Jurnal MAHANDIA
ISSN: 25448-8899 (Print)
https://mahandia.com/jurnal/index.php/jm/index

METODE BRACKETING VERSUS OPEN DALAM MENEMUKAN NOL FUNGSI DENGAN MATLAB

Muhamad Iradat Achmad[1*]
[1]Teknik Informatika, Fakultas Teknik, Universitas Dayanu Ikhsanuddin
[1]irad4t@gmail.com
*Penulis Korespondensi

diajukan: 3 Januari 2025, diterima: 1 Februari 2025.

Abstract

Finding the roots of a function is a fundamental problem in numerical analysis in various scientific and engineering applications. The Bracketing method (Bisection, Regula Falsi) and the Open method (Newton-Raphson, Secant) have been developed to solve this problem. This research aims to compare the performance of the four methods in finding the roots of polynomial and transcendental functions. Implementation was carried out using MATLAB to ensure computational efficiency and validity. Four numerical methods were tested on polynomial and transcendental functions with evaluation criteria including the number of iterations required to achieve a certain tolerance, the accuracy of the roots obtained, and stability against variations in initial values. The results found, the Secant Method has a speed almost equivalent to Newton-Raphson, without requiring function derivatives, making it more flexible. Regula Falsi is faster than Bisection but can experience stagnation under certain conditions. Bisection always converges with high stability, but requires more iterations than other methods. For polynomial functions, Newton-Raphson and Secant are more efficient, while for transcendental functions, Secant is a more practical alternative to Newton-Raphson if the derivative is difficult to calculate. No method is completely superior in all aspects. Newton-Raphson and Secant are recommended for computational efficiency, while Bisection and Regula Falsi are preferred if stability and certainty of convergence are preferred.

Keywords: Function Roots, Bisection, Regula Falsi, Newton-Raphson, Secant, MATLAB

Abstrak

Pencarian akar fungsi merupakan permasalahan fundamental dalam analisis numerik di berbagai aplikasi sains dan rekayasa. Metode Bracketing (Bisection, Regula Falsi) dan metode Open (Newton-Raphson, Secant) telah dikembangkan untuk menyelesaikan permasalahan tersebut. Penelitian ini bertujuan untuk membandingkan performa keempat metode tersebut dalam menemukan akar fungsi polinomial dan transendental. Implementasi dilakukan menggunakan MATLAB untuk memastikan efisiensi dan validitas komputasi. Empat metode numerik diuji pada fungsi polinomial dan transendental dengan kriteria evaluasi meliputi jumlah iterasi yang diperlukan untuk mencapai toleransi tertentu, akurasi akar yang diperoleh, serta stabilitas terhadap variasi nilai awal. Hasil yang ditemukan, Metode Secant memiliki kecepatan hampir setara dengan Newton-Raphson, tanpa memerlukan turunan fungsi, menjadikannya lebih fleksibel. Regula Falsi lebih cepat dari Bisection tetapi dapat mengalami stagnasi pada kondisi tertentu. Bisection selalu konvergen dengan kestabilan tinggi, tetapi membutuhkan iterasi lebih banyak dibanding metode lainnya. Untuk fungsi polinomial, Newton-Raphson dan Secant lebih efisien, sementara untuk fungsi transendental, Secant menjadi alternatif lebih praktis dibanding Newton-Raphson jika turunan sulit dihitung. Tidak ada metode sepenuhnya unggul dalam semua aspek. Newton-Raphson dan Secant direkomendasikan untuk efisiensi komputasi, sementara Bisection dan Regula Falsi lebih disarankan jika kestabilan dan kepastian konvergensi lebih diutamakan.

Kata Kunci : Akar Fungsi, Bisection, Regula Falsi, Newton-Raphson, Secant, MATLAB

1. PENDAHULUAN

Dalam banyak bidang ilmu pengetahuan dan rekayasa, permasalahan menemukan nol fungsi atau akar dari suatu persamaan non-linear sering muncul sebagai bagian dari analisis sistem yang lebih kompleks. Nol fungsi didefinisikan sebagai nilai x yang memenuhi persamaan $f(x) = 0$, di mana $f(x)$ merupakan suatu fungsi non-linear. Proses ini memiliki aplikasi luas dalam berbagai disiplin ilmu, seperti fisika, ekonomi, teknik, dan kecerdasan buatan. Sebagai contoh, dalam

bidang teknik sipil dan mekanika, penyelesaian sistem persamaan non-linear sering digunakan untuk analisis struktur, perhitungan deformasi material, dan desain sistem dinamis. Di bidang ekonomi dan keuangan, metode pencarian nol fungsi diaplikasikan dalam pemodelan pasar, perhitungan suku bunga implisit, serta optimasi portofolio. Sementara itu, dalam sains komputasi, metode ini digunakan dalam pemodelan fenomena kompleks, seperti prediksi cuaca dan simulasi dinamika fluida. Berbagai metode numerik telah dikembangkan untuk mencari nol fungsi, yang umumnya dapat diklasifikasikan menjadi dua kategori utama, yaitu metode Bracketing dan metode Open. Metode Bracketing, seperti Bisection dan Regula Falsi (Intep, 2018), bekerja dengan membatasi akar dalam suatu interval yang diketahui mengandung solusi dan secara iteratif mempersempit rentang interval hingga diperoleh nilai yang cukup dekat dengan akar sebenarnya. Metode ini bersifat konvergen secara global, namun sering kali lebih lambat dibandingkan metode lain. Sebaliknya, metode Open, seperti Newton-Raphson, Secant, dan Fixed-Point Iteration (Badr et al., 2022), menggunakan pendekatan yang lebih langsung dengan memanfaatkan informasi turunan atau pendekatan iteratif lainnya. Metode ini sering kali memiliki tingkat konvergensi yang lebih cepat dibandingkan metode Bracketing, tetapi lebih sensitif terhadap pemilihan titik awal dan dapat mengalami divergensi jika kondisi tertentu tidak terpenuhi. Oleh karena itu, pemilihan metode yang tepat sangat bergantung pada karakteristik fungsi yang dianalisis serta kebutuhan spesifik dari permasalahan yang dihadapi.

MATLAB, sebagai salah satu perangkat lunak komputasi numerik yang banyak digunakan, menyediakan berbagai fungsi bawaan untuk menyelesaikan masalah pencarian nol fungsi, seperti fzero (Mathworks, 2013) yang berbasis metode hybrid antara Bracketing dan Open Methods. Namun, pemahaman tentang karakteristik masing-masing metode tetap diperlukan agar pengguna dapat memilih algoritma yang paling sesuai dengan kebutuhan analisisnya. Dalam penelitian ini, dilakukan studi numerik untuk membandingkan metode Bracketing dan Open dalam pencarian nol fungsi menggunakan MATLAB. Perbandingan ini mencakup aspek kecepatan konvergensi, akurasi hasil, dan kestabilan metode terhadap variasi fungsi. Hasil studi ini diharapkan dapat memberikan panduan bagi akademisi dan praktisi dalam memilih metode yang paling efektif berdasarkan karakteristik permasalahan yang dihadapi. Pemecahan masalah pencarian nol fungsi atau root-finding merupakan aspek fundamental dalam analisis numerik yang digunakan secara luas dalam berbagai bidang ilmu pengetahuan dan teknik. Meskipun berbagai metode telah dikembangkan untuk mencari akar suatu fungsi non-linear, tidak ada metode tunggal yang secara universal optimal untuk semua jenis permasalahan. Oleh karena itu, studi perbandingan metode root-finding menjadi penting untuk memahami keunggulan dan keterbatasan masing-masing pendekatan, sehingga pengguna dapat memilih metode yang paling sesuai berdasarkan karakteristik masalah yang dihadapi. Metode Bracketing, seperti Bisection dan Regula Falsi, dikenal karena kestabilannya dalam menemukan akar dalam suatu interval yang diketahui mengandung solusi. Metode ini selalu konvergen, tetapi sering kali membutuhkan lebih banyak iterasi untuk mencapai tingkat akurasi yang diinginkan. Sebaliknya, metode Open, seperti Newton-Raphson, Secant, dan Fixed-Point Iteration, biasanya memiliki konvergensi yang lebih cepat, tetapi bergantung pada pemilihan titik awal yang tepat dan dapat mengalami divergensi jika tidak diterapkan dengan hati-hati.

Studi Perbandingan Diperlukan untuk Menentukan Efisiensi dan Kecepatan Konvergensi. Setiap metode memiliki laju konvergensi yang berbeda. Misalnya, metode Bisection memiliki laju konvergensi linear, sedangkan metode Newton-Raphson memiliki laju konvergensi kuadratik, yang berarti Newton-Raphson dapat mencapai solusi lebih cepat jika kondisi yang diperlukan terpenuhi. Dengan membandingkan berbagai metode, kita dapat mengevaluasi mana yang paling efisien dalam konteks perhitungan numerik. Selain itu, perbandingan diperlukan untuk Evaluasi Akurasi dan Presisi. Beberapa metode dapat memberikan solusi dengan tingkat akurasi yang lebih tinggi dalam jumlah iterasi yang lebih sedikit, sementara yang lain mungkin memerlukan lebih

banyak langkah untuk mencapai presisi yang sama. Faktor ini menjadi penting dalam aplikasi yang membutuhkan keakuratan tinggi, seperti dalam perhitungan struktural di bidang teknik sipil atau dalam simulasi sistem dinamis di bidang fisika dan keuangan. Lebih jauh lagi perbandingan diperlukan untuk mengetahui Stabilitas Metode terhadap Variasi Fungsi. Metode Bracketing umumnya lebih stabil, karena mereka tidak bergantung pada nilai turunan atau asumsi linearitas lokal. Sebaliknya, metode Open bisa mengalami divergensi jika titik awal tidak dipilih dengan baik atau jika fungsi memiliki karakteristik tertentu, seperti adanya titik stasioner di dekat akar. Studi perbandingan membantu mengidentifikasi situasi di mana metode tertentu mungkin lebih rentan terhadap kegagalan. Dampak dan Kontribusi Studi Perbandingan ini akan memberikan wawasan bagi peneliti, akademisi, dan praktisi dalam memilih metode pencarian nol fungsi yang paling sesuai dengan jenis permasalahan yang dihadapi. Dalam praktiknya, pemilihan metode yang tepat tidak hanya meningkatkan efisiensi komputasi, tetapi juga mengurangi risiko kesalahan dalam perhitungan numerik yang dapat berdampak pada pengambilan keputusan teknik dan ilmiah. Dengan demikian, studi ini diharapkan dapat memberikan kontribusi dalam pengembangan metode numerik yang lebih baik serta meningkatkan pemahaman tentang bagaimana algoritma pencarian akar dapat dioptimalkan dalam MATLAB dan aplikasi lainnya.

1.1. Metod Bisection

Metode Bisection (Osman et al., 2022) menggunakan pendekatan pembagian interval hingga mencapai toleransi tertentu. Jika akar berada dalam interval [a, b], maka titik tengahnya dihitung sebagai:

$$c_n = \frac{a_n + b_n}{2}$$

Jika $f(a_n) \cdot f(c_n) < 0$, maka akar ada dalam interval $[a_n, c_n]$ sehingga:

$$b_{n+1} = c_n$$

Jika $f(c_n) \cdot f(b_n) < 0$, maka akar ada dalam interval $[c_n, b_n]$ sehingga:

$$a_{n+1} = c_n$$

Iterasi berhenti jika:

$$|b_n - a_n| < \text{toleransi}$$

1.2. Metod Regula Falsi

Metode Regula Falsi (Nguyen, 2021)(Abu Bakar et al., 2012) mirip dengan Bisection, tetapi memilih titik baru menggunakan interpolasi linier antara (a, $f(a)$) dan (b, $f(b)$):

$$c_n = \frac{a_n f(b_n) - b_n f(a_n)}{f(b_n) - f(a_n)}$$

Jika $f(a_n) \cdot f(c_n) < 0$, maka akar ada dalam interval $[a_n, c_n]$ sehingga:

$$b_{n+1} = c_n$$

Jika $f(c_n) \cdot f(b_n) < 0$, maka akar ada dalam interval $[c_n, b_n]$ sehingga:

$$a_{n+1} = c_n$$

Iterasi berhenti jika:

$$|f(c_n)| < \text{toleransi}$$

1.3. Metod Newton-Raphson

Metode Newton-Raphson (Nadhim & Al-Jilawi, 2022) menggunakan pendekatan turunan fungsi untuk mencari akar. Iterasi dilakukan dengan rumus:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

Iterasi berhenti jika:

$$|f(x_n)| < \text{toleransi}$$

atau perubahan nilai *x* antar iterasi sangat kecil:

$$|x_{n+1} - x_n| < \text{toleransi}$$

1.4. Metod Secant

Metode Secant (Juhari, 2021) adalah varian dari Newton-Raphson yang tidak menggunakan turunan eksplisit, tetapi mendekati turunan dengan diferensiasi numerik. Rumus iterasinya adalah:

$$x_{n+1} = x_n - \frac{f(x_n)(x_n - x_{n-1})}{f(x_n) - f(x_{n-1})}$$

Iterasi berhenti jika:

$$|f(x_n)| < \text{toleransi}$$

atau perubahan nilai *x* antar iterasi sangat kecil:

$$|x_{n+1} - x_n| < \text{toleransi}$$

2. METODE

2.1 Pemilihan Fungsi Uji

Pemilihan fungsi uji merupakan tahap krusial dalam membandingkan efektivitas berbagai metode *root-finding*. Fungsi yang dipilih harus cukup beragam untuk menguji performa metode dalam berbagai skenario, termasuk tingkat kompleksitas fungsi, sifat turunan, serta sensitivitas terhadap inisialisasi awal. Dalam penelitian ini, fungsi uji dikelompokkan menjadi fungsi polinomial dan fungsi transendental, yang masing-masing memiliki karakteristik unik dalam konteks pencarian akar. Fungsi polinomial merupakan salah satu bentuk fungsi yang paling umum dalam berbagai aplikasi teknik dan sains. Fungsi ini berbentuk:

$$f(x) = a_n x^n + a_{n-1} x^{n-1} + \cdots + a_1 x + a_0$$

dengan $a_n, a_{n-1}, \ldots, a_0$ sebagai koefisien dan *n* sebagai derajat polinomial. Dalam penelitian ini, untuk menguji performa metode pencarian akar dipakai fungsi polinomial derajat 3 dengan akar real yang berbeda sebagai berikut.

$$f(x) = x^3 - 6x^2 + 11x - 6$$

Fungsi ini memiliki tiga akar real berbeda, yaitu *x* = 1, 2, 3. Pemilihan fungsi ini bertujuan untuk menguji bagaimana setiap metode beradaptasi terhadap akar yang terdistribusi secara merata dan apakah metode mengalami kesulitan dalam menemukan akar tertentu.

Fungsi transendental adalah fungsi yang tidak dapat dinyatakan dalam bentuk polinomial, seperti fungsi eksponensial, trigonometri, dan logaritmik. Fungsi-fungsi ini sering muncul dalam fenomena fisis dan teknik. Dalam penelitian ini, fungsi transendental yang digunakan dalam perbandingan adalah sebagai berikut.

$$f(x) = e^x - 3$$

Fungsi transendental sering kali memiliki turunan yang bervariasi secara signifikan, yang dapat menyebabkan beberapa metode mengalami kesulitan konvergensi jika dipilih titik awal yang tidak sesuai.

2.2 Pembandingan Metode Pencarian Akar dalam MATLAB

Implementasi metode numerik dalam perangkat lunak seperti MATLAB merupakan langkah penting untuk menguji dan menganalisis efektivitas berbagai teknik pencarian akar (*root-finding methods*)(Muhamad Iradat Achmad, 2006). MATLAB dipilih karena menyediakan kemudahan dalam manipulasi numerik, grafik visualisasi, serta alat bantu analisis konvergensi. Pada penelitian ini, empat metode pencarian akar yang diimplementasikan dalam MATLAB adalah Metode Bisection, Metode Regula Falsi (False Position), Metode Newton-Raphson, dan Metode Secant. Setiap metode dikodekan sebagai fungsi MATLAB yang menerima parameter

fungsi f(x) dan turunan fungsinya, interval atau titik awal, toleransi error, dan jumlah iterasi maksimum. Implementasi mencakup penanganan error, pengecekan konvergensi, serta visualisasi hasil dalam bentuk tabel dan grafik. Metode Bisection merupakan pendekatan numerik yang menggunakan interval iteratif untuk mempersempit rentang akar hingga mencapai toleransi tertentu. Metode ini memeriksa validitas interval untuk memastikan nilai fungsu batas interval $f(a)$ dan $f(b)$ memiliki tanda berlawanan. Selama iterasi, Interval terus dibagi dua hingga mencapai toleransi atau iterasi maksimum. Keadaan konvergen yaitu iterasi berhenti ketika setengah jarak interval $(b - a)/2$ lebih kecil dari toleransi.

Metode Newton-Raphson menggunakan turunan fungsi untuk mempercepat konvergensi dalam pencarian akar dengan iterasi. Metode ini memeriksa kegagalan dengan cara menghitung nilai turunan fungsi sama dengan nol. Metode ini memiliki kecepatan konvergensi kuadratis namun sensitif terhadap titik awal. Metode Secant adalah varian Newton-Raphson yang tidak memerlukan turunan eksplisit tetapi menggunakan pendekatan diferensial. Metode ini tidak memerluka turunan fungsi sehingga cocok untuk fungsi kompleks. Metode ini memiliki konvergensi yang superlinear namun dapat gagal jika dua iterasi awal terlalu dekat atau jika gradien sangat kecil. Metode Regula Falsi mirip dengan Bisection, tetapi menggunakan interpolasi linier untuk memperkirakan akar fungsi. Metode ini umumnya lebih cepat dibanding Bisection dalam beberapa kasus meskipun dapat melambat jika salah satu ujung interval tetap dalam iterasi berturut-turut.

Analisis Hasil Implementasi dilakukan untuk menguji keempat metode di atas. Analisis ini dilakukan melalui eksperimen numerik dengan dua fungsi uji, yaitu fungsi polinomial kubik $f(x) = x^3 - 6x^2 + 11x - 6$ dan fungsi transendental $f(x) = e^x - 5$. Hasil perbandingan dari metode di atas dicatat dalam tabel iterasi yang mencakup jumlah iterasi hingga konvergensi, nilai akar yang ditemukan, dan kesalahan relatif terhadap akar sebenarnya.

2.3 Eksperimen dan Pengaturan Parameter

Eksperimen dalam penelitian ini bertujuan untuk membandingkan performa berbagai metode *root-finding* dalam menyelesaikan permasalahan pencarian akar fungsi nonlinear. Agar hasil eksperimen dapat dianalisis secara objektif, dilakukan serangkaian uji dengan parameter yang telah ditentukan secara sistematis. Agar eksperimen berjalan secara konsisten, setiap metode dihentikan berdasarkan salah satu dari dua kondisi berikut, yaitu Error Mutlak Lebih Kecil dari Toleransi dan Perbedaan Nilai Akar pada Iterasi Berurutan Kecil. Eksperimen ini menggunakan nilai toleransi 10^{-6} untuk kedua kondisi tersebut. Jika salah satu dari kedua kriteria ini terpenuhi, iterasi dihentikan dan solusi diambil sebagai akar yang ditemukan.

Metode numerik sangat bergantung pada pemilihan parameter awal. Untuk menghindari bias, digunakan pendekatan sistematis sebagai berikut. Pada Metode Bracketing (Bisection dan Regula Falsi), dipilih interval awal $[a, b]$ yang memastikan $f(a) \times f(b) < 0$, sehingga akar berada dalam interval tersebut. Sementara pada Metode Open (Newton-Raphson dan Secant), Dipilih titik awal x_0 yang berdekatan dengan akar eksak berdasarkan grafik fungsi atau perkiraan numerik. Dalam eksperimen ini, untuk fungsi $f(x) = x^3 - 6x^2 + 11x - 6$, interval awal untuk metode *bracketing* sebagai [0.5,1.5], sementara titik awal untuk metode *open* dapat diambil sebagai x_0=0.5. Selain itu, terkait dengan jumlah iterasi maksimum, beberapa metode dapat mengalami konvergensi yang lambat atau bahkan divergen jika pemilihan parameter awal kurang tepat. Oleh karena itu, diterapkan batas maksimum iterasi untuk mencegah komputasi yang tidak efisien. Dalam eksperimen ini, jumlah iterasi maksimum ditetapkan sebagai Iterasi Maksimum $= 100$. Jika metode tidak mencapai solusi dalam jumlah iterasi ini, maka metode dianggap gagal konvergen untuk kasus tersebut.

3. HASIL DAN PEMBAHASAN

3.1 Implementasi Metode Dalam MATLAB

Implementasi metode Bisection dalam kode program MATLAB diperlihatkan dalam fungsi M-File "bisection.m" berikut ini.

```
function root = bisection(f, a, b, tol, max_iter)
    if f(a) * f(b) >= 0
        error('Fungsi harus memiliki tanda berlawanan pada a dan b.');
    end
    iter = 0;
    while (b - a) / 2 > tol && iter < max_iter
        c = (a + b) / 2; % Titik tengah
        if f(c) == 0
            root = c;
            return;
        elseif f(c) * f(a) < 0
            b = c;
        else
            a = c;
        end
        iter = iter + 1;
    end
    root = (a + b) / 2;
end
```

Implementasi metode Regula Falsi dalam kode program MATLAB diperlihatkan dalam fungsi M-File "regula_falsi.m" berikut ini.

```
function root = regula_falsi(f, a, b, tol, max_iter)
    % Memastikan bahwa f(a) dan f(b) memiliki tanda yang berlawanan
    if f(a) * f(b) > 0
        error('f(a) dan f(b) harus memiliki tanda yang berlawanan.');
    end

    iter = 0;
    while iter < max_iter
        % Menghitung titik baru dengan metode Regula Falsi
        c = (a * f(b) - b * f(a)) / (f(b) - f(a));

        % Jika akar ditemukan atau sudah cukup dekat, berhenti
        if abs(f(c)) < tol
            break;
        end

        % Menentukan interval baru berdasarkan tanda f(c)
        if f(a) * f(c) < 0
            b = c;
        else
            a = c;
        end

        iter = iter + 1;
    end

    root = c; % Mengembalikan akar yang ditemukan
end
```

Implementasi metode Newton-Raphson dalam kode program MATLAB diperlihatkan dalam fungsi M-File "newton.m" berikut ini.

```
function root = newton(f, df, x0, tol, max_iter)
    iter = 0;
    x = x0;
    while abs(f(x)) > tol && iter < max_iter
        x = x - f(x) / df(x);
        iter = iter + 1;
    end
    root = x;
end
```

Implementasi metode Secant dalam kode program MATLAB diperlihatkan dalam fungsi M-File "secant.m" berikut ini.

```
function root = secant(f, x0, x1, tol, max_iter)
    iter = 0;
    while abs(x1 - x0) > tol && iter < max_iter
        x_temp = x1 - (f(x1) * (x1 - x0)) / (f(x1) - f(x0));
        x0 = x1;
        x1 = x_temp;
        iter = iter + 1;
    end
    root = x1;
end
```

Untuk menggunakan fungsi M-File metode-metode tersebut diperlukan skrip M-File pemanggil "sccall01.m" untuk fungsi polinomial orde-3 dengan tiga akar real berbeda $f(x) = x^3 - 6x^2 + 11x - 6$, seperti berikut ini.

```
clear all; close all; clc;
f = @(x) x.^3 - 6*x.^2 + 11*x - 6; % Fungsi polinomial kubik
df = @(x) 3*x.^2 - 12*x + 11; % Turunan fungsi
a = 1.6;  % Batas bawah
b = 2.7;  % Batas atas
tol = 1e-6;  % Toleransi error
max_iter = 100; % Maksimum iterasi
% Bisection
root_bis = bisection(f, a, b, tol, max_iter);
disp(['Bisection Root: ', num2str(root_bis)]);
% Regula Falsi
root_rf = regula_falsi(f, a, b, tol, max_iter);
disp(['Regula Falsi Root: ', num2str(root_rf)]);
% Newton-Raphson
root_newton = newton(f, df, a, tol, max_iter);
disp(['Newton-Raphson Root: ', num2str(root_newton)]);
% Secant
root_sec = secant(f, a, b, tol, max_iter);
disp(['Secant Root: ', num2str(root_sec)]);
```

dan skrip M-File pemanggil "sccall02.m" untuk fungsi transendental $f(x) = e^x - 5$. seperti berikut ini.

```
clear all; close all; clc;
f = @(x) exp(x) - 5; % Fungsi yang digunakan dalam semua metode
df = @(x) exp(x);
a = 0;  % Batas bawah
b = 2;  % Batas atas
tol = 1e-6;  % Toleransi error
max_iter = 100; % Maksimum iterasi
akarbi = bisection(f, a, b, tol, max_iter);
disp(['Bisection Root: ', num2str(akarbi)]);
akarrf = regula_falsi(f, a, b, tol, max_iter);
disp(['Regula Falsi Root: ', num2str(akarrf)]);
akarnew = newton(f, df, a, tol, max_iter);
disp(['Newton-Raphson Root: ', num2str(akarnew)]);
akarsec = secant(f, a, b, tol, max_iter);
disp(['Secant Root: ', num2str(akarsec)]);
```

Untuk menjalankan fungsi pemanggil metode, pastikan seluruh M-File berada dalam satu direktori. Keluaran fungsi pemanggil "sccall01.m" adalah salah satu akar atau nol fungsi dari polinomial $f(x) = x^3 - 6x^2 + 11x - 6$ yang dapat dilihat pada jendela Command Window MATLAB seperti berikut ini.

```
Bisection Root: 2
Regula Falsi Root: 2
Newton-Raphson Root: 2
Secant Root: 2
>>
```

Sementara keluaran fungsi pemanggil "sccall02.m" adalah akar atau pembuat nol fungsi dari fungsi transendental $f(x) = e^x - 5$, yang dapat dilihat pada jendela Command Window MATLAB sesaat setelah fungsi tersebut dijalankan, seperti berikut ini.

```
Bisection Root: 1.6094
Regula Falsi Root: 1.6094
Newton-Raphson Root: 1.6094
Secant Root: 1.6094
>>
```

3.2 Hasil Perhitungan Fungsi Polinomial dan Pembahasan

Metode Bisection mempersempit interval [*a*, *b*] di setiap iterasi dengan mengambil titik tengah *c* dan menentukan di mana akar berada berdasarkan tanda *f*(*c*).

Tabel Iterasi Bisection untuk Fungsi Polinomial
Fungsi yang digunakan: $f(x) = x^3 - 6x^2 + 11x - 6$
Interval awal: a = 1.6, b = 2.7, Toleransi error: $\epsilon = 10^{-6}$, Batas iterasi: 100 iterasi.

Iterasi	*a*	*b*	$c = \frac{a+b}{2}$	*f*(*c*)	Interval Baru
1	1.6	2.7	2.15	0.1624	[1.6, 2.1500]
2	1.6	2.15	1.875	-0.0234	[1.8750, 2.1500]
3	1.875	2.15	2.0125	0.0252	[1.8750, 2.0125]
4	1.875	2.0125	1.9438	-0.0057	[1.9438, 2.0125]
5	1.9438	2.0125	1.9781	-0.0009	[1.9781, 2.0125]
6	1.9781	2.0125	1.9953	-0.0001	[1.9953, 2.0125]
7	1.9953	2.0125	2.0039	0	Akar ditemukan!

Akar ditemukan pada x = 2.0039 dalam 7 iterasi. Karena metode ini hanya mengurangi interval setengahnya setiap iterasi, konvergensi lebih lambat dibanding metode lainnya. Keandalan metode Bisection adalah meskipun lebih lambat, metode ini selalu konvergen selama interval awal mengandung akar. Dibandingkan dengan metode lainnya, metode ini membutuhkan relatif lebih banyak iterasi untuk mencapai toleransi yang sama. Metode Bisection sangat stabil namun relatif lebih lambat dibanding metode lainnya.

Metode Regula Falsi mirip dengan Bisection, tetapi menggunakan pendekatan garis lurus untuk memperkirakan akar lebih cepat.

Tabel Iterasi Regula Falsi untuk Fungsi Polinomial
Fungsi yang digunakan: $f(x) = x^3 - 6x^2 + 11x - 6$
Interval awal**:** a = 1.6, b = 2.7, Toleransi error: $\epsilon = 10^{-6}$, Batas iterasi: 100 iterasi.

Iterasi	*a*	*b*	*c* (Akar Perkiraan)	*f*(*c*)	Interval Baru
1	1.6	2.7	2.0373	0.0515	[1.6000, 2.0373]
2	1.6	2.0373	1.9817	-0.0033	[1.9817, 2.0373]
3	1.9817	2.0373	1.9994	-0.0003	[1.9994, 2.0373]
4	1.9994	2.0373	2	0	Akar ditemukan!

Akar ditemukan pada x = 2.0000 dalam 4 iterasi**.** Regula Falsi lebih cepat dibanding Bisection (7 iterasi) karena memperkirakan akar berdasarkan garis lurus antara *f*(*a*) dan *f*(*b*). Metode ini memiliki Konvergensi lebih cepat dibanding Bisection karena Regula Falsi menggunakan pendekatan garis lurus, perbaikannya lebih agresif dibanding metode Bisection yang hanya membagi interval. Kelemahan metode Regula Falsi terlihat jika turunan *f*(*x*) sangat kecil di salah satu ujung interval, metode ini bisa menjadi sangat lambat karena titik tetap pada salah satu sisi.

Namun, dalam kasus ini, konvergensi cepat karena akar ada dalam interval yang dipilih dengan baik. Metode Regula Falsi lebih efisien daripada Bisection, tetapi bisa terjebak jika kemiringan fungsi sangat kecil di salah satu sisi.

Berikut ini diberikan tabel iterasi metode Newton-Raphson untuk menghitung nol fungsi dari polinomial orde-3 dengan tiga akar real berbeda.

Tabel Iterasi Newton-Raphson untuk Fungsi Polinomial
Fungsi yang digunakan: $f(x) = x^3 - 6x^2 + 11x - 6$
Nilai awal: a = 1.6, Toleransi error: $\epsilon = 10^{-6}$, Batas iterasi: 100 iterasi.

Iterasi	x_n	$f(x_n)$	$f'(x_n)$	x_{n+1}
1	1.6	0.384	-0.92	2.016522
2	2.016522	-0.0655	-2.06408	1.984799
3	1.984799	0.025869	-2.09146	1.997183
4	1.997183	0.005664	-2.00564	1.999999
5	1.999999	0.000001	-2	2

Metode ini memiliki Konvergensi Cepat. Newton-Raphson mencapai akar x = 2.000000 hanya dalam 5 iterasi, jauh lebih cepat dibanding metode lainnya. Ini karena metode ini memiliki konvergensi kuadratik, yang berarti error berkurang secara eksponensial. Meski demikian, metode ini memiliki ketidakstabilan jika turunan kecil atau Nol. Jika $f'(x_n)$ mendekati nol, metode ini bisa divergen atau tidak terkendali. Pada x_0=1.6, kita mendapatkan $f'(1.6)$ = -0.92, yang relatif kecil, sehingga perbaikannya besar di iterasi pertama. Metode ini sangat bergantung pada titik awal. Jika titik awal dipilih jauh dari akar atau pada titik di mana turunan hampir nol, metode ini bisa melompat ke akar yang salah atau gagal konvergen. Dalam kasus ini, pemilihan x_0=1.6 tetap bekerja dengan baik.

Metode Secant mirip dengan Newton-Raphson, tetapi menggunakan dua titik awal x_0 dan x_1 untuk menghitung perkiraan akar tanpa memerlukan turunan eksplisit.

Tabel Iterasi Secant untuk Fungsi Polinomial
Fungsi yang digunakan: $f(x) = x^3 - 6x^2 + 11x - 6$
Interval awal: x_0 = 1.6, x_1 = 2.7, Toleransi error: $\epsilon = 10^{-6}$, Batas iterasi: 100 iterasi.

Iterasi	x_{n-1}	x_n	$f(x_n)$	x_{n+1} (Akar Perkiraan)
1	1.6	2.7	0.1624	2.0384
2	2.7	2.0384	0.0538	1.9806
3	2.0384	1.9806	-0.0037	2.0003
4	1.9806	2.0003	0.0006	1.9999
5	2.0003	1.9999	-0.0001	2
6	1.9999	2	0	Akar ditemukan!

Akar ditemukan pada x = 2.0000 dalam 6 iterasi. Secant lebih cepat dibanding Bisection (7 iterasi) tetapi sedikit lebih lambat dibanding Regula Falsi (4 iterasi) dan Newton-Raphson (5 iterasi). Konvergensinya superlinear tetapi tidak secepat Newton-Raphson (yang kuadratik). Keunggulan metode Secant adalah metode ini tidak memerlukan turunan fungsi seperti Newton-Raphson. Lebih cepat dibanding metode Bracketing (Bisection dan Regula Falsi) dalam kebanyakan kasus. Kelemahan metode Secant adalah metode ini bisa gagal konvergen jika perkiraan awal buruk. Tidak selalu stabil jika nilai x_n melompat jauh dari akar yang sebenarnya. Metode Secant sering kali lebih cepat daripada Bisection dan Regula Falsi, tetapi tidak seandal Newton-Raphson jika turunan tersedia.

3.3 Hasil Perhitungan Fungsi Transendental dan Pembahasan

Di bagian ini fungsi transendental $f(x) = e^x - 5$ dipakai sebagai fungsi uji untuk membandingkan metode-metode perhitungan nol fungsi. Untuk kebutuhan ini metode Bisection diterapkan dengan interval awal a=0 dan b=2.

Tabel Iterasi Bisection untuk Fungsi Transendental
Fungsi yang digunakan: $f(x) = e^x - 5$
Interval awal: a = 0, b = 2, Toleransi error: $\epsilon = 10^{-6}$, Batas iterasi: 100 iterasi.

Iterasi	a	b	$c = \frac{a+b}{2}$	$f(c)$	Interval Baru
1	0	2	1	-2.2817	[1.0000,2.0000]
2	1	2	1.5	-0.5183	[1.5000,2.0000]
3	1.5	2	1.75	0.7546	[1.5000,1.7500]
4	1.5	1.75	1.625	0.017	[1.5000,1.6250]
5	1.5	1.625	1.5625	-0.2527	[1.5625,1.6250]
6	1.5625	1.625	1.5938	-0.1192	[1.5938,1.6250]
7	1.5938	1.625	1.6094	-0.0511	[1.6094,1.6250]
8	1.6094	1.625	1.6172	-0.0172	[1.6172,1.6250]
9	1.6172	1.625	1.6211	0	Akar ditemukan!

Akar ditemukan pada x = 1.6211 dalam 9 iterasi. Nilai ini mendekati $\ln(5) \approx 1.6094$, yang merupakan akar eksak dari $e^x - 5 = 0$. Keunggulan Metode Bisection adalah selalu konvergen**,** karena interval selalu mengecil. Metode ini stabil dalam menemukan akar selama ada perubahan tanda pada $f(x)$. Kelemahan metode ini adalah lambat dibanding metode lain seperti Newton-Raphson atau Secant. Metode ini memerlukan banyak iterasi untuk mencapai akurasi tinggi.

Penerapan metode Regula Falsi untuk menghitung akar atau nol dari fungsi transendental dilakukan secara iteratif di bagian ini dan hasilnya ditunjukkan pada tabel iterasi berikut ini.

Tabel Iterasi Regula Falsi untuk Fungsi Transendental
Fungsi yang digunakan**:** $f(x) = e^x - 5$
Interval awal**:** a = 0, b = 2, Toleransi error: $\epsilon = 10^{-6}$, Batas iterasi: 100 iterasi.

Iterasi	a	b	c (Akar Perkiraan)	$f(c)$	Interval Baru
1	0	2	1.4556	-1.1522	[1.4556,2.0000]
2	1.4556	2	1.5707	-0.3489	[1.5707,2.0000]
3	1.5707	2	1.6045	-0.1037	[1.6045,2.0000]
4	1.6045	2	1.6139	-0.0302	[1.6139,2.0000]
5	1.6139	2	1.6166	-0.0087	[1.6166,2.0000]
6	1.6166	2	1.6174	-0.0025	[1.6174,2.0000]
7	1.6174	2	1.6176	-0.0007	[1.6176,2.0000]
8	1.6176	2	1.6177	-0.0002	[1.6177,2.0000]
9	1.6177	2	1.6178	0	Akar ditemukan!

Akar ditemukan pada x=1.6178x = 1.6178 dalam 9 iterasi. Nilai ini sangat dekat dengan akar eksak $\ln(5) \approx 1.6094$. Keunggulan Metode Regula Falsi lebih cepat dibanding metode Bisection karena menggunakan interpolasi linier. Metode ini juga memiliki Konvergensi lebih cepat daripada Bisection dalam banyak kasus. Meskipun demikian, kelemahan metode ini bisa terjebak dalam

konvergensi lambat jika satu sisi interval tetap tidak berubah dalam banyak iterasi. Metode Regula Falsi lebih cepat daripada Bisection dan cukup andal jika interval dipilih dengan baik.

Dengan fungsi transendental yang sama, metode Newton-Raphson diterapkan untuk menemukan akar atau nol fungsi secara iteratif, dan progress pemutakhiran nilai akar selama iterasi diperlihatkan dalam tabel itersi berikut ini.

Tabel Iterasi Newton-Raphson untuk Fungsi Transendental
Fungsi yang digunakan: $f(x) = e^x - 5$
Nilai awal: $x_0 = 0$, Toleransi error: $\epsilon = 10^{-6}$, Batas iterasi: 100 iterasi.

Iterasi	x_n	$f(x_n)$	$f'(x_n)$	x_{n+1}
1	0	-4	1	4
2	4	49.5982	54.5982	3.0904
3	3.0904	16.0324	22.0324	2.3613
4	2.3613	5.608	10.608	1.8327
5	1.8327	1.2561	6.2561	1.6311
6	1.6311	0.1019	5.1019	1.6111
7	1.6111	0.0044	5.0044	1.6094
8	1.6094	0	5	Akar ditemukan!

Akar ditemukan pada x=1.6094x = 1.6094 hanya dalam 8 iterasi. Nilai ini sama dengan $\ln(5) \approx 1.6094$. Metode ini jauh lebih cepat dibandingkan metode Bisection dan Regula Falsi. Keunggulan Metode Newton-Raphson adalah sangat cepat dengan konvergensi kuadratik dan Biasanya lebih efisien dibanding metode lain. Kelemahan metode ini adalah membutuhkan turunan $f'(x)$, yang bisa saja sulit dihitung dalam beberapa kasus. Pemilihan nilai awal yang tidak tepat dapat menyebabkan metode ini divergen. Metode Newton-Raphson sangat efisien, tetapi perlu turunan fungsi dan nilai awal yang baik agar tidak gagal konvergen.

Berbeda halnya dengan metode Newton-Raphson yang mensyaratkan adanya turunan fungsi, metode secant membutuhkan tebakan dua nilai awal yang baik untuk mencapai konvergensi optimal. Pemutakhiran nilai awal konvergen menuju ke nilai akar fungsi diperlihatkan dalam tabel iterasi berikut ini.

Tabel Iterasi Secant untuk Fungsi Transendental
Fungsi yang digunakan: $f(x) = e^x - 5$
Nilai awal: $x_0 = 0$, $x_1 = 2$, Toleransi error: $\epsilon = 10^{-6}$, Batas iterasi: 100 iterasi.

Iterasi	x_n	$f(x_n)$	x_{n+1}
1	0	-4	1.3333
2	2	2.3891	1.5264
3	1.3333	-1.2741	1.5974
4	1.5264	-0.366	1.6079
5	1.5974	-0.0778	1.6093
6	1.6079	-0.0075	1.6094
7	1.6093	-0.0002	Akar ditemukan!

Akar ditemukan pada x = 1.6094 hanya dalam 7 iterasi. Hasilnya sangat dekat dengan $\ln(5) \approx 1.6094$. Metode ini lebih cepat dibanding Bisection dan Regula Falsi, dan bahkan untuk kasus ini lebih cepat dari Newton-Raphson. Keunggulan Metode Secant adalah Tidak memerlukan turunan seperti Newton-Raphson. Konvergensi lebih cepat dibandingkan Bisection dan Regula Falsi. Kelemahan metode ini adalah bisa gagal jika dua nilai awal tidak dipilih dengan baik. Metode ini

tidak selalu lebih cepat dari Newton-Raphson. Metode Secant cukup cepat dan tidak membutuhkan turunan, tetapi membutuhkan dua nilai awal yang baik untuk konvergensi optimal.

4. KESIMPULAN

4.1 Kesimpulan

Dalam penelitian ini, empat metode numerik utama untuk pencarian akar fungsi, yaitu Bisection, Regula Falsi, Newton-Raphson, dan Secant, telah dianalisis dan dibandingkan berdasarkan kecepatan konvergensi, akurasi solusi, serta kestabilan terhadap variasi fungsi dan nilai awal, dengan kesimpulan sebagai berikut.

1. Dari aspek Konvergensi dan Efisiensi, Newton-Raphson memiliki konvergensi tercepat dengan sifat konvergensi kuadratik, tetapi membutuhkan turunan fungsi yang dapat menjadi kendala dalam beberapa kasus. Metode Secant menawarkan kecepatan konvergensi yang hampir setara dengan Newton-Raphson, namun tanpa memerlukan turunan fungsi, menjadikannya pilihan yang lebih fleksibel. Regula Falsi memiliki tingkat konvergensi lebih cepat dibanding Bisection, tetapi dapat mengalami stagnasi pada kasus tertentu jika salah satu batas interval tetap tidak berubah dalam beberapa iterasi. Bisection adalah metode yang paling stabil dan selalu konvergen, tetapi memiliki tingkat konvergensi paling lambat dibandingkan metode lainnya.
2. Dari aspek Akurasi dan Kestabilan, Semua metode berhasil menemukan akar dengan tingkat akurasi yang tinggi, tetapi kecepatan mencapai akurasi tersebut sangat bergantung pada metode yang digunakan. Metode Newton-Raphson dan Secant lebih sensitif terhadap nilai awal, sehingga pemilihan titik awal yang tidak tepat dapat menyebabkan divergensi atau perlambatan konvergensi. Bisection dan Regula Falsi lebih stabil, tetapi memerlukan iterasi lebih banyak untuk mencapai toleransi yang diinginkan.
3. Dari penerapan pada Fungsi Polinomial dan Transendental, untuk fungsi polinomial, Newton-Raphson dan Secant terbukti lebih efisien karena dapat dengan cepat mempersempit estimasi akar. Sementara Untuk fungsi transendental**,** Newton-Raphson dapat tetap efisien jika turunannya mudah dihitung, tetapi jika tidak, metode Secant menjadi alternatif yang lebih praktis. Bisection dan Regula Falsi tetap menjadi pilihan yang dapat diandalkan untuk fungsi apa pun, terutama ketika kestabilan dan kepastian konvergensi lebih diutamakan dibanding kecepatan.

4.2 Implikasi dan Rekomendasi

Jika kecepatan sangat penting dan turunan tersedia**,** Newton-Raphson adalah pilihan terbaik. Jika turunan sulit dihitung**,** metode Secant menjadi alternatif yang efisien. Jika kestabilan dan kepastian konvergensi lebih diutamakan**,** Bisection atau Regula Falsi lebih direkomendasikan, terutama untuk fungsi yang kompleks atau memiliki banyak akar. Dalam praktiknya, pemilihan metode yang optimal harus mempertimbangkan sifat fungsi yang diuji, ketersediaan turunan, serta kecepatan dan stabilitas yang diinginkan.

RFERENSI

Abu Bakar, N., Monsi, M., & Hassan, N. (2012). An improved parameter regula falsi method for enclosing a zero of a function. *Applied Mathematical Sciences*, *6*(25–28), 1347–1361.

Badr, E., Attiya, H., & El Ghamry, A. (2022). Novel hybrid algorithms for root determining using advantages of open methods and bracketing methods. *Alexandria Engineering Journal*, *61*(12), 11579–11588. https://doi.org/10.1016/j.aej.2022.05.007

Intep, S. (2018). A review of bracketing methods for finding zeros of nonlinear functions. *Applied Mathematical Sciences*, *12*(3), 137–146. https://doi.org/10.12988/ams.2018.811

Juhari, J. (2021). On the Modification of Newton-Secant Method in Solving Nonlinear Equations for Multiple Zeros of Trigonometric Function. *CAUCHY: Jurnal Matematika Murni Dan Aplikasi*, *7*(1), 84–96. https://doi.org/10.18860/ca.v7i1.12934

Mathworks. (2013). *Matlab Function Reference, R2013a*. *2*, 10–11.

Muhamad Iradat Achmad. (2006). Perbandingan Teknik Mencari Nol Fungsi. *JTE, UGM, Penerbit MAHANDIA*. https://doi.org/https://doi.org/10.5281/zenodo.14791321

Nadhim, A. I., & Al-Jilawi, A. S. (2022). The bridge between Newton's method and Newton-Raphson's method in numerical optimization. *International Journal of Health Sciences*, *6*(March), 5249–5267. https://doi.org/10.53730/ijhs.v6ns1.6042

Nguyen, T. (2021). *The convergence of the Regula Falsi method*. http://arxiv.org/abs/2109.03523

Osman, S. A. A., Abdel Rahman, A. R. A. R. A. G., Mohammed, A. O. A., & Ali, A. A. A. (2022). Solution of Non-linear Equations using Bisection Method by New Technical Method.)26(*6*, □□□□□□□□ □□□□□□□□ □□□□□□ □□□□□□□ □□□□□□, 261–274. https://doi.org/10.21608/jasep.2022.216291