Indonesian Journal of Multidisciplinary on Social and Technology
Homepage: https://journal.ilmudata.co.id/index.php/ijmst
Vol. 5 No. 1 (2026) pp: 826-835
P-ISSN: 2986-6790, e-ISSN: 2986-6782

Perbandingan Akurasi dan Konvergensi Metode Bisection dan Newton-Raphson pada Persamaan Nonlinear

Yuliadi Erdani, Muhammad Ihsan Hanafiah, Maliha Khumaira, Rainier Timothy, Reffy Danadyaksa
Program Studi Teknologi Rekayasa Otomasi, Jurusan Teknik Otomasi Manufaktur dan Mekatronika, Politeknik Manufaktur Bandung
yul_erdani@yahoo.com, 225441038@mhs.polman-bandung.ac.id, 225441036@mhs.polman-bandung.ac.id, 225441045@mhs.polman-bandung.ac.id, 225441046@mhs.polman-bandung.ac.id

Abstrak

Pencarian akar persamaan nonlinear merupakan masalah dasar dalam metode numerik yang sering muncul di bidang teknik, sains, dan pemodelan komputer karena banyak persamaan tidak dapat diselesaikan dengan rumus langsung. Penelitian ini bertujuan membandingkan metode Bisection dan Newton-Raphson dari segi ketelitian, konvergensi, jumlah iterasi, dan biaya evaluasi fungsi, serta menguji ketahanannya terhadap pilihan nilai awal dan akar ganda. Kedua metode dibuat dengan bahasa pemrograman Python dan diuji pada tiga fungsi nonlinear dengan tiga tingkat toleransi, yaitu 10−3, 10−6, dan 10−9. Pada setiap pengujian dicatat akar hampiran, jumlah iterasi, error mutlak terhadap akar acuan, dan waktu hitung, sedangkan orde konvergensi Newton-Raphson ditaksir dengan aritmetika presisi tinggi. Pada toleransi 10−9, Newton-Raphson memerlukan 4–6 iterasi, sedangkan Bisection memerlukan 30–31 iterasi yang sama dengan prediksi teoretisnya. Newton-Raphson juga memerlukan evaluasi fungsi yang lebih sedikit, yaitu 8–12 dibandingkan 32–33. Orde konvergensi Newton-Raphson yang terukur mendekati 2, tetapi metode ini peka terhadap nilai awal: pada rentang −3 sampai 3, hanya 53,4% nilai awal pada fungsi ketiga yang menuju akar acuan. Pada akar ganda, konvergensi Newton-Raphson turun menjadi linear dengan rasio error sekitar 0,5, sedangkan Bisection tidak dapat dijalankan karena tidak ada perubahan tanda. Bisection cocok bila keandalan menjadi prioritas, sedangkan Newton-Raphson cocok bila turunan tersedia dan nilai awal dekat dengan akar. Hasil ini dapat dipakai sebagai pedoman memilih metode.

Kata kunci: Metode Bisection, Metode Newton-Raphson, Akar Persamaan Nonlinear, Konvergensi, Metode Numerik

Abstract

Finding the roots of nonlinear equations is a fundamental numerical problem in engineering, science, and computer modeling, because many equations cannot be solved with direct formulas. This study aims to compare the Bisection and Newton-Raphson methods in terms of accuracy, convergence, number of iterations, and function evaluation cost, and to test their robustness to the choice of initial values and to multiple roots. Both methods were implemented in Python and tested on three nonlinear functions at tolerances of 10−3, 10−6, and 10−9. Each test recorded the approximate root, number of iterations, absolute error relative to a reference root, and computation time, while the convergence order of Newton-Raphson was estimated with high-precision arithmetic. At a tolerance of 10−9, Newton-Raphson required 4–6 iterations, whereas Bisection required 30–31, matching its theoretical prediction. Newton-Raphson also needed fewer function evaluations (8–12 versus 32–33). Its measured convergence order was close to 2, but it was sensitive to the initial value: over the range −3 to 3, only 53.4% of the initial values for the third function led to the reference root. For a multiple root, its convergence dropped to linear with an error ratio of about 0.5, whereas Bisection could not be run because there was no sign change. Bisection suits cases where reliability is the priority, whereas Newton-Raphson suits cases where the derivative is available and the initial value is close to the root. These results can guide method selection.

Keywords: Bisection Method, Newton-Raphson Method, Nonlinear Equation Roots, Convergence, Numerical Methods

1. Pendahuluan

Banyak masalah di bidang teknik, sains, dan pemodelan komputer berujung pada persamaan nonlinear yang akarnya sulit, atau bahkan tidak mungkin, dicari dengan rumus langsung (Maure & Tulan, 2024; Wulandari et al., 2026). Pada kondisi seperti ini, metode numerik dipakai untuk mendapatkan nilai perkiraan akar melalui perhitungan berulang (iterasi) sampai hasilnya cukup teliti (Pandia & Sitepu, 2021).

Dua metode yang paling umum dipakai adalah Bisection dan Newton-Raphson. Beberapa penelitian di Indonesia sudah membandingkan metode pencarian akar. Maure dan Tulan (2024) menguji Bisection, Newton-Raphson, dan Secant pada fungsi polinomial, trigonometri, dan eksponensial, dan menemukan bahwa Newton-Raphson memberi error mutlak paling kecil pada fungsi polinomial dan trigonometri. Wulandari et al. (2026) menunjukkan bahwa Secant berhasil menemukan akar pada semua fungsi ujinya, dengan 5 sampai 7 iterasi. Erviana et al. (2023) meneliti efisiensi Newton-Raphson dan Secant menggunakan Python, sedangkan Mumtazi et al. (2024) membandingkan metode Steffensen dan Secant menggunakan PHP. Rozi dan Rarasati (2022) serta Sholihah et al. (2025) memanfaatkan Microsoft Excel (dan Maple) untuk menyelesaikan persamaan nonlinear, dan Syaharuddin et al. (2026) meneliti perbandingan metode untuk sistem persamaan nonlinear.

Di tingkat internasional, perhatian banyak tertuju pada metode gabungan (hibrida) yang memadukan kestabilan metode pengapit seperti Bisection dengan kecepatan metode terbuka seperti Newton-Raphson (Badr et al., 2021; Badr et al., 2022). Beberapa penelitian lain mengusulkan algoritma hibrida baru dan algoritma iteratif yang lebih cepat (Qureshi et al., 2024; Vivas-Cortez et al., 2023), mengubah metode klasik menjadi program Python yang efisien (Romadani, 2025), menggabungkan varian metode Newton dan Halley (Ali & Fathi, 2025), membandingkan Bisection dengan Newton-Raphson (Molawade et al., 2022), serta menggabungkan Newton dan Secant (Sharma & Chouhan, 2026). Hampir semua penelitian tersebut memakai metode klasik seperti Bisection dan Newton-Raphson sebagai pembanding, sehingga pemahaman yang jelas tentang kinerja keduanya tetap menjadi dasar yang penting.

Penelitian ini membandingkan kedua metode tersebut secara terukur pada tiga fungsi uji dan tiga tingkat toleransi. Pembandingan dilakukan berdasarkan empat hal, yaitu akar hampiran, jumlah iterasi, error mutlak terhadap akar acuan, dan waktu hitung. Hasilnya diharapkan dapat membantu pembaca memilih metode yang tepat untuk suatu persoalan pencarian akar.

Selain perbandingan dasar tersebut, penelitian ini menambahkan tiga kajian pendukung agar kesimpulan tidak bertumpu pada jumlah iterasi saja. Pertama, jumlah iterasi Bisection dibandingkan dengan prediksi teoretisnya dan orde konvergensi Newton-Raphson ditaksir dari data, sehingga hasil program dapat dicocokkan dengan teori. Kedua, biaya kedua metode dinyatakan dalam jumlah evaluasi fungsi, yaitu ukuran yang tidak bergantung pada komputer, karena waktu hitung pada fungsi sederhana mudah terganggu. Ketiga, ketahanan Newton-Raphson diuji terhadap pilihan nilai awal dan terhadap akar ganda. Dengan demikian, pertanyaan yang dijawab adalah seberapa besar perbedaan kedua metode dalam iterasi dan ketelitian, apakah perilakunya sesuai dengan teori konvergensi, dan dalam kondisi apa metode yang cepat menjadi kurang andal.

2. Metode Penelitian

Penelitian ini bersifat eksperimen komputasi. Kedua metode dibuat dengan bahasa pemrograman Python dan diuji pada tiga fungsi nonlinear yang berbeda bentuknya, seperti terlihat pada Tabel 1.

Tabel 1. Fungsi uji, interval awal, dan nilai awal.

Kode	Fungsi uji	Interval awal	Nilai awal x_0
f_1	$f(x) = x^3 - x - 2$	[1, 2]	1,5
f_2	$f(x) = \cos(x) - x$	[0, 1]	0,5
f_3	$f(x) = x^2 - 4\sin(x)$	[1, 3]	1,5

2.1. Metode Bisection

Metode Bisection (bagi dua) bekerja dengan membagi dua sebuah interval berulang kali. Jika fungsi f(x) kontinu pada interval [a, b] dan nilai fungsi di kedua ujungnya berbeda tanda, maka di dalam interval itu pasti ada minimal satu akar. Pada setiap iterasi, titik tengah interval dihitung dengan

$$c = \frac{a+b}{2} \quad (1)$$

DOI: https://doi.org/10.69693/ijmst.v5i1.14655
Lisensi: Creative Commons Attribution 4.0 International (CC BY 4.0)

dengan *c* adalah titik tengah interval, serta *a* dan *b* masing-masing adalah batas bawah dan batas atas interval, dengan syarat awal

$$f(a) \cdot f(b) < 0 \tag{2}$$

Nilai f(c) kemudian dipakai untuk memilih setengah interval yang masih memuat perubahan tanda, sehingga panjang interval mengecil separuhnya pada setiap langkah. Karena itu, jumlah iterasi yang dibutuhkan untuk mencapai toleransi tol dapat diperkirakan dengan

$$n \geq log_2\left(\frac{b-a}{tol}\right) \tag{3}$$

dengan *n* adalah jumlah iterasi dan tol adalah toleransi. Metode ini konvergen secara linear (orde 1). Artinya, iterasinya relatif banyak, tetapi hasilnya pasti mendekati akar selama syarat awal terpenuhi (Burden et al., 2016).

2.2. Metode Newton-Raphson

Metode Newton-Raphson memakai garis singgung kurva di titik terakhir untuk menebak akar berikutnya. Rumus iterasinya adalah

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \tag{4}$$

dengan x_n adalah hampiran akar pada iterasi ke-*n*, $f(x_n)$ adalah nilai fungsi di titik tersebut, dan $f'(x_n)$ adalah turunan pertama fungsi. Jika nilai awal cukup dekat dengan akar dan turunan f′(x) di sekitar akar tidak nol, metode ini konvergen secara kuadratik (orde 2) (Sharma & Chouhan, 2026; Burden et al., 2016). Kelemahannya, metode ini membutuhkan rumus turunan fungsi dan bisa gagal (divergen) jika nilai awal jauh dari akar atau turunannya mendekati nol (Sharma & Chouhan, 2026).

2.3. Rancangan Pengujian

Bisection memakai interval awal dan Newton-Raphson memakai nilai awal x_0. Setiap metode diuji pada tiga tingkat toleransi, yaitu 10^{-3}, 10^{-6}, dan 10^{-9}, dengan batas 200 iterasi untuk mengantisipasi kasus yang tidak konvergen. Iterasi Bisection berhenti ketika panjang interval lebih kecil dari toleransi:

$$b - a < tol \tag{5}$$

dengan *a* dan *b* adalah batas bawah dan batas atas interval pada iterasi berjalan. Sedangkan iterasi Newton-Raphson berhenti ketika selisih dua hampiran berurutan lebih kecil dari toleransi:

$$|x_{n+1} - x_n| < tol \tag{6}$$

dengan x_{n+1} dan x_n adalah dua hampiran berurutan. Akar acuan diperoleh dari metode Bisection dengan 200 iterasi, sehingga jauh lebih teliti daripada hasil kedua metode yang diuji. Ketelitian setiap metode dinyatakan dengan error mutlak terhadap akar acuan:

$$E_{abs} = |x_n - x_{acuan}| \tag{7}$$

dengan E_{abs} adalah error mutlak, x_n adalah akar hampiran, dan x_{acuan} adalah akar acuan. Waktu hitung diukur dengan penghitung kinerja (*performance counter*) Python dalam skala mikrodetik, hanya untuk proses iterasi tanpa waktu mencetak atau menyimpan data, lalu dinyatakan dalam milidetik. Empat hal dicatat pada setiap pengujian: (1) akar hampiran, (2) jumlah iterasi, (3) error mutlak, dan (4) waktu hitung. Semua percobaan dijalankan di lingkungan komputer yang sama agar perbandingan waktu adil.

2.4. Algoritma

Kedua metode ditulis dalam bentuk algoritma pada Algoritma 1 dan Algoritma 2 agar dapat direproduksi tanpa listing program. Pada Bisection, hampiran akar yang dilaporkan adalah titik tengah *c* pada iterasi terakhir,

DOI: https://doi.org/10.69693/ijmst.v5i1.14655
Lisensi: Creative Commons Attribution 4.0 International (CC BY 4.0)

sedangkan pada Newton-Raphson hampiran yang dilaporkan adalah x_{n+1} terakhir. Turunan fungsi uji dihitung secara analitik, yaitu $f_1'(x) = 3x^2 - 1$, $f_2'(x) = -\sin(x) - 1$, dan $f_3'(x) = 2x - 4\cos(x)$.

Algoritma 1. Metode Bisection.

Masukan: fungsi *f*, batas bawah *a*, batas atas *b*, toleransi tol, batas iterasi n_{maks}

Keluaran: akar hampiran *c* dan jumlah iterasi *n*

jika $f(a)\cdot f(b) \geq 0$ maka hentikan (syarat perubahan tanda tidak terpenuhi)

$n \leftarrow 0$; $c \leftarrow (a + b)/2$

selama $(b - a) \geq$ tol dan $n < n_{maks}$ lakukan

    $c \leftarrow (a + b)/2$; $n \leftarrow n + 1$

    jika $f(a)\cdot f(c) < 0$ maka $b \leftarrow c$, selain itu $a \leftarrow c$

kembalikan *c* dan *n*

Algoritma 2. Metode Newton-Raphson.

Masukan: fungsi *f*, turunan *f'*, nilai awal x_0, toleransi tol, batas iterasi n_{maks}

Keluaran: akar hampiran *x* dan jumlah iterasi *n*

$n \leftarrow 0$; $x \leftarrow x_0$

selama $n < n_{maks}$ lakukan

    jika $f'(x) = 0$ maka hentikan (gagal)

    $x_{baru} \leftarrow x - f(x)/f'(x)$; $n \leftarrow n + 1$

    jika $|x_{baru} - x| <$ tol maka kembalikan x_{baru} dan *n*

    $x \leftarrow x_{baru}$

hentikan: tidak konvergen dalam n_{maks} iterasi

Semua perhitungan memakai bilangan titik-mengambang presisi ganda (*double precision*) bawaan Python. Dengan menyimpan nilai *f* di ujung interval, satu iterasi Bisection hanya memerlukan satu evaluasi *f* baru, ditambah dua evaluasi awal untuk memeriksa Persamaan (2). Satu iterasi Newton-Raphson memerlukan satu evaluasi *f* dan satu evaluasi *f'*. Perbedaan ini dipakai untuk menghitung biaya evaluasi pada Subbab 2.5.

2.5. Rancangan Pengujian Tambahan

Pengujian tambahan dilakukan dalam empat langkah. Langkah pertama membandingkan jumlah iterasi Bisection dengan prediksi Persamaan (3). Langkah kedua menaksir orde konvergensi Newton-Raphson dari barisan error. Jika $e_k = |x_k - r|$ adalah error pada iterasi ke-*k* terhadap akar *r*, orde konvergensi ditaksir dengan

$$p_k \approx \frac{\ln\left(\frac{e_{k+1}}{e_k}\right)}{\ln\left(\frac{e_k}{e_{k-1}}\right)} \tag{8}$$

DOI: https://doi.org/10.69693/ijmst.v5i1.14655
Lisensi: Creative Commons Attribution 4.0 International (CC BY 4.0)

dengan p_k adalah taksiran orde konvergensi pada iterasi ke-*k*, serta e_{k-1}, e_k, dan e_{k+1} adalah tiga error berurutan. Untuk akar sederhana, teori menyatakan bahwa Newton-Raphson berorde 2 dan rasio error mendekati sebuah konstanta (Burden et al., 2016), yaitu

$$\lim_{k\to\infty} \frac{e_{k+1}}{e_k^2} = \left|\frac{f''(r)}{2f'(r)}\right| \tag{9}$$

dengan $f''(r)$ dan $f'(r)$ adalah turunan kedua dan turunan pertama fungsi di akar *r*. Karena error Newton-Raphson mencapai ketelitian mesin hanya dalam beberapa iterasi, taksiran ini dihitung ulang dengan aritmetika presisi tinggi (60 digit desimal) mulai dari nilai awal yang sama dengan Tabel 1.

Langkah ketiga menghitung biaya dalam jumlah evaluasi fungsi. Bila c_f dan $c_{f'}$ adalah biaya satu evaluasi *f* dan *f'*, Newton-Raphson lebih murah daripada Bisection selama

$$\frac{c_{f'}}{c_f} < \frac{n_B + 2}{n_N} - 1 \tag{10}$$

dengan n_B dan n_N masing-masing adalah jumlah iterasi Bisection dan Newton-Raphson, dan konstanta 2 berasal dari dua evaluasi awal $f(a)$ dan $f(b)$. Langkah keempat menguji ketahanan Newton-Raphson. Untuk sensitivitas terhadap nilai awal, setiap fungsi uji dijalankan dari 601 nilai awal x_0 pada rentang −3 sampai 3 dengan jarak 0,01, toleransi 10^{-9}, dan batas 200 iterasi. Hasil setiap titik digolongkan menuju akar acuan, menuju akar lain, atau tidak konvergen dalam batas iterasi. Untuk akar ganda dipakai fungsi tambahan $f_4(x) = x^3 - 3x + 2 = (x - 1)^2(x + 2)$, yang mempunyai akar ganda pada x = 1 dan akar sederhana pada x = −2, dengan nilai awal $x_0 = 2$.

3. Hasil dan Pembahasan

3.1. Hasil Pengujian

Akar acuan fungsi f_1: $x^3 - x - 2 = 0$ pada interval [1, 2] adalah 1,52137971. Hasil kedua metode ditampilkan pada Tabel 2.

Tabel 2. Hasil pengujian metode Bisection dan Newton-Raphson pada fungsi f_1.

Toleransi	Metode	Akar hampiran	Iterasi	Error absolut	Waktu (ms)
1×10^{-3}	Bisection	1,52050781	10	$8{,}719\times10^{-4}$	0,0110
	Newton-Raphson	1,52137981	2	$9{,}916\times10^{-8}$	0,0038
1×10^{-6}	Bisection	1,52138042	20	$7{,}177\times10^{-7}$	0,0090
	Newton-Raphson	1,52137971	3	$7{,}772\times10^{-15}$	0,0025
1×10^{-9}	Bisection	1,52137971	30	$3{,}548\times10^{-10}$	0,0098
	Newton-Raphson	1,52137971	4	$2{,}220\times10^{-16}$	0,0019

Akar acuan fungsi f_2: $\cos(x) - x = 0$ pada interval [0, 1] adalah 0,73908513. Hasilnya ditampilkan pada Tabel 3.

Tabel 3. Hasil pengujian metode Bisection dan Newton-Raphson pada fungsi f_2.

Toleransi	Metode	Akar hampiran	Iterasi	Error absolut	Waktu (ms)
1×10^{-3}	Bisection	0,73925781	10	$1{,}727\times10^{-4}$	0,0044
	Newton-Raphson	0,73908513	3	$7{,}056\times10^{-10}$	0,0038
1×10^{-6}	Bisection	0,73908520	20	$6{,}423\times10^{-8}$	0,0055
	Newton-Raphson	0,73908513	4	0	0,0016
1×10^{-9}	Bisection	0,73908513	30	$2{,}769\times10^{-11}$	0,0079
	Newton-Raphson	0,73908513	4	0	0,0016

DOI: https://doi.org/10.69693/ijmst.v5i1.14655
Lisensi: Creative Commons Attribution 4.0 International (CC BY 4.0)

Akar acuan fungsi f_3: $x^2 - 4\sin(x) = 0$ pada interval [1, 3] adalah 1,93375376. Hasilnya ditampilkan pada Tabel 4.

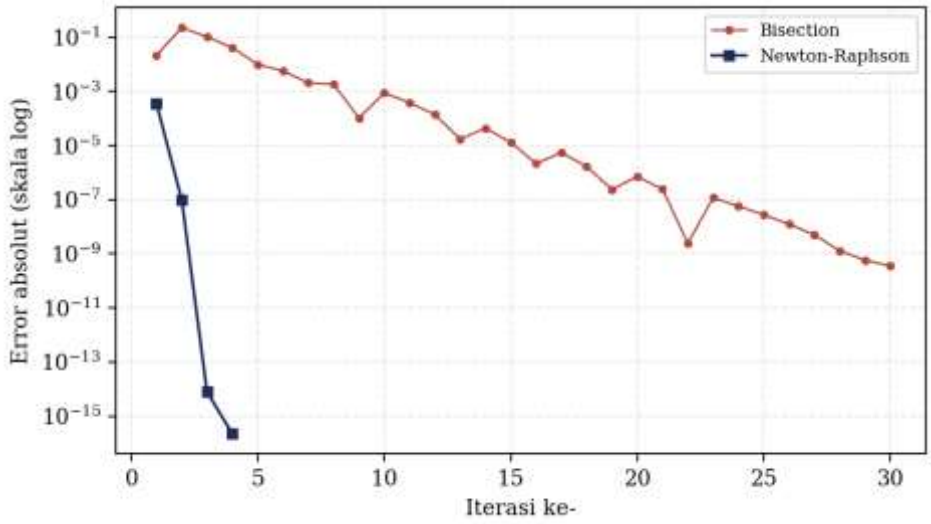
Tabel 4. Hasil pengujian metode Bisection dan Newton-Raphson pada fungsi f_3.

Toleransi	Metode	Akar hampiran	Iterasi	Error absolut	Waktu (ms)
1×10^{-3}	Bisection	1,93457031	11	$8{,}165\times10^{-4}$	0,0049
	Newton-Raphson	1,93375378	4	$1{,}696\times10^{-8}$	0,0031
1×10^{-6}	Bisection	1,93375301	21	$7{,}492\times10^{-7}$	0,0075
	Newton-Raphson	1,93375376	5	0	0,0025
1×10^{-9}	Bisection	1,93375376	31	$4{,}985\times10^{-10}$	0,0104
	Newton-Raphson	1,93375376	6	$2{,}220\times10^{-16}$	0,0025

Pada ketiga tabel, akar yang dihasilkan sama dengan akar acuan setelah toleransi diperketat. Hasil untuk f_1 (akar 1,52138) dan f_2 (akar 0,739085) juga sama dengan akar yang dilaporkan Wulandari et al. (2026) untuk fungsi polinomial dan trigonometri, sehingga program yang kami buat dapat dipercaya.

3.2. Analisis Konvergensi

Gambar 1 memperlihatkan bagaimana error mengecil dari iterasi ke iterasi pada fungsi f_1 dengan toleransi 10^{-9}. Sumbu vertikal memakai skala logaritmik, dan error yang sudah mencapai ketelitian mesin ditampilkan pada batas bawah grafik (10^{-16}).



Gambar 1. Error absolut (skala log) terhadap iterasi pada fungsi f_1 dengan toleransi 10^{-9}.

Newton-Raphson menurunkan error paling tajam, sesuai dengan sifat konvergensi kuadratiknya, dan hanya butuh 4 iterasi untuk mencapai ketelitian mesin. Bisection menurun jauh lebih lambat dan bergelombang. Error Bisection tidak selalu turun di setiap langkah karena yang selalu mengecil separuh adalah panjang intervalnya, sedangkan letak titik tengah terhadap akar sebenarnya bisa berubah-ubah. Pola ini sejalan dengan hasil Sharma dan Chouhan (2026), yang juga menemukan bahwa Newton-Raphson paling cepat.

Orde konvergensi Newton-Raphson ditaksir dengan Persamaan (8) dan hasilnya disajikan pada Tabel 5. Taksiran orde pada ketiga fungsi menuju 2,00 setelah dua sampai tiga iterasi, sesuai dengan orde kuadratik menurut teori. Taksiran awal belum stabil pada f_2 dan f_3 (2,10 dan 3,27) karena nilai awal masih relatif jauh dari akar sehingga perilaku asimtotik belum terbentuk.

DOI: https://doi.org/10.69693/ijmst.v5i1.14655
Lisensi: Creative Commons Attribution 4.0 International (CC BY 4.0)

Tabel 5. Taksiran orde konvergensi Newton-Raphson dengan presisi 60 digit.

Fungsi	p_1	p_2	p_3	p_4	e_5/e_4^2 (teramati)	$\lvert f''(r)/2f'(r)\rvert$ (teoretis)
f_1	2,01	2,00	2,00	2,00	0,7679	0,7679
f_2	2,10	2,00	2,00	2,00	0,2208	0,2208
f_3	3,27	1,91	2,00	2,00	0,5427	0,5427

Rasio e_5/e_4^2 pada Tabel 5 sama dengan konstanta teoretis pada Persamaan (9) sampai empat angka desimal, sehingga program yang dipakai konsisten dengan teori. Orde 1 pada Bisection terlihat dari jumlah iterasinya: setiap pengetatan toleransi sebesar 10^3 kali menambah tepat 10 iterasi (Tabel 2–4), sesuai dengan $\log_2(10^3) \approx 9{,}97$, sedangkan pada Newton-Raphson penambahannya paling banyak satu iterasi.

3.3. Perbandingan Efisiensi

Berdasarkan Tabel 2–4, pada toleransi 10^{-9} Newton-Raphson membutuhkan 4–6 iterasi dan Bisection 30–31 iterasi pada semua fungsi uji. Hal ini menegaskan keunggulan Newton-Raphson dalam jumlah iterasi ketika turunan tersedia dan nilai awal cukup dekat dengan akar. Pada toleransi 10^{-3}, Newton-Raphson juga memberi error mutlak terkecil pada ketiga fungsi, sejalan dengan temuan Maure dan Tulan (2024).

Dari sisi waktu, kedua metode sangat cepat, yaitu hanya beberapa hingga belasan mikrodetik, karena fungsi ujinya sederhana. Dengan waktu sekecil ini, hasil pengukuran mudah terpengaruh gangguan kecil di komputer, sehingga perbedaan waktu antarmetode yang tipis, misalnya selisih waktu Newton-Raphson dan Bisection pada f_2 dengan toleransi 10^{-3} yang sangat tipis padahal jumlah iterasinya berbeda jauh, tidak perlu ditafsirkan berlebihan. Secara umum, waktu total lebih ditentukan oleh jumlah iterasi daripada beban hitung tiap iterasi. Untuk fungsi yang jauh lebih rumit, perbedaan beban per iterasi akan lebih terasa, misalnya ketika menghitung turunan sangat mahal sehingga beban hitung tiap iterasi Newton-Raphson jauh lebih besar daripada Bisection (Chapra & Canale, 2015).

Karena waktu hitung pada fungsi sederhana mudah terganggu, biaya kedua metode juga dinyatakan dalam jumlah evaluasi fungsi pada Tabel 6. Jumlah iterasi Bisection pada semua sembilan kasus sama dengan prediksi Persamaan (3), yaitu 10, 20, dan 30 iterasi untuk f_1 dan f_2 serta 11, 21, dan 31 iterasi untuk f_3. Pada toleransi 10^{-9}, Bisection memerlukan 32 sampai 33 evaluasi fungsi, sedangkan Newton-Raphson memerlukan 8 evaluasi untuk f_1 dan f_2 serta 12 evaluasi untuk f_3.

Tabel 6. Jumlah iterasi dan evaluasi fungsi kedua metode.

Fungsi	Toleransi	Iterasi Bisection	Evaluasi f Bisection	Iterasi Newton	Evaluasi f dan f' Newton	Batas $c_{f'}/c_f$
f_1	10^{-3}	10	12	2	4	5,00
f_1	10^{-6}	20	22	3	6	6,33
f_1	10^{-9}	30	32	4	8	7,00
f_2	10^{-3}	10	12	3	6	3,00
f_2	10^{-6}	20	22	4	8	4,50
f_2	10^{-9}	30	32	4	8	7,00
f_3	10^{-3}	11	13	4	8	2,25
f_3	10^{-6}	21	23	5	10	3,60
f_3	10^{-9}	31	33	6	12	4,50

Kolom terakhir Tabel 6 menunjukkan batas rasio $c_{f'}/c_f$ dari Persamaan (10). Pada toleransi 10^{-9}, Newton-Raphson tetap lebih murah selama satu evaluasi turunan tidak lebih mahal dari 7 kali (f_1 dan f_2) atau 4,5 kali (f_3) satu evaluasi fungsi. Batas ini lebih ketat pada toleransi longgar, yaitu 2,25 sampai 5,0 kali pada 10^{-3}, karena selisih jumlah iterasi kedua metode belum sebesar pada toleransi ketat. Jadi, turunan analitik yang sederhana seperti pada ketiga fungsi uji membuat Newton-Raphson lebih murah, tetapi bila turunan harus dihampiri secara numerik atau sangat mahal dihitung, keunggulan itu dapat hilang.

Bisection unggul dalam keandalan: metode ini selalu konvergen pada semua kasus uji, tidak butuh turunan, dan tidak butuh nilai awal yang dekat dengan akar. Karena itu, Bisection cocok dipakai sebagai langkah awal untuk

DOI: https://doi.org/10.69693/ijmst.v5i1.14655
Lisensi: Creative Commons Attribution 4.0 International (CC BY 4.0)

mempersempit lokasi akar sebelum dilanjutkan dengan metode yang lebih cepat, sebuah gagasan yang juga mendasari metode hibrida (Badr et al., 2021; Badr et al., 2022). Newton-Raphson paling efisien dalam jumlah iterasi, tetapi nilai awal yang buruk dapat membuatnya divergen atau menuju akar yang tidak diinginkan (Molawade et al., 2022; Sharma & Chouhan, 2026).

Pola yang sama muncul pada ketiga fungsi uji yang bentuknya berbeda, yaitu polinomial pangkat tiga, gabungan trigonometri dan linear, serta gabungan kuadrat dan trigonometri. Ini menunjukkan bahwa perbandingan kinerja kedua metode bukan kebetulan pada satu bentuk fungsi saja. Meski begitu, ketiga fungsi utama hanya mempunyai akar sederhana, sehingga ketahanan kedua metode terhadap nilai awal dan akar ganda diperiksa terpisah pada Subbab 3.4 dan 3.5.

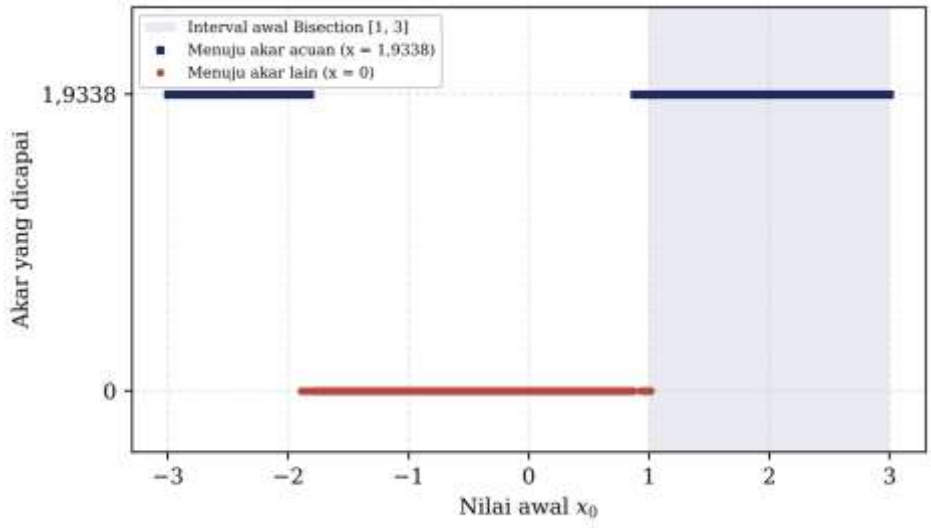
3.4. Sensitivitas terhadap Nilai Awal

Hasil pada Tabel 2–4 diperoleh dari satu nilai awal untuk setiap fungsi. Tabel 7 merangkum perilaku Newton-Raphson dari 601 nilai awal pada rentang −3 sampai 3 dengan toleransi 10^{-9} dan batas 200 iterasi. Pada f_1, seluruh nilai awal menuju akar acuan, walaupun jumlah iterasinya berkisar antara 3 dan 73 dengan median 10.

Tabel 7. Hasil Newton-Raphson dari 601 nilai awal pada rentang −3 sampai 3.

Fungsi	Titik uji	Menuju akar acuan	Menuju akar lain	Tidak konvergen	Iterasi menuju akar acuan (min / median / maks)
f_1	601	601 (100,0%)	0	0	3 / 10 / 73
f_2	601	573 (95,3%)	0	28 (4,7%)	3 / 6 / 199
f_3	601	321 (53,4%)	280 (46,6%)	0	3 / 6 / 16

Pada f_2, 573 dari 601 titik (95,3%) menuju akar acuan dan 28 titik (4,7%) tidak konvergen dalam 200 iterasi. Semua kegagalan terjadi pada $x_0 \leq -0{,}75$ dan tersebar di sekitar $x = -\pi/2 \approx -1{,}571$, yaitu titik dengan $f_2'(x) = 0$; di titik itu garis singgung mendatar sehingga langkah pertama melempar hampiran jauh dari akar. Pada f_3 tidak ada kegagalan, tetapi hanya 321 titik (53,4%) menuju akar acuan 1,9338, sedangkan 280 titik (46,6%) menuju akar lain, yaitu $x = 0$. Seluruh nilai awal dari −1,81 sampai 0,87 menuju $x = 0$, ditambah beberapa titik tersebar di sekitar batasnya sampai $x_0 = 1{,}01$, seperti terlihat pada Gambar 2.



Gambar 2. Akar yang dicapai Newton-Raphson pada fungsi f_3 untuk nilai awal x_0 dari −3 sampai 3.

Hasil pada f_3 penting karena akar $x = 0$ berada di luar interval awal Bisection [1, 3]. Bila nilai awal dipilih di dalam interval yang sama (201 titik dengan jarak 0,01), 199 titik menuju akar acuan dan hanya $x_0 = 1{,}00$ dan $x_0 = 1{,}01$ yang menuju $x = 0$; pada f_1 dan f_2, seluruh 101 titik dalam interval masing-masing menuju akar acuan. Dengan

DOI: https://doi.org/10.69693/ijmst.v5i1.14655
Lisensi: Creative Commons Attribution 4.0 International (CC BY 4.0)

demikian, interval yang sudah diapit memberi petunjuk nilai awal yang relatif aman, tetapi tidak menjamin karena satu langkah Newton-Raphson dari dekat batas interval dapat keluar dari interval. Bisection tidak mengalami masalah ini karena titik tengahnya selalu berada di dalam interval yang masih memuat akar.

3.5. Kasus Akar Ganda

Ketiga fungsi uji utama hanya mempunyai akar sederhana. Untuk melihat perilaku pada akar ganda, kedua metode diterapkan pada $f_4(x) = x^3 - 3x + 2$. Bisection tidak dapat dijalankan pada interval yang hanya memuat akar ganda $x = 1$, misalnya [0, 2], karena $f_4(0) \cdot f_4(2) = 8 > 0$ sehingga syarat Persamaan (2) tidak terpenuhi walaupun $f_4(1) = 0$. Pada interval [−3, 2] syarat terpenuhi, tetapi Bisection menemukan akar sederhana $x = -2$ dalam 33 iterasi pada toleransi 10^{-9}, bukan akar ganda.

Tabel 8. Hasil Newton-Raphson pada fungsi f_4 dengan akar ganda ($x_0 = 2$).

Toleransi	Iterasi	Akar hampiran	Error absolut
1×10^{-3}	11	1,0006377880	$6{,}378\times10^{-4}$
1×10^{-6}	21	1,0000006230	$6{,}230\times10^{-7}$
1×10^{-9}	31	0,9999999964	$3{,}569\times10^{-9}$

Newton-Raphson tetap dapat dijalankan dari $x_0 = 2$, tetapi konvergensinya turun dari kuadratik menjadi linear. Rasio error berturut-turut turun dari 0,536 pada iterasi ke-2 menjadi sekitar 0,500 pada iterasi ke-12, sesuai dengan teori bahwa untuk akar ganda dua rasio ini mendekati 1/2 (Burden et al., 2016). Akibatnya jumlah iterasi naik menjadi 11, 21, dan 31 pada ketiga toleransi (Tabel 8), jauh lebih banyak daripada 2–6 iterasi pada akar sederhana dan sebanding dengan Bisection. Pada toleransi 10^{-9}, error mutlaknya ($3{,}57\times10^{-9}$) melebihi toleransi. Hal ini sejalan dengan sifat akar ganda yang sulit ditentukan secara akurat dengan aritmetika mesin karena fungsi mendatar di sekitar akar.

3.6. Implikasi Pemilihan Metode

Hasil di atas memperlihatkan bahwa kecepatan dan keandalan tidak dapat diperoleh sekaligus dari satu metode. Newton-Raphson unggul dalam jumlah iterasi dan biaya evaluasi bila turunan tersedia dan nilai awal baik, tetapi hasil akhirnya bergantung pada nilai awal (Tabel 7) dan keunggulannya hilang pada akar ganda (Tabel 8). Bisection lambat, tetapi pada semua kasus uji hasilnya terjamin dan jumlah iterasinya dapat diprediksi dari Persamaan (3) sebelum perhitungan dimulai. Pola ini mendukung gagasan metode hibrida yang memakai Bisection untuk mempersempit lokasi akar lalu Newton-Raphson untuk mempercepat (Badr et al., 2021; Badr et al., 2022). Karena Tabel 7 dan Gambar 2 menunjukkan bahwa nilai awal di dalam interval pun dapat menghasilkan langkah yang keluar dari interval, rancangan seperti itu perlu mengamankan langkah Newton-Raphson, misalnya dengan beralih ke langkah Bisection bila hampiran baru jatuh di luar interval.

Perbandingan ini juga perlu dibaca dengan memperhatikan perbedaan kriteria berhenti. Pada Bisection, panjang interval yang lebih kecil dari toleransi menjamin error tidak melebihi toleransi, sesuai dengan rasio error terhadap toleransi yang berkisar 0,03 sampai 0,87 pada sembilan kasus uji. Pada Newton-Raphson, selisih dua hampiran berurutan bukan jaminan error; pada akar sederhana errornya jauh di bawah toleransi (rasio kurang dari 10^{-4}), tetapi pada akar ganda errornya dapat melampaui toleransi. Dengan demikian, toleransi yang sama tidak berarti jaminan ketelitian yang sama.

Keterbatasan penelitian ini adalah jumlah fungsi uji yang kecil (tiga fungsi dengan akar sederhana dan satu fungsi dengan akar ganda), nilai awal yang hanya diuji pada rentang −3 sampai 3, dan waktu hitung yang diukur pada satu komputer sehingga tidak dapat dibandingkan langsung dengan komputer lain. Metode lain yang sering dipakai sebagai pembanding, seperti Secant dan Brent, juga belum diikutkan.

4. Kesimpulan

Metode Bisection dan Newton-Raphson punya sifat yang berbeda dan saling melengkapi. Newton-Raphson lebih cepat, yaitu hanya 4–6 iterasi pada toleransi 10^{-9}, tetapi membutuhkan turunan fungsi dan nilai awal yang tepat. Bisection membutuhkan iterasi lebih banyak (30–31 iterasi), tetapi lebih andal karena pasti konvergen selama

DOI: https://doi.org/10.69693/ijmst.v5i1.14655
Lisensi: Creative Commons Attribution 4.0 International (CC BY 4.0)

syarat perubahan tanda di interval awal terpenuhi. Pengujian tambahan menunjukkan bahwa jumlah iterasi Bisection sama dengan prediksi teoretis pada semua sembilan kasus dan orde konvergensi Newton-Raphson yang terukur mendekati 2,00 pada akar sederhana. Keunggulan Newton-Raphson tidak berlaku mutlak: dari 601 nilai awal pada rentang −3 sampai 3, 95,3% menuju akar acuan pada f_2 dan 53,4% pada f_3, sedangkan pada akar ganda konvergensinya menjadi linear dengan rasio error sekitar 0,5 dan memerlukan 31 iterasi pada toleransi 10^{-9}, hampir sama dengan Bisection. Pemilihan metode sebaiknya mengikuti kebutuhan: Bisection bila keandalan menjadi prioritas dan Newton-Raphson bila turunan tersedia dan kecepatan diutamakan. Penelitian selanjutnya dapat menguji lebih banyak fungsi, termasuk fungsi dengan banyak akar atau sangat bergelombang dan akar kompleks, serta membandingkannya dengan metode Secant, metode Brent, atau metode hibrida terbaru.

Referensi

1. Ali, K. M., & Fathi, B. G. (2025). Enhancing nonlinear equation solutions through the combination of variant Newton's and Halley's methods. *Science Journal of University of Zakho*, *13*(4), 590–599. https://doi.org/10.25271/sjuoz.2025.13.4.1594
2. Badr, E., Almotairi, S., & El Ghamry, A. (2021). A comparative study among new hybrid root finding algorithms and traditional methods. *Mathematics*, *9*(11), 1306. https://doi.org/10.3390/math9111306
3. 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
4. Burden, R. L., Faires, J. D., & Burden, A. M. (2016). *Numerical analysis* (10th ed.). Cengage Learning.
5. Chapra, S. C., & Canale, R. P. (2015). *Numerical methods for engineers* (7th ed.). McGraw-Hill Education.
6. Erviana, B. S., Amrullah, Triutami, T. W., & Subarinah, S. (2023). Efisiensi penyelesaian numerik persamaan non-linear dengan metode Newton Raphson dan metode Secant menggunakan program software berbasis Python. *Pendas: Jurnal Ilmiah Pendidikan Dasar*, *8*(3), 1719–1729. https://doi.org/10.23969/jp.v8i3.10964
7. Maure, O. P., & Tulan, H. (2024). Studi komparasi beberapa metode numerik dalam mengaproksimasi akar-akar persamaan non linear. *Asimtot: Jurnal Kependidikan Matematika*, *6*(1), 1–12. https://doi.org/10.30822/asimtot.v6i01.4008
8. Molawade, M., Patil, T., & Mehtre, V. V. (2022). Comparative study of bisection and Newton-Raphson methods of root-finding problems. *International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering*. https://doi.org/10.17148/IJIREEICE.2022.101103
9. Mumtazi, Y., Amrullah, Hikmah, N., & Prayitno, S. (2024). Comparison of Steffensen and Secant methods in determining non-linear function roots using Hypertext Preprocessor (PHP) programmer. *Jurnal Pendidikan, Sains, Geologi, dan Geofisika (GeoScienceEd Journal)*, *5*(3), 418–424. https://doi.org/10.29303/geoscienceed.v5i3.393
10. Pandia, J. W., & Sitepu, I. (2021). Penentuan akar persamaan non linier dengan metode numerik. *Jurnal Mutiara Pendidikan Indonesia*, *6*(2), 122–129. https://doi.org/10.51544/mutiarapendidik.v6i2.2326
11. Qureshi, S., Argyros, I. K., Soomro, A., Gdawiec, K., Shaikh, A. A., & Hincal, E. (2024). A new optimal root-finding iterative algorithm: Local and semi-local analysis with polynomiography. *Numerical Algorithms*, *95*, 1715–1745. https://doi.org/10.1007/s11075-023-01625-7
12. Romadani, M. N. H. (2025). From theory to code: Transforming classical root-finding methods into efficient Python implementations. *ICCK Journal of Applied Mathematics*, *1*(3), 154–189. https://doi.org/10.62762/JAM.2025.840767
13. Rozi, S., & Rarasati, N. (2022). Template metode numerik pada Excel untuk menemukan solusi dari persamaan nonlinier. *AXIOM: Jurnal Pendidikan dan Matematika*, *11*(1), 33–47. https://doi.org/10.30821/axiom.v11i1.11254
14. Sharma, R., & Chouhan, V. S. (2026). Computational performance of classical and hybrid root-finding methods for real-world nonlinear models. *Discover Computing*, *29*, Article 252. https://doi.org/10.1007/s10791-026-10141-w
15. Sholihah, K. I., Qoiriyah, I., & Wibowo, A. (2025). Implementasi metode Secant dalam penyelesaian persamaan nonlinear menggunakan Microsoft Excel dan Maple. *Griya Journal of Mathematics Education and Application*, *5*(3), 748–761. https://doi.org/10.29303/griya.v5i3.672
16. Syaharuddin, Hidayah, H., Mahsup, M., Mehmood, S., & Raza, W. (2026). Accuracy comparison of multivariate Newton-Raphson and Newton-Kantorovich methods through numerical simulation in nonlinear systems. *Jurnal ELTIKOM*, *10*(1), 37–46. https://doi.org/10.31961/eltikom.v10i1.1971
17. Vivas-Cortez, M., Ali, N. Z., Khan, A. G., & Awan, M. U. (2023). Numerical analysis of new hybrid algorithms for solving nonlinear equations. *Axioms*, *12*(7), 684. https://doi.org/10.3390/axioms12070684
18. Wulandari, R., Umara, U., & Tarigan, I. M. B. (2026). Penerapan metode Secant dalam menyelesaikan persamaan non linear. *Jurnal Surya Informatika*, *16*(2), 186–193. https://doi.org/10.48144/suryainformatika.v16i2.2487

DOI: https://doi.org/10.69693/ijmst.v5i1.14655
Lisensi: Creative Commons Attribution 4.0 International (CC BY 4.0)