Національний технічний університет України
"Київський політехнічний інститут імені Ігоря Сікорського"
Факультет Електроніки
Кафедра мікроелектроніки

ЗВІТ
Про виконання розрахункової роботи №1
з дисципліни: «Теорія поля»

Виконавець:
Студент 3-го курсу _____ А. С. Мнацаканов
(підпис)

Превірила: _____ Т. А. Саурова
(підпис)

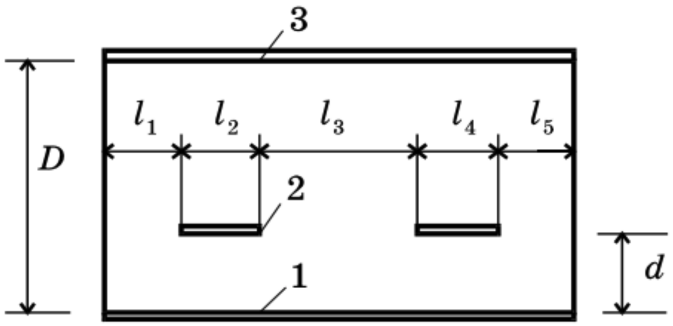
2020

ЗАВДАННЯ

1. Розрахувати розподіл потенціалу в міжелектродному просторі польового транзистора (варіант конструкції вибирається за передостанньою цифрою номера залікової книжки) з точністю до 0,01 В. Номер варіанту обирається за останньою цифрою номера залікової книжки.

2. Побудувати картини поля за допомогою еквіпотенціалей і векторів напруженості електричного поля. Побудувати косокутну проекцію потенціального рельєфу U(x,y) $= -e \cdot V(x, y)$

варіант 1



$l_1 = l2 = l_3/2 = l_4 = l_5 = 1,0$ мкм, d = 1,0 мкм,
D = 3,0 мкм 1 – витік, 2 – затвор, 3 – стік

вар. №	4
$-V_{3B}, B$	0.5
V_{CB}, B	3

РОЗРАХУНКИ

Одним з найбільш розповсюджених методів розрахунку розподілу потенціалу в міжелектродному просторі польового транзистора є мотод скінченних різниць. В основу методу покладена заміна похідних в рівнянії Лапласа малими приростами. Розглянемо його для плоского поля V (x, y). Надаючи по черзі аргументам x і y деякої неперервної функції V (x, y) малого приросту з кроком h, можна приблизно (з точністю до величин другого порядка малості відносно h^2) замінити приватні похідні функціїотношеніямі різниць рис.1 :

$$\frac{\partial V}{\partial x} \approx \begin{cases} [V(x,y) - V(x-h,y)] \,/\, h & -\text{лів оруч}, \\ [V(x+h,y) - V(x,y)] \,/\, h & -\text{прав оруч}, \end{cases}$$

$$\frac{\partial V}{\partial y} \approx \begin{cases} [V(x,y) - V(x,y-h)] \,/\, h & -\text{унизу}, \\ [V(x,y+h) - V(x,y)] \,/\, h & -\text{угорі}\,. \end{cases}$$

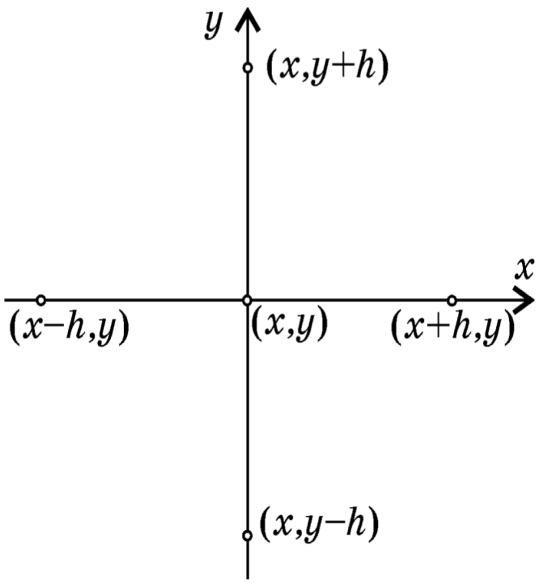


Рис. 1: П'ятиточкова схема для обчислення похідних через кінцеві прирости і розрахунку потенціалів..

Представимо всю площину транзистора як дискретний масив точок розмірністю 25x13.

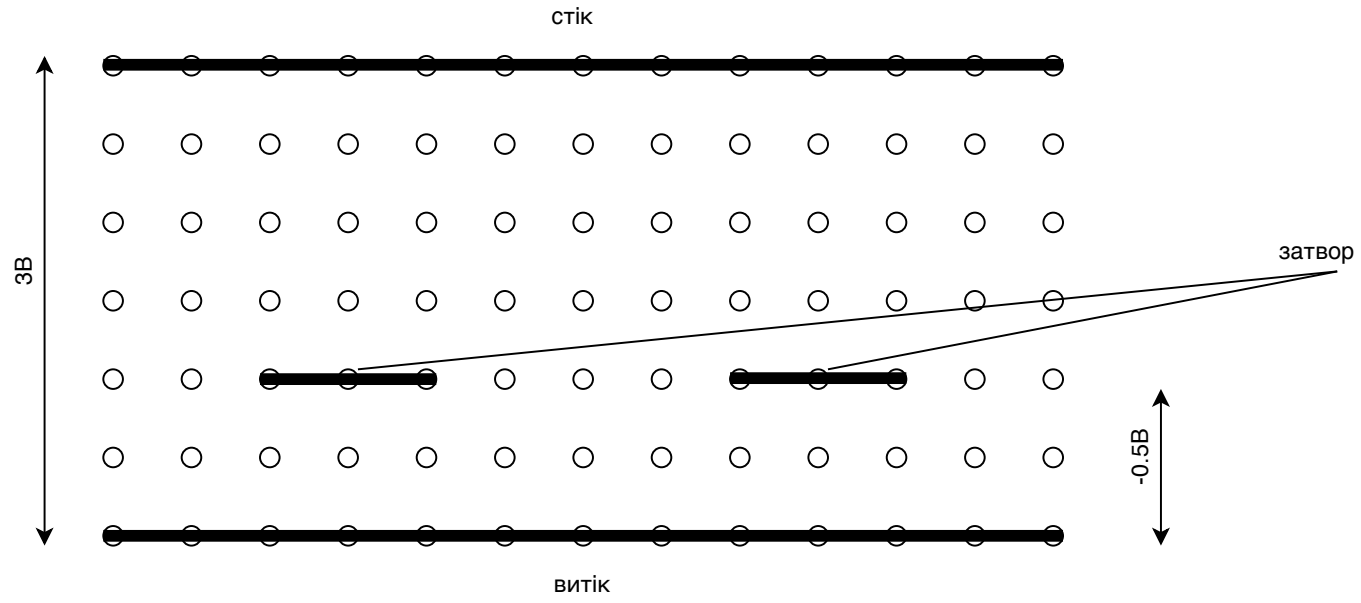


Рис. 2: Дисткретизація простору інтегрування за допомогою сітки з постійним кроком 0.5 мкм[1]

При цьому двомірне рівняння Лапласа

$$\frac{\partial^2 V}{\partial x^2} + \frac{\partial^2 V}{\partial y^2} = 0 \tag{1}$$

наближено заміняється наступним алгебраічним рівнянням:

$$V(x+h,y) + V(x-h,y) + V(x,y+h) + V(x,y-h) - 4V(x,y) = 0 \tag{2}$$

[1]Зауважимо, що для подальших розрахунків крок було зменшено вдвічі, тобто 0.25 мкм.

Спочатку програмно виведемо табл. 1 нульового наближення, а також табл.2 першого наближення.

Табл. 1: Нульове наближення

3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	-0.5	-0.5	-0.5	-0.5	-0.5	0.0	0.0	0.0	0.0	0.0	0.0	-0.5	-0.5	-0.5	-0.5	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

Табл. 2: Перше наближення

3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0
0.0	0.75	0.94	0.98	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
0.0	0.19	0.28	0.32	0.33	0.33	0.33	0.33	0.33	0.33	0.33	0.33	0.33	0.33	0.33	0.33	0.33	0.33	0.33	0.33	0.33	0.33	0.33
0.0	0.05	0.08	0.1	0.11	0.11	0.11	0.11	0.11	0.11	0.11	0.11	0.11	0.11	0.11	0.11	0.11	0.11	0.11	0.11	0.11	0.11	0.11
0.0	0.01	0.02	0.03	0.03	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.04
0.0	0.0	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	-0.12	-0.16	-0.16	-0.16	-0.17	-0.04	-0.01	-0.0	0.0	0.0	0.0	-0.12	-0.15	-0.16	-0.17	-0.04	-0.01	-0.0	-0.0
0.0	0.0	0.0	-0.12	-0.5	-0.5	-0.5	-0.5	-0.5	-0.14	-0.04	-0.01	-0.0	-0.0	-0.12	-0.5	-0.5	-0.5	-0.5	-0.14	-0.04	-0.01	-0.01
0.0	0.0	0.0	-0.03	-0.13	-0.16	-0.16	-0.17	-0.17	-0.08	-0.03	-0.01	-0.0	-0.0	-0.03	-0.13	-0.16	-0.16	-0.17	-0.08	-0.03	-0.01	-0.01
0.0	0.0	0.0	-0.01	-0.04	-0.05	-0.05	-0.05	-0.06	-0.03	-0.02	-0.01	-0.0	-0.0	-0.01	-0.04	-0.05	-0.05	-0.06	-0.03	-0.02	-0.01	-0.01
0.0	0.0	0.0	-0.0	-0.01	-0.01	-0.02	-0.02	-0.02	-0.01	-0.01	-0.0	-0.0	-0.0	-0.0	-0.01	-0.01	-0.02	-0.02	-0.01	-0.01	-0.0	-0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

Результат після 39 ітерацій:

Табл. 3: Розподіл потенціалу в польовому транзисторі

3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0
2.55	2.56	2.56	2.56	2.56	2.56	2.56	2.56	2.56	2.57	2.57	2.57	2.58	2.57	2.57	2.56	2.56	2.56	2.56	2.56	2.57	2.57	2.57
2.12	2.12	2.13	2.12	2.12	2.12	2.12	2.12	2.13	2.14	2.15	2.16	2.16	2.16	2.15	2.14	2.13	2.13	2.13	2.14	2.14	2.15	2.15
1.7	1.71	1.71	1.7	1.7	1.69	1.69	1.7	1.71	1.73	1.75	1.76	1.76	1.76	1.73	1.72	1.71	1.7	1.71	1.72	1.73	1.74	1.74
1.31	1.32	1.32	1.3	1.28	1.27	1.27	1.28	1.3	1.33	1.36	1.38	1.39	1.38	1.34	1.31	1.29	1.28	1.3	1.32	1.34	1.35	1.35
0.95	0.96	0.95	0.92	0.88	0.86	0.85	0.86	0.9	0.95	1.0	1.03	1.04	1.03	0.96	0.91	0.87	0.86	0.89	0.94	0.97	1.0	1.0
0.64	0.64	0.61	0.55	0.47	0.43	0.42	0.44	0.49	0.58	0.67	0.72	0.74	0.72	0.59	0.5	0.45	0.43	0.49	0.57	0.64	0.67	0.67
0.37	0.38	0.33	0.21	0.04	-0.01	-0.03	-0.01	0.05	0.24	0.38	0.45	0.48	0.46	0.25	0.06	-0.0	-0.02	0.05	0.22	0.34	0.4	0.4
0.18	0.18	0.11	-0.07	-0.5	-0.5	-0.5	-0.5	-0.5	-0.05	0.16	0.25	0.28	0.26	-0.04	-0.5	-0.5	-0.5	-0.5	-0.06	0.12	0.2	0.2
0.06	0.07	0.01	-0.1	-0.27	-0.33	-0.34	-0.32	-0.26	-0.08	0.05	0.13	0.15	0.13	-0.08	-0.26	-0.32	-0.34	-0.27	-0.1	0.02	0.08	0.08
0.01	0.01	-0.02	-0.08	-0.15	-0.19	-0.21	-0.19	-0.15	-0.06	0.01	0.06	0.07	0.06	-0.06	-0.15	-0.19	-0.21	-0.15	-0.08	-0.01	0.02	0.02
-0.0	-0.0	-0.02	-0.04	-0.07	-0.09	-0.1	-0.09	-0.07	-0.03	-0.0	0.02	0.03	0.02	-0.03	-0.07	-0.09	-0.1	-0.07	-0.04	-0.01	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

Візуалізація цих данних було наведено на Рис. 3, Рис. 4.

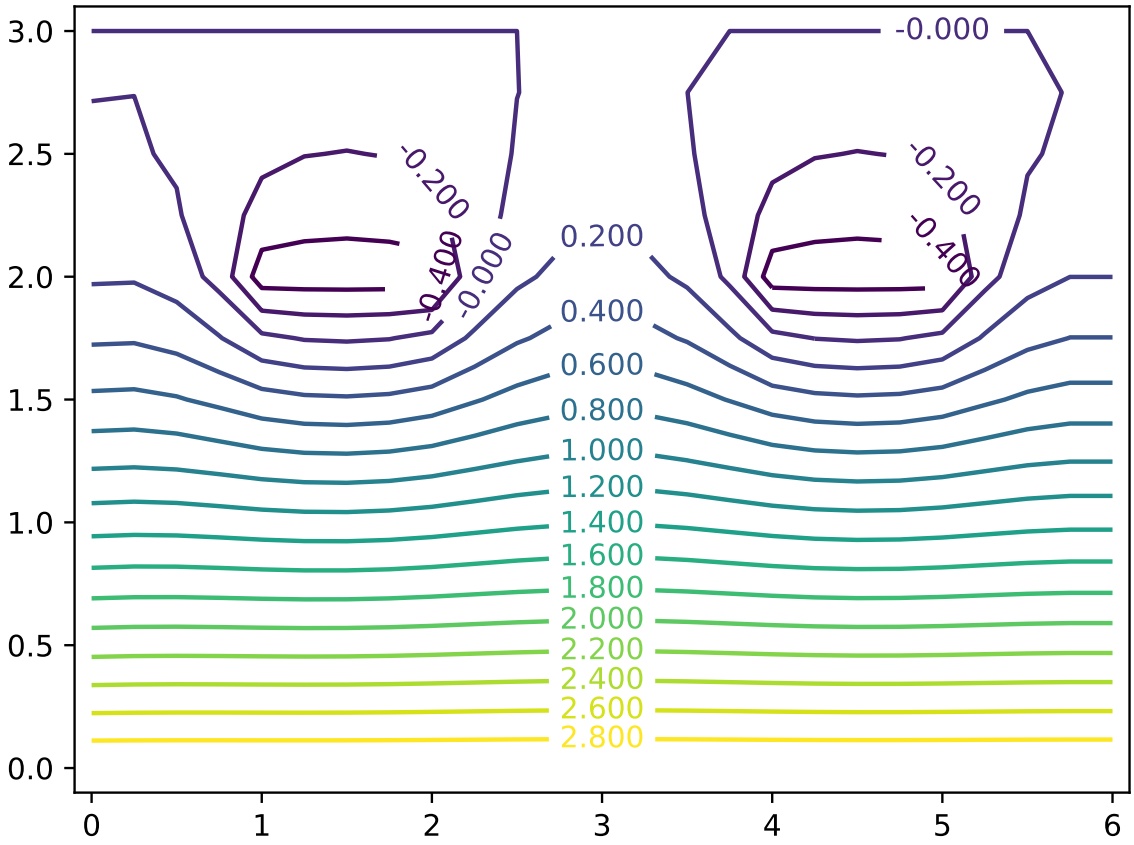


Рис. 3: Картина поля за допомогою еквіпотенціалей.

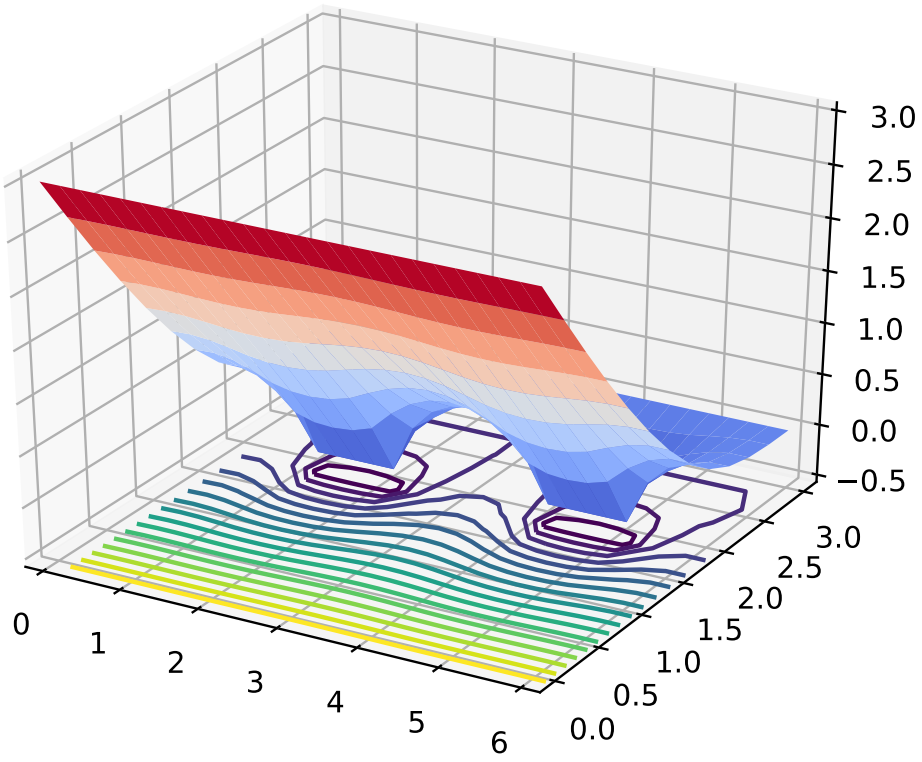


Рис. 4: Косокутна проекція потенціального рельєфу U(x,y).

Розрахунок напруженості електричного поля провадиться за співвідношення $E_x{=}-\dfrac{\partial V}{\partial x}$, $E_y{=}\dfrac{\partial V}{\partial y}$що наближено обчислюються через кінцеві різниці. Використовуючи значення похідних зліва і справа, можна знайти середнє арифметичне

$$E_{x\ i,j} = \frac{V_{(i-1),j} + V_{(i+1),j}}{2} \tag{3}$$

$$E_{y\ i,j} = \frac{V_{i,(j-1)} + V_{i,(j+1)}}{2} \tag{4}$$

Наступним кроком я графічно зобразив веркори напруженості електричного поля в нашому польовому транзисторі (рис. 5). Дивлячись на цей рисунок можна чітко побачити що електричне поле яке прямує із стоку до затвору набагато сильніше ніж те що йде йому на зустріч, про що свідчать довжини стрілок.

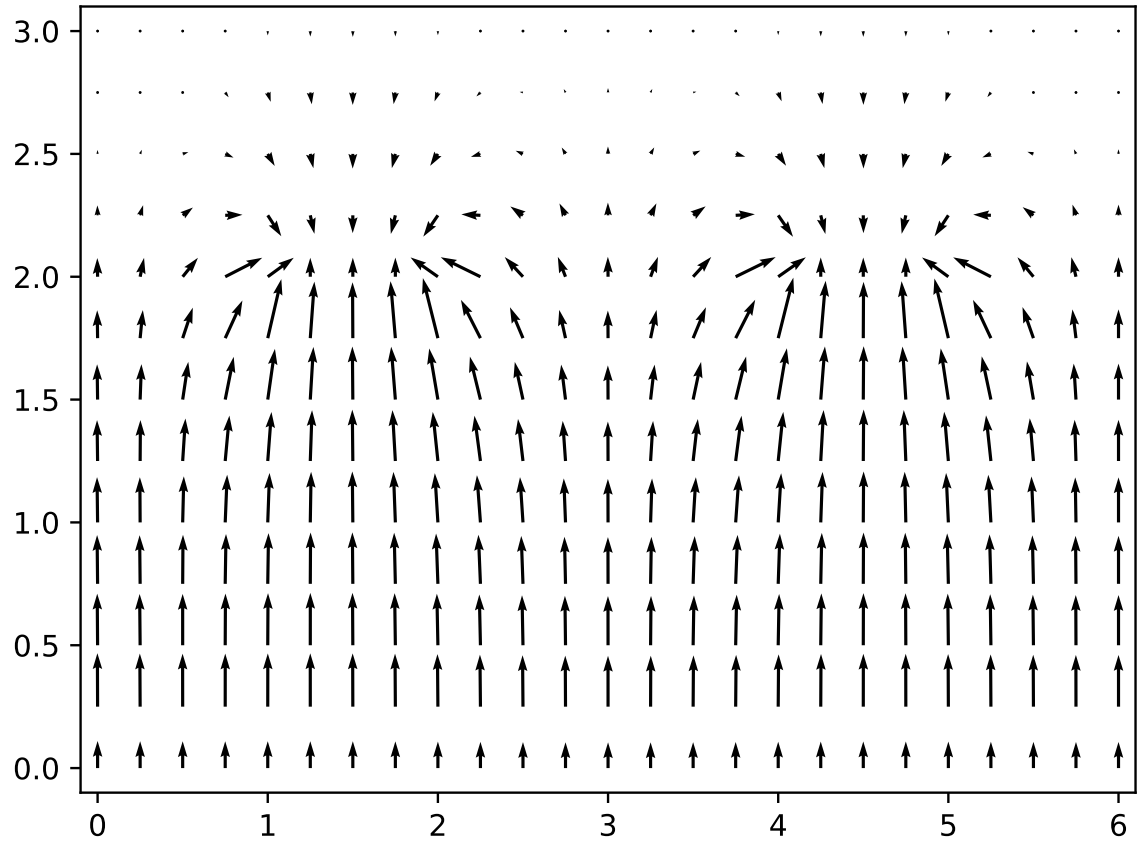


Рис. 5: Картина поля за допомогою векторів напруженості електричного поля.

Останнім кроком я побудував косокутку проекцію потенціального рельєфу $U(x, y) = -eV(x, y)$ рис. 6

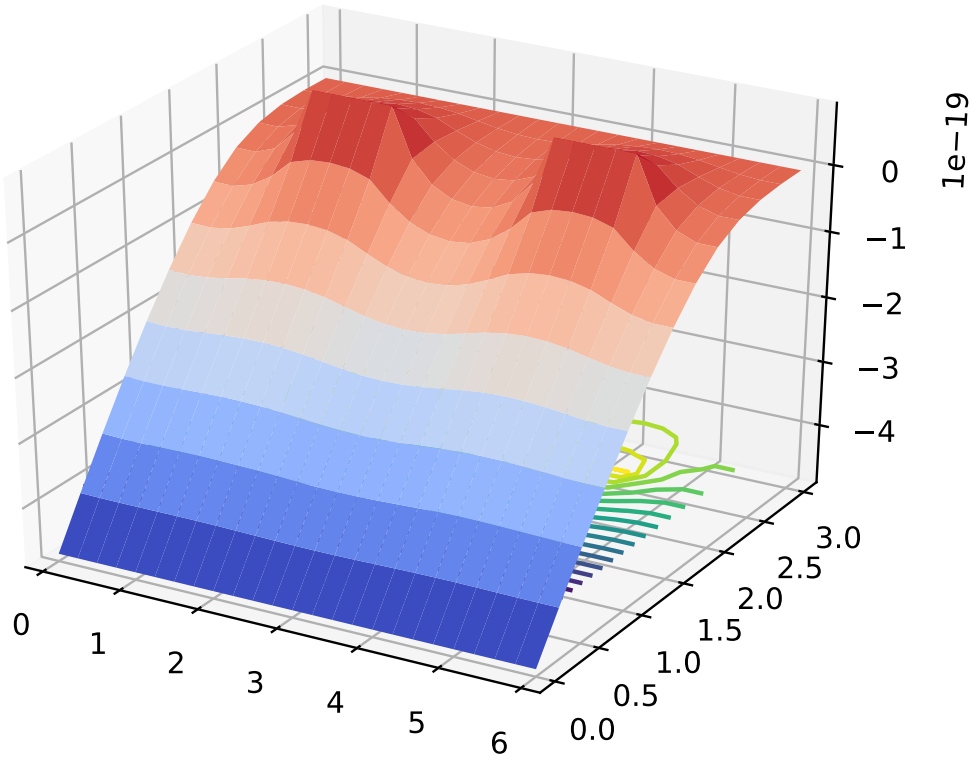


Рис. 6: Косокутна проекція потенціального рельєфу U(x,y) = -eV(x,y).

Висновок:

У цій розрахунковій роботі я засвоїв принципи та методи чисельного інтегрування рівнянь Лапласа, а також способи графічно зображувати результати. Для визначення потенціалів дискретної сітки був застосований метод скінченних різниць, а також використані граничні умови Діріхле та Неймана. За допомогою ітераційного методу була досягнута задана в завданні точність 0.01В також були побудовані картини розподілу поля за допомогою еквіпотенціальних ліній та векторів напруженості електричного поля, а також була побудована косокутна проекція потенціального рельєфу. З отриманого графіка еквіпотенціалей помітні 2 ділянки, де є викривлення їх форми. Тут різко змінюється потенціал, оскільки це є затвор, на якому потенціал є постійним (в моєму випадку -0.5). При використанні векторів напруженості електричного поля, отримана картина покаже нам розподіл електричного поля в міжелектродному просторі. На отриманій же просторовій сітці, можемо побачити як вузли із розрахованим значенням потенціалу з'єднуються між собою прямими лініями, які і є еквіпотенціальними лініями.

ДОДАТОК

Для обрахунку всіх значень, графіків та поверхонь я написав програму на мові Python:

```
#!/usr/bin/env python
#coding=utf8
from pylab import figure, axes, pie, title, show
import matplotlib.cm as cm
import matplotlib
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
import math


accuracy = 0.01
e = -1.6*10**(-19)
V1 = 3.
V2 = -0.5
ah = 0.25
xmax = 6
ymax = 3
xset = int(xmax/ah)
yset = int(ymax/ah)
points_in_mkm = int(1.0/ah)

def max_delta (mas1, mas2):
    res = 0.
    for i in np.arange(yset+1):
        for j in np.arange(xset+1):
            if math.fabs(mas1[i][j] - mas2[i][j]) > res:
                res = math.fabs(mas1[i][j] - mas2[i][j])
    return res

a = np.zeros((int(yset+1), int(xset+1)))
for i in range(xset+1):
    a[0][i] = V1

for i in range(points_in_mkm, 2*points_in_mkm+1):
    a[2*points_in_mkm][i] = V2
    a[2*points_in_mkm][i+3*points_in_mkm] = V2
while True:
    b = np.copy(a)
    for i in np.arange(1,yset):
        for j in np.arange(xset+1):
            if i  == 2*points_in_mkm and (j in range(points_in_mkm, 2*points_in_mkm+1) o
                continue
            elif j == 0:
                a[i][j] = a[i][j+1]
            elif j == xset:
                a[i][j] = a[i][j-1]
            else:
                 a[i][j] = (a[i+1][j] + a[i][j+1] + a[i-1][j] + a[i][j-1])/4
    eps = max_delta(a,b)
    print (eps)
    if eps < accuracy:
        break
print (a)

Ex = np.zeros((yset+1, xset+1))
Ey = np.zeros((yset+1, xset+1))
```

```
for i in np.arange(yset+1):
    for j in np.arange(xset+1):
        if i == 0 and j == 0:
            Ex[i][j] = (a[i][j] - a[i+1][j])/(2*ah*10**(-6))
            Ey[i][j] = (a[i][j] - a[i][j+1])/(2*ah*10**(-6))
        elif i == 0 and j == xset:
            Ex[i][j] = (a[i][j] - a[i+1][j])/(2*ah*10**(-6))
            Ey[i][j] = (a[i][j-1] - a[i][j])/(2*ah*10**(-6))
        elif i == yset and j == 0:
            Ex[i][j] = (a[i-1][j] - a[i][j])/(2*ah*10**(-6))
            Ey[i][j] = (a[i][j] - a[i][j+1])/(2*ah*10**(-6))
        elif i == yset and j == xset:
            Ex[i][j] = (a[i-1][j] + a[i][j])/(2*ah*10**(-6))
            Ey[i][j] = (a[i][j-1] + a[i][j])/(2*ah*10**(-6))
        elif i == 0:
            Ex[i][j] = (a[i][j] - a[i+1][j])/(2*ah*10**(-6))
            Ey[i][j] = (a[i][j-1] - a[i][j+1])/(2*ah*10**(-6))
        elif i == yset:
            Ex[i][j] = (a[i-1][j] - a[i][j])/(2*ah*10**(-6))
            Ey[i][j] = (a[i][j-1] - a[i][j+1])/(2*ah*10**(-6))
        elif j == 0:
            Ex[i][j] = (a[i-1][j] - a[i+1][j])/(2*ah*10**(-6))
            Ey[i][j] = (a[i][j] - a[i][j+1])/(2*ah*10**(-6))
        elif j == xset:
            Ex[i][j] = (a[i-1][j] - a[i+1][j])/(2*ah*10**(-6))
            Ey[i][j] = (a[i][j-1] - a[i][j])/(2*ah*10**(-6))
        else:
             Ex[i][j] = (a[i-1][j] - a[i+1][j])/(2*ah*10**(-6))
             Ey[i][j] = (a[i][j-1] - a[i][j+1])/(2*ah*10**(-6))


print (a)
print ('####################')
print(Ex)
print(Ey)

X = []
Y = []

for i in np.arange(yset+1):
    xline = []
    yline = []
    for j in np.arange(xset+1):
        xline.append(ah*j)
        yline.append(ah*i)
    X.append(xline)
    Y.append(yline)

U = Ex
V = Ey

fig, ax = plt.subplots()
widths = np.linspace(0, 1, V.size)
w = 0.003
plt.quiver(X, Y, V, U, width = w)
axes = plt.gca()
axes.set_xlim([-0.1,6.1])
axes.set_ylim([-0.1,3.1])
fig.savefig('/Users/antonmnacakanov/Desktop/ToF/rgr1/1.pdf', dpi=1000)
plt.close(fig)

fig, ax = plt.subplots()
CS = ax.contour(X, Y, a, levels = np.arange(-1, 3, 0.2))
```

```
ax.clabel(CS, inline=True, fontsize=10)
ax.set_xlim([-0.1,6.1])
ax.set_ylim([-0.1,3.1])
fig.savefig('/Users/antonmnacakanov/Desktop/ToF/rgr1/2.pdf', dpi=1000)
plt.close(fig)

fig = plt.figure()
ax = fig.gca(projection='3d')
CS = ax.plot_surface(X, Y, a, cmap = cm.coolwarm)
ax.contour(X, Y, a, zdir='z', offset=-0.7, levels = np.arange(-1, 3, 0.2))
ax.clabel(CS, inline=True, fontsize=10)
ax.set_xlim([-0.1,6.1])
ax.set_ylim([-0.1,3.1])
fig.savefig('/Users/antonmnacakanov/Desktop/ToF/rgr1/3.pdf', dpi=1000)
plt.close(fig)

f = open("/Users/antonmnacakanov/Desktop/ToF/rgr1/data.csv", "w")
for j in np.arange(xset+1):
    f.write(str(j)+";")
f.write('\n')
for i in np.arange(yset+1):
    for j in np.arange(xset+1):
        f.write(str(round(a[i][j],3))+";")
    f.write('\n')
f.close()

for i in np.arange(yset+1):
    for j in np.arange(xset+1):
        a[i][j] = e * a[i][j];
fig = plt.figure()
ax = fig.gca(projection='3d')
CS = ax.plot_surface(X, Y, a, cmap = cm.coolwarm)
ax.contour(X, Y, a, zdir='z', offset=-4*10**(-19), levels = np.arange(-4*10**(-19),10**(
ax.clabel(CS, inline=True, fontsize=10)
ax.set_xlim([-0.1,6.1])
ax.set_ylim([-0.1,3.1])
fig.savefig('/Users/antonmnacakanov/Desktop/ToF/rgr1/3.1.pdf', dpi=1000)
plt.close(fig)
```