur

Équation de la diffusion :

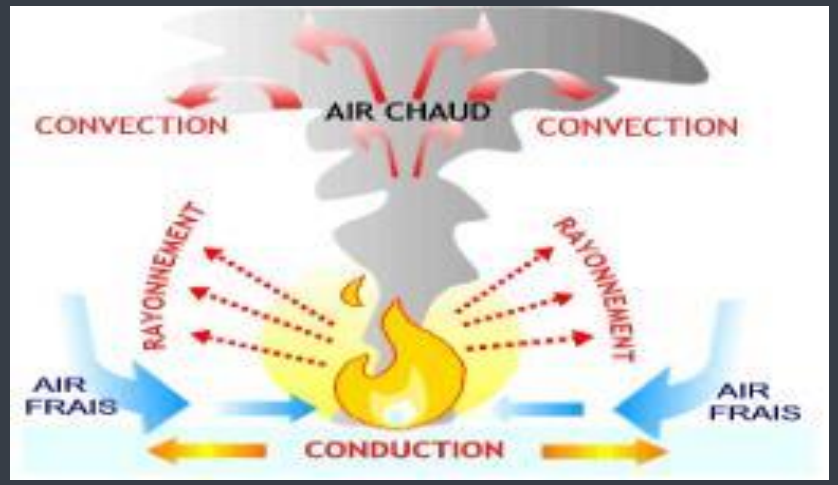
$$\rho C v \frac{\partial T}{\partial t} = -\lambda \Delta T + \sigma$$

Les transferts thermiques par conduction s'effectuent de façon irréversible et a tendance à uniformiser la température est régi par la loi:

$$\vec{j}(M,t) = -\lambda \vec{\nabla} \vec{T}(M,t)$$

La source de chaleur (le feu) le feu va transformer son énergie sous forme de lumière et cette lumière par rayonnement va être transmise dans les zones proches.

La convection est caractérisé par le déplacement du gaz chaud ascendant.



Dans une forêt de dimension 250m en longueur et 250m en largeur la propagation se fait des manière déjà citez  et dans la forêt on suppose que les arbres sont disposés de manière homogène et plane

Dans l'équation de la chaleur les termes utilisés sont:

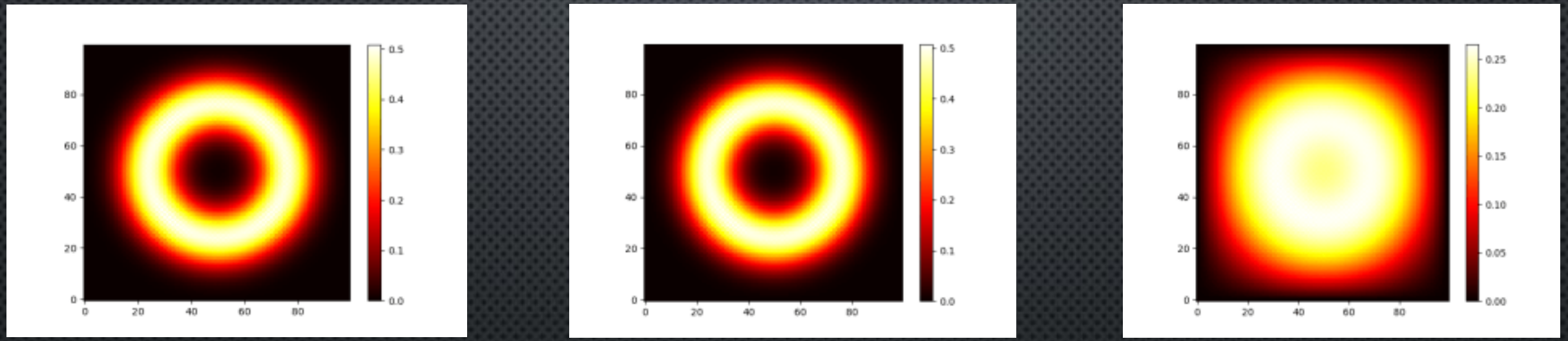
Masse volumique(toutes les arbres ont la même masse volumique dans l'hypothèse citer)

Coefficient de diffusion du bois: 0.15 **W m$^{-1}$ K$^{-1}$ à 20 °C**

Taux de production d'énergie chimique due à la combustion de l'arbre par unité de temps et de volume

## B-résolution de l'équation de chaleur:

La résolution de l'équation de chaleur peut se faire analytiquement mais en connaissant les conditions limites de la propagation




**Graphe 1**

La résolution numérique de l'équation de chaleur  donner les schémas suivant, ces schémas vérifient le résultats de l'expérience 2 que la propagation de du feu est circulaire

# 3.2-MODÈLE PROBABILISTE

Le modèle probabiliste se base sur les probabilités (Percolation)  est pour ce modèle on doit citer quelques hypothèses :

**La structure de la forê**t :

la structure de la forêt est homogène

→ Les arbres sont espacés de manière régulière

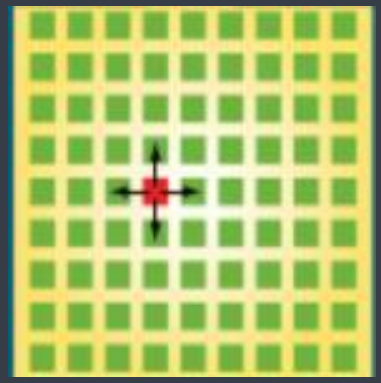
→ Les arbres sont de même nature

→ Les arbres sont de même taille (modèle à 2D)

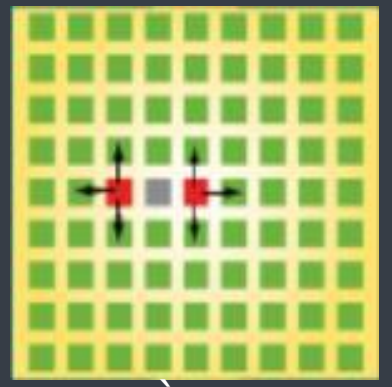
**La structure du feu** :Le feu a la même probabilité de se propager dans chacune des directions  Si la combustion a lieu, elle est complète.

**La discrétisation du temps** : La combustion d'un arbre n'est pas continue mais instantanée

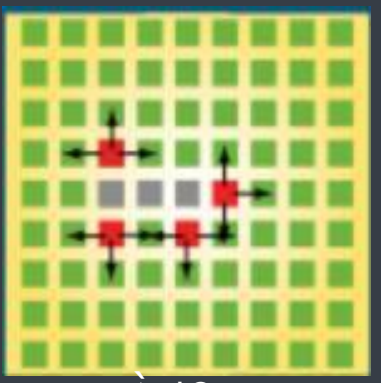
1er modèle



À t1

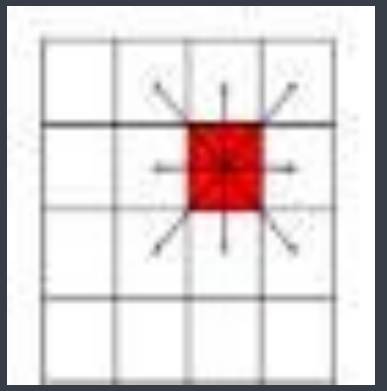


À t2



À t3

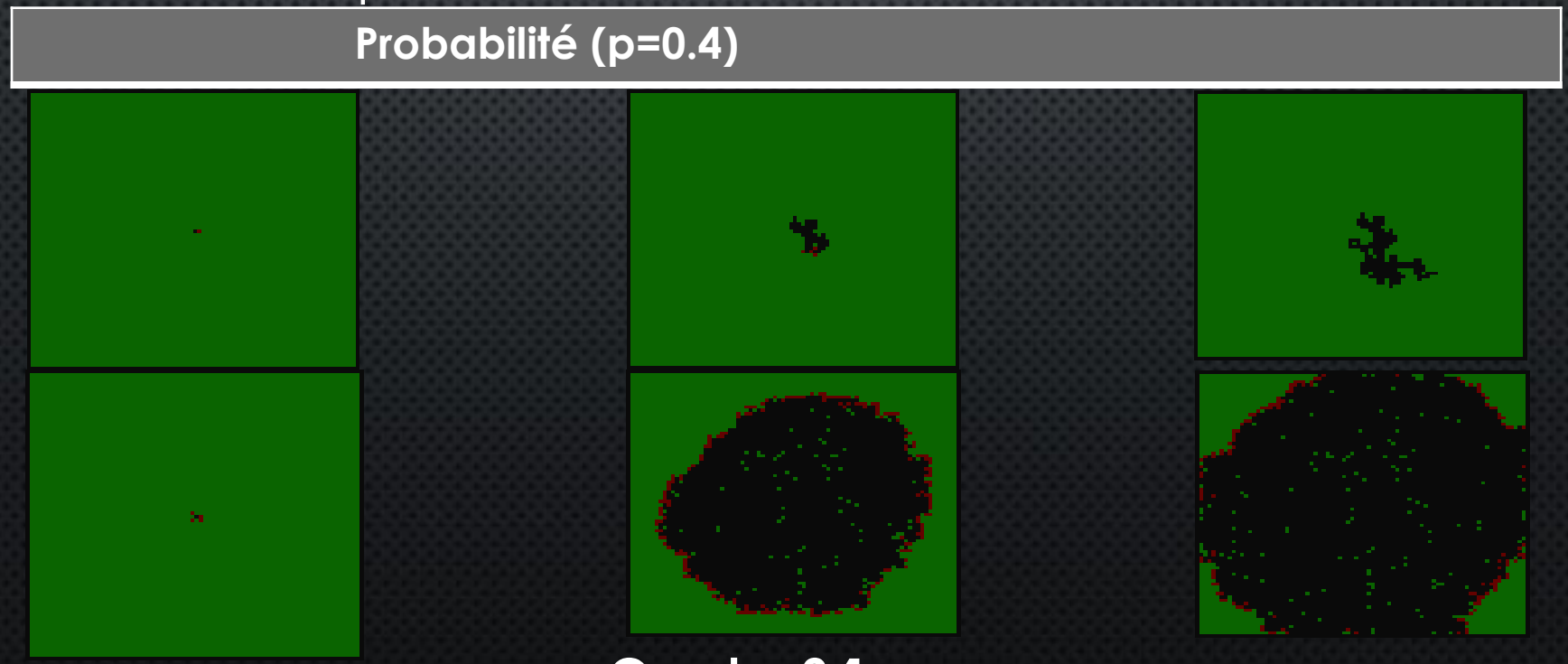
2eme modèle



Le 1er modèle utilise la méthode de proches voisins c'est-à-dire qu'une fois un arbre prend feu , le feu se propage vers  les 4 directions plus proche de l'arbre.

Le 2eme modèle utilise la méthode super diffusion c'est-à-dire  que le feu se propage vers  les directions plus proche de l'arbre.

Dans la théorie de percolation l'existence d'un seuil est fondémentale  mais en dimension 2 ce seuil et toujours égale à 0.5  donc pour une probabilité inférieur à cet valeur le feu reste localisé et pour une valeur  supérieur le feu se propage dans la foret pour le 1er modèle.

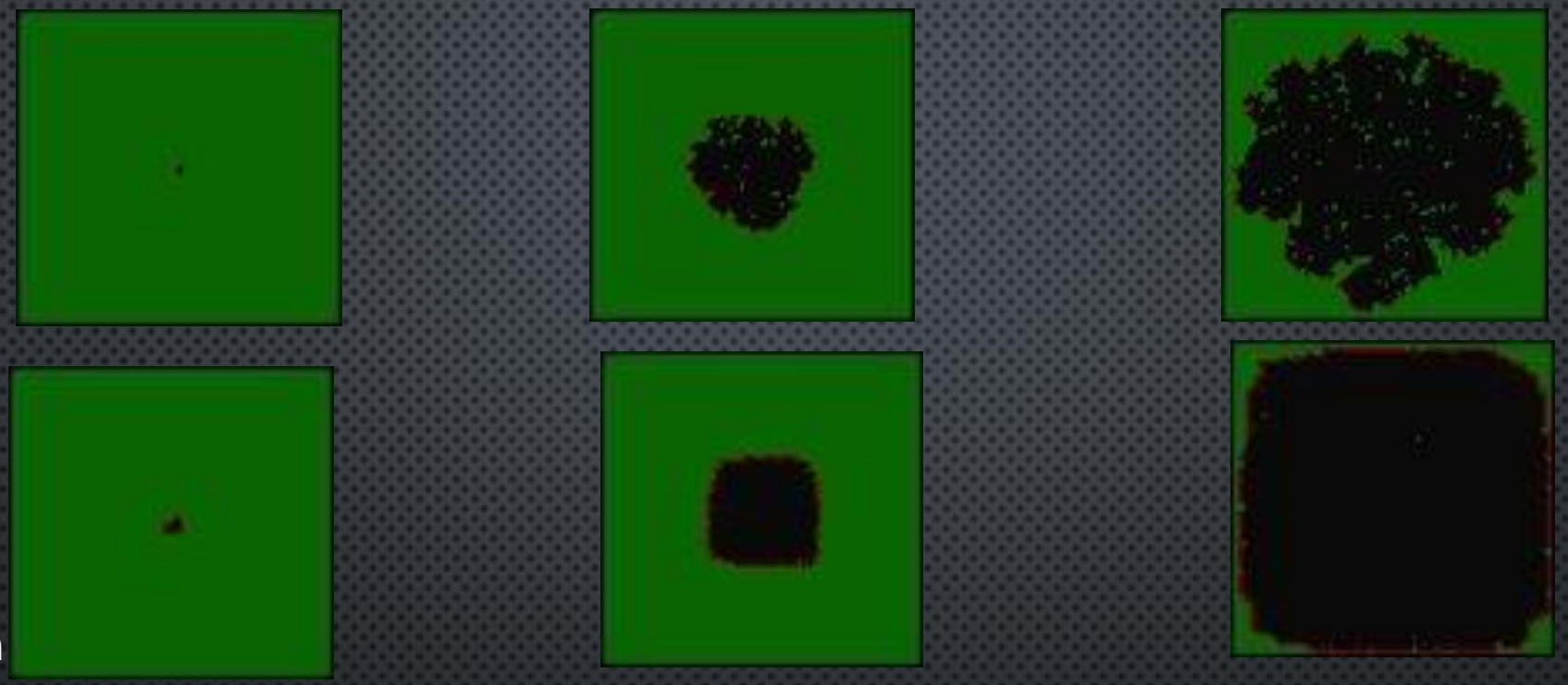



Graphe 2.1

**Probabilité(p=0.6)**

Modèle de proches voisins

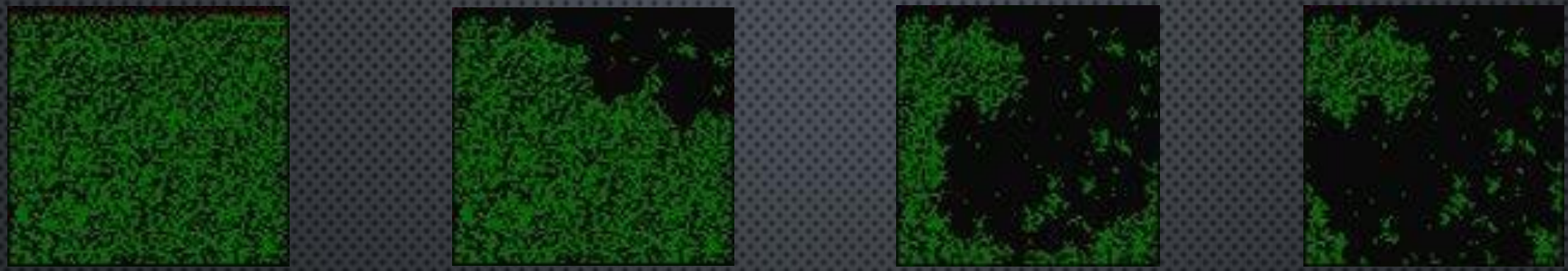
Modèle de super diffusion



**Graphe 2.2**

Dans le modèle proches voisins le résultats que seuil de percolation est bien verifier et que si le seuil augmente plus que la probabilité que  le feu reste localisé dans une petite région augmente et la simulation numérique

Le graphe précédent est fait dans une forêt de très grande densité d'arbre, voilà un autre graphe dans une forêt de très faible densité :



**Graphe 3**

# 4-RESULTATS(CONCLUSION)

L'expérience  2 nous a montrer que le feu se propage circulairement dans une forêt, La simulation  numérique d'un feu de forêt avec la percolation montre que le feu aussi se propage circulairement  comme dans les graphe (2.1 et 2.2). Le graphe 1 aussi de la résolution de l'équation de la chaleur aussi montre la propagation circulaire de la chaleur

La propagation en conclusion de passe circulairement en absence du vent

Chaque modélisation présente de avantages et des inconvénients :
- Le modèle déterministe à des avantages au fait qu'il tient compte de presque tous les paramètres qui influencent le comportement du feu
 et son inconvénient est qu'il ne tient pas compte de la probabilité que l'arbre peut se bruler

-Les avantages du modèle probabiliste se résume au fait qu'il tient compte de la probabilité que l'arbre peut se bruler et son inconvénient et qu'il se base sur beaucoup de approximation notamment sur la structure de la forêt  et la structure du feu

La propagation du feu ne peut se passer que si la percolation de la forêt est supérieur  à une certaine valeur critique appelé seuil de percolation et cette valeur dépend du:

Mode de propagation du feu :

- Modèle de proche voisins
- Modèle de super diffusion

De la structure de la forêt:

- densité des arbres
-dimension du réseau

Donc l'augmentation du seuil limite la propagation du feu dans forêt

# Merci pour votre attention

# ANNEXE1(GRAPHE1)

```
import numpy as np
import matplotlib
from pylab import *
dx=0.01
dy=0.01
a=1.66 # Diffusion constant.
timesteps=700
nx = int(1/dx)
ny = int(1/dy)

dx2=dx**2 # To save CPU cycles, we'll

dy2=dy**2 # and Delta y^2 only once and


# For stability, this is the largest

# for the size of the time-step:
dt = dx2*dy2/( 2*a*(dx2+dy2) )
ui = np.zeros([nx,ny])
u = np.zeros([nx,ny])
for i in range(nx):
    for j in range(ny):
        if ( ( (i*dx-0.5)**2+(j*dy-0.5)**2 <=0.1) and ((i*dx-0.5)**2+(j*dy-0.5)**2>=.05) ):
            ui[i,j] = 1

def evolve_ts(u, ui):
    global nx, ny
    for i in range(1,nx-1):
        for j in range(1,ny-1):
            uxx = ( ui[i+1,j] - 2*ui[i,j] +ui[i-1, j] )/dx2
            uyy = ( ui[i,j+1] - 2*ui[i,j] +ui[i, j-1] )/dy2
            u[i,j] = ui[i,j]+dt*a*(uxx+uyy)


    ui = u.copy()
    return(ui,u)
m=100
p=0
for i in range(timesteps):
    if i <= m:
        ui, u = evolve_ts(u, ui)
        p=+1
fig = plt.figure(1)
img = plt.plot(111)
im = plt.imshow( ui, cmap=cm.hot,interpolation='nearest', origin='lower')
```

```
# For stability, this is the largest

# for the size of the time-step:
dt = dx2*dy2/( 2*a*(dx2+dy2) )
ui = np.zeros([nx,ny])
u = np.zeros([nx,ny])
for i in range(nx):
    for j in range(ny):
        if ( ( (i*dx-0.5)**2+(j*dy-0.5)**2 <=0.1) and ((i*dx-0.5)**2+(j*dy-0.5)**2>=.05) ):
            ui[i,j] = 1

def evolve_ts(u, ui):
    global nx, ny
    for i in range(1,nx-1):
        for j in range(1,ny-1):
            uxx = ( ui[i+1,j] - 2*ui[i,j] +ui[i-1, j] )/dx2
            uyy = ( ui[i,j+1] - 2*ui[i,j] +ui[i, j-1] )/dy2
            u[i,j] = ui[i,j]+dt*a*(uxx+uyy)


    ui = u.copy()
    return(ui,u)
m=100
p=0
for i in range(timesteps):
    if i <= m:
        ui, u = evolve_ts(u, ui)
        p=+1
fig = plt.figure(1)
img = plt.plot(111)
im = plt.imshow( ui, cmap=cm.hot,interpolation='nearest', origin='lower')
manager = get_current_fig_manager()
fig.colorbar( im ) # Show the colorb

plt.show()
```

# ANNEXE2(GRAPHE2.1 GRAPHE2.2 GRAPHE3)

```python
from numpy import *
from random import *
from PIL import Image


def foret_avant_incendie(n):
    "fonction permettant de créer la foret de taille n"
    A=zeros((n,n),int) #matrice remplie de 0 pour representer chaque arbre
    return A

def conditions_aux_limites(A,n):
    "fonction permettant de considerer le bord comme brule : conditions aux limites"
    ligne=2+zeros((1,n),int) #on ajoute des arbres brules sur la face ouest et est
    A=concatenate((A,ligne))
    A=concatenate((ligne,A))
    colonne=2+zeros((n+2,1),int) #on ajoute des arbres brules sur la face nord et sud
    A=concatenate((A,colonne),1)
    A=concatenate((colonne,A),1)
    return A

def deboiser(A,n,p):
    "Déboise au hazard avec une probabilité de p pour chaque zone"
    for i in range (1,n+1):
        for j in range (1,n+1):
            if (random()<p):
                A[i,j]=2
    return A

def enflamer(A,n):
    "Allume une zone au hazard"
    x=randint(1,n)
    y=randint(1,n)
    R=[]
    R.append([x,y])
    A[x,y]=1
    return A,R

def enflamer_centre (A,n):
    "Allume la zone au centre"
    A[int(n/2),int(n/2)]=1
    R=[]
    R.append([int(n/2),int(n/2)])
    return A,R


def enflamer_front(A,n):
```

```
Python
Type 'help' for help, type '?' for a list of *mag
Running script: "C:\Users\bon\azazaazazzaaz.py"

>>> Film_front(80,0.6)
étape : 0  avec  31  zones en feu
étape : 1  avec  17  zones en feu
étape : 2  avec  12  zones en feu
étape : 3  avec  13  zones en feu
étape : 4  avec  9  zones en feu
étape : 5  avec  6  zones en feu
étape : 6  avec  1  zones en feu
étape : 7  avec  0  zones en feu
array([[2, 2, 2, ..., 2, 2, 2],
       [2, 2, 2, ..., 2, 2, 2],
       [2, 2, 2, ..., 2, 2, 2],
       ...,
       [2, 2, 2, ..., 2, 0, 2],
       [2, 0, 2, ..., 0, 2, 2],
       [2, 2, 2, ..., 2, 2, 2]])

>>>
```

Structure du pro... | Explorateur de fichiers

foret_avant_in...
conditions_au...
deboiser()
enflamer()
enflamer_cent...
enflamer_front()
une_etape()
une_etape_car...
compte_zone()
compte_fronti...
feu_sur_fronti...
obtention_ima...
nom_image()
Nom_de_fichi...
ZEF()
Film()
Film_front()

Cliquez sur l'étoile pour ajouter le ré

C:\Users\bon

Figure77474.png
Figure_3.png
Figure_5.png
Foret00000.png
Foret00001.png
Foret00002.png
Foret00003.png
Foret00004.png
Foret00005.png
Foret00006.png
Foret00007.png

!*.pyc



Taper ici pour rechercher

```python
def enflamer_front(A,n):
    "Allume un front de feu au nord"
    R=[]
    for i in range (1,n+1):
        A[1,i]=1
        R.append([1,i])
    return A,R

def une_etape(A,R,n,p):
    B=copy(A)
    S=[]
    for feu in R:
        i,j=feu[0],feu[1]
        for k in [-1,1]:
            if ((random()<p) and B[i+k, j]==0):
                B[i+k,j]=1
                S.append([i+k,j])
            if ((random()<p) and B[i,j+k]==0):
                B[i,j+k]=1
                S.append([i, j+k])
        B[i,j]=2
    A=copy(B)
    return A,S

def une_etape_carre_8(A,R,n,p):
    "calcul pour une étape en sortie avec une proba p d'inflamation 8 directions"
    B=copy(A)
    S=[]
    for feu in R:
        i,j=feu[0],feu[1]
        for k in range(-1,2):
            for l in range(-1,2):
                if((k!=0 or l!=0)):
                    if((random()<p) and B[i+k,j+l]==0):
                        B[i+k,j+l]=1
                        S.append([i+k,j+l])
        B[i,j]=2
    A=copy(B)
    return A,S

def compte_zone(A,n,k):
    "Compte le nombre de zone de A qui contient la valeur k"
    conteur=0
    for i in range (1,n+1):
        for j in range (1,n+1):
```

```
Type 'help' for help, type '?' for a list of *magic* commands.
Running script: "C:\Users\bon\azazaazazzaaz.py"

>>> Film_front(80,0.6)
étape : 0  avec  31  zones en feu
étape : 1  avec  17  zones en feu
étape : 2  avec  12  zones en feu
étape : 3  avec  13  zones en feu
étape : 4  avec  9  zones en feu
étape : 5  avec  6  zones en feu
étape : 6  avec  1  zones en feu
étape : 7  avec  0  zones en feu
array([[2, 2, 2, ..., 2, 2, 2],
       [2, 2, 2, ..., 2, 2, 2],
       [2, 2, 2, ..., 2, 2, 2],
       ...,
       [2, 2, 2, ..., 2, 0, 2],
       [2, 0, 2, ..., 0, 2, 2],
       [2, 2, 2, ..., 2, 2, 2]])

>>>
```

Structure du pro...

- foret_avant_in...
- conditions_au...
- deboiser()
- enflamer()
- enflamer_cent...
- enflamer_front()
- une_etape()
- une_etape_car...
- compte_zone()
- compte_fronti...
- feu_sur_fronti...
- obtention_ima...
- nom_image()
- Nom_de_fichi...
- ZEF()
- Film()
- Film_front()

Explorateur de fichiers

Cliquez sur l'étoile pour ajouter le répertoire à la liste des projets

C:\Users\bon

- Figure77474.png
- Figure_3.png
- Figure_5.png
- Foret00000.png
- Foret00001.png
- Foret00002.png
- Foret00003.png
- Foret00004.png
- Foret00005.png
- Foret00006.png
- Foret00007.png



!*.pyc

Rechercher dans les fichiers

```
85
86  def compte_zone(A,n,k):
87      "Compte le nombre de zone de A qui contient la valeur k"
88      conteur=0
89      for i in range (1,n+1):
90          for j in range (1,n+1):
91              "condition logique"
92              if (A[i,j] == k):
93                  conteur+=1
94      return conteur
95
96  def compte_frontiere_atteinte(liste, longueur):
97      "Compte dans la liste de réponse de longueur le nombre de fois où la frontière est atteinte avant la fin"
98      resultat=0
99      for i in range(longueur):
100         if (liste[i][1]==1):resultat+=1
101     return resultat
102
103 def feu_sur_frontiere(A,n):
104     "retourne 1 si un point de la frontière est en feu"
105     reponse=0
106     for i in range (1,n+1):
107         if ((A[i,1]==1) or (A[i,n]==1) or (A[1,i]==1) or (A[n,i]==1)):
108             reponse=1
109     return reponse
110
111 def obtention_image(A,n,nom_de_fichier):
112     "Donne l'image de la forêt à partir de la matrice :fichier png"
113     "join: convert a list of characters into a string"
114     colors = [(10,100,0),(100,2,0),(10,10,10)]
115     colors=[''.join([chr(x) for x in color]) for color in colors]
116
117     img_str=''
118
119     for line in range (n):
120         for col in range (n):
121
122             img_str += colors[A[line,col]]
123     img_str=img_str.encode()
124
125
126     img = Image.frombytes('RGB',(n,n),img_str)
127     img.save(nom_de_fichier,'PNG')
128
129     return True
130
```

```
def Nom_de_fichier_csv(n,proba):
    "Crée un nom de fichier"
    Nom='100_tirages_sur_carte'+str(n)+'x'+str(n)+'_avec_proba_de'+str(proba)+'csv'
    return Nom

def ZEF(n,proba,nb_de_cas):
    Nom=str(nb_de_cas)+'_etude_de_zones_en_feu_avec_proba_de'+str(proba)+'.csv'
    return Nom

def Film(n,p,carte):
    "obtention des images pour une carte nxn et une proba p"
    A=foret_avant_incendie(n)
    A=conditions_aux_limites(A,n)
    "Il y a un probleme par rapport a la fonction enflamer_centre"
    "[A,R]=enflamer_front(A,n)"
    [A,R]=enflamer_centre(A,n+2)
    print(A)
    print(R)
    i=-1
    while(R!=[]):
        i+=1
        if carte==0:
            [A,R]=une_etape(A,R,n,p)
            print( "etape :" , i ," avec ", compte_zone(A,n,1))
            obtention_image(A,n+2,nom_image(i,'Foret'))
        if carte==1:
            [A,R]=une_etape_carre_8(A,R,n,p)
            print( "etape :",i,"avec ",compte_zone(A,n,1)," zones en feu et R : ",R)
            obtention_image(A,n+2,nom_image(i,'Foret'))
    return A

def Film_front(n,p):
    A=foret_avant_incendie(n)
    A=conditions_aux_limites(A,n)
    A=deboiser(A,n,p)
    A,R=enflamer_front(A,n)
    i=-1
    while (R!=[]):
        i+=1
        A,R=une_etape(A,R,n,1)
        print("étape :",i," avec ",compte_zone(A,n,1), " zones en feu")
        obtention_image(A,n+2,nom_image(i,'Front'))
    return A

```

# ANNEXE3(ÉQUATION DE CHALEUR)

On raisonne sur un volume V, fixe dans un référentiel tridimensionnelle délimité par la surface fermée (S), l'énergie interne, à l'instant t , de ce système s'écrit: $U(t)=\iiint_{(v)} u(M,t)dm=\iiint_{(v)} \mu u(M,t)d\tau$

À l'instant t+dt : $U(t+dt)=\iiint_{(v)} \mu\frac{\partial u(M,t+dt)}{\partial t}d\tau$ ce qui donne $dU=U(t+dt)-U(t)=\iiint_{(v)} \mu\frac{\partial u}{\partial t}dt\,d\tau$

Soit : $\frac{dU}{dt}=\iiint_{(v)} \mu\frac{\partial u}{\partial t}d\tau=\iiint_{(v)} \mu c\frac{\partial T(M,t)}{\partial t}dt$

Le bilan de puissance s'écrit notant: $\sigma$ la puissance "produite" par unité de volume du système par des mécanismes autre que la conduction: $\frac{dU}{dt}=-\oiint_{(S)} \vec{j}\,d\vec{S}+\iiint_{(v)} \sigma\,d\tau$

En combinant les équations et en utilisant le théorème de Green-Ostogradsky, on écrit:

$$\iiint_{(v)} \mu c\frac{\partial T(M,t)}{\partial t}d\tau=\iiint_{(v)} (-dive(\vec{j})+\sigma)d\tau$$

Comme aucune hypothèse n'a été faite sur le volume V , on en déduit l'équation locale suivante:

$$\mu c\frac{\partial T(M,t)}{\partial t}=-dive((\vec{j}))+\sigma$$

L'introduction de la loi de Fourier : $\vec{j}(M,t)=-\lambda\nabla\vec{T}(M,t)$ et l'utilisation de l'égalité $dive(\nabla T)=\Delta T$

Permet d'écrire l'équation de la chaleur: $\mu c\frac{\partial T(M,t)}{\partial t}=\lambda\Delta T(M,t)+\sigma(M,t)$

Ici dive signifie l'opérateur divergence