תרגיל בית 4 – מציאת שורשים

שאלה 1 – false position

נתונה הפונקציה: $f(x) = x^2 - 1/2$.

הגדירו את תחום ה-bracketing להיות בין 0 ל-1.

בצעו 4 צעדים בשיטת false position למציאת שורש של הפונקציה. עשו זאת ידנית או בעזרת קוד (שמדפיס את x ואת $f(x)$ עבור כל צעד), לבחירתכם.

(בתרגול ראינו דוגמת ריצה של מספר צעדים עבור מציאת השורשים של הפונקציה בשיטת ה-Secant).

שאלה 2 – שיטת החציה

א. כתבו במחשב פונקציה המוצאת שורש של $f(x)$ בשיטת החציה (Bisection). הפונקציה תקבל כקלט:

- פונקציה f
- קצה שמאלי של תחום bracketing – a
- קצה ימני של תחום bracketing – b
- דיוק מבוקש

הפונקציה תרוץ עד למציאת שורש, או להגעה לסביבה קרובה מספיק לשורש, בהתאם לדיוק המבוקש. הפונקציה תחזיר את השורש, וגם תחזיר (או תדפיס) את מספר הצעדים שלקח לה לרוץ.

ב. בעזרת הפונקציה, מצאו את השורש של $f(x) = x^2 - 1/2$ עד כדי דיוק של 10^{-10}. התחילו מתחום בין 0 ל-1.

ג. כמה צעדים נדרשים לפונקציה במקרה זה? השוו את מספר הצעדים שקיבלתם בסעיף ב, למספר הצעדים שאמורים להתקבל לפי החישוב האנליטי, $n = \log_2 \frac{\Delta}{\epsilon}$.

שאלה 3 – Newton Fractal

הקדמה

(קודם כל – לא להיבהל, ההקדמה ארוכה אבל השאלה קצרה, ויש בסוף קטעי קוד שיעזרו לכם).

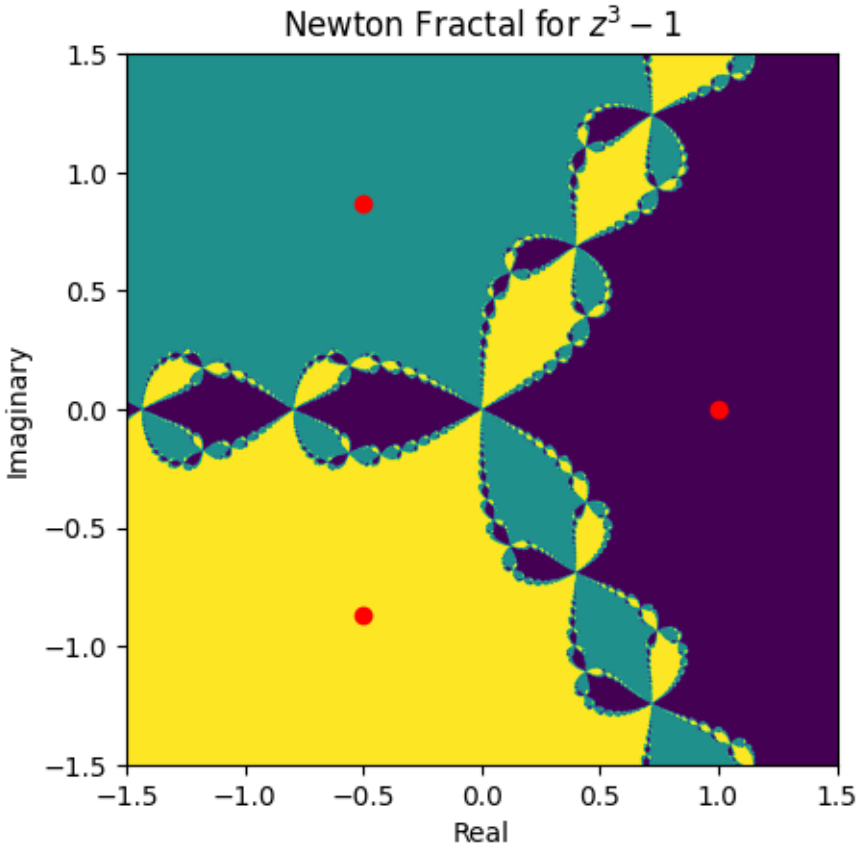
כאשר משתמשים בשיטת Newton-Raphson, האלגוריתם מתחיל מנקודה התחלתית x_0, ומעדכן אותה באמצעות האלגוריתם שלמדנו, בכל צעד $x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)}$, עד להתכנסות לשורש.

כעת נניח שיש לנו פונקציה מרוכבת, למשל $f(z) = z^3 - 1$. לפונקציה יש שלושה שורשים במישור המרוכב: $z = 1, z = e^{i\frac{2}{3}\pi}, z = e^{-i\frac{2}{3}\pi}$.

עבור כל נקודה במישור המרוכב, ניתן לבדוק: אם נתחיל ממנה, ונרוץ בעזרת שיטת Newton-Raphson למשך n צעדים, לאיזה אחד מהשורשים נהיה הכי קרובים בסוף.

נבחר צבע עבור כל שורש, ונצבע בו את כל הנקודות, שהשורש הזה הוא השורש הכי קרוב לתוצאה של הריצה שלהן.

מתברר שאנו מקבלים תמונות יפות ומורכבות (השורשים מסומנים באדום):



התמונה שמתקבלת היא פרקטל, תוכלו לקרוא על פרקטלים בויקיפדיה.

למעוניינים, ניתן לצפות בסרטון בקישור זה, בו הנושא מוסבר בצורה יפה.

המשימה

א. כתבו פונקציה המוצאת שורשים בשיטת ניוטון-רפסון במימד אחד. הפונקציה תקבל:

- פונקציה f
- פונקציית הנגזרת f'
- נקודה התחלתית x_0
- מספר צעדים רצוי n

היא תתחיל מ-x_0, ותריץ את האלגוריתם למשך n צעדים.

(ספציפית עבור השאלה שלנו, אנו לא נתייחס לדיוק הרצוי במציאת השורש, אלא רק למספר הצעדים שהפונקציה רצה).

ב. ציירו את פרקטל ניוטון של הפונקציה $f(z) = z^4 - 1$, עבור המישור המרוכב בתחום שבין -1.5 לבין 1.5, (גם בציר הממשי וגם בציר המרוכב).

ציירו את התמונה המתקבלת מאלגוריתם שרץ 2 צעדים, ואת הפרקטל שמתקבל מאלגוריתם שרץ 30 צעדים.

שימו לב: למרות שמדובר במישור המרוכב, אפשר להשתמש בשיטת ניוטון רפסון עבור מימד אחד, כי פייתון ומטלב יודעים לעבוד עם מרוכבים.

הערה חשובה

ניתן, ורצוי, להיעזר בקוד המצורף (בפייתון ובמטלב), שיוצר את פרקטל ניוטון בתחום הרצוי עבור $f(z) = z^3 - 1$.

(במקום לרוץ על כל נקודה אחת אחרי השניה ולהריץ את האלגוריתם עבורה, הקוד משתמש ב-vectorization: אנו שולחים את כל הנקודות שאנו רוצים לעבוד עליהם אל הפונקציה, ביחד. מטלב וספריית numpy בפייתון שניהם יודעים לעשות חישובים עבור כמה נקודות במקביל. כתיבת הקוד בצורה כזו חוסכת זמן ריצה באופן משמעותי.)

לא צירפתי לכם את הקוד עבור האלגוריתם לשיטת ניוטון רפסון עצמו, אותו אתם מתבקשים לכתוב בעצמכם. בקוד שלי, אני קראתי לפונקציה NR(f,dfdx,x0,n).

קוד בפייתון:

```
def f(z):
  return (z**3)-1

def dfdx(z):
  return 3*(z**2)

# the true roots of the function:
roots = np.array([1.0, np.exp((2/3)*np.pi*1j), np.exp((-
2/3)*np.pi*1j)])

# number of steps for NR method:
num_steps = 30

# define the complex plane:
lim = [-1.5,1.5]
mesh = 1000
x = np.linspace(lim[0],lim[1],mesh)
plane = (x.reshape(1,mesh) + 1j *
x.reshape(mesh,1)).reshape(mesh,mesh,1)

# run the algorithm:
NR_results = NR(f,dfdx,plane,num_steps)

# find the closest root to each point, and label it for color:
r = np.argmin(np.abs(roots - NR_results), axis=2)

# show the fractal:
plt.imshow(r, extent=(lim[0],lim[1], lim[0],lim[1]), origin='lower')
plt.xlabel('Real')
plt.ylabel('Imaginary')
plt.title(r'Newton Fractal for $z^3-1$')
plt.plot(np.real(roots), np.imag(roots), 'o', color='red')
plt.show()
```

קוד במטלב:

```
f = @(z) (z.^3) - 1;
dfdx = @(z) 3 * (z.^2);

% the true roots of the function:
roots = [1.0, exp((2/3)*pi*1i), exp((-2/3)*pi*1i)];

% number of steps for NR method:
num_steps = 30;

% Define the complex plane:
lim = [-1.5, 1.5];
mesh = 1000;
x = linspace(lim(1), lim(2), mesh);
[X, Y] = meshgrid(x, x);
plane = X + 1i * Y;

% Apply Newton-Raphson method
result = NR(f, dfdx, plane, num_steps);

% find the closest root to each point, and label it for color:
[~, r] = min(abs(reshape(roots, 1, 1, []) - result), [], 3);

% Plot the result
imagesc(lim, lim, r);
axis xy; % Ensure correct orientation
xlabel('Real');
ylabel('Imaginary');
title('Newton Fractal for $z^3-1$', Interpreter='latex');
hold on;
% Plot the roots
plot(real(roots), imag(roots), 'r*');
hold off;
```