EFISIENSI PENYELESAIAN NUMERIK PERSAMAAN NON-LINEAR DENGAN METODE NEWTON RAPHSON DAN METODE SECANT MENGGUNAKAN PROGRAM SOFTWARE BERBASIS PYTHON

Baiq Sinta Erviana[1], Amrullah[2*], Tabita Wahyu Triutami[3], Sri Subarinah[4]
[1] Mahasiswa Pendidikan Matematika, FKIP, Universitas Mataram, Mataram
[234] Pendidikan Matematika, FKIP, Universitas Mataram, Mataram
baiqsintaerviana@gmail.com

ABSTRACT

Usually in our daily life, we often encounter complex problems that require the theory of solving non-linear equations. This research aims to obtain programs for the Newton Raphson method and the Secant method in Python and to compare the efficiency of non-linear equations from the Newton and the Secant method in terms of the number of iterations, errors and program execution time. The functions that will be tested are polynomial functions, exponential functions and trigonometric functions. The research used is application type research. The program test was carried out three times by changing the coefficients and initial values. The format of a polynomial function is $f(x) = a_1x^{b_1} + a_2x^{b_2} + a_3x^{b_4} + a_4$, *exponential function* $f(x) = a_1x + a_2ex^{b_1x}$), *trigonometric function* $f(x) = a_1\,sin(b_1x) + a_2\,cos(b_2x) + a_3$. *The output of this research is program code for polynomial functions, exponential functions and trigonometric functions. in one program it consists of the declaration of Newton's formula, Secant's formula, error, iteration table and execution time formula. From the six experiments carried out, results were obtained using the Newton method on polynomial functions, exponential functions and trigonometric functions, the number of iterations obtained was fewer with smaller errors and less time than the Secant method. So it was concluded that the Newton Raphson method was more efficient than the Secant method.*

Keywords: *efficiency; non-liniar equations; Newton Raphson method; Secant method; Python.*

ABSTRAK

Pada kehidupan sehari-hari sering ditemui suatu permasalahan kompleks yang membutuhkan penyelesaian persamaan non-linear. Penelitian ini bertujuan untuk mendapatkan program meetode Newton Raphson dan metode Secant dalam bahasa Python dan membandingkan efisiensi persamaan non-linear dari metode Newton Raphson dan metode Secant ditinjau dari jumlah iterasi, galat dan waktu eksekusi program. Fungsi yang akan di uji adalah fungsi polinomial, fungsi eksponen dan fungsi trigonometri. Penelitian yang digunakan adalah penelitian jenis penerapan. Uji program dilakukan sebanyak tiga kali dengan mengganti koefisien dan nilai awalnya. Format fungsi polinomial adalah $f(x) = a_1x^{b_1} + a_2x^{b_2} + a_3x^{b_4} + a_4$, fungsi eksponen $f(x) = a_1x + a_2ex^{b_1x}$, fungsi trigonometri $f(x) = a_1\sin(b_1x) + a_2\cos(b_2x) + a_3$. Output dari penelitian ini adalah kode program untuk fungsi polinomial, fungsi eksponen dan fungsi trigonometri. dalam satu program tersbut terdiri dari deklarasi rumus Newton, rumus Secant, galat, tabel iterasi dan rumus waktu eksekusi. Dari enam percobaan yang dilakukan diperoleh hasil dengan menggunakan metode Newton Raphson pada fungsi polinomial, fungsi eksponen

dan fungsi trigonometri jumlah iterasi yang didapatkan lebih sedikit dengan galat yang kecil dan waktu yang lebih sedikit dibandingkan metode Secant. Sehingga didapatkan kesimpulan bahwa metode Newton Raphson lebih efisien dibanding metode Secant.

Kata Kunci: efisiensi; persamaan non-liniar; metode Newton Raphson; metode Secant; Python.

A. Pendahuluan

Banyak pemodelan yang membutuhkan teori dari persamaan terutama persamaan non-linear. Persamaan non-linear dapat muncul dalam berbagai bidang seperti fisika, ekonomi dan biologi. Fungsi non-linear yang paling sering ditemui dalam matematika terapan adalah persamaan yang berbentuk $f(x) = 0$. Fungsi $f(x)$ dapat berbentuk fungsi aljabar, fungsi transenden atau fungsi campuran (Subarinah, 2022).

Persamaan non-linear yang sederhana seperti persamaan kuadrat $ax^2 + bx + c = 0$ dapat diselesaikan dengan tiga cara, yaitu pemfaktoran, melengkapkan kuadrat dan rumus abc $x_{1.2} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$ (Arjudin, 2011). Persamaan non-linear yang kompleks tidak dapat diselesaikan dengan cara analitik biasa karena membutuhkan proses yang rumit dan panjang. Oleh karena itu, persamaan non-linear yang kompleks membutuhkan pendekatan numerik seperti metode Newton Raphson dan metode Secant. Kedua metode ini merupakan metode yang paling mudah konvergen. Peneliti mengkaji lebih dalam mengenai perbandingan efisiensi dari dua metode ini ditinjau dari jumlah iterasi, galat dan waktu eksekusi yang digunakan.

Seiring dengan berkembangnya teknologi saat ini memberikan perangkat *sogtware* yang dapat membantu menyelesaikan masalah pengerjaan metode numerik dengan lebih cepat (Firdaus, Amrullah, Wulandari dan Hikmah, 2023). Biasanya, perhitungan numerik ini menggunakan bantuan aplikasi seperti Matlab, Pascal, Java karena perhitungannya yang panjang. Seperti penelitian dari Hutagalung (2017) melakukan penelitian tentang metode Newton Raphson menggunakan Matlab, Ritonga & Suryana (2019) yang membandingkan tingkat konvergensi metode titik tetap dengan metode Newton Raphson menggunakan Matlab. Ada banyak aplikasi bantuan selain Matlab yang dapat digunakan untuk metode

numerik. Salah satunya yaitu Python. Python merupakan bahasa pemrograman yang terbilang baru di bidang pendidikan dan merupakan bahasa pemrograman yang populer di dunia programmer. Kepopulerannya ini karena Python memiliki algoritma kode yang lebih mudah dipahami dan lebih pendek dibanding bahasa pemrograman lain. Dengan menggunakan Python untuk perhitungan numerik, kode program yang digunakan dapat lebih pendek dan lebih cepat.

Metode numerik yang paling sering digunakan adalah metode Newton Raphson dan metode Secant karena kedua metode ini yang paling mudah konvergen dibanding metode numerik yang lain. Pada penelitian sebelumnya, belum ada yang membandingkan efisiensi dari kedua metode ini. Oleh karena itu peneliti melakukan uji efisiensi metode Newton Raphson dan metode Secant menggunakan Python ditinjau dari jumlah iterasi, galat dan waktu eksekusi program untuk menguji metode mana yang lebih efisien diantara keduanya.

Tujuan dari penelitian ini adalah untuk mendapatkan program metode Newton Raphson dan metode Secant pada penyelesaian non-linear menggunakan Python dan memperoleh perbandingan efisiensi metode Newton Raphson dan metode Secant dalam penyelesaian soal numerik fungsi polinomial, fungsi eksponen dan fungsi trigonometri.

B. Metode Penelitian

Penelitian ini menggunakan penelitian jenis penerapan, yaitu untuk menerapkan metode Newton Raphson dan metode Secant yang diaplikasikan dalam bahasa pemrograman Python untuk mecari akar persamaan non-linear ditinjau dari jumlah iterasi, galat dan waktu eksekusi program. Fungsi yang di uji coba adalah fungsi polinomial, fungsi eksponen dan fungsi trigonometri.

2.1. Algoritma metode Newton Raphson

Metode Newton Raphson merupakan metode iterasi untuk menyelesaikan persamaan $f(x) = 0$ dengan mengasumsikan f mempunyai turunan kontinu f' (Subarinah, 2022:29). Adapun algoritma metode Newton Raphson menggunakan bahasa pemrograman Python adalah sebagai berikut.

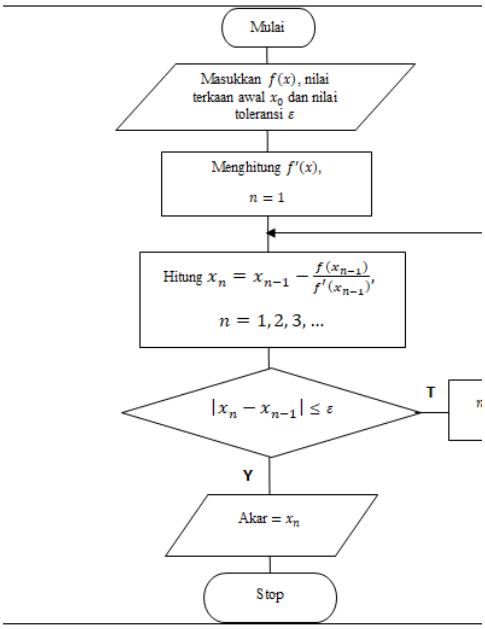
1. Ditentukan fungsi $f(x)$ yang akan dicari akarnya.
2. Ditentukan iterasi maksimum epsilon (ε) sebagai toleransi kesalahan.

3. Ditentukan nilai x_{n-1} sebagai terkaan awal
4. Dihitung $f(x_{n-1})$ dan $f'(x_{n-1})$.
5. Dihitung x_n dengan cara

$$x_n = x_{n-1} - \frac{f(x_{n-1})}{f'(x_{n-1})}; f'(x_{n-1}) \neq 0, n = 1,2,3, \ldots$$

6. Jika nilai $|x_n - x_{n-1}| \leq \varepsilon$ maka tulis $x_{hampiran} = x_n$, jika tidak lanjut ke iterasi $n = n + 1$ dan kembali ke langkah 4.

Berikut diagram alir dari metode Newton Raphson



Gambar 2.1 Diagram Alir Metode Newton Raphson

2.2 Algoritma metode Secant

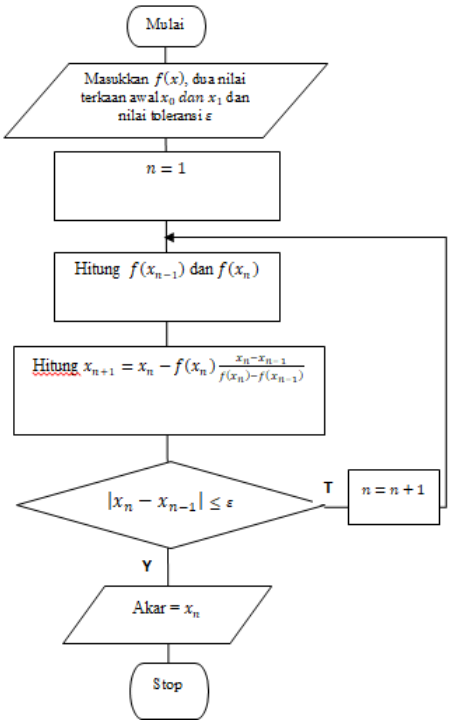
Metode Secant merupakan perbaikan dari metode Newton Raphson yang menggunakan turunan untuk mencari aproksimasi akarnya (Subarinah, 2022:36). Adapun algoritma dari metode Secant menggunakan Python adalah sebagai berikut.

1. Ditentukan fungsi $f(x)$ yang akan dicari akar persamaannya.
2. Ditentukan iterasi maksimum epsilon (ε) sebagai toleransi kesalahan.
3. Ditentukan nilai x_0 dan x_1 sebagai terkaan awal.
4. Dihitung $f(x_0)$ dan $f(x_1)$.
5. Dihitung x_{n+1} dengan cara

$$x_{n+1} = x_n - f(x_n)\frac{x_n - x_{n-1}}{f(x_n) - f(x_{n-1})}, n = 1,2,3, \ldots$$

6. Jika nilai $|x_n - x_{n-1}| \leq \varepsilon$ maka tulis $x_{hampiran} = x_n$, jika tidak lanjut ke iterasi $n = n + 1$ dan kembali ke langkah 4.

Berikut diagram alir dari metode Secant.



Gambar 2.2 Diagram alir metode Secant

C.Hasil Penelitian dan Pembahasan

3.1 Persamaan yang diselesaikan

Fungsi yang digunakan dalam penelitian ini adalah fungsi polinomial, fungsi eksponen dan fungsi trigonometri. Program aplikasi dengan bahasa pemrograman Python dengan menggunakan metode Newton Raphson dan metode Secant dilakukan uji validitas berdasarkan beberapa fungsi yang telah diketahui akar-akarnya seperti yang tertera pada Tabel 3.1. Adapun fungsi yang dimaksud adalah sebagai berikut.

Tabel 3.1 Uji Coba Program

No	Fungsi	Eksak	Newton Raphson			Secant		
			Hasil	Galat	Iterasi	Hasil	Galat	Iterasi
1	$f(x) = x^2 + 3x + 2$	$x = -1$ $x = -2$	-1	0.00000	5	-1	0.00000	7
2	$f(x) = x^3 - 3x^2 - 6x + 8$	$x = 1$ $x = -2$ $x = 4$	-2	0.00000	6	-2	0.00000	6
3	$f(x) = x - e^{-x}$	0.56714	0.56714	0.00000	3	0.56714	0.00000	4
4	$f(x) = e^x - 4x$	0.35740	0.35740	0.00000	3	0.35740	0.00000	4

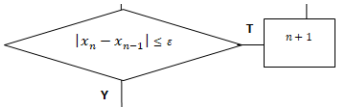
5	$f(x) = \sin(x) + \cos(x) + 1$	-1.57080	-1.57080	0.00000	5	-1.57080	0.00000	4
6	$f(x) = 2\sin(x) - \cos(x) + 1$	4.06889	4.06889	0.00000	4	4.06889	0.00000	3

Hasil yang diperoleh menunjukkan bahwa hasil dari program hampir 99% sama dengan nilai eksaknya. Oleh karena itu, program yang diperoleh dianggap valid. Selanjutnya untuk menentukan perbandingan efisiensi antara metode Newton Raphson dengan metode Secant dilakukan uji program dengan cara menaikkan koefisien dan nilai awal dari masing-masing fungsi.

3.2 Program Fungsi Polinomial

Program ini digunakan untuk menentukan akar dari persamaan fungsi polinomial menggunakan metode Newton Raphson dan Metode Secant dalam satu program Python. Kode program yang digunakan terdiri dari beberapa deklarasi diantaranya yaitu deklarasi input dimana user dapat menginput nilai koefisien dan nilai awal yang ingin dicari akar persamaannya dengan format $f(x) = a_1x^{b_1} + a_2x^{b_2} + a_3x^{b_4} + a_4$adalah, deklarasi turunan untuk metode Newton dan deklarasi untuk menentukan akar persamaan menggunakan metode Newton Raphson dan metode Secant.

Output dari program ini adalah tabel yang berisi iterasi metode Newton dan Secant hingga ditemukannya nilai akar. Program aplikasi yang dihasilkan menggunakan algoritma seperti pada Gambar 2.1 dan Gambar 2.2 pada bagian *decision*



untuk melakukan iterasi diperoleh *soruce code* sebagai berikut.

```
def metode_newton_raphson(x, fungsi, turunan, x0, tol=0.00001, max_iter=100):
    iterasi = 0
    galat = tol + 1
    print("Metode Newton-Raphson")
    print("{:<10s} {:<10s} {:<10s}".format("Iterasi", "x", "Galat"))
    while galat > tol and iterasi < max_iter:
        x_baru = x0 - (fungsi.subs(x, x0) / turunan.subs(x, x0))
        galat = abs(x_baru - x0)
        print("{:<10d} {:<10.5f} {:<10.5f}".format(iterasi, x_baru, galat))
        x0 = x_baru
```

```
        iterasi += 1


def metode_secant(x, fungsi, x0, x1,
tol=0.00001, max_iter=100):

    iterasi = 0

    galat = tol + 1

    print("Metode Secant")

    print("{:<10s} {:<10s}
{:<10s}".format("Iterasi", "x", "Galat"))

    while galat > tol and iterasi <
max_iter:

        f_x0 = fungsi.subs(x, x0)

        f_x1 = fungsi.subs(x, x1)

        if f_x1 - f_x0 == 0:

            print("Pembagian oleh nol
terdeteksi. Metode Secant tidak dapat
dilanjutkan.")

            break

        x_baru = x1 - (f_x1 * (x1 - x0)) /
(f_x1 - f_x0)

        galat = abs(x_baru - x1)

        print("{:<10d} {:<10.5f}
{:<10.5f}".format(iterasi, x_baru, galat))

        x0 = x1

        x1 = x_baru

        iterasi += 1
```

Setelah dilakukan uji validitas selanjutnya dilakukan pencarian akar fungsi dengan menggunakan koefisien a_i dan b_i yang berbeda-beda. Berikut adalah hasil uji program fungsi polinomial metode Newton Raphson dan metode Secant menggunakan Python.

Tabel 3.2 Fungsi Polinomial menggunakan metode Newton

No	a_1	b_1	a_2	b_2	a_3	b_3	a_4	x_0	Iterasi	Galat	Akar	Waktu (detik)
1	1	3	-2	2	7	1	15	0	5	0.00000	-1.31858	0.00407
2	2	3	0	2	7	1	15	0	5	0.00000	-1.38454	0.00340
3	2	3	2	2	8	1	16	0	4	0.00000	-1.60752	0.00310
4	3	3	2	2	9	1	16	1	5	0.00001	-1.35568	0.00333
5	3	3	3	2	9	1	17	1	6	0.00000	-1.50615	0.00330

Tabel 3.3 Fungsi Polinomial menggunakan metode Secant

No	a_1	b_1	a_2	b_2	a_3	b_3	a_4	x_0	x_1	Iterasi	Galat	Akar	Waktu (detik)
1	1	3	-2	2	7	1	15	0	1	7	0.00000	-1.31858	0.00348
2	2	3	0	2	7	1	15	0	1	6	0.00000	-1.38454	0.00414
3	2	3	2	2	8	1	16	0	1	5	0.00001	-1.60752	0.00362
4	3	3	2	2	9	1	16	1	2	8	0.00000	-1.35568	0.00433
5	3	3	3	2	9	1	17	1	2	8	0.00000	-1.50615	0.00398

3.3 Program Fungsi Eksponen

Perbedaan antara masing-masing program fungsi adalah pada rumus atau format yang digunakan. Pada fungsi eksponen format yang digunakan adalah $f(x) = a_1x - a_2e^{b_1x}$ dengan menggunakan nilai $a_i\ dan\ b_i =$ (bilangan bulat). Sehingga dalam program Python yang perlu diubah hanya pada bagian *input user* dan fungsinya saja yaitu:

fungsi = a_satu * x + a_dua * sp.exp(b_satu * x)

Berikut adalah hasil uji program fungsi eksponen metode Newton Raphson dan metode Secant menggunakan Python.

Tabel 3.4 Fungsi eksponen menggunakan metode Newton

No	a_1	a_2	b_1	x_0	Iterasi	Galat	Akar	Waktu (detik)
1	-4	-1	1	0	2	0.00000	-0.20389	0.00197
2	-4	1	1	0	3	0.00000	0.35740	0.00116
3	2	1	1	0	3	0.00000	-0.35173	0.00312
4	1	1	2	1	5	0.00000	-0.42630	0.00290
5	2	2	2	2	7	0.00000	-0.42630	0.00414

Tabel 3.5 Fungsi Eksponen menggunakan metode Secant

No	a_1	a_2	b_1	x_0	x_1	Iterasi	Galat	Akar	Waktu (detik)
1	-4	-1	1	0	1	3	0.00001	-0.20389	0.00205
2	-4	1	1	0	1	4	0.00000	0.35740	0.00370
3	2	1	1	0	1	4	0.00000	-0.35173	0.00381
4	1	1	2	1	2	9	0.00001	-0.42630	0.00325
5	2	2	2	2	3	11	0.00000	-0.42630	0.00445

3.4 Program Fungsi Trigonometri

Format untuk fungsi trigonometri adalah $f(x) = a_1sin(b_1x) + a_2cos(b_2x) + a_3$ dengan menggunakan nilai $a_i\ dan\ b_i =$ (bilangan bulat). Program Python yang perlu diubah hanya pada bagian *input user* dan fungsinya saja yaitu:

fungsi = a_satu * sp.sin(b_satu * x) + a_dua * sp.cos(b_dua * x) + a_tiga

Berikut adalah hasil uji program fungsi trigonometri metode Newton Raphson dan metode Secant menggunakan Python.

Tabel 3. 6 Fungsi Trigonometri menggunakan metode Newton

No	a_1	b_1	a_2	b_2	a_3	x_0	Iterasi	Galat	Akar	Waktu (detik)
1	1	1	1	1	1	-1	3	0.00000	-1.57080	0.00218
2	2	1	-1	1	1	1	3	0.00000	0.00000	0.00498
3	3	2	1	1	2	0	4	0.00000	-0.61089	0.00432
4	3	1	2	1	3	2	3	0.00001	3.53638	0.00313
5	4	3	-2	2	3	3	2	0.00000	3.22841	0.00212

Tabel 3.7 Fungsi Trigonometri menggunakan metode Secant

No	a_1	b_1	a_2	b_2	a_3	x_0	x_1	Iterasi	Galat	Akar	Waktu (detik)
1	1	1	1	1	1	-1	0	4	0.00000	-1.57080	0.00485
2	2	1	-1	1	1	1	2	6	0.00000	0.00000	0.00501
3	3	2	1	1	2	0	0.5	4	0.00000	-0.61089	0.00386
4	3	1	2	1	3	2	2.5	5	0.00000	3.53638	0.00480
5	4	3	-2	2	3	3	3.5	5	0.00000	3.22841	0.00348

3.5 Impilkasi Hasil

Berdasarkan beberapa percobaan diatas, penentuan akar fungsi pada penelitian ini menggunakan metode Newton dan metode Secant. Hasil dari percobaan ini adalah metode Newton memilki jumlah iterasi, galat dan waktu eksekusi yang lebih singkat dibanding metode Secant. Orde dari fungsi yang digunakan juga mempengaruhi jumlah iterasi pada masing-masing metode. Hal ini sejalan dengan pendapat dari Herfina, Amrullah dan Junaidi (2019) mengenai kenaikan orde fungsi. Semakin besar orde fungsi yang digunakan maka akan semakin kecil galat yang ditimbulkan sehingga hasilnya pun lebih akurat. Penjelasan masing-masing tabel adalah sebagai berikut.

1. Tabel 4.2 dan Tabel 4.3 jika dibandingkan, iterasi metode Newton Raphson lebih pendek daripada metode Secant. Galat untuk metode Newton juga lebih kecil dari metode Secant. Waktu rata-rata yang dibutuhkan metode Newton lebih sedikit dari metode Secant. jika ditinjau dari jumlah iterasi, galat dan waktu rata-rata eksekusi, metode Newton dapat dikatakan lebih efisien dibanding metode Secant.
2. Tabel 4.6 dan Tabel 4.7 untuk fungsi eksponen, jumlah iterasi, galat dan waktu eksekusi metode Newton Raphson lebih kecil daripada metode Secant walaupun koefisien dan nilai awalnya berbeda-beda. Iterasi metode Newton 22% lebih efisien dan 12% lebih efisien jika ditinjau

dari waktu dibanding metode Secant.

3. Hasil uji program untuk fungsi trigonometri dengan koefisien dan nilai awal yang berbeda-beda ditunjukkan pada Tabel 4.10 dan Tabel 4.11 memperoleh iterasi metode Newton Raphson lebih pendek dari pada metode Secant. sedangkan untuk galat, terdapat perbedaan 0.00001 pada metode Newton Raphson. Waktu yang digunakan metode Newton lebih singkat dibanding metode Secant.

D. Kesimpulan

Berdasarkan hasil dari pembahasan, diperoleh kesimpuan bahwa:

1. Penggunaan bahasa pemrograman Python diperoleh program aplikasi metode Newton Raphson dan metode Secant untuk menyelesaikan penentuan akar persamaan fungsi polinomial, fungsi eksponen dan fungsi trigonometri yang sesuai.
2. Penentuan akar fungsi menggunakan metode Newton lebih efisien 16% dibanding metode Secant pada fungsi polinomial. Kemudian pada fungsi eksponen metode Newton lebih efisien 22% dibanding metode Secant. Untuk fungsi Trigonometri metode Newton lebih efisien 24% dibanding metode Secant jika ditinjau dari jumlah iterasinya. Oleh karena itu, dapat disimpulkan bahwa metode Newton lebih efisien dibanding metode Secant pada tiga jenis fungsi tersebut.

Pada penelitian ini yang ditentukan akar persamaannya adalah fungsi polinomial, fungsi eksponen dan fungsi trigonometri. diharapkan penelitian selanjutnya dapat membahas mengenai fungsi polinomial pangakat empat dan seterusnya.

DAFTAR PUSTAKA

Arjudin. (2011). *Sifat Akar Polinom Dan Penerapannya Pada Sistem Persamaan Non Linier*. http://eprints.uny.ac.id/7270/

Firdaus, A., Amrullah, Wulandari, N. P., & Hikmah, N. (2023). *Analisis Efisiensi Integral Numerik Metode Simpson 1/3 dan Simpson 3/8 Menggunakan Program Software Berbasis Pascal*. *9*. https://doi.org/https://doi.org/10.37012/jtik.v9i2.1737

Herfina, N., Amrullah, & Junaidi. (2019). Efektivitas Metode Trapesium dan Simpson Dalam Penentuan Luas Menggunakan Pemrograman Pascal. *Mandalika Mathematics and Educations Journal*, *1*(1), 53–65. https://doi.org/10.29303/jm.v1i1.1242

Hutagalung, S. N. (2017). Pemahaman metode numerik (Studi kasus metode New-

Rhapson) menggunakan pemprogrman Matlab. *Jurnal Teknologi Informasi*, *1*(1), 95. https://doi.org/10.36294/jurti.v1i1.109

Ritonga, J., & Suryana, D. (2019). Perbandingan kecepatan konvergensi akar persamaan non linier metode titik tetap dengan metode Newton Raphson menggunakan Matlab. *INFORMASI (Jurnal Informatika Dan Sistem Informasi)*, *11*(2), 51–64. https://doi.org/10.37424/informasi.v11i2.17

Subarinah, S. (2022). *Metode Numerik*. FKIP Press.