Protection contre la foudre des structures et zones urbanisées

Par Quentin ROUCH
Numéro de candidat : **32448**

Problématique

Comment évaluer la protection fournie par un paratonnerre ?

Objectifs

S'approprier le phénomène

Développer un modèle

Étude de cas concrets

I — II — III — IV

I. Préliminaire

Principe du paratonnerre
Déroulement d'un orage

II. Simulation

Modélisation et mise en équation du problème
Effet de pointe
Modèle de propagation d'un éclair
Optimisation

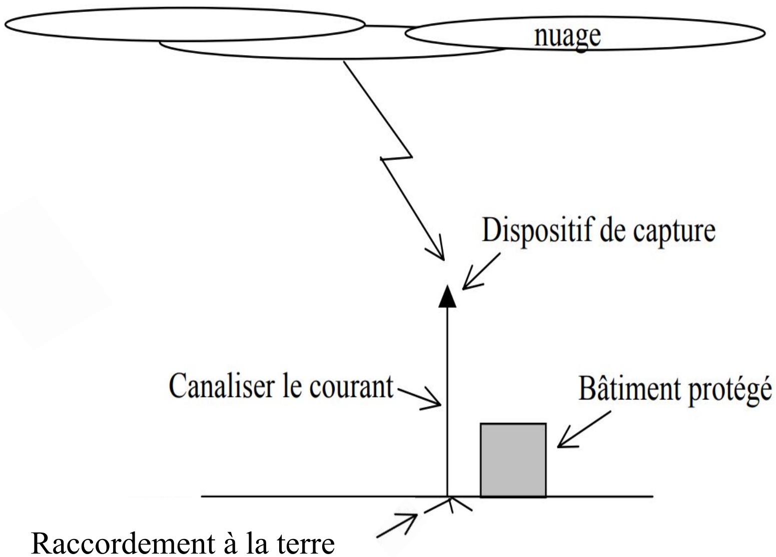
III. Validation

Densité de foudroiement
Maquette
Protocole expérimental
Caractéristiques électriques de la maquette
Comparaison maquette simulation

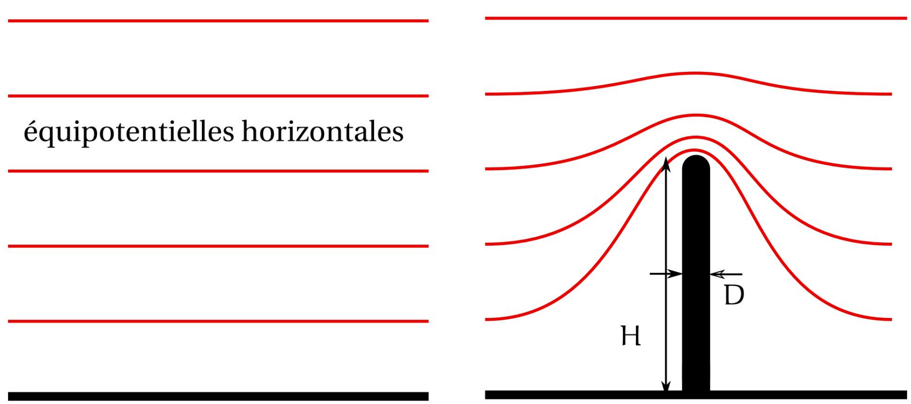
IV. Application concrète

Cas concret n°1
géométrie simple
Cas concret n°2
géométrie quelconque

Principe du paratonnerre



Comment modéliser numériquement ce phénomène ?



Situation au repos — *Situation perturbée*

PTS

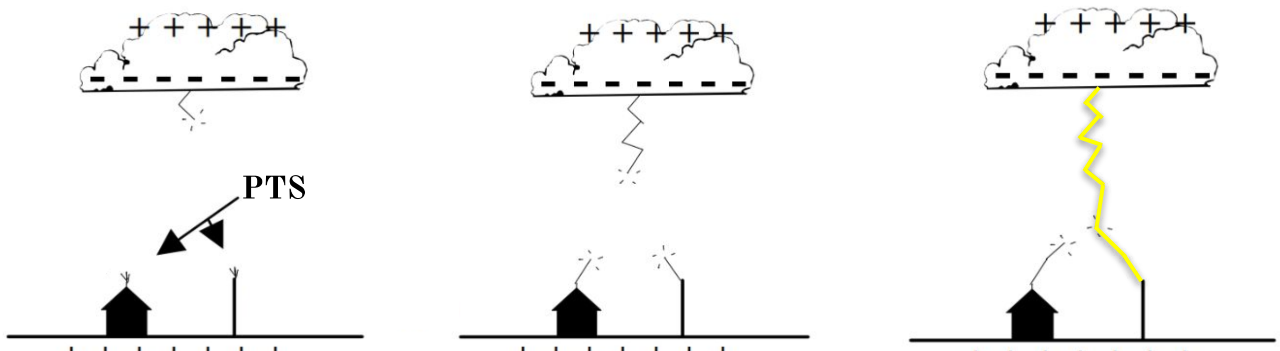
Vocabulaire Technique

Quelle différence lors d'un foudroiement ?

PDA

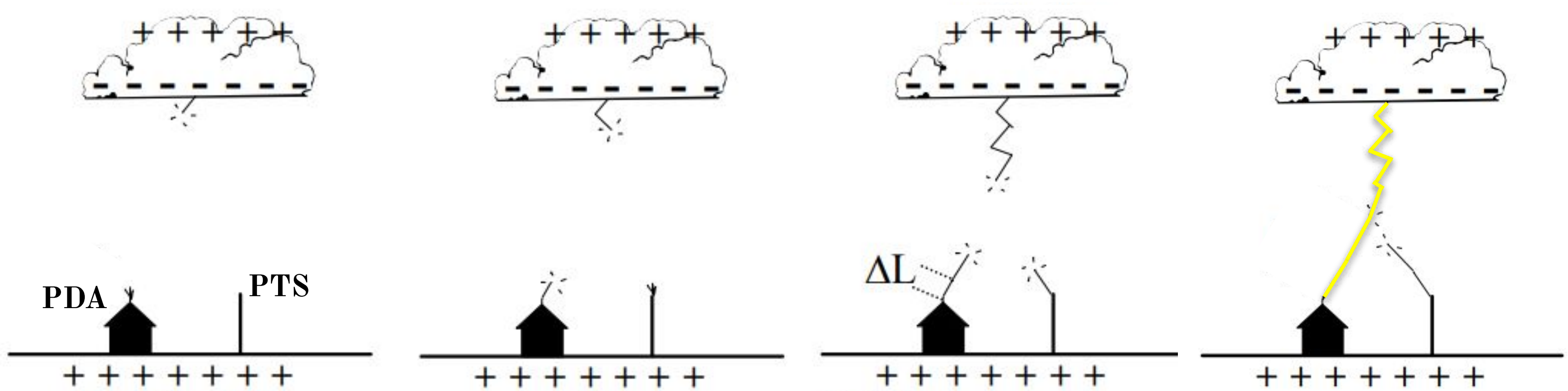
Vocabulaire Technique

Déroulement d'un orage



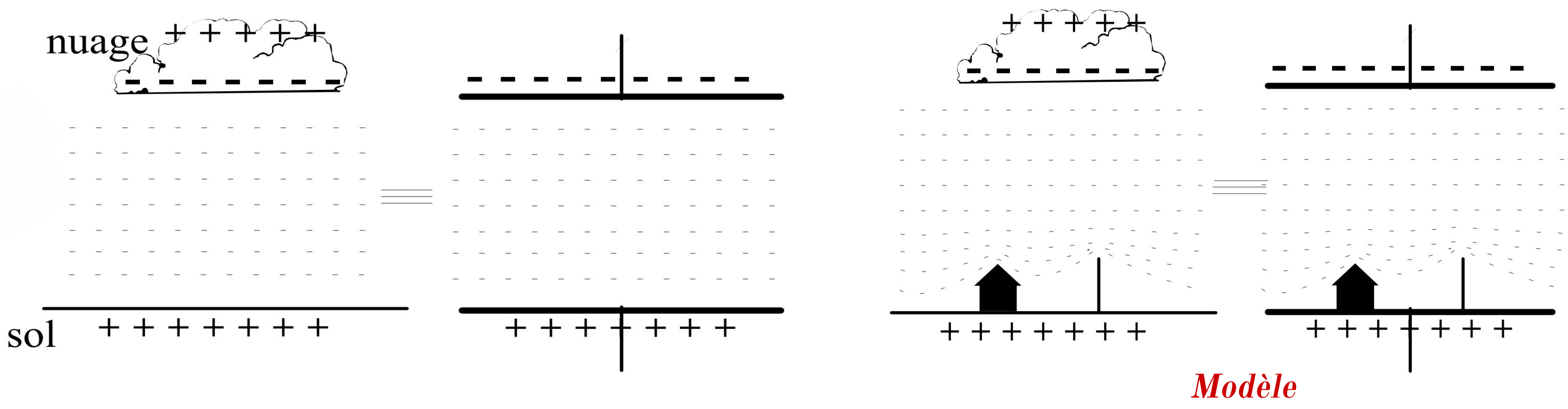
Départ de traceurs ascendants — *Compétition* — *Jonction et décharge*

Traceur descendant → E_{sol} ↗ → Pré-décharges → $E_{sol} = E_{cri}$ → Traceurs ascendants



Départ du traceur ascendant du PDA — *Départ d'autres traceurs ascendants* — *Compétition des traceurs* — *Jonction des traceurs et décharge*

Mise en équations du problème

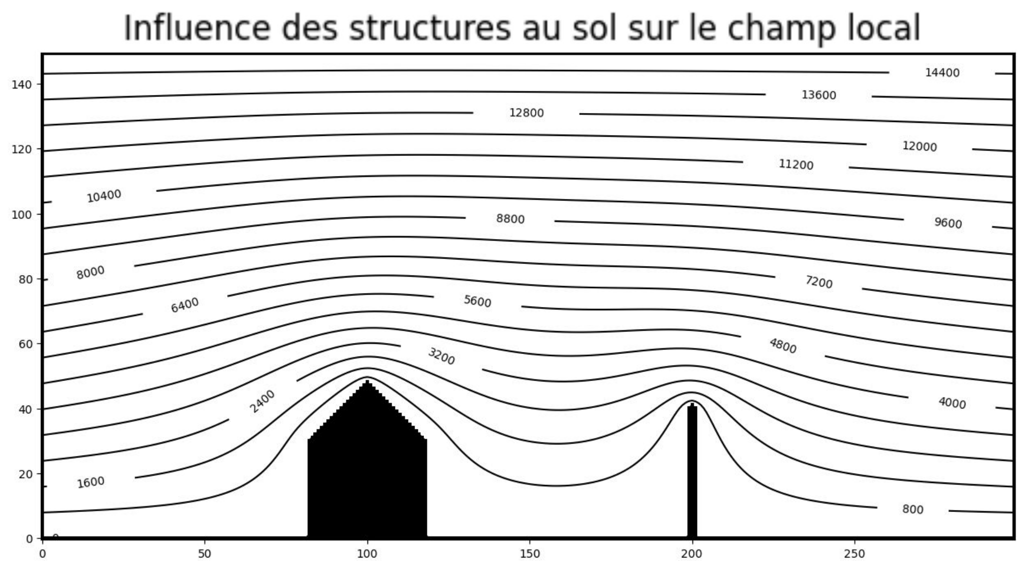


- Discrétisation
- Conditions aux limites
- Équation de Laplace

$$\Delta V = 0 \xrightarrow{\text{(Méthode de Jacobi)}} V \text{ en tout point}$$

\+ *Tracé des équipotentielles*

Annexe 1



Implémentation Python

Effet de pointe

Annexe 2

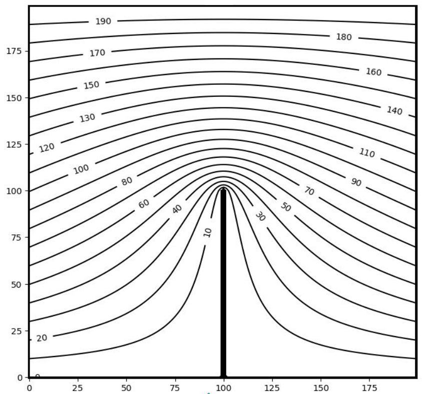
$$-\overrightarrow{\text{grad}\ V}$$

Approximation discrète

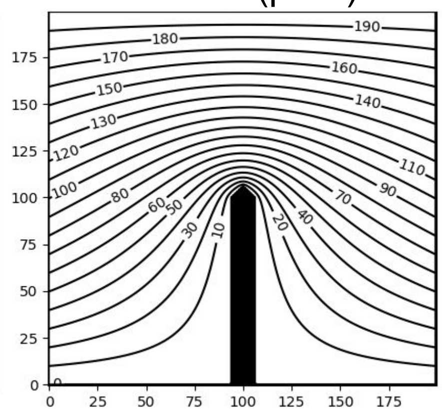
$$\downarrow$$

Norme de $\overrightarrow{E}$

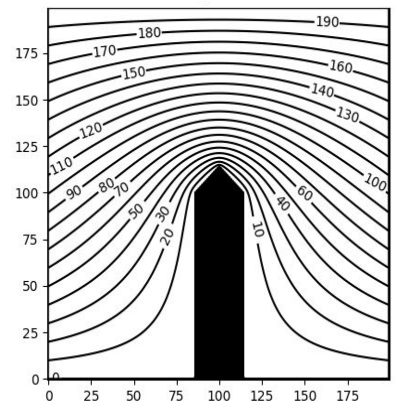
d = 3 (pixel)

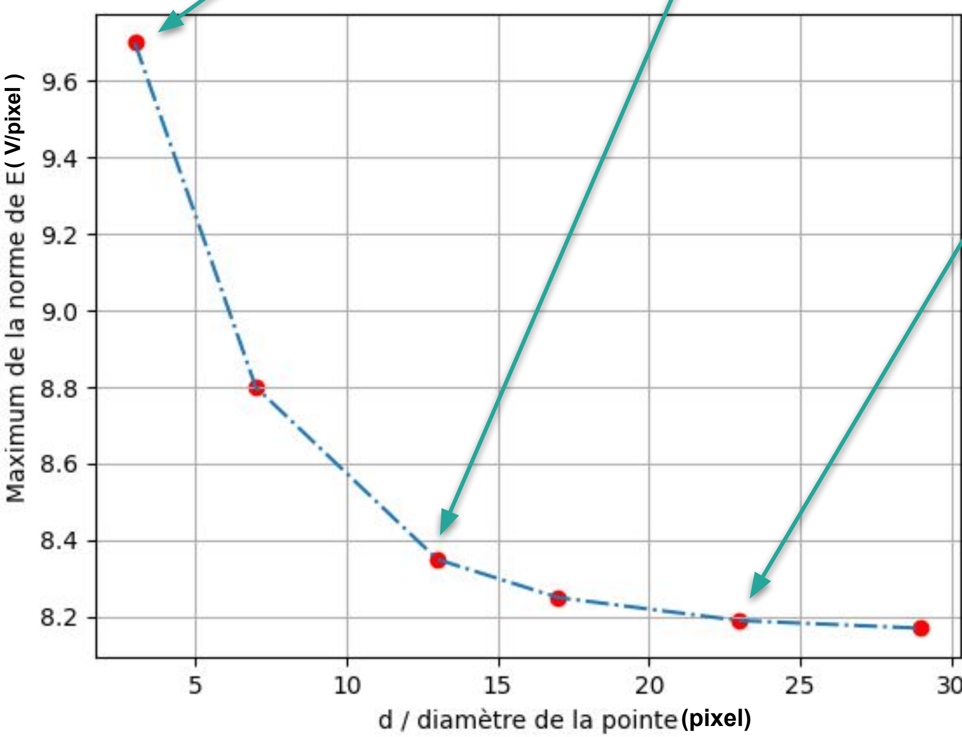


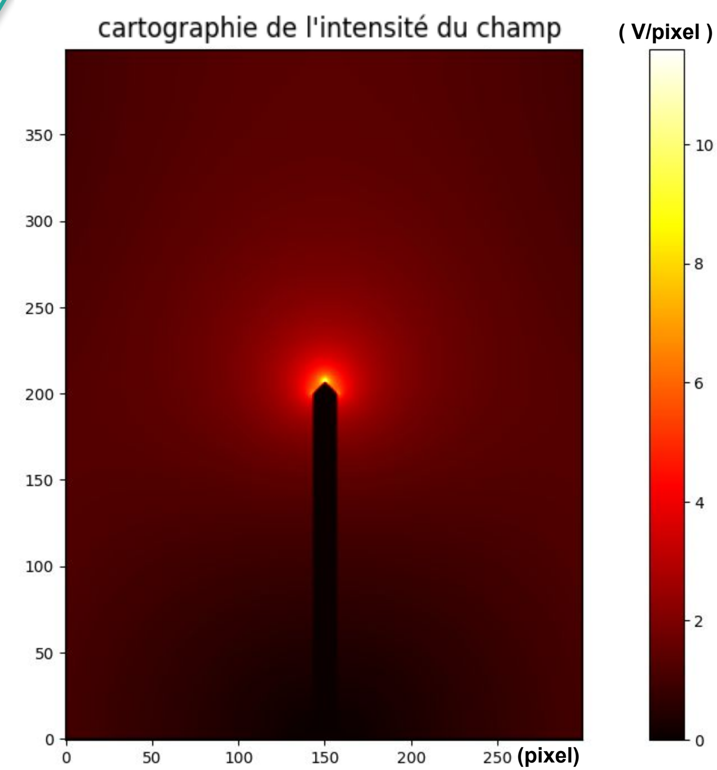
d = 13 (pixel)



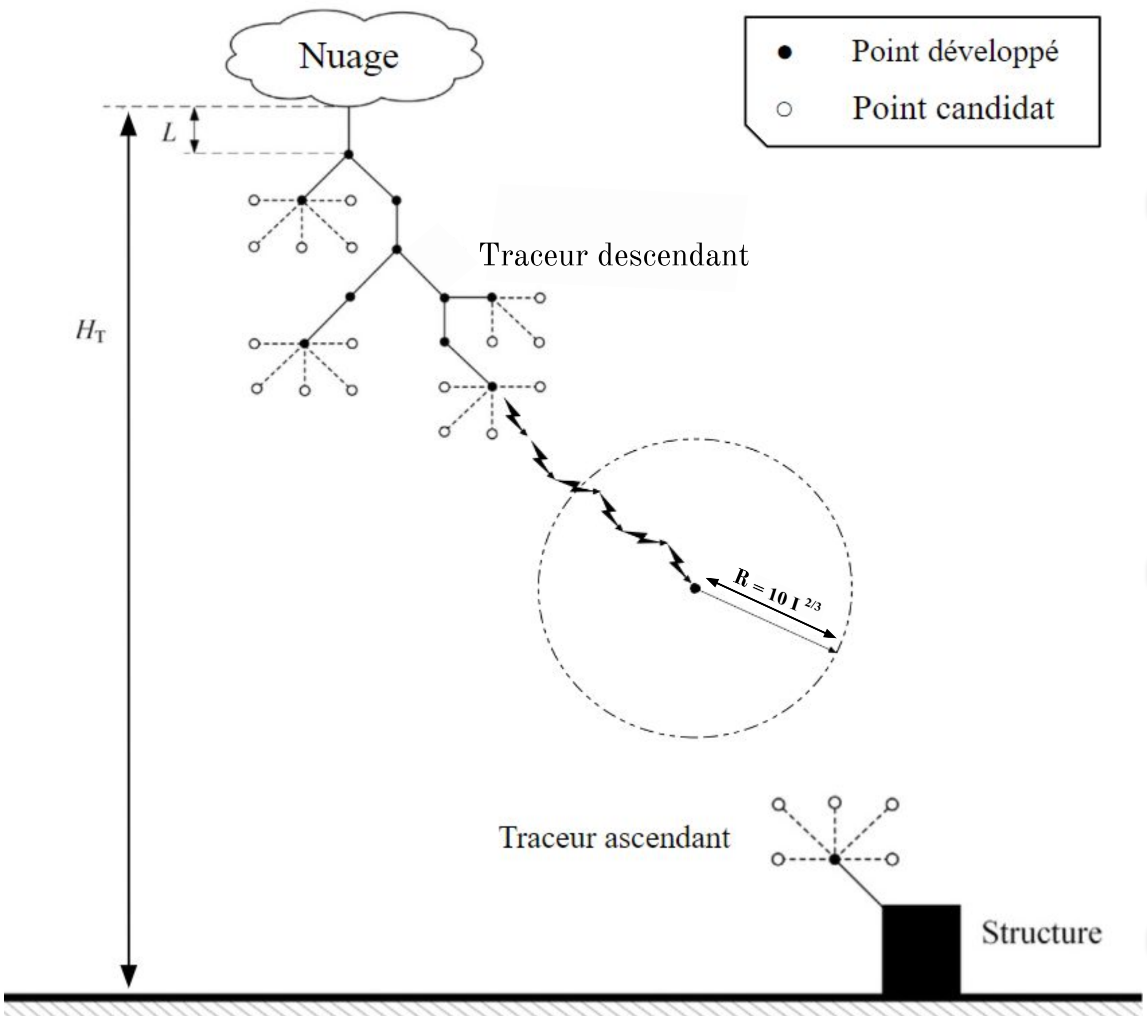
d = 23 (pixel)







Modèle de propagation d'un éclair



Comportement de la foudre ?

Modèle de développement fractal de foudre non déterministe

Développement de traceurs ascendants ?

Champ électrique limite au delà duquel on observe l'apparition de traceurs

Attachement des traceurs ?

Modèle de la sphère fictive

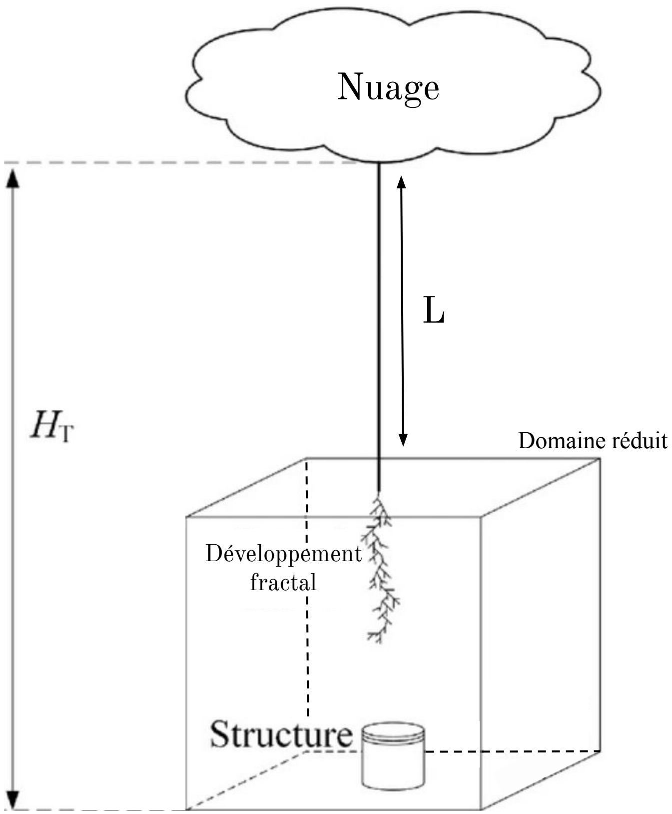
Annexe 3

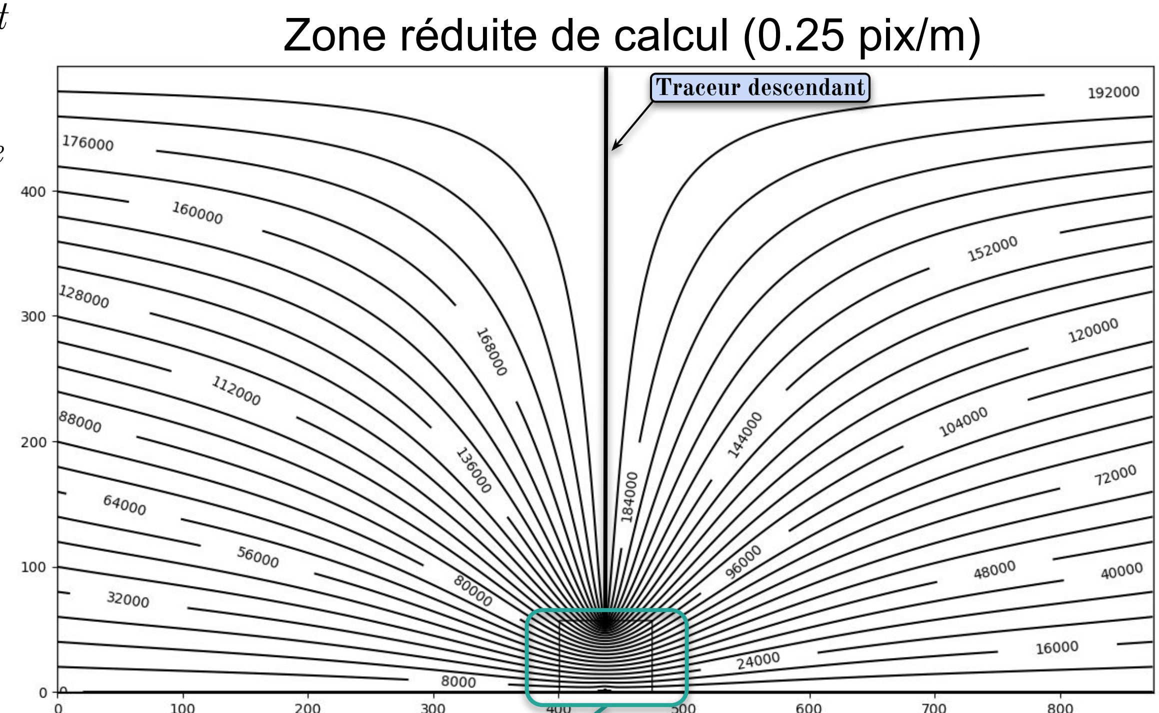
Développement fractal du nuage jusqu'au sol impossible (Complexité exponentielle)

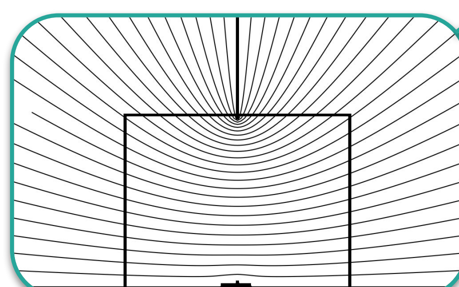
Environnement de travail

Réduire le nombre de développement

Réduction du domaine et hypothèse sur la propagation du traceur



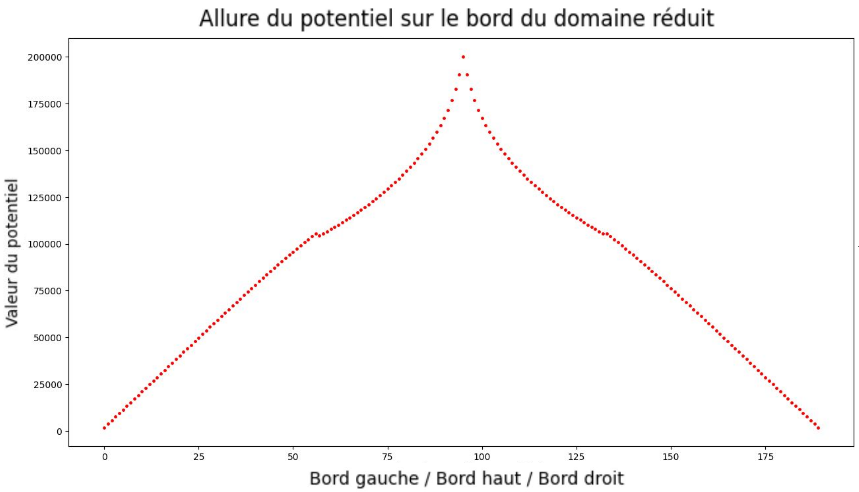


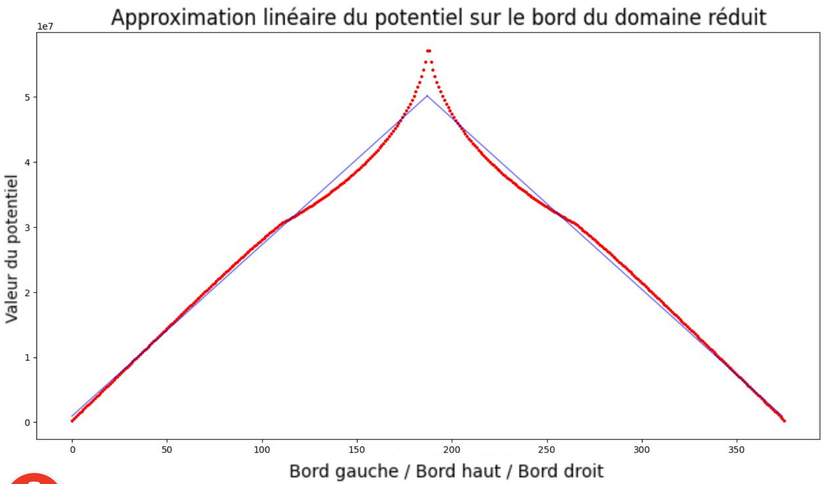


Conditions aux limites inconnues

Impossible de résoudre

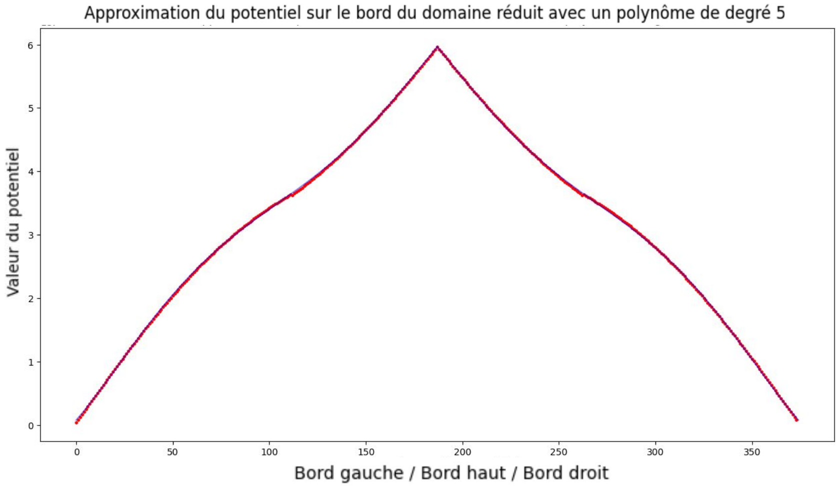
Conditions de convergence

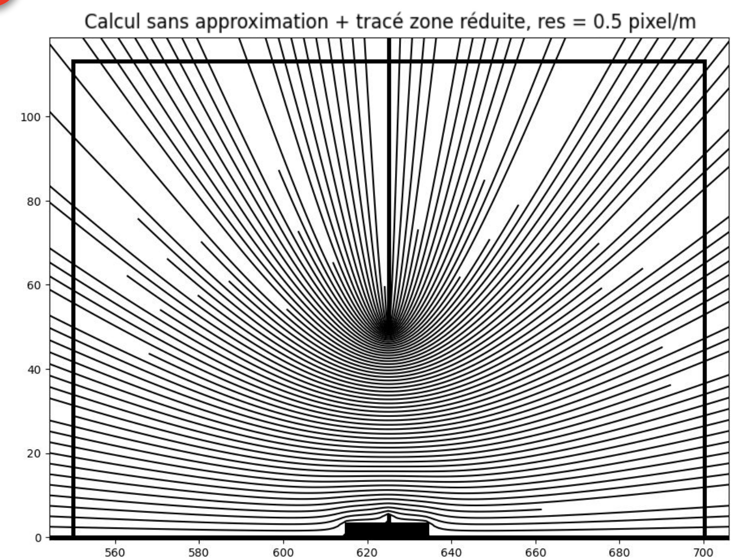




Approximation linéaire

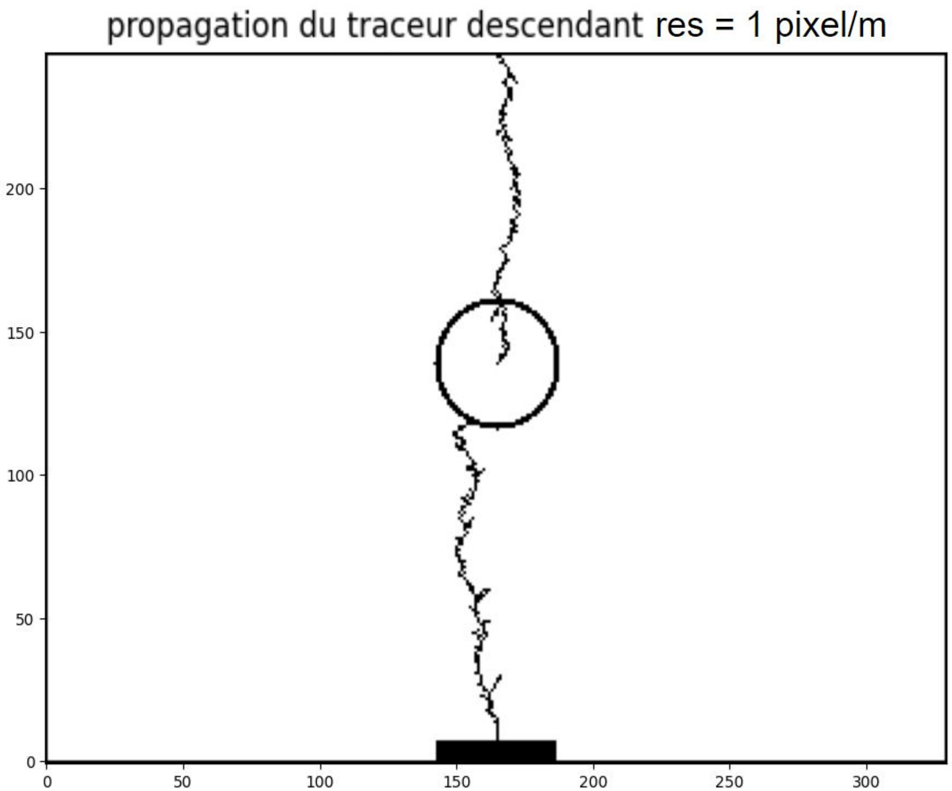
Conditions aux limites changent lors de la propagation





Approximation polynômiale

Optimisation du code



Augmenter la résolution

~ 250 itérations

Méthode de Gauss Seidel

Annexe 4

~ 3h

Cython

Annexe 5

Calculs récurrents qui pourraient-être évités

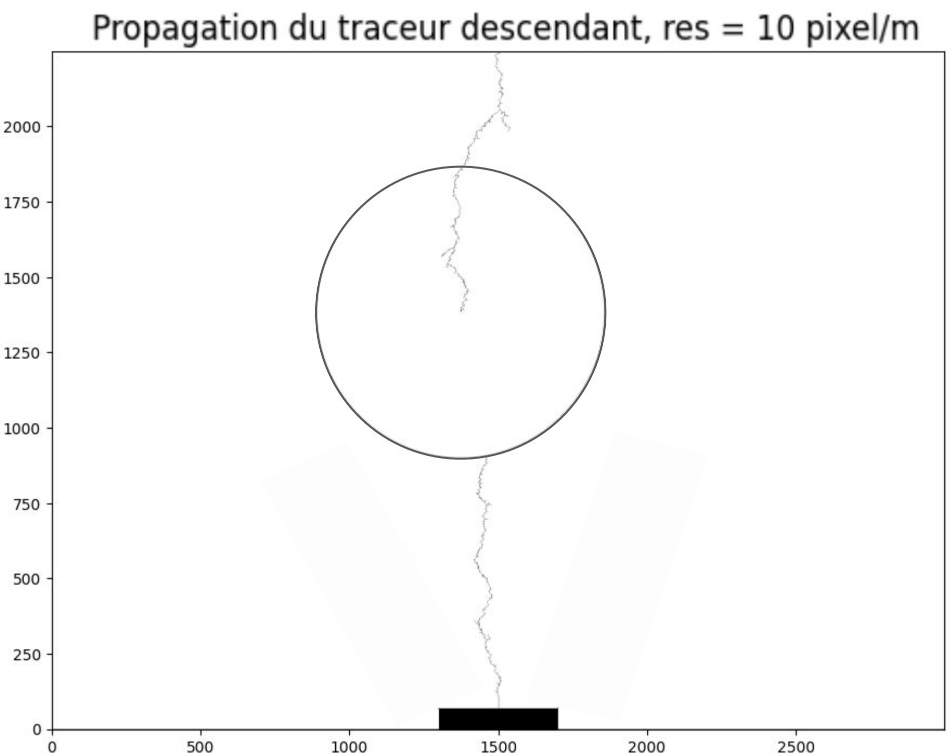
Nouvelle conception de la convergence

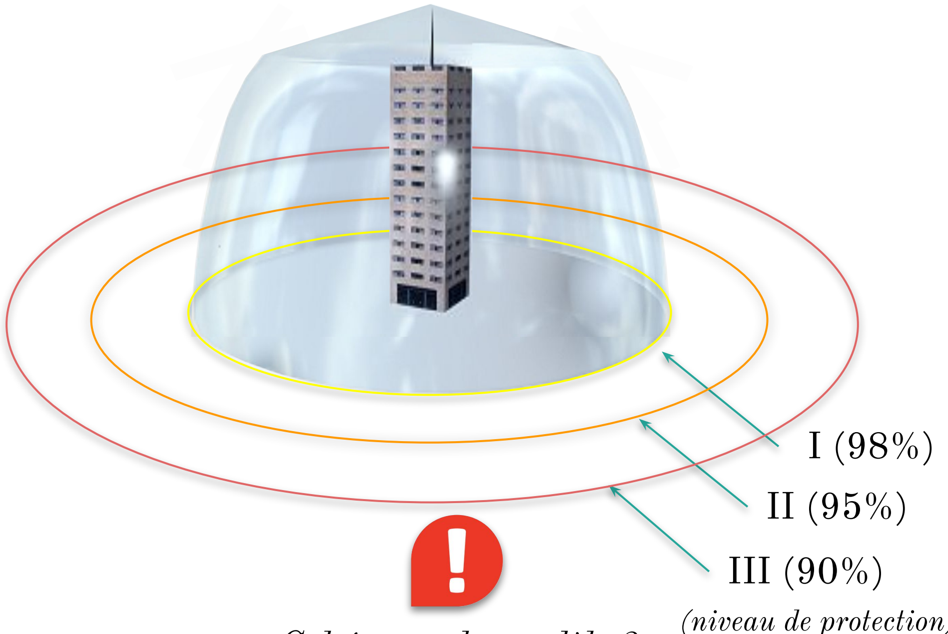
Annexe 6

Densité de foudroiement

Évaluer la protection fournie par un paratonnerre ?

Interpréter la notion de rayon de protection en terme de densité de foudroiement





Estimation numérique de la densité de foudroiement

Approche statistique dite de Monte-Carlo

Cohérence du modèle ?

Validation expérimentale des résultats du modèle

Maquette

Garantir la cohérence du modèle numérique

Reproduire le phénomène à échelle réduite

Annexe 7

Plateau conducteur suspendu

Générateur de Marx (1MV)

Laboratoire Ampère 22/02/2023

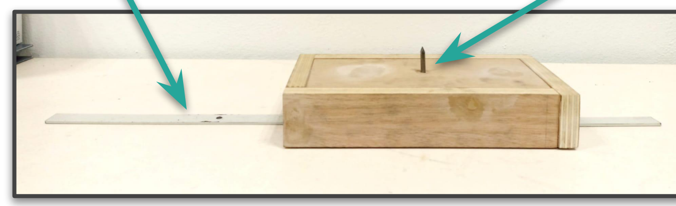
Plateau de terre

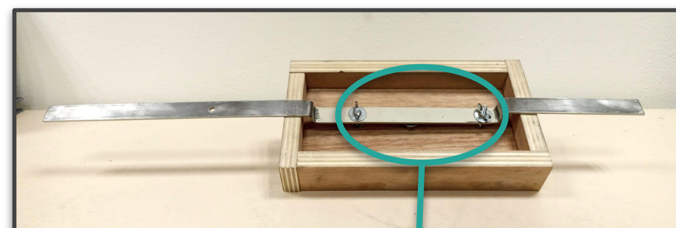
Mise à la terre

Mise à la terre de la maquette

Annexe 8

Pointe

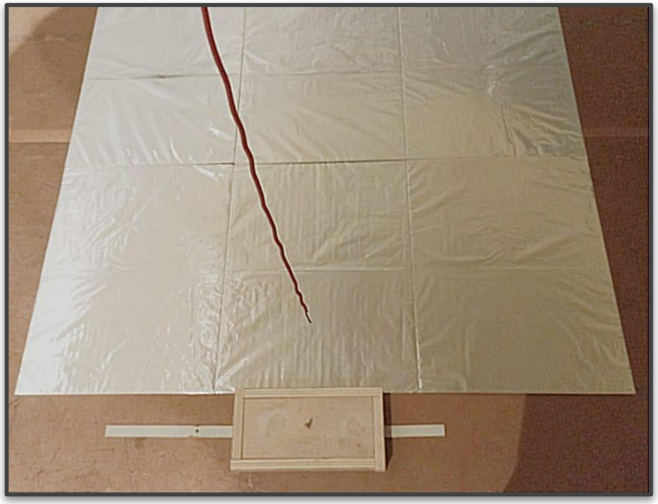




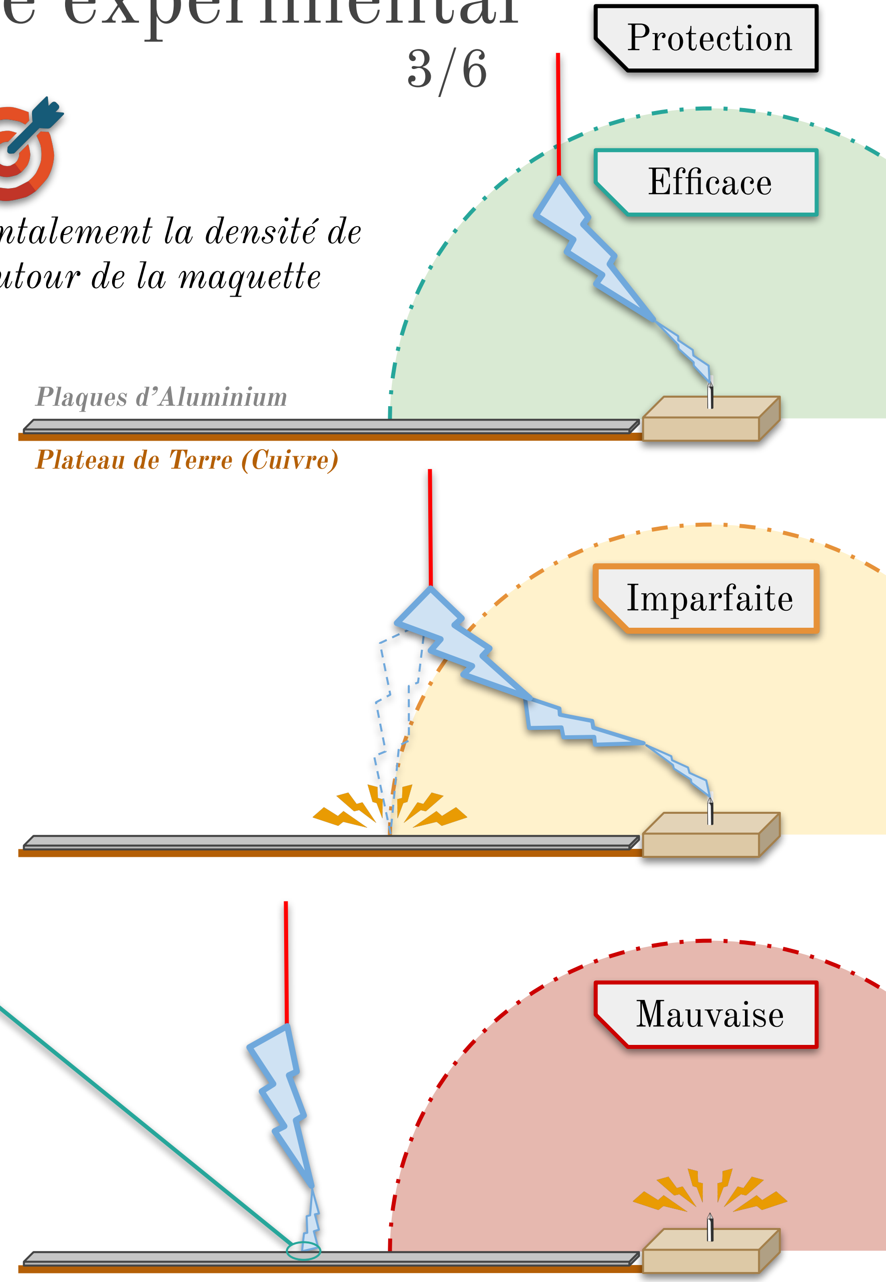
Système de pression pour assurer le contact entre la pointe et la terre

Protocole expérimental

3/6

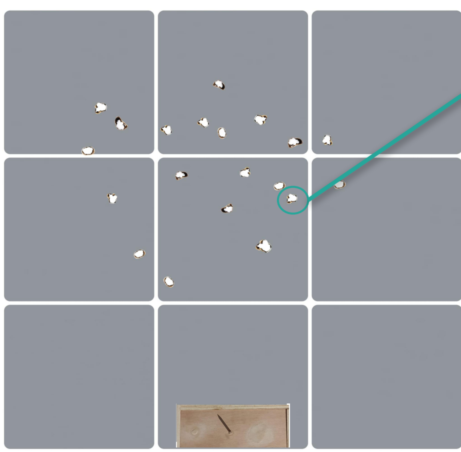


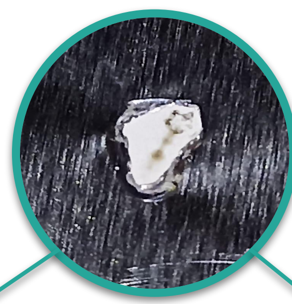
Évaluer expérimentalement la densité de foudroiement autour de la maquette



Visualiser les impacts

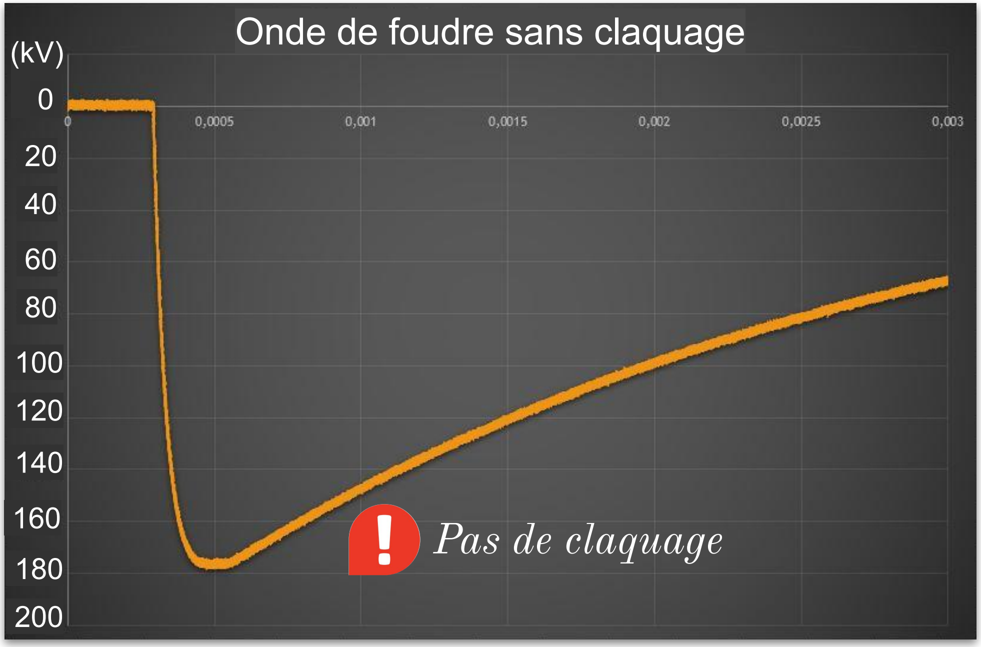
Plaques d'Aluminium

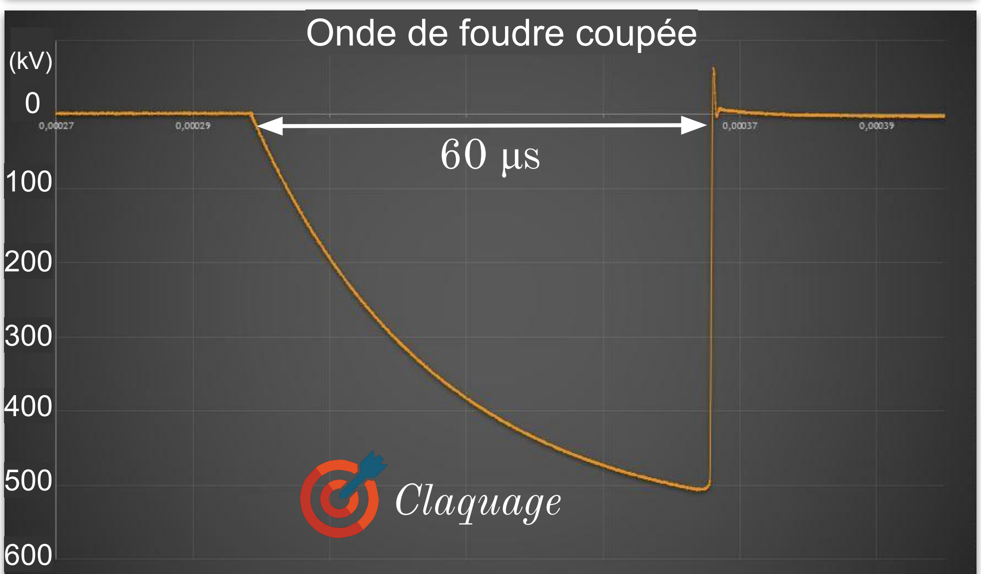




Champs électriques critiques inconnus

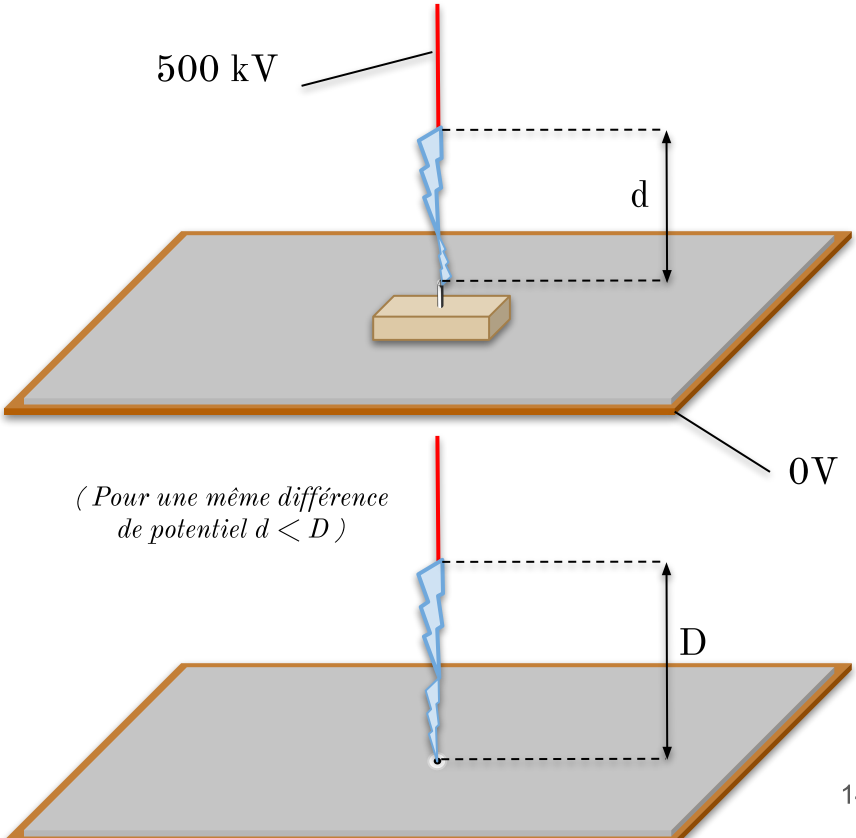
Caractéristiques de la maquette



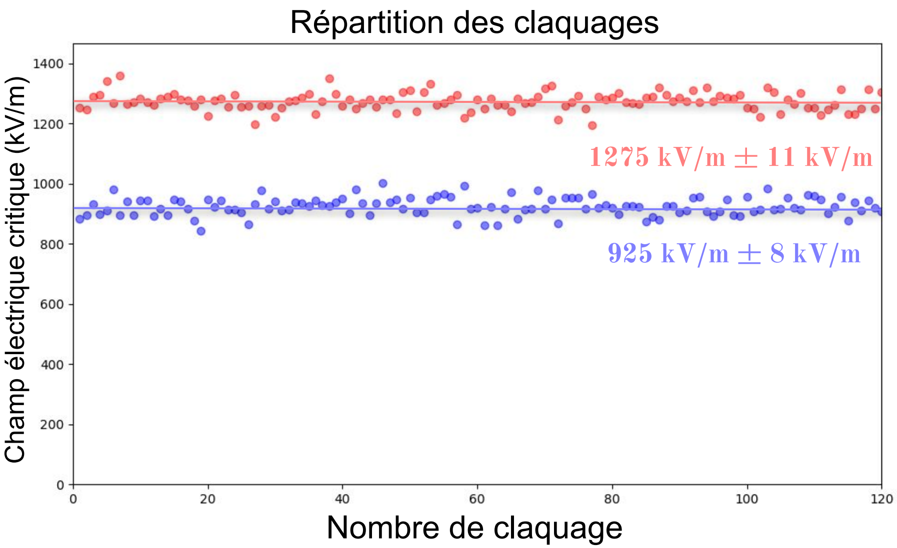


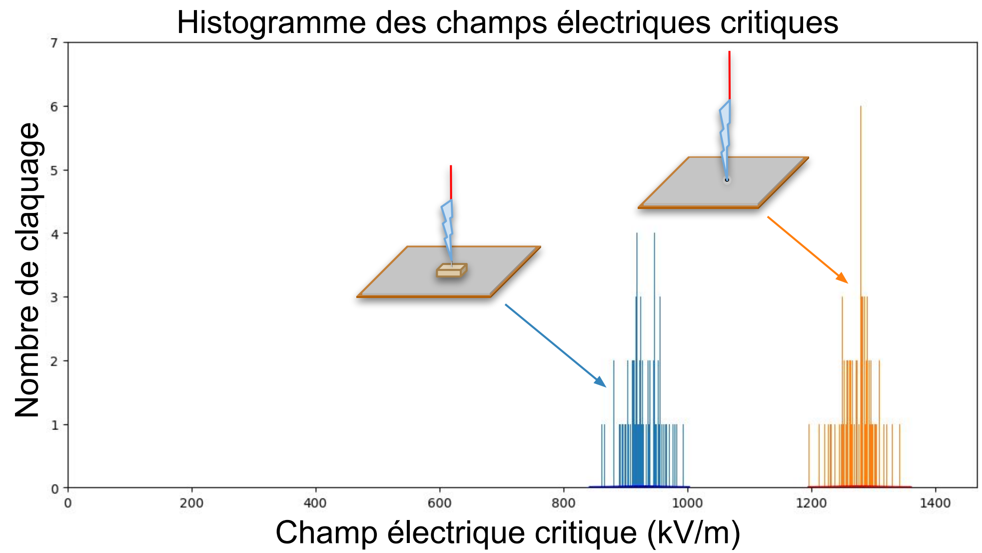
Déterminer les tensions de claquage entre :

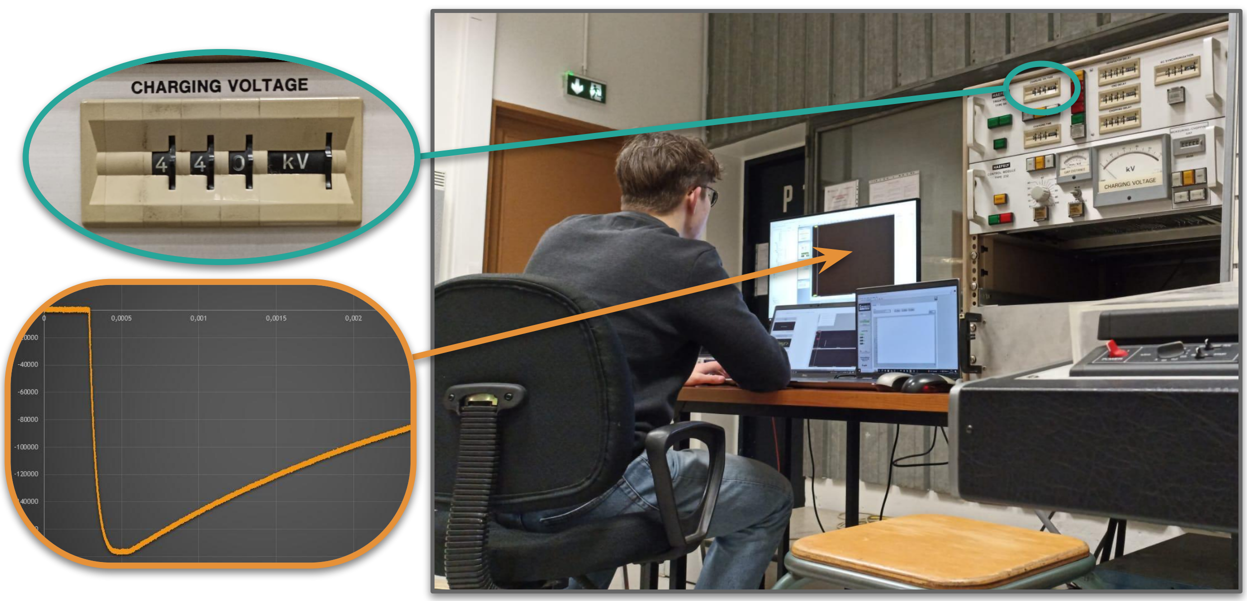
- ❑ *le fil et la pointe*
- ❑ *le fil et le plan*



Caractéristiques de la maquette



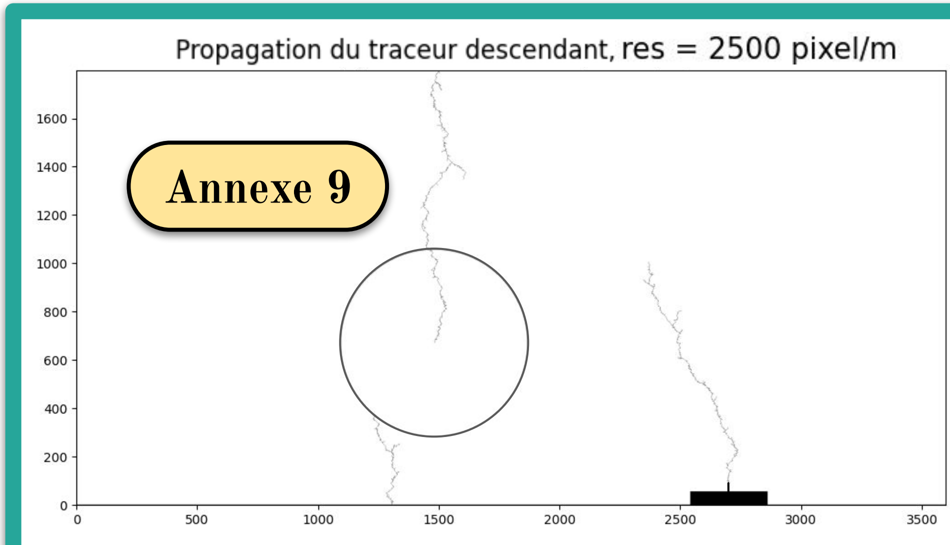




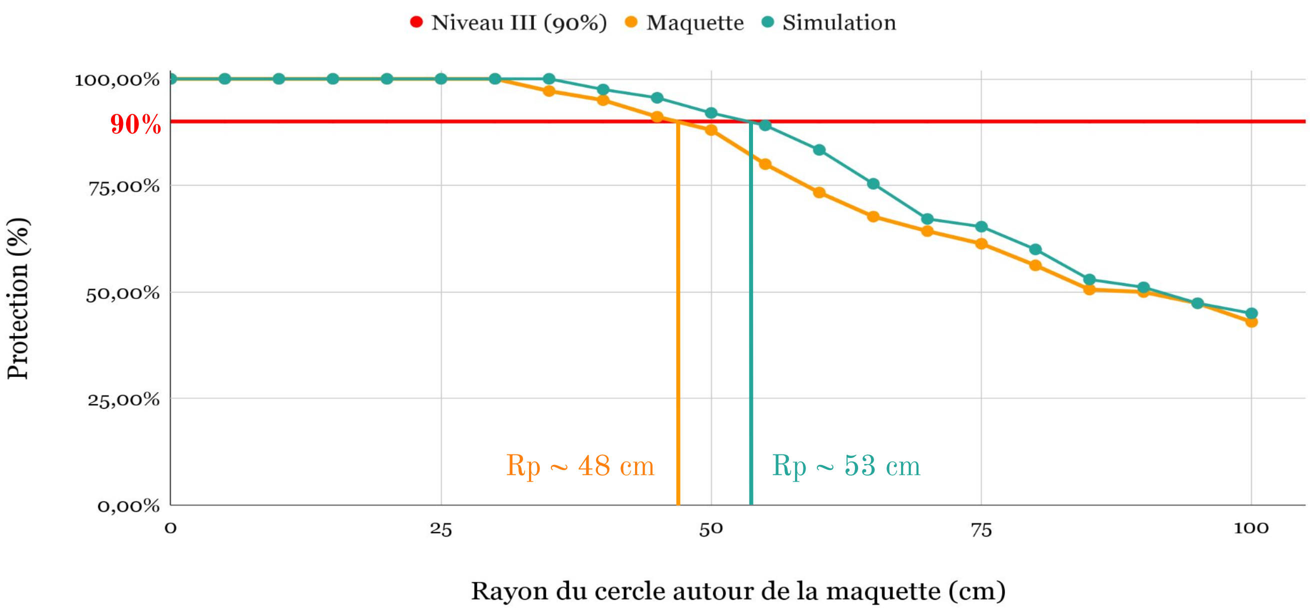
Estimer numériquement la densité de foudroiement autour de la maquette

Simuler les mêmes claquages avec les caractéristiques de la maquette

Comparaison maquette/simulation



Pourcentage de protection estimé grâce à la densité de foudroiement autour de la maquette



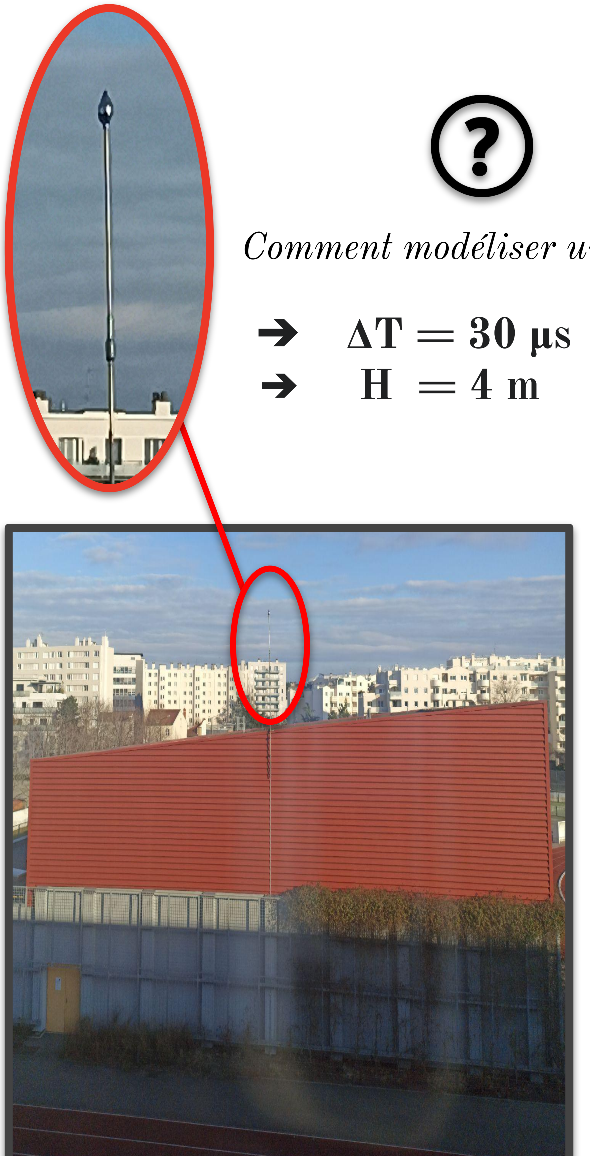
Rayon de protection estimé par la **Simulation**

Rp ~ 53 cm

Rayon de protection estimé avec la **Maquette**

Rp ~ 48 cm

Cas concret 1 : Gymnase du lycée



Gymnase et PDA depuis l'internat

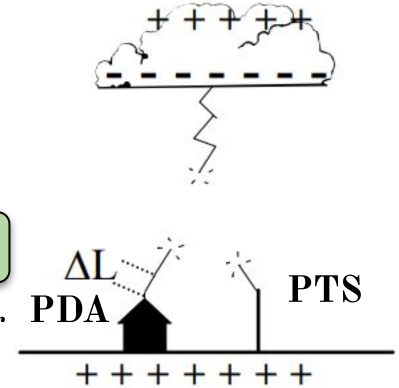
Comment modéliser un PDA ?

➔ $\Delta T = 30\ \mu s$
➔ $H = 4\ m$

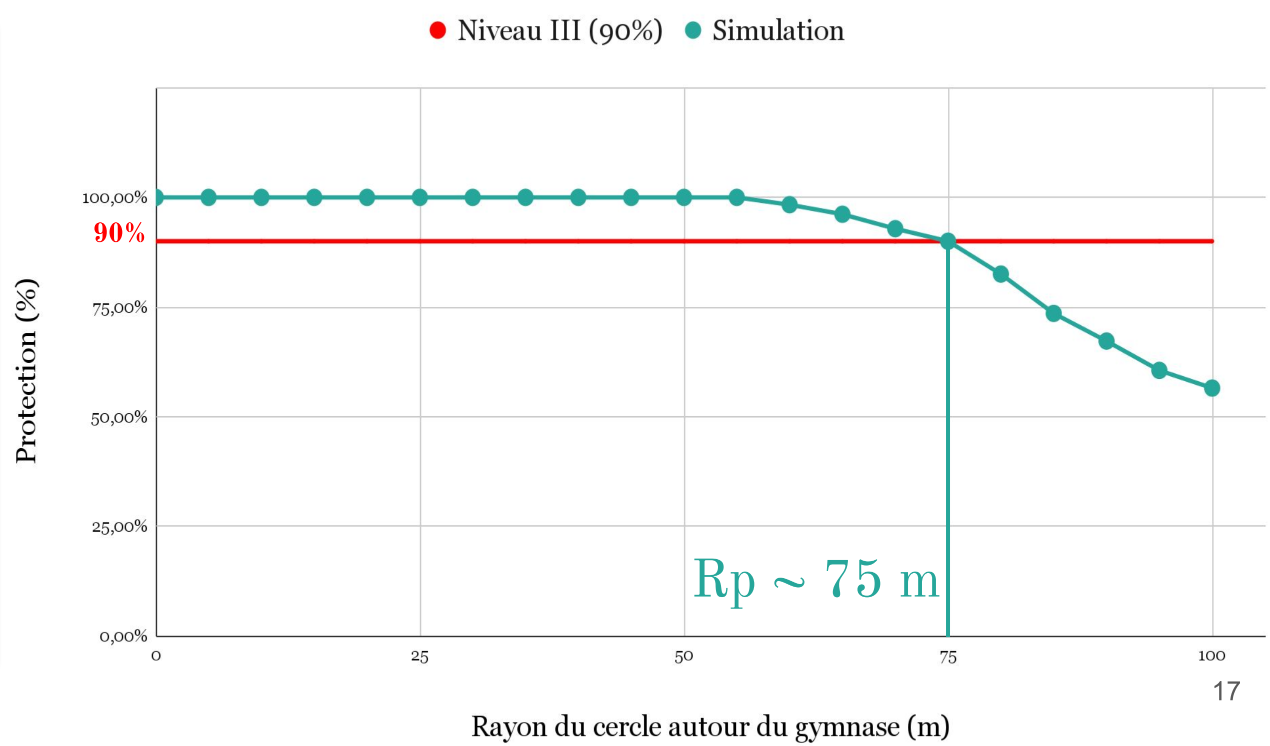
Approximation :

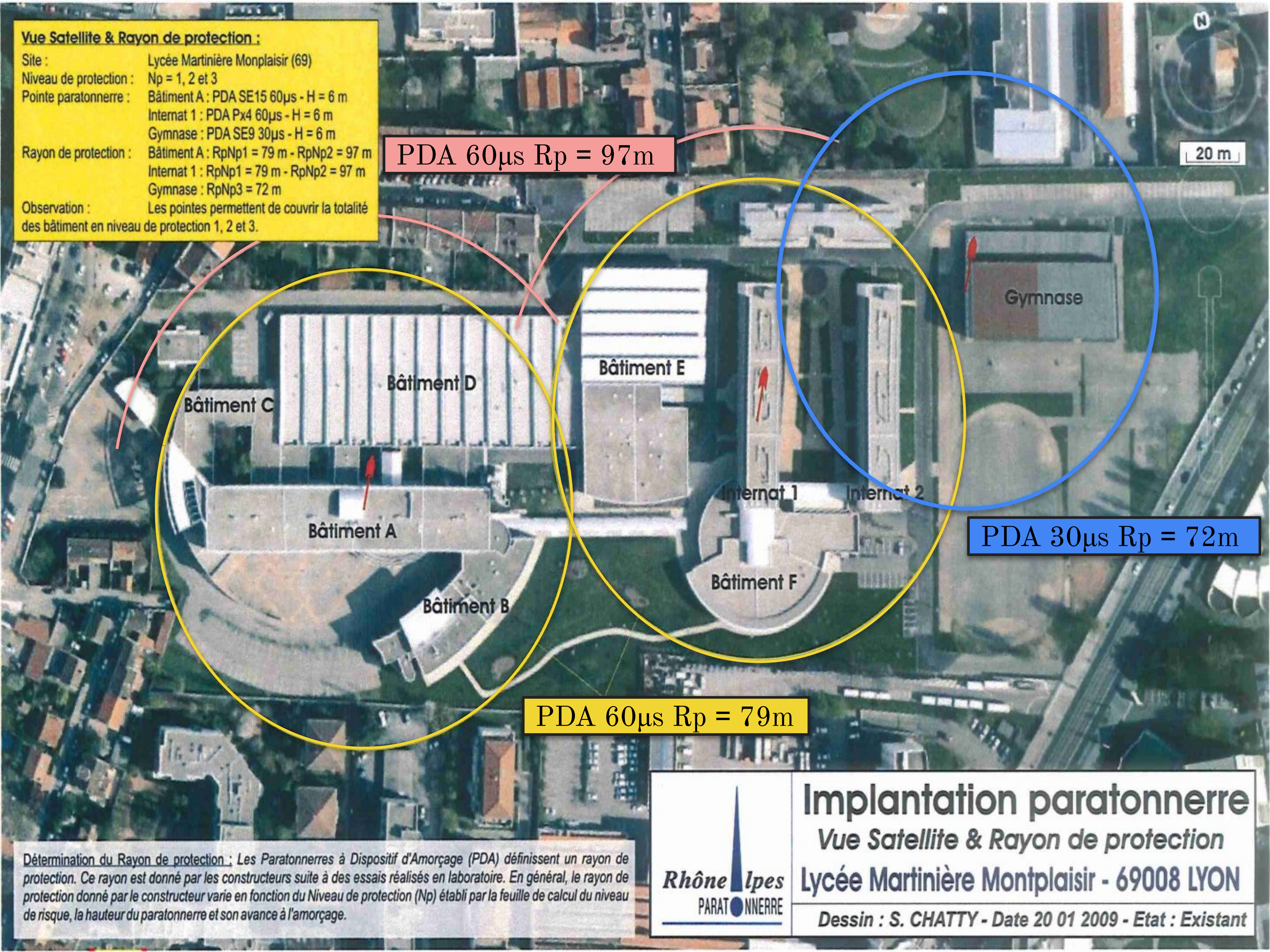
$$PDA = PTS + \Delta L \text{ avec } \Delta L = V \times \Delta T$$

V : vitesse de propagation d'un traceur



Protection dans un cercle autour du gymnase



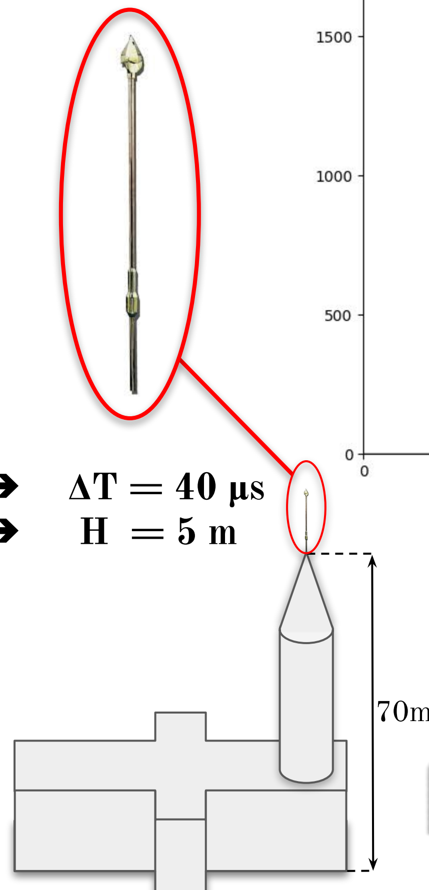

Vue Satellite & Rayon de protection :
Site : Lycée Martinière Monplaisir (69)
Niveau de protection : Np = 1, 2 et 3
Pointe paratonnerre : Bâtiment A : PDA SE15 60µs - H = 6 m
Internat 1 : PDA Px4 60µs - H = 6 m
Gymnase : PDA SE9 30µs - H = 6 m
Rayon de protection : Bâtiment A : RpNp1 = 79 m - RpNp2 = 97 m
Internat 1 : RpNp1 = 79 m - RpNp2 = 97 m
Gymnase : RpNp3 = 72 m
Observation : Les pointes permettent de couvrir la totalité des bâtiment en niveau de protection 1, 2 et 3.
PDA 60µs Rp = 97m
N
20 m
Gymnase
Bâtiment D
Bâtiment C
Bâtiment E
Internat 1
Internat 2
PDA 30µs Rp = 72m
Bâtiment A
Bâtiment F
Bâtiment B
PDA 60µs Rp = 79m
Détermination du Rayon de protection : Les Paratonnerres à Dispositif d'Amorçage (PDA) définissent un rayon de protection. Ce rayon est donné par les constructeurs suite à des essais réalisés en laboratoire. En général, le rayon de protection donné par le constructeur varie en fonction du Niveau de protection (Np) établi par la feuille de calcul du niveau de risque, la hauteur du paratonnerre et son avance à l'amorçage.
Rhône lpes PARAT NNERRE
Implantation paratonnerre
Vue Satellite & Rayon de protection
Lycée Martinière Montplaisir - 69008 LYON
Dessin : S. CHATTY - Date 20 01 2009 - Etat : Existant

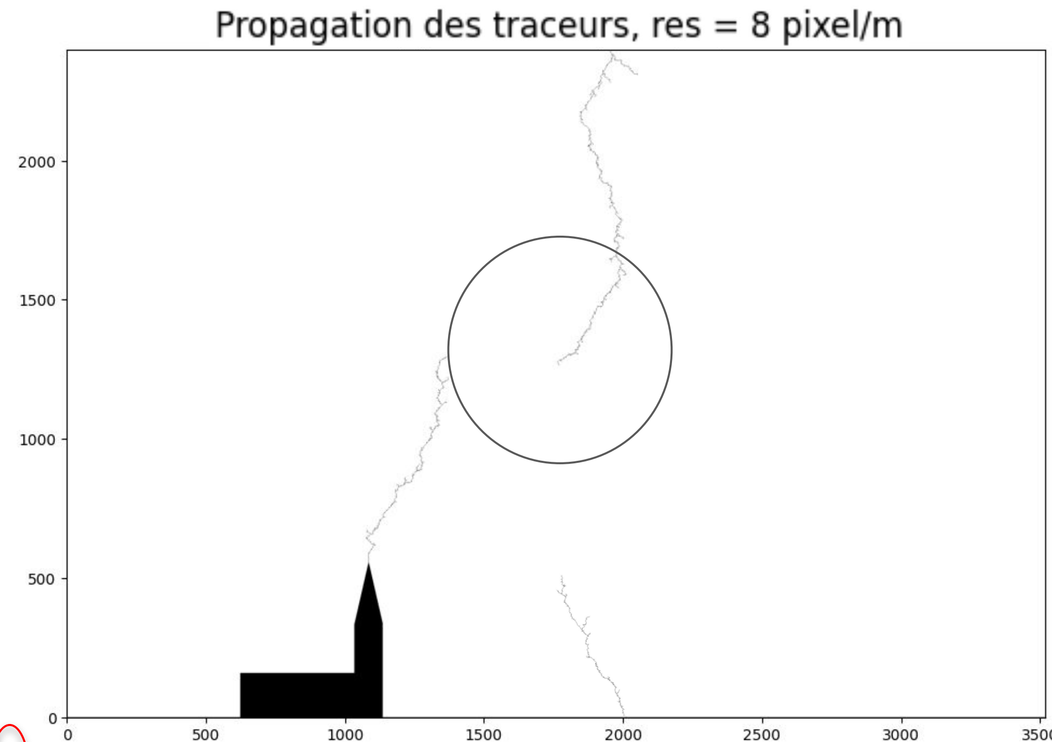
Cas concret 2 : Cathédrale de Gap
3/3

Protection souhaitée : **Niveau I** (98%)

Cathédrale de Gap

➔ **$\Delta T = 40$ µs**
➔ **H = 5 m**





Rayon de protection **simulé ~** 64m

Indelec

Rayon de protection **estimé ~** 58m

Conclusion

Rhône Alpes Paratonnerre

Gymnase

90% - Niveau III | 72m

90% - Niveau III | 75m | Rayon de protection surestimé
Erreur : +5%

Indelec

Cathédrale

98% - Niveau I | 58m

98% - Niveau I | 64m | Rayon de protection surestimé
Erreur : +10%

Pistes d'amélioration

- Réaliser plus de simulations
- Reconsidérer certaines hypothèses simplificatrices
- Prendre en compte les potentiels traceurs ascendants venant du bâtiment lui-même

Vocabulaire Technique

- **Traceur** ⇔ Canal d'air ionisé qui se propage de la base du nuage jusqu'au sol
- **Attachement des traceurs** ⇔ jonction entre un traceur ascendant et un traceur descendant qui permet la décharge entre le nuage et le sol
- **Amorçage** ⇔ Ionisation de l'air / Départ d'un traceur ascendant
- **PTS** ⇔ Paratonnerre à Tige Simple
- **PDA** ⇔ Paratonnerre à Dispositif d'Amorçage
- **Avance à l'amorçage** ⇔ capacité du PDA à amorcer avant un PTS
- **Champ électrique critique** ⇔ Champ électrique nécessaire à l'amorçage
- **Claquage** ⇔ Amorçage puis décharge simultanée
- **Densité de foudroiement** ⇔ rapport du nombre de traceur interceptés sur le nombre de coup de foudre total dans une zone donnée

Annexe 1

Méthode de Jacobi

```python
def ecart(V1,V2):

    Nx = V1.shape[0]
    Ny = V1.shape[1]
    diff_carre = 0
    for i in range(Nx):
        for j in range(Ny):
            diff_carre += (V1[i,j]-V2[i,j])**2
    return (diff_carre/(Nx*Ny))**(1/2)


def iteration_jacobi():

    V_copie = V.copy()
    for i in range(Nx):
        for j in range(Ny):
            if B[i,j]: # si [i,j] est un point du bord ne rien faire
                V[i,j] = V_copie[i,j]
            else:
                V[i,j] = (V_copie[i+1,j]+V_copie[i-1,j]+V_copie[i,j+1]+V_copie[i,j-1])/4
    return ecart(V_copie,V)

def jacobi(eps):

    compteur = 0
    while iteration_jacobi() > eps:
        iteration_jacobi()
        compteur += 1
        print(iteration_jacobi() - eps)
    return compteur

```

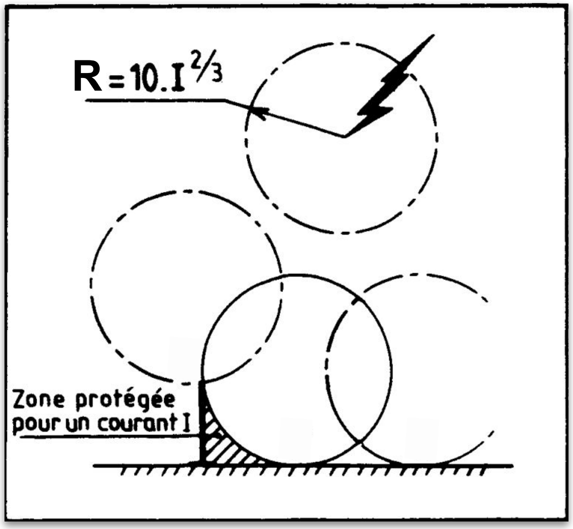
Annexe 2

Approximation discrète du gradient

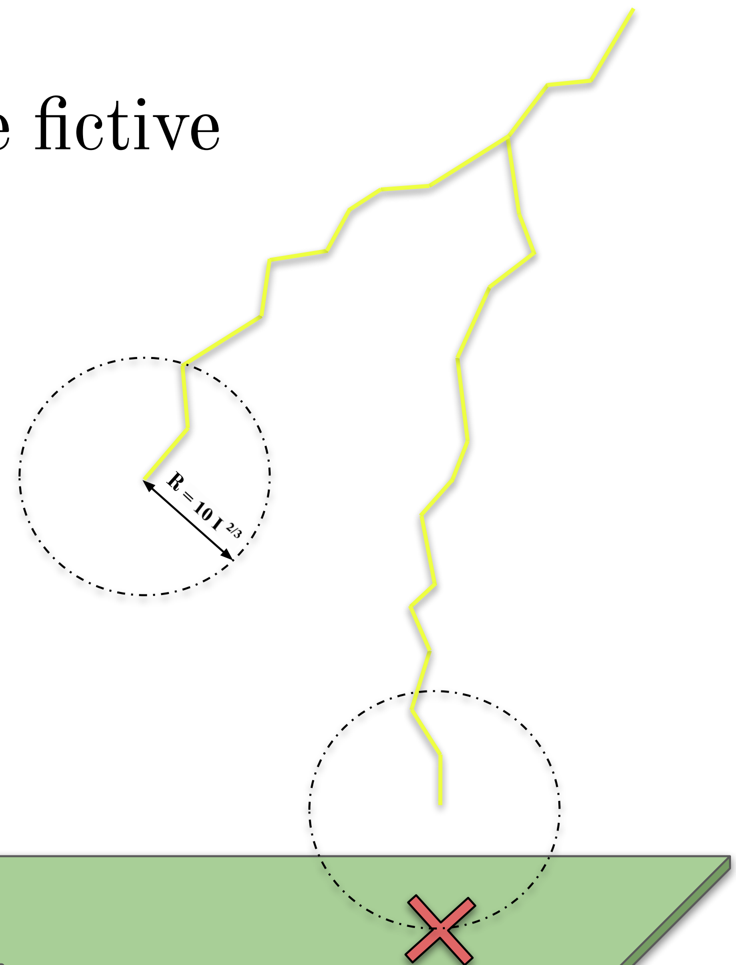
```python
def calcul_E(V,res_pix_par_m):
    Ny,Nx = V.shape
    Ex = numpy.zeros((Ny,Nx))
    Ey = numpy.zeros((Ny,Nx))
    norme_E = numpy.zeros((Ny,Nx))
    for i in range(1,Ny-1): # on évite les bords
        for j in range(1,Nx-1): # on évite les bords
            Ey[i,j] = -(V[i+1,j]-V[i-1,j])/(2.*res_pix_par_m)
            Ex[i,j] = -(V[i,j+1]-V[i,j-1])/(2.*res_pix_par_m)
            norme_E = numpy.sqrt(Ex**2+Ey**2)
    return Ex, Ey, norme_E
```

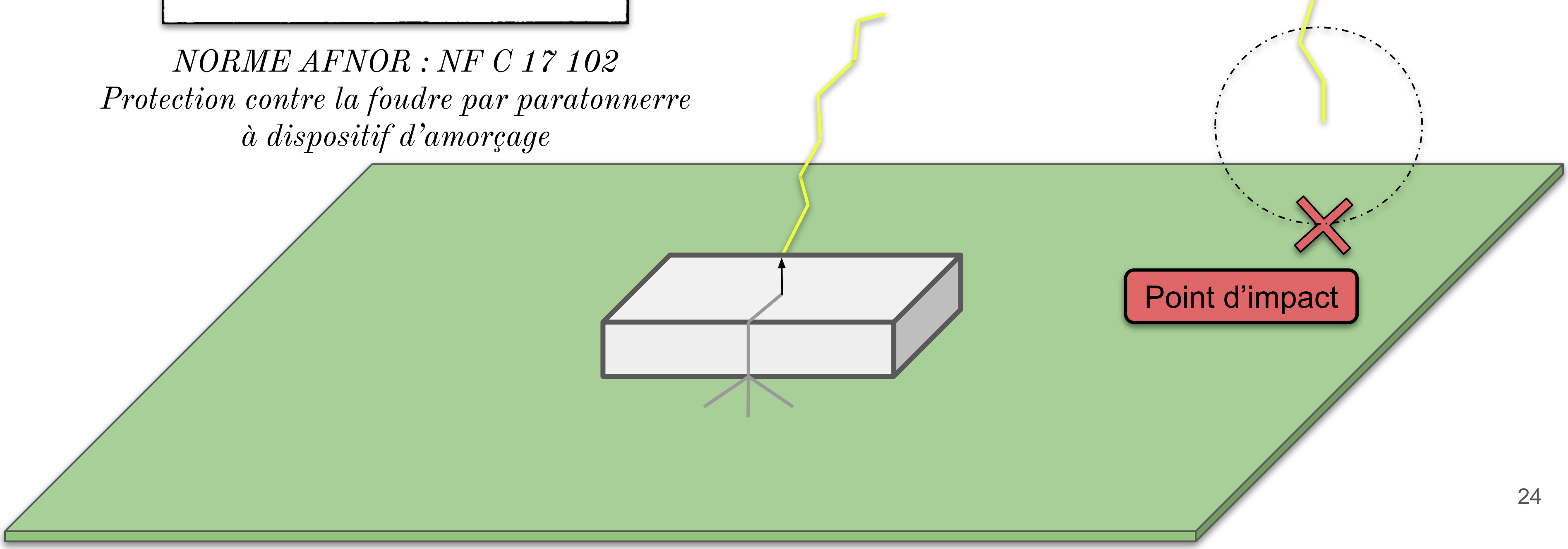
Annexe 3

Modèle de la sphère fictive



NORME AFNOR : NF C 17 102
Protection contre la foudre par paratonnerre
à dispositif d'amorçage





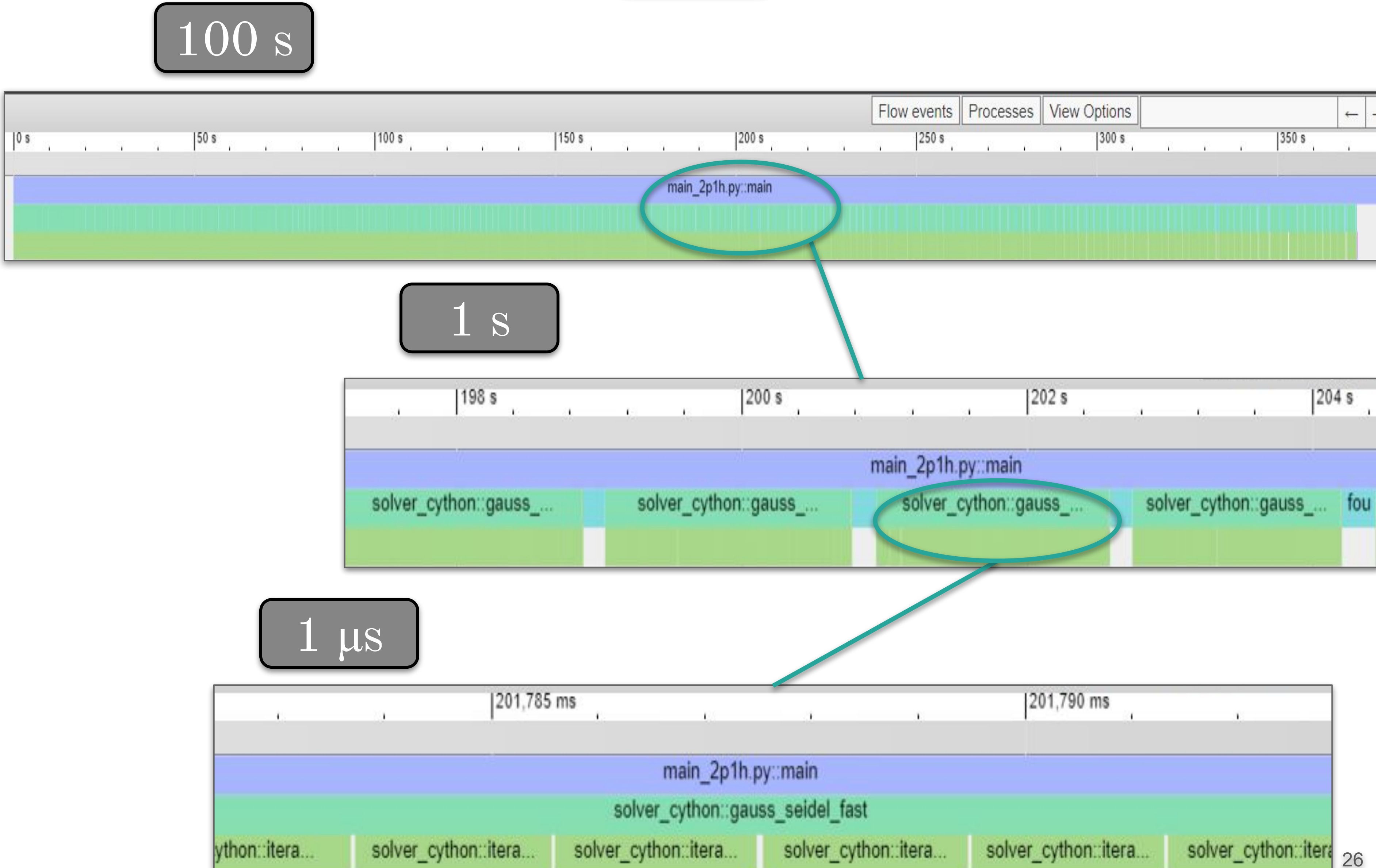
Annexe 4

Méthode de Gauss-Seidel

```
def iteration_gauss_seidel_fast(B,V):

    cdef numpy.ndarray[numpy.uint8_t,ndim=2 ] B_cython = B.copy()
    cdef numpy.ndarray[numpy.double_t,ndim=2] V_update = V.copy()
    cdef numpy.ndarray[numpy.double_t,ndim=2] V_copie = V.copy()
    cdef int i,j
    cdef float diff
    cdef float omega = 2/(1+numpy.pi/numpy.max(V.shape))
    cdef int status_vide = StatutB.VIDE

    for i in range(V_update.shape[0]):
        for j in range(V_update.shape[1]):
            if B_cython[i,j] != status_vide: # si [i,j] est un point du bord ne rien faire
                V_update[i,j] = V_copie[i,j]
            else:
                V_update[i,j] = (1-omega)*V_copie[i,j]+omega*(V_copie[i+1,j]+V_update[i-1,j] + V_copie[i,j+1]+V_update[i,j-1])/4

    diff = ecart(V_update,V_copie)
    return V_update, diff
def gauss_seidel_fast(B,V,eps):

    start = time.perf_counter()
    cdef numpy.ndarray[numpy.uint8_t,ndim=2] B_cython = B.copy()
    V, diff = iteration_gauss_seidel_fast(B_cython,V)
    while diff > eps:

        V, diff = iteration_gauss_seidel_fast(B_cython,V)

    end = time.perf_counter()
    print(f"{end - start} s")

    return numpy.asarray(V)
```

Ordre de grandeur :

Annexe 5



Annexe 6

Temps de calcul
de chaque itération

```
>>> (executing file "main_2p1h.py")
885.4335143000353 s
0.1765552000142634 s
0.1567390999989584 s
0.1695864003576875 s
0.1655382999451831 s
0.1605918999941736 s
0.1733449993748218 s
0.1606687000213385 s
0.1957518996976405 s
0.1619852000148967 s
0.167127700568807 s
0.1707308001864582 s
0.1609607000136748 s
0.197213500039652 s
0.1624908002307892 s
0.360855699824941 s
0.320059800054878 s
0.490554600227332 s
0.516825700623271 s
0.4953806009945184 s
0.1661823996023476 s
1.002148500061594 s
1.21922850003466 s
0.692104700214458 s
0.341609900023678 s
0.1886099995376915 s
0.3606977994431883 s
0.846493999394119 s
0.519099700318021 s
```

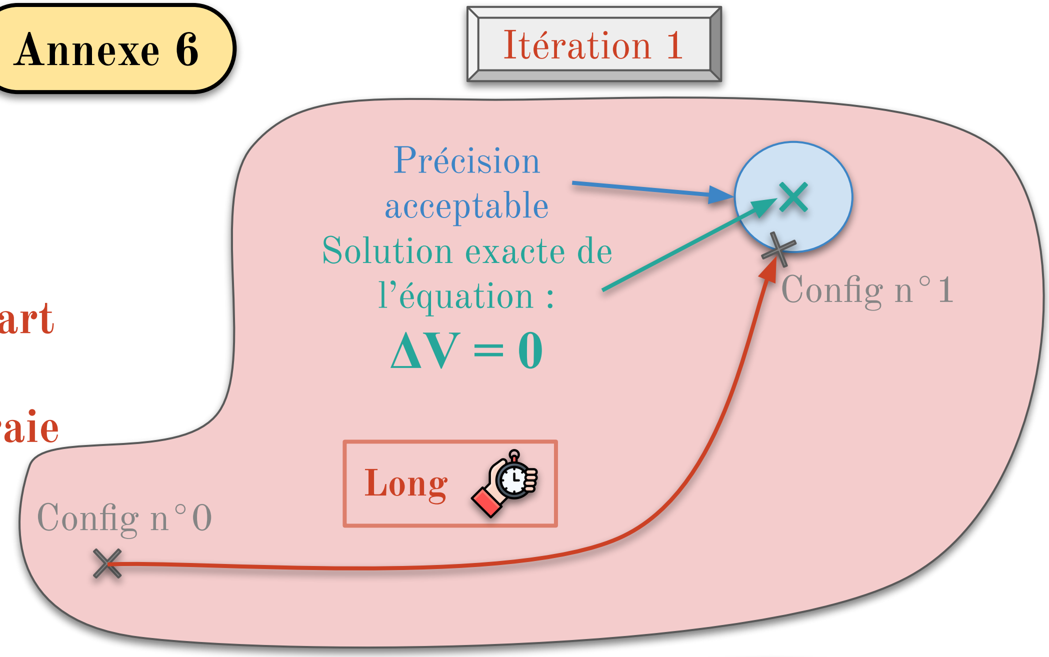
1

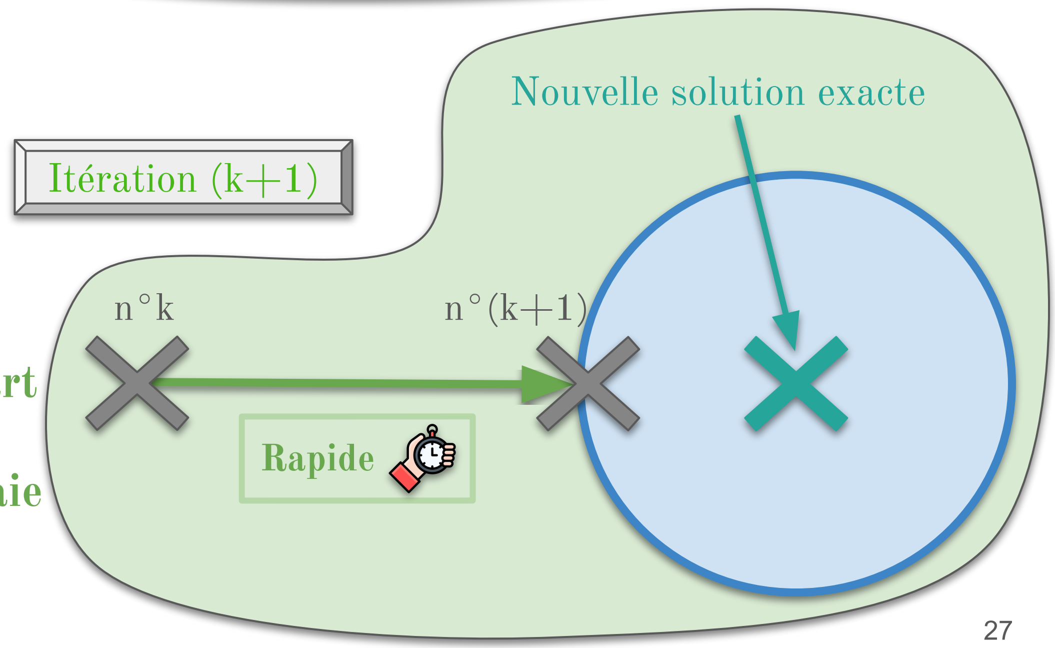
Configuration de **départ**
très loin
de la Configuration **vraie**

Nouvelles
Conditions Limites

2

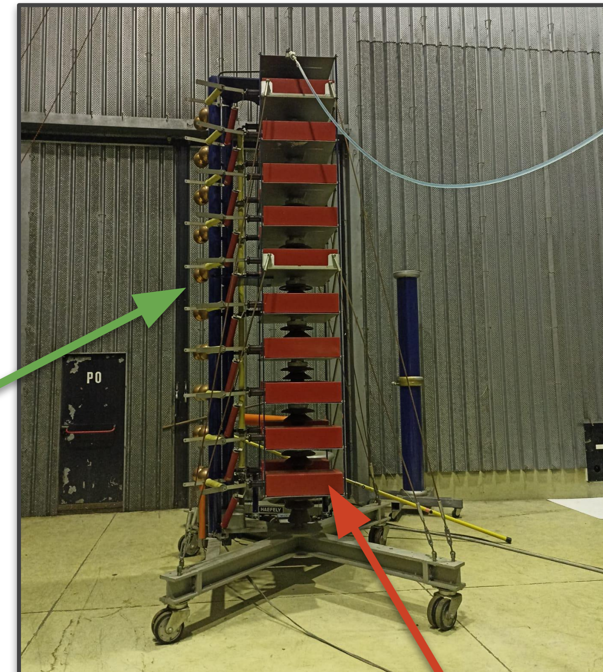
Configuration de **départ**
très proche
de la Configuration **vraie**



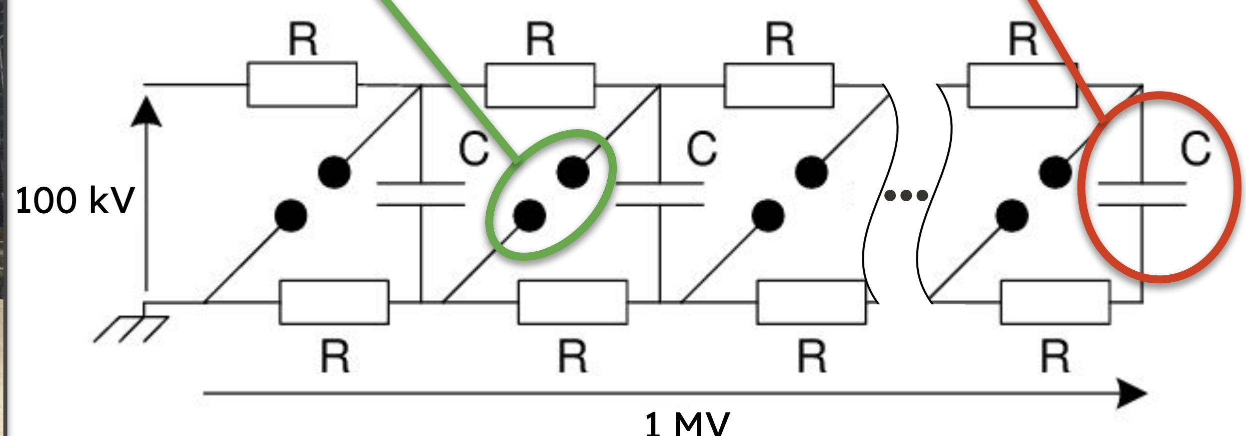


Annexe 7

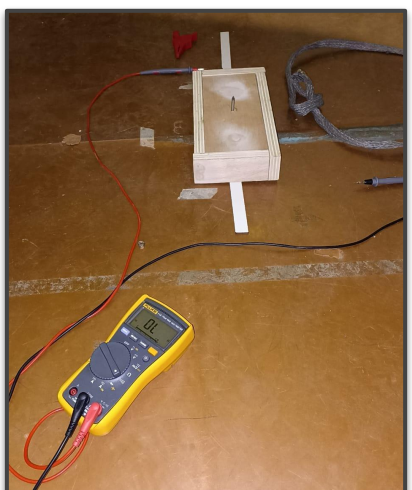
Générateur de Marx (1MV)



Éclateur

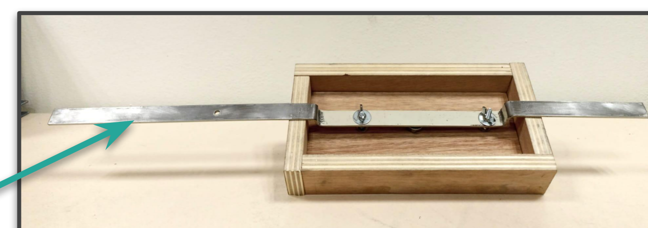


Annexe 8

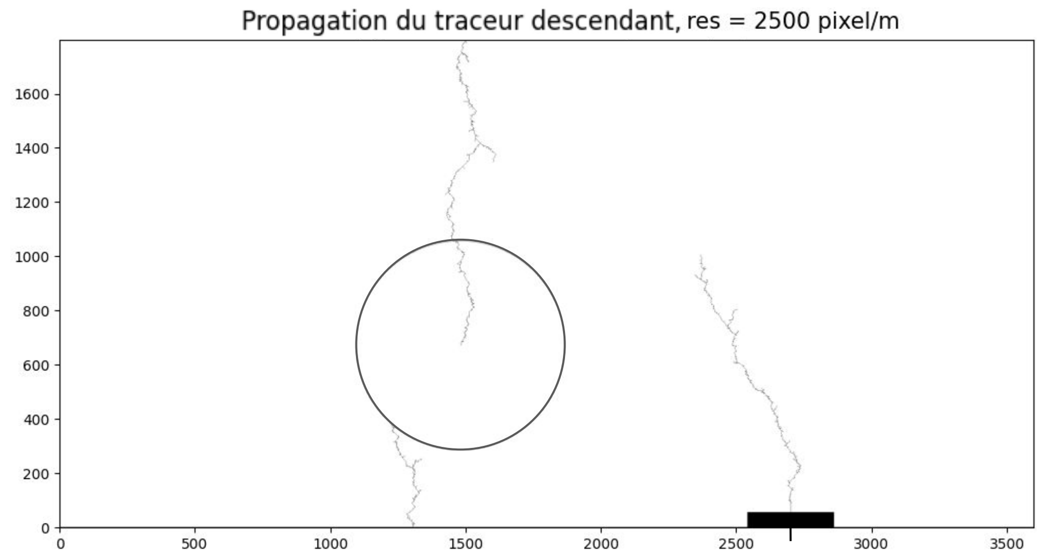


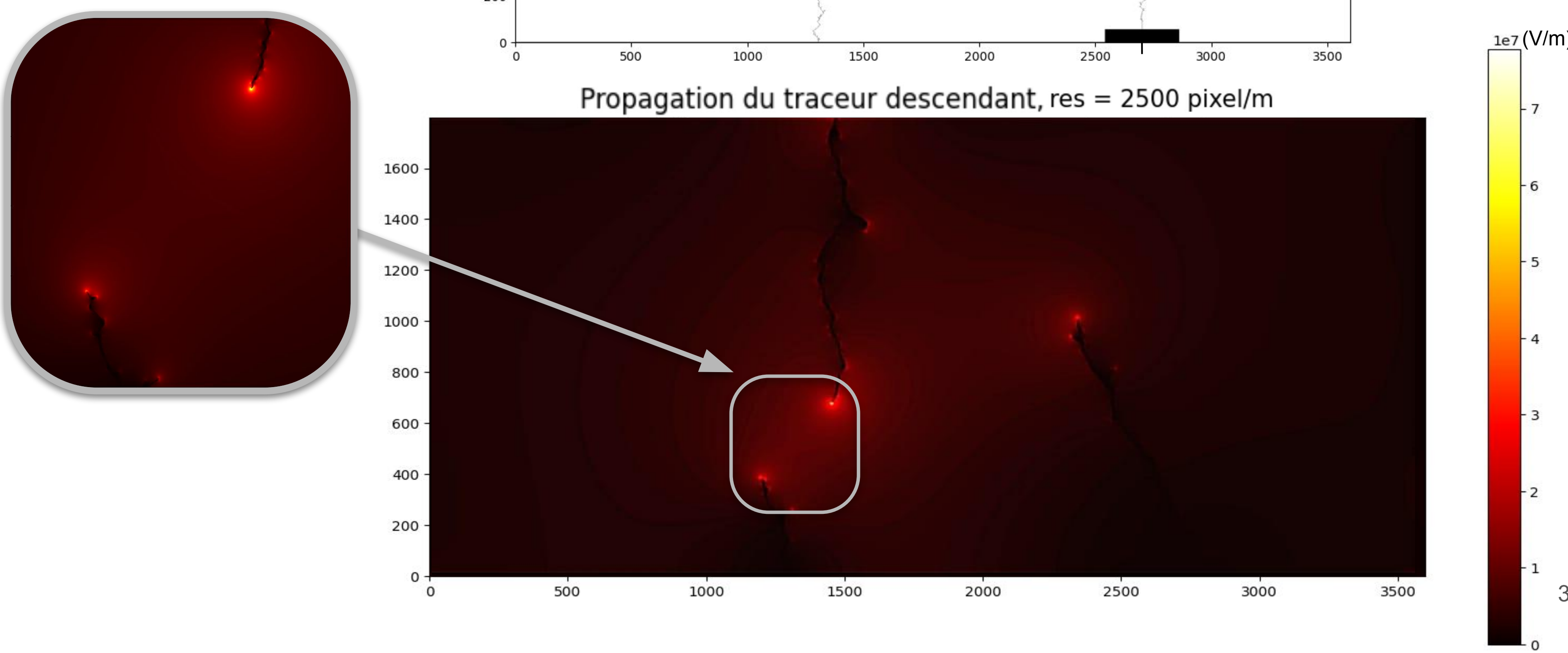
Mauvaise conductivité avec la terre

Ponçage



Annexe 9


Propagation du traceur descendant, res = 2500 pixel/m
1600
1400
1200
1000
800
600
400
200
0
0
500
1000
1500
2000
2500
3000
3500


Propagation du traceur descendant, res = 2500 pixel/m
1600
1400
1200
1000
800
600
400
200
0
0
500
1000
1500
2000
2500
3000
3500
1e7 (V/m)
7
6
5
4
3
2
1
0