IMT Atlantique

Électrostatique : modélisation de l'effet de pointe par méthode des éléments finis

BLOCH Maxime
LEHEL Eliaz

Mars 2025

Sommaire

Introduction

Nous nous proposons dans ce travail d'utiliser la Méthode des Eléments Finis (MEF) pour modéliser l'effet de pointe dans un champ électrostatique.[1]

1 Situation physique

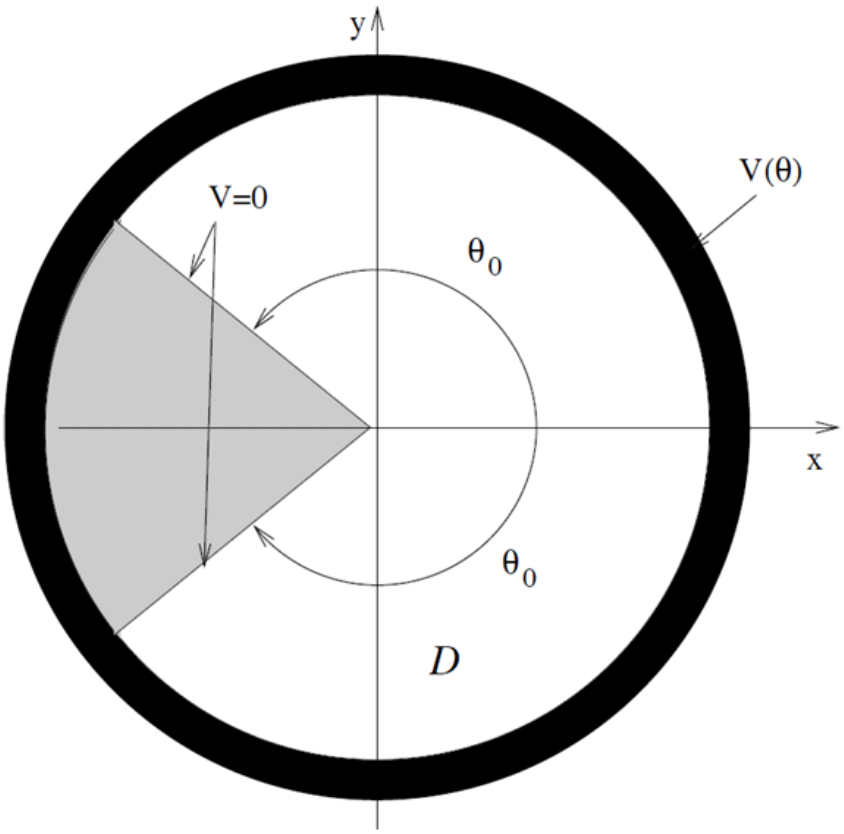

Figure 1: Modélisation de la pointe

La figure 1 représente une pointe (en gris) dont les bords sont maintenus à un potentiel $V = 0$ et les contours de la cavité (en noir) à un potentiel constant $V(\theta)$ non nul.

$$V(\theta) = 1 - \frac{\theta^2}{\theta_0^2}, \text{ avec } \theta_0 = \frac{3\pi}{4} \tag{1}$$

Dans ces conditions, le potentiel électrostatique V est solution de l'équation de Laplace :

$$\Delta V = 0 \tag{2}$$

2 Modélisation par éléments finis

2.1 Equation à résoudre

Nous cherchons à résoudre l'équation de Laplace sur un domaine Ω avec des conditions aux limites de Dirichlet sur le domaine Γ_D.

On pose u le potentiel électrostatique. Nous voulons résoudre:

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = 0 \tag{3}$$

Nous introduisons les fonctions v sur Ω le domaine de définition de u, notre potentiel, avec $v \in V = \{H^1(\Omega); v|_{\Gamma_D} = 0\}$, où $H^1(\Omega)$ est l'espace de Hilbert des fonctions dérivables au sens des distributions et carrés intégrables sur Ω.

Pour appliquer la méthode des éléments finis, nous utilisons Green pour écrire:

$$\int_\Omega \Delta u(x)v(x)dx = \int_\Omega \nabla u(x)\nabla v(x)dx - \int_\Gamma \partial_n u(s)v(s)ds \tag{4}$$

Par définition, les fonctions tests v sont nulles sur le bord Γ. Nous avons donc:

$$\int_\Omega \nabla u(x)\nabla v(x)dx = 0 \tag{5}$$

L'équation 5 est appelée la formulation faible de notre problème.

Ainsi, nous cherchons u telle que pour tout v dans l'espace de Hilbert $H^1(\Omega)$, l'équation 5 soit vérifiée.

On peut réécrire cette équation sous la forme suivante:

$$a(u, v) = 0 \tag{6}$$

On ajoute maintenant la condition de Dirichlet sur le bord Γ_D:

$$u(x) = g(x), \quad \forall x \in \Gamma_D \Leftrightarrow u - g \in V \tag{7}$$

Dès lors, nous pouvons approcher le problème avec la solution u^h dans un espace de dimension finie V^h.

On a donc:

$$a(u^h, v^h) = 0 \tag{8}$$

On décompose u^h et v^h sur la base des fonctions chapeau $\{B_i^h\}_{i=1}^{i=N^h}$

$$u^h = g^h + \sum_{i=1}^{N^h} x_i B_i^h \quad \text{et} \quad v^h = \sum_{i=1}^{N^h} y_i B_i^h \tag{9}$$

Avec x_i les inconnues du problème.

On revient au problème équivalent suivant:

$$a(u^h - g^h, v^h) = -a(g^h, v^h) \tag{10}$$

On pose maintenant les coefficients $A_{ij} = a(B_j^h, B_i^h)$. Ainsi, nous pouvons réécrire le problème sous forme matricielle:

$$\sum_{i,j} y_i A_{i,j} x_j \tag{11}$$

En posant $g_i = a(g^h, B_i^h)$, ce qui donne $a(g^h, v^h) = \sum_i y_i g_i$, on obtient le système linéaire suivant:

$$\sum_{i,j} y_i A_{i,j} x_j = -\sum_i y_i g_i \tag{12}$$

Que l'on peut réécrire sous forme matricielle en posant b le vecteur colonne des $-g_i$ et x le vecteur colonne des x_i:

$$y^T \cdot A \cdot x = y^T \cdot b \tag{13}$$

Ce qui équivaut à résoudre:

$$A \cdot x = b \tag{14}$$

2.2 Méthode des éléments finis

Nous devons maintenant assurer le raccord entre les différents éléments. Pour ce faire, nous considérons dorénavant les fonctions tests polynomiales $v \in X^h$.

Dès lors, on montre que chaque fonction v est définie par les valeurs qu'elle prend sur les nœuds du maillage et nous pouvons les écrire comme une combinaison linéaire des fonctions B_i, les fonctions chapeau généralisées.

Ainsi, nous assurons directement la continuité de v sur les éléments en ayant simplement les mêmes valeurs aux nœuds.

2.3 Maillage du domaine

Nous allons discrétiser le domaine en éléments finis et résoudre le problème de Laplace par la méthode des éléments finis. Dans un premier temps, nous avons choisi de considérer une géométrie simple, mais les calculs qui vont suivre demeurent généraux jusqu'à la section 2.7.

La figure 2 représente la discrétisation du domaine en éléments finis. Nous nous placerons en deux dimensions.

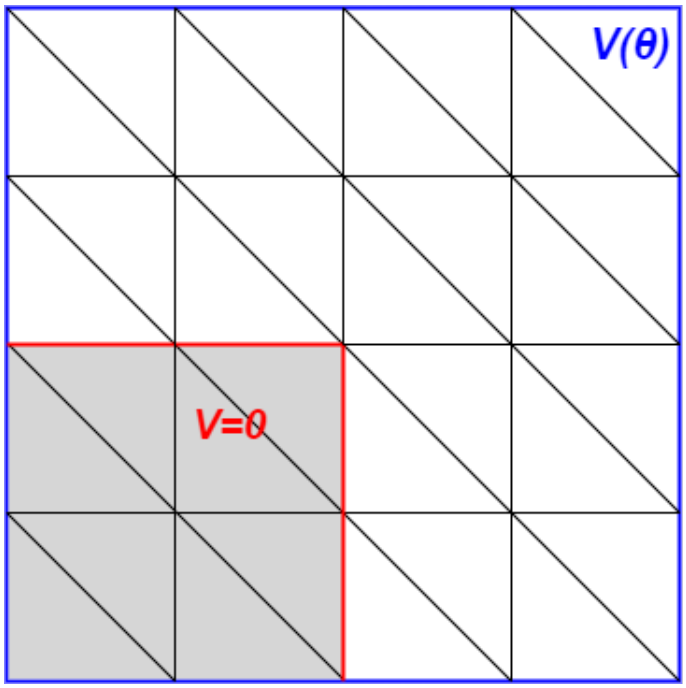

Figure 2: Discretisation du domaine

On rappelle qu'on doit résoudre le problème suivant:

$$\begin{cases} u^h \in X^h, \quad u^h - g^h \in V^h \\ a(u^h, v^h) = 0, \quad \forall v^h \in V^h \end{cases} \tag{15}$$

2.4 Décomposition des polynômes sur un élément

On réécrit le problème avec des sommes sur chaque élément $[e]$:

$$a(u^h, v^h) = \sum_{T^{[e]} \in \mathcal{T}^h} \int_{T^{[e]}} \sum_{i,j=1}^{N} a_{ij} \partial_j u^{[e]} \partial_i v^{[e]} dT^{[e]} \tag{16}$$

Chaque élément $T^{[e]}$ est défini par trois nœuds (Figure 3).

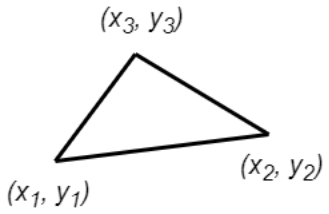

Figure 3: Un élément du maillage

Dans la suite, l'idée sera de décomposer les polynômes sur chaque élément et de réécrire le problème sous forme matricielle.

Pour ce faire, on note simplement $u^{[e]}$ et $v^{[e]}$ les fonctions u et v sur l'élément $T^{[e]}$, ainsi que $u_i^{[e]}$ et $v_i^{[e]}$ les valeurs de $u^{[e]}$ et $v^{[e]}$ au nœud i de l'élément $T^{[e]}$.

2.5 Matrice de raideur partielle

On introduit la matrice de raideur partielle $K^{[e]}$ telle que pour un élément [e]:

$$\int_{T^{[e]}} \sum_{i,j=1}^{2} a_{ij} \partial_j u^{[e]} \partial_i v^{[e]} dT^{[e]} = \left[v_1^{[e]}, v_2^{[e]}, v_3^{[e]}\right] \cdot K^{[e]} \cdot \left[u_1^{[e]}, u_2^{[e]}, u_3^{[e]}\right]^T \tag{17}$$

Si nous considérons un seul élément [e], nous pouvons chercher à expliciter les termes de la matrice de raideur $K^{[e]}$.

On réécrit le problème comme suit:

$$\left[v_1^{[e]}, v_2^{[e]}, v_3^{[e]}\right] K^{[e]} \left[u_1^{[e]}, u_2^{[e]}, u_3^{[e]}\right]^T = \int_{T^{[e]}} \left[\partial_x v^{[e]}, \partial_y v^{[e]}\right] A^{[e]} \left[\partial_x u^{[e]}, \partial_y u^{[e]}\right]^T dT^{[e]} \tag{18}$$

Avec $A^{[e]}$ la matrice des coefficients a_{ij}.

Nous allons chercher à réécrire ce problème sous forme matricielle afin d'être capable de calculer les coefficients de $K^{[e]}$. De cette manière, nous serons capable d'avoir accès au terme de l'intégrale de l'équation 17.

En sélectionnant des polynômes de degré 1 sur un élément $T^{[e]}$, l'intégrale se réduit à la mesure de l'aire du triangle élémentaire noté $|T^{[e]}|$ et à moyenner les valeurs de $A^{[e]}$:

$$\begin{bmatrix} v_1^{[e]} & v_2^{[e]} & v_3^{[e]} \end{bmatrix} K^{[e]} \begin{bmatrix} u_1^{[e]} \\ u_2^{[e]} \\ u_3^{[e]} \end{bmatrix} = |T^{[e]}| \cdot \begin{bmatrix} \partial_x v^{[e]} & \partial_y v^{[e]} \end{bmatrix} \overline{A^{[e]}} \begin{bmatrix} \partial_x u^{[e]} \\ \partial_y u^{[e]} \end{bmatrix} \tag{19}$$

2.6 Réécriture sous forme matricielle

Puisque $u, v \in V^h$, on sait qu'on peut les écrire sous forme de polynômes de degré 1 en deux dimensions:

$$u^{[e]}(x,y) = \alpha_0^{[e]} + \alpha_1^{[e]}x + \alpha_2^{[e]}y = [1\ x\ y]\left[\alpha_0^{[e]}\ \alpha_1^{[e]}\ \alpha_2^{[e]}\right]^T \tag{20}$$

$$v^{[e]}(x,y) = \beta_0^{[e]} + \beta_1^{[e]}x + \beta_2^{[e]}y = [1\ x\ y]\left[\beta_0^{[e]}\ \beta_1^{[e]}\ \beta_2^{[e]}\right]^T \tag{21}$$

Puisqu'on cherche la décomposition de u sur la base des fonctions chapeau, on exprime les coefficients de u par rapport à cette base grâce aux sommets de l'élément sous la forme suivante:

$$\begin{cases} \alpha_0^{[e]} + \alpha_1^{[e]}x_1 + \alpha_2^{[e]}y_1 = u_1^{[e]} \\ \alpha_0^{[e]} + \alpha_1^{[e]}x_2 + \alpha_2^{[e]}y_2 = u_2^{[e]} \\ \alpha_0^{[e]} + \alpha_1^{[e]}x_3 + \alpha_2^{[e]}y_3 = u_3^{[e]} \end{cases} \quad \Leftrightarrow \quad P^{[e]}\left[\alpha^{[e]}\right] = \left[u^{[e]}\right] \tag{22}$$

Avec la matrice $P^{[e]}$ définie par:

$$P^{[e]} = \begin{bmatrix} 1 & x_1 & y_1 \\ 1 & x_2 & y_2 \\ 1 & x_3 & y_3 \end{bmatrix} \tag{23}$$

Par ailleurs, en dérivant les formes générales de u et v, nous pouvons écrire:

$$\begin{bmatrix} \partial_x u^{[e]} \\ \partial_y u^{[e]} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \alpha_0^{[e]} \\ \alpha_1^{[e]} \\ \alpha_2^{[e]} \end{bmatrix} \tag{24}$$

$$\begin{bmatrix} \partial_x v^{[e]} \\ \partial_y v^{[e]} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \beta_0^{[e]} \\ \beta_1^{[e]} \\ \beta_2^{[e]} \end{bmatrix} \tag{25}$$

On pose alors la matrice $D = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$ qui nous permet de dériver simplement les fonctions u et v. On pose les vecteurs suivants:

$$\begin{bmatrix} \partial_x u^{[e]} \\ \partial_y u^{[e]} \end{bmatrix} = \left[\partial u^{[e]}\right] \text{ et } \begin{bmatrix} \partial_x v^{[e]} \\ \partial_y v^{[e]} \end{bmatrix} = \left[\partial v^{[e]}\right] \tag{26}$$

$$\begin{bmatrix} \beta_0^{[e]} \\ \beta_1^{[e]} \\ \beta_2^{[e]} \end{bmatrix} = \left[\beta^{[e]}\right] \text{ et } \begin{bmatrix} \alpha_0^{[e]} \\ \alpha_1^{[e]} \\ \alpha_2^{[e]} \end{bmatrix} = \left[\alpha^{[e]}\right] \tag{27}$$

$$\begin{bmatrix} u_1^{[e]} \\ u_2^{[e]} \\ u_3^{[e]} \end{bmatrix} = \left[u^{[e]}\right] \text{ et } \begin{bmatrix} v_1^{[e]} \\ v_2^{[e]} \\ v_3^{[e]} \end{bmatrix} = \left[v^{[e]}\right] \tag{28}$$

Avec ces nouvelles matrices, muni de l'équation 22 et de l'équation 19 que l'on réécrit comme suit:

$$\left[v^{[e]}\right]^T K^{[e]} \left[u^{[e]}\right] = |T^{[e]}| \left[\partial v^{[e]}\right] \overline{A^{[e]}} \left[\partial u^{[e]}\right] \tag{29}$$

On peut alors réécrire le problème sous forme matricielle:

$$[v]^T K^{[e]} [u] = |T^{[e]}| [\partial v]^T \overline{A^{[e]}} [\partial u] \tag{30}$$

$$\Leftrightarrow$$

$$\left(P^{[e]} \left[\beta^{[e]}\right]\right)^T K^{[e]} P^{[e]} \left[\alpha^{[e]}\right] = |T^{[e]}| \left(D \left[\beta^{[e]}\right]\right)^T \overline{A^{[e]}} D \left[\alpha^{[e]}\right] \tag{31}$$

En passant les termes à droite, en simplifiant les matrices $[\alpha]$ et $[\beta]$ par la droite et la gauche, et en écrivant $H^{[e]} = \left(P^{[e]}\right)^{-1}$, on obtient:

$$K^{[e]} = \left(H^{[e]}\right)^T D^T \left(|T|\overline{A^{[e]}}\right) D H^{[e]} \tag{32}$$

Notez qu'ici, D est de taille 2x3, $\overline{A^{[e]}}$ de taille 2x2 et $H^{[e]}$ de taille 3x3.

2.7 Expression des matrices dans le cadre du problème

En pratique, avec notre problème, nous savons que:

$$\overline{A^{[e]}} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{33}$$

Car, rappelons-le, notre formulation faible d'origine est

$$\int_\Omega \nabla u(x) \nabla v(x) dx = 0 \tag{34}$$

Nous avons maintenant besoin de connaître la matrice $H^{[e]}$ et la valeur de $|T|$.
Notons h la longueur d'un côté de l'élément [e]. Nous avons donc $h = x_2 - x_1 = y_2 - y_1$. Dès lors,

$$|T| = \frac{1}{2} h^2 \tag{35}$$

Par ailleurs, nous pouvons inverser la matrice $P^{[e]}$ pour obtenir la matrice $H^{[e]}$. Par calcul, on trouve:

$$H^{[e]} = \frac{1}{h^2} \begin{bmatrix} x_2 y_2 - x_1 y_1 & -h x_1 & -h y_1 \\ -h & h & 0 \\ -h & 0 & h \end{bmatrix} \tag{36}$$

Ces simplifications pourraient être faites dans notre code pour accélérer les calculs, mais en pratique, on constate que le temps de calcul est principalement occupé par la construction du maillage plutôt que par la résolution elle-même.

Nous avons donc choisi de garder dans notre solveur une forme générale du problème telle que présentée à la section précédente. (La forme de la matrice $A^{[e]}$ est cependant conservée égale à l'identité, car cette matrice dépend du problème physique et non de la géométrie.)

Ceci nous permet, comme nous le verrons dans la suite, d'adopter au besoin de nouvelles géométries.

2.8 Assemblage

Maintenant dotés des matrices de rigité élémentaires $K^{[e]}$ sur chaque élément, l'équation 16 nous permet de connaitre les coefficients de K, la matrice de rigidité globale du problème.

En effet, nous avons la correspondance suivante:

$$\begin{bmatrix} v_1^h & \cdots & v_N^h \end{bmatrix} K \begin{bmatrix} u_1^h \\ \vdots \\ u_N^h \end{bmatrix} = \sum_{T^{[e]} \in \mathcal{T}^h} \begin{bmatrix} v_1^{[e]} & v_2^{[e]} & v_3^{[e]} \end{bmatrix} K^{[e]} \begin{bmatrix} u_1^{[e]} \\ u_2^{[e]} \\ u_3^{[e]} \end{bmatrix} \tag{37}$$

Ce que nous dit cette équation, c'est que pour chaque nœud, la valeur de K est la somme de toutes les valeurs de $K^{[e]}$ correspondantes à ce nœud. En effet, chaque élément [e] est défini par trois nœuds, et chaque nœud est partagé par trois éléments, donc une matrice $K^{[e]}$ contient plusieurs fois les contributions de chaque nœud.

2.9 Conditions limites

Il ne reste plus qu'à définir les conditions limites de notre système.

Si aucune contrainte n'existait, le système à résoudre serait:

$$K \left[u^h\right]^T = [0] \tag{38}$$

Mais certaines contributions doivent être retirées de la matrice K. En effet, il ne faut pas calculer les variations du nœud i lorsque celui-ci est sur le bord, donc les contributions des autres nœud à la valeur en i doivent être éliminées. Pour ce faire, il suffit de fixer à 0 la colonne i de la matrice K (sauf la valeur en (i, i), ainsi le nœud i sera le seul à contribuer à sa propre valeur, et on élimine la variation) et définir une valeur fixe dans le second membre à la place de 0.

Admettons dans l'exemple suivant que le nœud i se trouve au bord et est fixée à la valeur U_i Voici les modifications à apporter à la matrice:

$$\begin{bmatrix} K_{1,1} & \cdots & K_{i,1} = 0 & \cdots & K_{1,N} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \vdots & \ddots & K_{i,i} = 1 & \ddots & \vdots \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ K_{N,1} & \cdots & K_{i,N} = 0 & \cdots & K_{N,N} \end{bmatrix} \begin{bmatrix} u_1^h \\ \vdots \\ u_N^h \end{bmatrix} = \begin{bmatrix} 0 \\ \vdots \\ U_i \\ \vdots \\ 0 \end{bmatrix} \tag{39}$$

En notant le membre de droite F et K^* la nouvelle matrice de rigidité prenant en compte les conditions aux limites, il en découle simplement un système linéaire à résoudre:

$$K^* \begin{bmatrix} u_1^h \\ \vdots \\ u_N^h \end{bmatrix} = F \tag{40}$$

Et c'est bien ce calcul qui sera réalisé dans notre code.

3 Présentation du code

L'architecture du code, la présentation des classes et de leurs méthodes sont déjà décrits extensivement dans le `README` du projet ainsi que sur le wiki associé. (Lien vers GitHub)

3.1 Solveur

Dans cette section est détaillé le fonctionnement mathématique du solveur et comment il implémente les équations présentées précédemment.

Il y a quatre méthodes principales dans le solveur:

- `solve_mesh()`
- `compute_rigidity_matrix()`
- `apply_boundary_conditions()`
- `compute_element_stiffness_matrix()`

La première résout le problème en appelant les autres. La deuxième calcule la matrice de raideur K nécessaire à la résolution du problème, la troisième applique les conditions limites à la matrice de raideur pour obtenir K^* et F, et la dernière calcule une matrice élémentaire $K^{[e]}$.

Bien évidemment, elles ne sont pas appelées dans cet ordre. Voici comment se déroule l'algorithme:

1. Pour chaque éléments du maillage, on calcule la matrice de raideur élémentaire $K^{[e]}$.
2. Pour chaque noeud, on ajoute ses contributions en "ajoutant" $K^{[e]}$ à l'endroit approprié de K.
3. On applique les conditions limites à K pour obtenir K^* et on détermine F.
4. On résout le système linéaire $K^* \cdot u = F$.

Une version de pseudo-code simplifiée et commentée de chaque méthode est présentée au listing 1 ci-après.

Listing 1: Structure simplifiée du solveur

```
def solve_mesh():
    # On cree la matrice avec toutes les contributions
    K = compute_rigidity_matrix()
    F = np.zeros(mesh.size())
    # On s'assure de retirer les contributions des noeuds a valeur fixe
    apply_boundary_conditions(K, F, mesh)
    # On appel numpy pour resoudre le systeme lineaire
    u = np.linalg.solve(K, F)
    return u

def compute_rigidity_matrix():
    K = np.zeros((n_nodes, n_nodes))
    for element in mesh.elements():
        # Pour chaque element, on rempli Ke
        Ke = compute_element_stiffness_matrix(element)
        for i, node_i in element.nodes:
            for j, node_j in element.nodes:
                # On ajoute le coeff de Ke correspondant a la
                # contribution entre deux noeuds a la matrice K
                K[node_i.index, node_j.index] += Ke[i, j]
    return K

def compute_element_stiffness_matrix(element):
    x = [node.x for node in element]
    y = [node.y for node in element]

    Pe = np.array([[1, x[0], y[0]],
                   [1, x[1], y[1]],
                   [1, x[2], y[2]]])
    Ae = np.array([[1, 0], [0, 1]])
    He = np.linalg.inv(Pe) # Inversement de la matrice Pe
    Te = 0.5 * np.abs(np.linalg.det(Pe)) # Calcul de l'aire du triangle
    D = np.array([[0, 1, 0],
                  [0, 0, 1]]) # Matrice de derivation
    DT = np.array([[0, 0],
                   [1, 0],
                   [0, 1]]) # Transpose de D

    Ke = np.transpose(He) @ DT @ Ae @ D @ He * Te # Solution elementaire
    return Ke

def apply_boundary_conditions(K):
    for i in range(mesh.size()):
        if mesh[i].value is not None:
            # Si on a une condition limite au noeud i
            K[i, :] = 0
            # Alors on retire toutes les contributions des autres noeuds
            K[i, i] = 1
            # On devient le seul noeud a participer a notre valeur
            F[i] = mesh[i].value
            # Et la solution pour ce noeud sera nous-meme
```

3.2 Maillage disponibles et architecture

Le code a été pensé selon une programmation orientée objet avec le langage de programmation `Python`.

Cette architecture a l'avantage de permettre une configuration simple de nouveaux maillages, et c'est la raison pour laquelle nous avons proposé deux options: un maillage carré comme présenté dans la section 2.3, et un maillage circulaire plus proche de la situation étudiée d'après le schéma figure 1.

Chacun de ces maillages implémente une classe abstraite `Mesh` qui définit les propriétés que doit avoir le maillage afin d'être traité par le solveur.

Puisque le solveur est codé de manière générale, un utilisateur pourrait définir son propre maillage grâce aux classe de notre code et le résoudre simplement.

Nous faisons également remarquer que la création d'un maillage circulaire n'est pas optimisée et peut rapidement prendre du temps. Cependant, le solveur est plutôt rapide grâce à l'utilisation du package `numpy` qui permet une résolution optimisée de systèmes matriciels.

3.3 Forme des résultats

Le code offre la possibilité d'afficher de nombreux graphiques différents, que ce soit pour assister la construction de maillage ou pour sauvegarder des résultats. La figure 4 montre un exemple d'affichage de résultats et de maillage pour un nombre de noeuds faible.

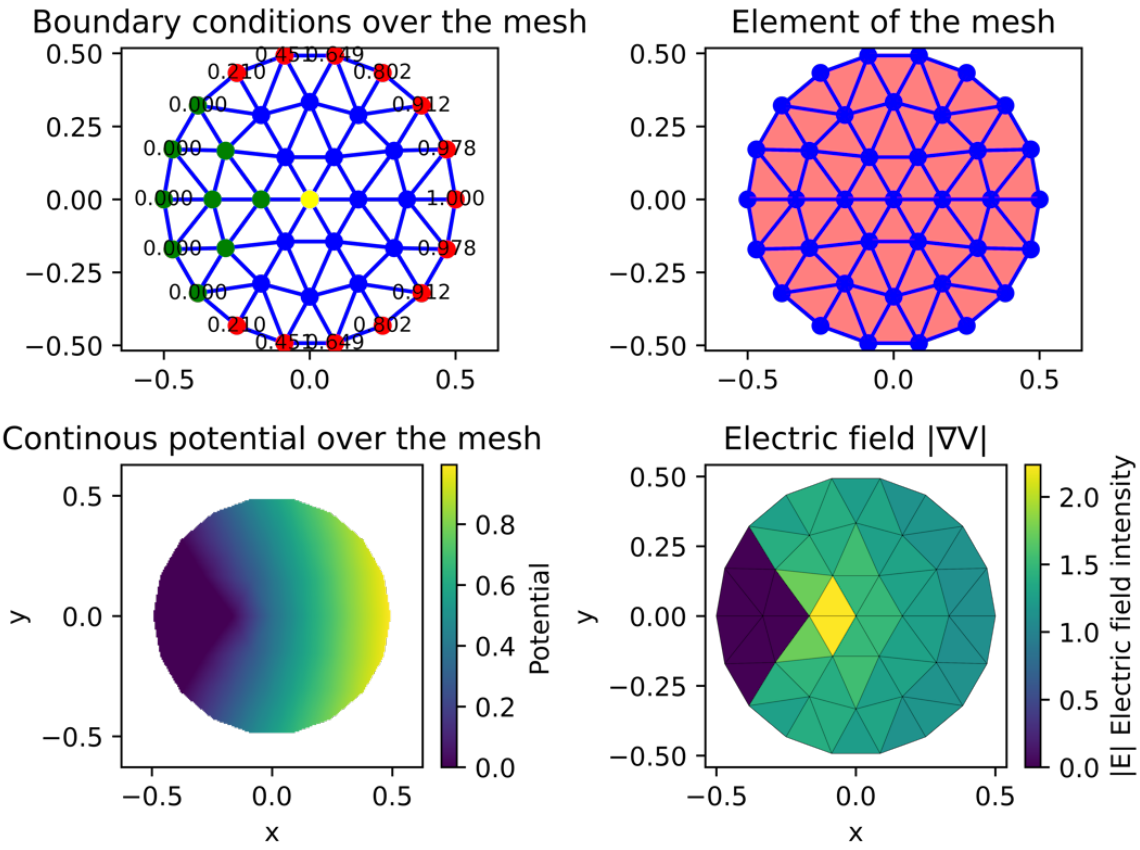

Figure 4: Exemple d'affichage disponible dans le code

4 Comparaison des résultats avec la théorie

Pour comparer notre code avec la théorie, nous proposonq de revenir à la situation initiale d'une pointe dans une géométrie circulaire.

Nous utiliserons donc le maillage circulaire proposé par notre code et nous comparerons les résultats avec la solution analytique.

4.1 Solution analytique

Nous avons posé la condition limite suivante:

$$V_1(\theta) = 1 - \frac{\theta^2}{\theta_0^2} \tag{41}$$

En notant V_1 la fonction de potentiel en $r = 1$.

Au vu de la géométrie, on cherche une décomposition de la fonction potentiel en coordonnées sphériques avec des solutions connus de l'équation de Laplace, c'est-à-dire les harmoniques sphériques (car leurs laplaciens sont nuls).

Ces fonctions notées $\Phi_n(r, \theta)$ s'écrivent:

$$\Phi_n(r, \theta) = r^{\frac{(2n+1)\pi}{2\theta_0}} cos\left(\frac{(2n+1)\pi}{2\theta_0}\theta\right) \tag{42}$$

Ces fonctions sont solutions de l'équation de Laplace et forment une base de l'espace des fonctions harmoniques sphériques, tout en vérifiant les propriétés qui nous intéressent, à savoir qu'elles s'annulent bien en $\theta = \pm\theta_0$.

Dès lors, il existe une suite (a_n) telle que:

$$V(r, \theta) = \sum_{n=0}^{+\infty} a_n \Phi_n(r, \theta) \tag{43}$$

Pour calculer les coefficients a_n, nous allons exploiter la condition limite en $r = 1$.

$$1 - \frac{\theta^2}{\theta_0^2} = \sum_{n=0}^{+\infty} a_n cos\left(\frac{(2n+1)\pi}{2\theta_0}\theta\right) \tag{44}$$

On reconnait ici la décomposition en série de Fourier de la fonction V_1. On peut donc d'après Fourier exprimer les coefficients a_n comme suit:

$$a_n = \frac{1}{\theta_0} \int_{-\theta_0}^{\theta_0} \left(1 - \frac{\theta^2}{\theta_0^2}\right) cos\left(\frac{(2n+1)\pi}{2\theta_0}\theta\right) d\theta \tag{45}$$

Nous allons résoudre cette équation pour tout n en réalisant deux intégrations par partie.

$$
\begin{aligned}
a_n &= \frac{1}{\theta_0}\int_{-\theta_0}^{\theta_0}\left(1-\frac{\theta^2}{\theta_0^2}\right)cos\left(\frac{(2n+1)\pi}{2\theta_0}\theta\right)d\theta \\
&= \frac{1}{\theta_0}\left(\left[\frac{-2\theta_0}{(2n+1)\pi}sin\left(\frac{(2n+1)\pi}{2\theta_0}\right)\frac{2\theta}{\theta_0^2}\right]_{-\theta_0}^{\theta_0}+\int_{-\theta_0}^{\theta_0}\frac{2\theta_0}{(2n+1)\pi}sin\left(\frac{(2n+1)\pi}{2\theta_0}\right)\frac{2\theta}{\theta_0^2}\right) \\
&= \frac{2}{(2n+1)\pi}\left(0+\left[\frac{-2\theta_0}{(2n+1)\pi}cos\left(\frac{(2n+1)\pi}{2\theta_0}\right)\frac{2\theta}{\theta_0^2}\right]_{-\theta_0}^{\theta_0}+\int_{-\theta_0}^{\theta_0}\frac{2\theta_0}{(2n+1)\pi}cos\left(\frac{(2n+1)\pi}{2\theta_0}\right)\frac{2}{\theta_0^2}\right) \\
&= \frac{2}{\left((2n+1)\pi\right)^2}\left(0+\left[\frac{2\theta_0}{(2n+1)\pi}sin\left(\frac{(2n+1)\pi}{2\theta_0}\right)\frac{4}{\theta_0}\right]_{-\theta_0}^{\theta_0}\right) \\
&= \frac{16}{\left((2n+1)\pi\right)^3}\left(sin\left(\frac{(2n+1)\pi}{2}\right)-sin\left(\frac{-(2n+1)\pi}{2}\right)\right) \\
&= (-1)^n\frac{32}{\left((2n+1)\pi\right)^3}
\end{aligned}
$$

Ainsi, on peut écrire la forme générale du potentiel V:

$$
V(r,\theta)=\sum_{n=0}^{+\infty}(-1)^n\frac{32}{\left((2n+1)\pi\right)^3}r^{\frac{(2n+1)\pi}{2\theta_0}}cos\left(\frac{(2n+1)\pi}{2\theta_0}\theta\right) \tag{46}
$$

Au vu de la forme des coefficients, on peut considérer que les dix premiers termes de la série suffisent à obtenir une très bonne approximation de la solution.

4.2 Résultats

Nous avons donc résolu le problème présenté en section 1 avec notre code et comparé les résultats avec un calcul analytique à l'ordre 10 en utilisant la formule l'équation 46. Les résultats sont visible figure 5.

4.2.1 Comparaison du potentiel

La figure 5 montre la comparaison entre les solutions numérique (à gauche) et analytique (à droite).

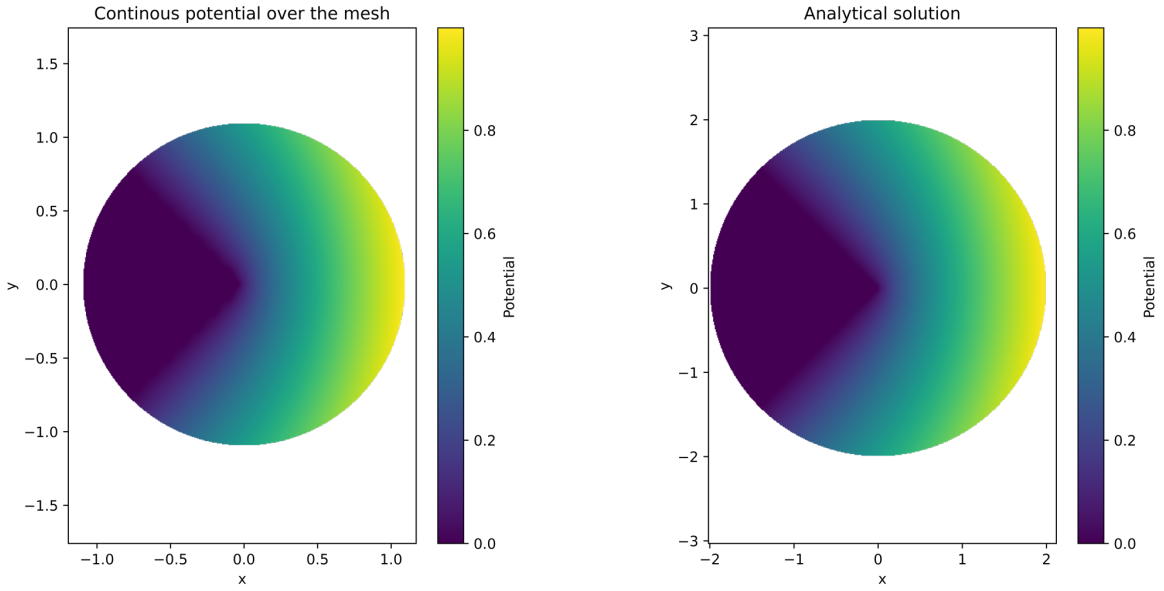

Figure 5: Comparaison des résultats analytiques et numériques

Les formes des potentiels se ressemblent beaucoup.

Pour emphaser les différences, on trace l'erreur relative du code par rapport aux valeurs théoriques en chaque point du plan (figure 6) en utilisant la simple différence relative $(V_{th} - V_{calc})/V_{th}$ (en valeur absolue).

Ces différences sont surtout visibles aux alentours de la pointe et le long de celle-ci. Cela s'explique très simplement par la forme du maillage dans la simulation. En effet, lors de la définition de la géométrie, le maillage de la pointe n'est pas contraint de suivre le bord de celle-ci.

On aurait pu ajouter cette contrainte, mais nous avons préféré garder une construction de maillage simple et automatique, qui introduit donc des discontinuités aux arrêtes de la pointe.

Malgré tout, les différences de potentiels restent satisfaisante sur l'ensemble du maillage. L'échelle a néanmoins été forcée à 100%, des différences plus élevées apparaissent au niveau de la pointe elle-même, principalement à cause de la géométrie approximée.

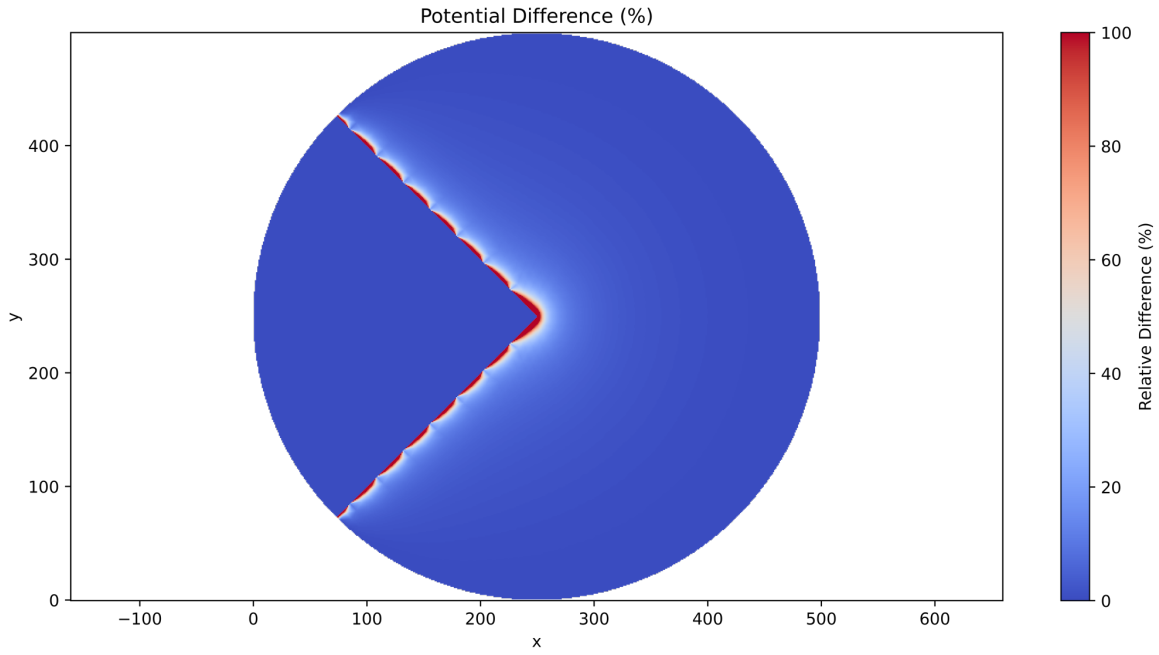

Figure 6: Différence entre les solutions analytiques et numériques

4.2.2 Comparaison du champ électrique

Nous allons maintenant comparer les champs électriques calculés par notre code et par la théorie.

Précisons déjà que lors d'une résolution par MEF avec des polynômes de degré 1, le champ électrique est nécessairement constant au sein de chaque élément triangulaire. En effet, si chaque équation qui décrit le potentiel dans un élément est de degré 1 par rapport à chaque coordonnée, alors le champ électrique est de degré 0, c'est-à-dire constant, dans chaque élément.

La figure 7 montre la comparaison entre les champs électriques numériques (à gauche) et analytiques (à droite).

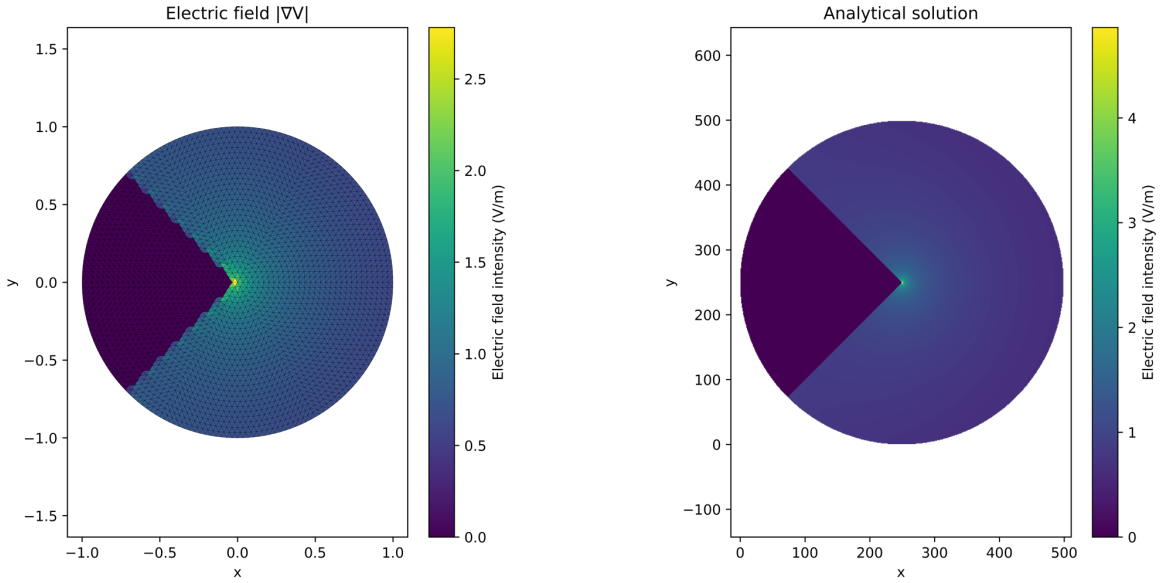

Figure 7: Comparaison des champs électriques analytiques et numériques

La figure 8 montre la différence relative entre les champs électriques numériques et analytiques $(E_{th} - E_{calc})/E_{th}$ (en valeur absolue). Une fois de plus, les différences sont minimes expecté sur la pointe, principalement à cause de la géométrie du maillage.

On observe également les mêmes discontinuités sur le bord de la pointe que pour le potentiel.

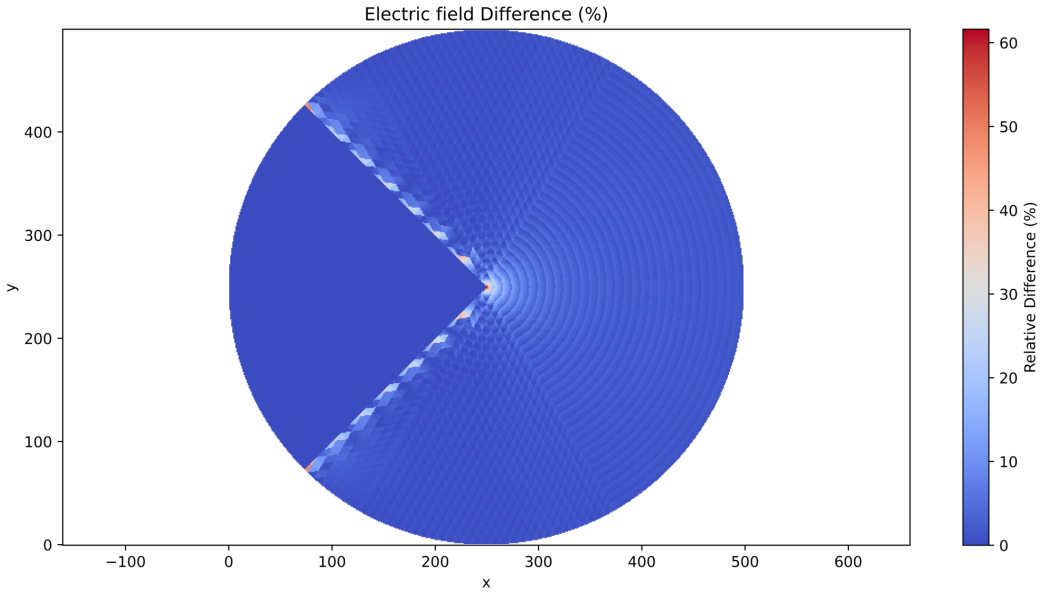

Figure 8: Différence relative entre les champs électriques analytiques et numériques

4.2.3 Influence du nombre de nœuds

Lorsqu'on augmente le nombre de nœuds, on s'attend à ce que les écarts diminuent. Et c'est bien ce qu'on observe en pratique.

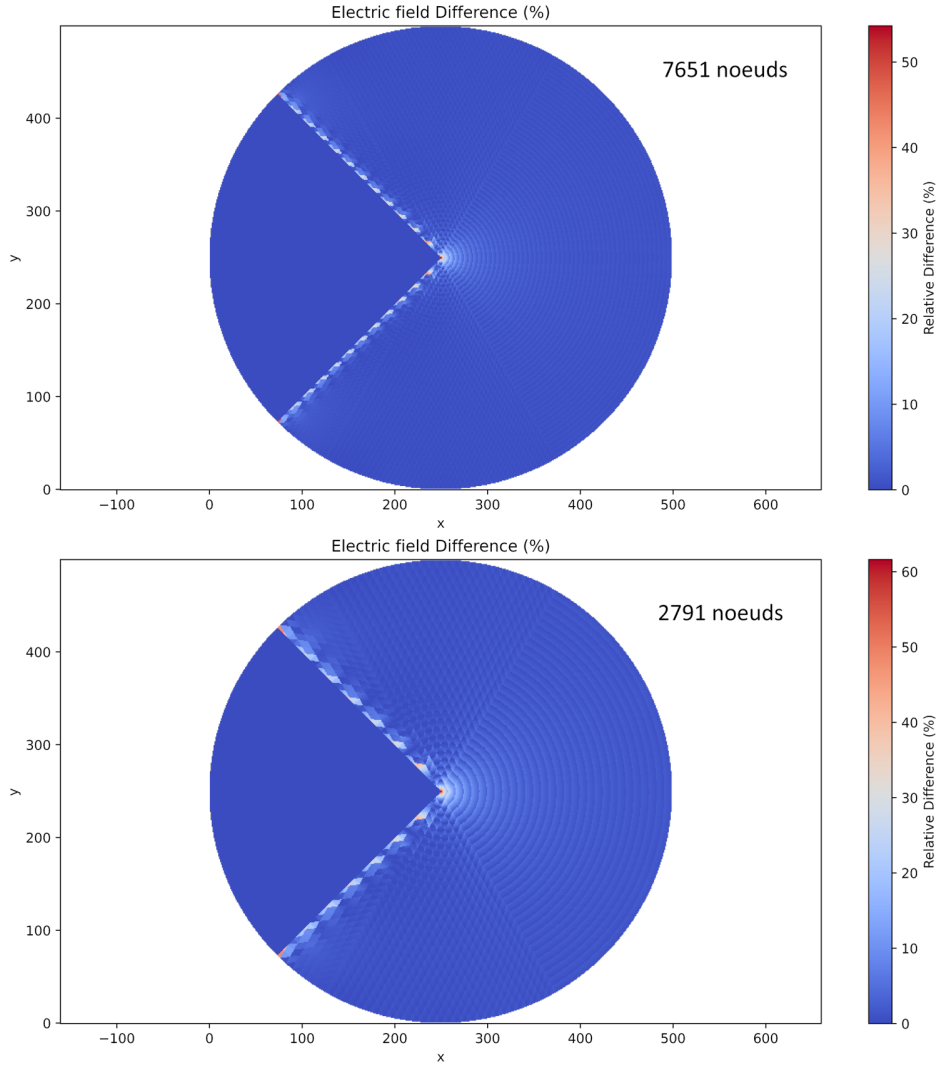

Figure 9: Illustration de l'impact du nombre de nœuds

La figure 9 montre la différence relative du champ électrique pour deux nombres de nœuds différents, 7651 et 2791 nœuds. On remarque que les différences visibles sur l'ensemble du maillage diminuent, de même que la valeur maximale de la différence, ce qui signifie que nous nous approchons d'une meilleure solution.

On peut aussi voir que la valeur de champ à la pointe se rapproche de la valeur calculée (non affiché dans ce document).

Cette tendance devrait bien entendu être étudiée pour un plus grand nombre d'échantillon pour être validée.

Un maillage adaptatif pourrait également aider, en offrant une meilleure précision dans les zones à forte variation, mais une fois encore, nous avons préféré rester simple dans la réalisation du maillage.

5 Conclusion

D'après les résultats de la section 4.2, nous pouvons dire que le code a un comportement satisfaisant vis-à-vis de la situation simulée. De plus, il renvoie des valeurs cohérentes de potentiel et de champ électrique.

Ces résultats peuvent être améliorés, notamment en définissant une géométrie plus adaptée avec un maillage "lisse" au bord de la pointe. L'architecture actuelle du code permet en tout cas une implémentation facilitée de nouvelles géométries.

Références

[1] Pol-Bernard Gossiaux. *Introduction à la méthode des éléments finis*. UV Outils du Numérique, DEMIN* - IMT Atlantique. 2025.