דוח סיכום אנליזה נומרית

קבוצה מס': 3

חברי הקבוצה הנוכחים: איתי כלף ,בר-אילן קימברובסקי ,שי מנשרוב ,אריאל גואטה,יואב יונג.

1.**הגדרת הבעיה** –אחד השימושים העיקריים של ניטור קרינה באוויר הוא במיפוי של בשל מגבלות מתמטיות ,ניטור אווירי(ניתוח איכות האוויר) של שדה רדיו אקטיבי אינו יכול לספק התפלגות תלת-ממדית של הזיהום בענן הרדיואקטיבי. עם זאת ניתן להשיג הערכה מדויקת יותר של התפלגות זיהום דו ממדית על הקרקע ,ניתן להשיג זאת ע"י מדידת שדה רדיואקטיבי מגובה קבוע של 100-500 מטר. יש להשתמש בשתי גישות שתלויות בעצמן על מנת להשיג את הפתרון לבעיה זו:

- המטריצה האלמנטרית
- יישום המבוסס על משוואת שטיין

את הנתונים אנחנו אוספים ע"י מסוק בעל גלאי קרינת גמא. המסוק נמצא בגובה קבוע של 200 מטר מעל פני הקרקע ,המסוק מרחף תמיד מעל לנקודת האמצע של כל ריבוע ושם הוא מבצע את המדידה.

עוצמת הקרינה היא: D=C(1+kR) e^{-mu*R}/R^2

מקרא:

- C הינו מקדם פרופורציה של המוניטור
- K פקטור הקרינה באוויר
- mu מקדם דילול הקרינה שבאוויר
- R מרחק במטרים

האזור של הדגימה הוא ריבועי N*N המסוק (הגלאי) נמצא בגובה H מעל פני הקרקע.

המדדים שאנו מקבלים ע"י הגלאי שבמסוק הם:

$$R_{i,j}^2 = (X_i - X_j)^2 + (Y_i - Y_j)^2 + Z^2$$

R שלנו הוא סכום הקורדינטות הקרטזיות Z,Y,X

2.השיטה והצגת הכלים לפתרון סעיף ב':

שיטת המיתר:

```
def secant(f,a,b,n):
    for i in range(0,n):  # בלולאה ריצה N איארציות
        if f(a)-f(b) == 0: # הכרחי עצירה תנאי
            return a
        res = (a - (f(a)*(b-a)*1.0)/(f(b)-f(a)))  # הבא הערך
באיטרציה
        a = b
        b = res
        print(b)
    return b

print(secant(lambda x: 16*x**3 - 16*x**2 + 1, -0.226, 0, 10))
print(secant(lambda x: 16*x**3 - 16*x**2 + 1, 0.298, 0, 10))
print(secant(lambda x: 16*x**3 - 16*x**2 + 1, 0.927, 0, 10))
```

לפתרון הבעיה השתמשנו בשיטת המיתר בקוד בפייטון אשר חברי הקבוצה כתבו יחדיו.

ניקח שני ערכים התחלתיים שקרובים יחסית לשורש, נבנה ישר שעובר דרך הנקודות ובכל איטרציה חדשה נקדם אותו להיות החיתוך של הישר.

ככל שנחשב יותר איטרציות כך נקבל ערך קרוב יותר לערך השורש האמיתי.

שיטת החצייה :

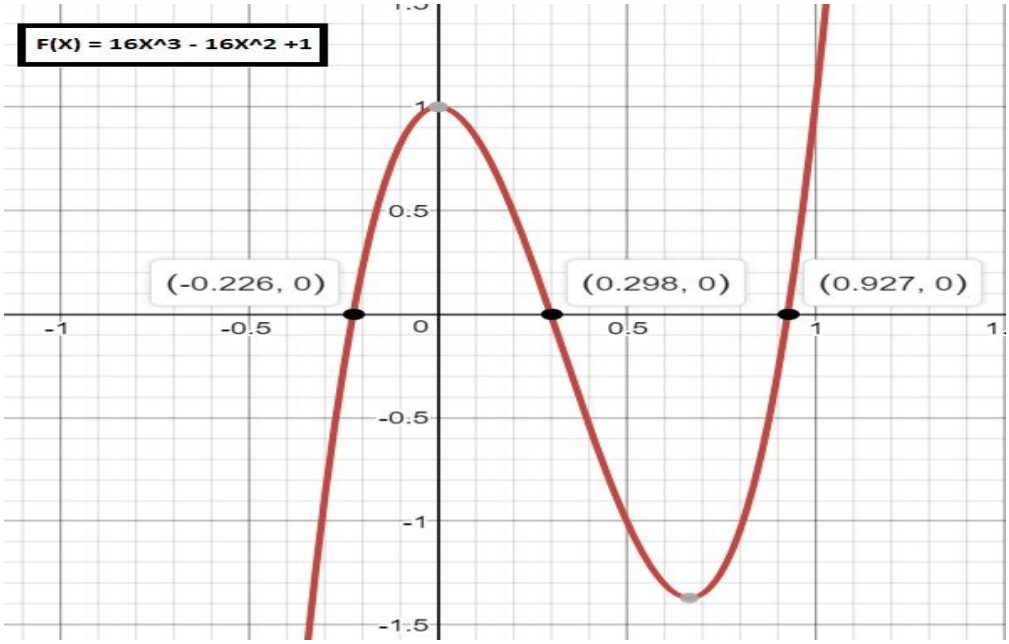
```
import math

def splithalf_method(a, b,func, tol):#function gets a and b is
intervals , polynom and tolerance
    i=0
    if func(a) * func(b) <= 0:#If it is a continuous function
in the interval, we will print an error
        print("error")
    middle = a #turn the middle to a value
    while math.fabs(a-b) >= tol:#turn a-b to absolute number
and check if its bigger or equal to tolerance
        middle = (a + b) / 2
        if func(middle) == 0:
            return middle
        if func(a) * func(middle) < 0:
            b = middle
        else:
            a = middle
        i+=1
        print("Operation number:",i,"\n","Middle:",middle)
    return middle
```

```
y = lambda x:16*(x*3)-16(x**2)+1 #create polynom function
print(splithalf_method(-0.226, 0, y, 0.01))
print(splithalf_method(0.298, 0, y, 0.01))
print(splithalf_method(0.927, 0, y, 0.01))
```

שיטת החצייה היא אלגוריתם למציאת שורש של פונקציה שעושה שימוש בחלוקת המרווח לשורש ובוחר מרווח קטן יותר שהשורש נמצא שם.

3. הצגת נתונים – סעיף ג':

גרף הפונקציה :



קיבלנו 3 שורשים – (0.927,0) , (0.298,0) , (-0.226,0).

הפתרונות שקיבלנו עבור השורשים:

עבור השורש הראשון (-0.226,0)

-0.22556987974418571

-0.22607931589982802

-0.2258028027739527

-0.2258029813409571

-0.22580298147788838

-0.22580298147788833

-0.22580298147788833

-0.22580298147788833

עבור השורש השני (0.298,0)

0.2987628826555001

0.29832419708785524

0.2984841274975575

0.298484141619375

0.2984841416186576

0.2984841416186576

0.2984841416186576

עבור השורש השלישי (0.927,0)

0.9235861742844047

0.8855858133823167

0.9277375638638898

0.9272722755591011

0.9273187919785912

0.9273188398647102

0.9273188398592308

0.9273188398592307

0.9273188398592307

0.9273188398592307

המומצע החשבוני של השורשים החיוביים של המשוואה:

בשורש הראשון התוצאות יהיו שליליות לכן ניקח, את שני השורשים השניים ונחשב ביניהם.

נעזרנו בקוד הקודם וחישבנו את הממוצע.

```python
def secant(f,a,b,n):
    sum = 0
    times = 0
    for i in range(0,n):  # איארציות N בלולאה ריצה
        if f(a)-f(b) == 0: # הכרחי עצירה תנאי
            return a
        res = (a - (f(a)*(b-a)*1.0)/(f(b)-f(a)))  # באיטרציה הבא הערך
        a = b
        b = res
        if (b > 0):
            sum = sum + b
            times += 1
    return sum/times


print(secant(lambda x: 16*x**3 - 16*x**2 + 1, 0.298, 0, 10))
print(secant(lambda x: 16*x**3 - 16*x**2 + 1, 0.927, 0, 10))
```

קיבלנו ממוצע בשורש הראשון ובשורש השני :

0.2984841416186576

0.9273188398592307

ומכאן נקבל שהממוצע של כל השורשים החיוביים יהיה 0.6129014907389442