Perancangan Program Perhitungan Solusi Numerik Menggunakan Metode Bolzano pada Bahasa Python

Wahyu Widyanto, Fa'iz Abiyyu Rizqullah Saputra, dan Andy Rachman

Jurusan Teknik Informatika, Institut Teknologi Adhi Tama Surabaya

ABSTRACT

Numerical computing is a method that utilizes computer technology to perform complex numerical calculations. The purpose of this paper is to develop a numerical calculation program for the Bisection method in the Python programming language to obtain the calculation results in the form of a table and the approximate value of the root of the equation. The pre-test and post-test methods were used to test the effectiveness of the Bisection algorithm in the program that was built, then manual calculations were carried out using MS. Excel which could later be used as a benchmark for the results of calculations in the program. The results of the pre-test (CLI) and post-test (GUI) produce similar or even the same output. Changes in appearance from the CLI to the GUI have no major effect on the program output. With this research, it is hoped that it can help people, both the general public and students who are dealing with finding the solution value of numerical equations, because with this numerical solution calculation program users do not have to bother to write complex formulas, it is enough to enter the formula for the numerical problem that you want to find a solution for and other supporting data, this program is able to get the solution you are looking for quickly.

Article History

Received 23 – 1 – 2022
Revised 27 – 6 – 2022
Accepted 12 – 9 – 2022

Key words

Root of Equation
Bisection Method
Numerical Computing
Phyton

ABSTRAK

Komputasi numerik merupakan metode yang memanfaatkan teknologi komputer untuk melakukan perhitungan numerik yang rumit. Tujuan tulisan ini adalah untuk membangun program perhitungan numerik metode Bisection dalam Bahasa Pemrograman Python guna untuk mendapatkan hasil perhitungan dalam bentuk berupa tabel dan nilai hampiran akar persamaan. Metode *pre-test* dan *post-test* digunakan untuk menguji keefektifan algoritma Bisection pada program yang dibangun kemudian dilakukan perhitungan manual terlebih dahulu menggunakan MS. Excel yang nantinya dapat digunakan sebagai tolak ukur dari hasil perhitungan pada program. Hasil dari dari *pre-test* (CLI) dan *post-test* (GUI) menghasilkan keluaran yang mirip bahkan sama. Perubahan tampilan dari CLI ke GUI tidak berpengaruh besar kepada keluaran program. Dengan adanya penelitian ini diharapkan dapat membantu orang-orang baik itu masyarakat umum maupun mahasiswa yang sedang berhadapan dengan pencarian nilai solusi dari persamaan numerik, dikarenakan dengan adanya program perhitungan solusi numerik ini para pengguna tidak perlu repot-repot untuk menuliskan formula – formula yang rumit, cukup memasukan rumus permasalahan numerik yang ingin dicari solusinya dan data-data pendukung lainnya, program ini sudah mampu mendapatkan solusi yang dicari secara cepat.

PENDAHULUAN

Salah satu topik matematika yang sering digunakan untuk analisis adalah Komputasi Numerik yang memanfaatkan teknologi komputer dalam membantu melakukan iterasi perhitungan numerik yang rumit [5]. Metode Numerik akan sangat membantu dalam menyelesaikan setiap permasalahan apabila polanya berbentuk matematis dan memiliki hubungan antar variable [6]. Secara umum metode ini digunakan untuk menyelesaikan masalah dalam mencari solusi permasalahan linear dan nonlinear yang kompleks, kalkulus turunan, dan integral yang semua itu merupakan ilmu matematika analisis [7] dan bisa digunakan dalam Teknik Mesin, Teknik Elektro, kedokteran, Sosial Ekomoni, dan bidang ilmu lainnya [8].

Metode numerik disajikan ke dalam bentuk algoritma yang dapat dihitung dengan cepat dan mudah dengan pendekatan pengulangan proses perhitungan algoritmanya atau disebut dengan *iterasi* guna untuk mencari solusi persamaan linear [9]. Iterasi adalah proses yang dilakukan secara berulang hingga kondisi berhenti yang telah ditentukan. Hasil dari penyelesaian dari metode ini adalah bentuk hampiran yang mendekati penyelesaian yang sebenarnya (eksak) [10]. Metode Bisection merupakan salah satu metode numerik yang sering digunakan dalam menentukan analisis dari permasalahan numerik [11] dan hasil perhitungannya akan selalu menghasilkan akar atau solusi dari metode ini selalu konvergen [12]. Pada prinsipnya mencari nilai solusi dari metode ini dibutuhkan dua nilai a dan b yang dimana kedua nilai ini harus menghimpit nilai solusinya [13].

Solusi dari metode numerik adalah solusi hampiran, maka dari itu akan muncul istilah kata galat (*error*). Galat adalah kondisi yang dimana terjadi perbedaan nilai antara hasil solusi persoalan model numerik analisis dengan nilai sejatinya (eksak) [14]. Solusi maupun nilai sejati dari permasalahan analisis ini akan selalu berbentuk angka maka dari itu kedua hal ini akan memiliki hubungan dalam menentukan nilai errornya dan diharapkan nilai error yang ditemukan haruslah relatif kecil. Pada hakekatnya mencari solusi dari perhitungan numerik yang menghasilkan error ini bertujuan untuk menemukan cara perhitungan dan metode yang paling efisien untuk memperoleh solusi dari beberapa permasalahan [15].

Algoritma yang dapat digunakan dalam metode Bisection adalah algoritma iterasi yang dimana Algoritma adalah kemampuan sesorang dalam beripikir menggunakan akal tentang permasalahan yang menghasilkan suatu hal yang dapat dibuktikan dan dapat diterima oleh akal [16]. Dalam dunia komputer, algotima memiliki peran sangat penting dalam pembangunan suatu software [17]. Mempelajari algoritma hampir sama dengan belajar matematika, jika dalam matematika diharuskan paham terlebih dahulu mengenai rumus sebelum mendapatkan hasil permasalahan, begitu juga dengan pada algoritma, harus terlebih dahulu memahami kode – kode atau Bahasa program (*Coding)* agar dapat menjalankan program dan menuangkannya kedalam bebarapa simbol yang dapat dipahami dan dibaca [18].

Pada penelitian sebelumnya [19] dan [20] mencari solusi persamaan, keduanya menggunakan aturan nilai toleransi error, dengan nilai ini proses pencarian solusinya akan dibatasi sebesar yang telah ditentukan, dan kedua penelitian tersebut juga menggunakan sebuah Bahasa pemrograman dalam membantu mencari sebuah solusi persamaan yang ingin dicari. Pada penelitian ini, penulis ingin menerapkan algoritma dari metode Bisection kedalam Bahasa pemrograman Python guna untuk mendapatkan hasil solusi numerik yang dilakukan perhitungan berulang-ulang kedalam bentuk berupa tabel dan nilai hampiran akar persamaan, tetapi sebelum itu akan dilakukan komparasi hasil perhitungan (output) pada program python dengan hasil pada perhitungan di excel, setelah itu akan dilakukan tes perhitungan menggunakan beberapa permasalahan matematis untuk memastikan program dapat digunakan dengan baik.

TINJAUAN PUSTAKA

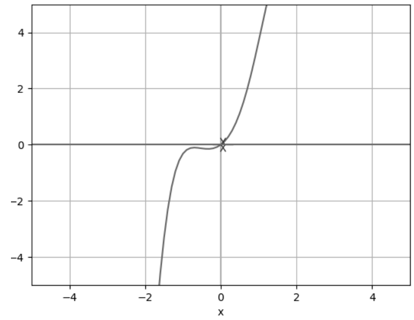
Umum

Metode Numerik adalah Teknik matematika yang digunakan untuk memecahkan masalah matematika yang tidak dapat diselesaikan atau sulit dipecahkan seperti halnya pada Gambar 1. Solusi yang dihasilkan pada metode ini adalah solusi numerik atau solusi hampiran (perkiraan). Meskipun hasil solusinya adalah perkiraan tetapi solusi ini bisa sangat akurat. Dalam banyak metode numerik, perhitungan dijalankan secara iteratif sampai akurasi yang diinginkan tercapai. – Contoh: mulai dari satu nilai x kemudian ubah nilainya sedikit demi sedikit. Perubahan tanda pada f(x) akan menunjukkan bahwa ada akar di dalam penambahan nilai terakhir. Saat ini, metode numerik digunakan dalam komputer digital yang memungkinkan untuk melakukan banyak perhitungan yang membosankan dan berulang-ulang yang menghasilkan solusi yang akurat (walaupun tidak tepat) dalam waktu yang sangat singkat [21]. Untuk setiap jenis masalah matematika ada beberapa teknik numerik yang dapat digunakan. Setiap teknik berbeda dalam akurasi, panjang perhitungan, dan kesulitan dalam pemrograman. Misalnya, untuk menyelesaikan persamaan nonlinier bentuk $f(x) = 0$ dapat menggunakan metode-metode numerik seperti Bisection, Regula Falsi, Newton's, Secant dan lain sebagainya.

Kesalahan (Error)

Ada dua jenis kesalahan yang umum dalam perhitungan numerik yakni kesalahan pembulatan dan pemotongan. Kesalahan pembulatan disebabkan oleh fakta bahwa bilangan *floating point* diwakili oleh presisi yang terbatas. Kesalahan pemotongan terjadi ketika membuat pendekatan diskrit ke fungsi kontinu [22]. Kesalahan Pembulatan (*Roundoff Error*) adalah setiap perhitungan yang melibatkan bilangan real muncul kesalahan pembulatan. Bahkan ketika tidak ada kesalahan aproksimasi[1] yang terlibat, kesalahan pembulatan akan tetap ada. Hal ini muncul karena representasi presisi terbatas dari bilangan real pada komputer mana pun yang memengaruhi representasi data dan aritmatika komputer [23]. Kesalahan pemotongan (*Truncation error*) dihasilkan dari penggunaan pendekatan di tempat prosedur matematika yang tepat [24]. Kesalahan pemotongan dapat dikatakan sebagai kesalahan diskritisasi[2] dan disebabkan oleh perkiraan yang dibuat, bukan arsitektur komputer yang digunakan.

$$f(x) = x^3 + 2sin(x^2) + x = 0$$



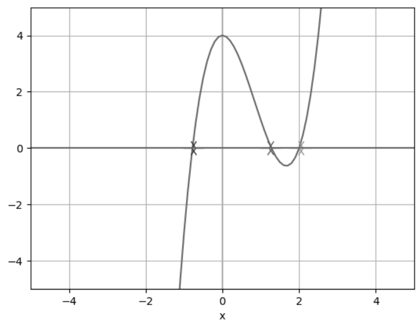
Gambar 1. Grafik akar persamaan

Metode Bisection

Metode Bisection merupakan salah satu metode yang paling sederhana dan paling dapat diandalkan dari metode iterasi untuk mencari solusi persamaan non liniear yang bergantung pada fakta bahwa *f(x)* adalah rill dan kontinu dalam interval $a < x < b$ dan *f(a)* dan *f(b)* saling berlawanan tanda (positif dan negatif) $f(a)\ x\ f(b) < 0$ [25], Maka paling sedikit akan ada satu nilai akar antara a dan b. Oleh karena itu, dengan menguji tanda pada fungsi di titik tengah dapat disimpulkan bagian mana dari interval yang memiliki akar. Yang penting di sini adalah satu akar didalam interval. Jika lebih dari satu akar dalam rentang ini, proses masih dapat dilanjutkan tetapi akan melalui proses yang sedikit rumit, untuk itu interval dapat dipersempit lagi hingga menghasilkan satu akar saja didalam interval tersebut.

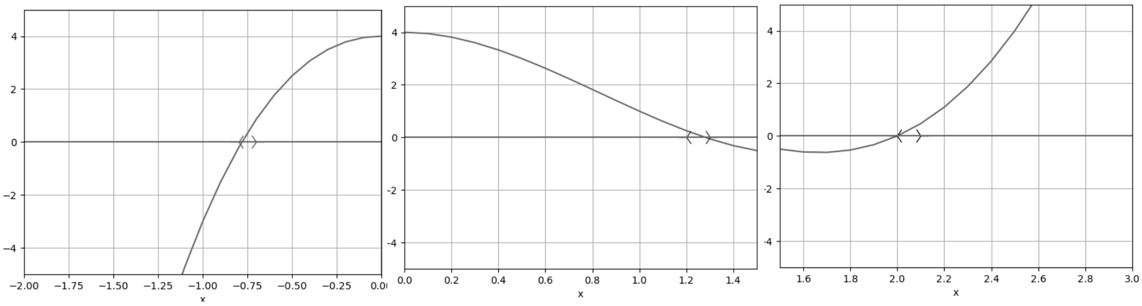
$$f(x) = 2x^3 - 5x^2 + 4 = 0$$

Pada Gambar 2, persamaan $f(x) = 2x^3 - 5x^2 + 4 = 0$ menghasilkan tiga buah akar persamaan dalam interval -4 $<$ x $<$ 4, kondisi tersebut dapat diperkecil intervalnya seperti pada Gambar 3, baik itu di bagian kiri (x bilangan negatif) maupun bagian kanan (x bilangan positif). Tanda $<$ $>$ yang berwarna merah, biru, atau hijau adalah posisi yang menghimpit akar solusi numerik yang akan didapatkan, jadi metode Bisection akan melakukan proses perhitungan secara berulang-ulang pada salah satu interval diatas.



Gambar 2. Grafik akar persamaan

[1] Aproksimasi adalah metode pembulatan angka terhadap hasil pengukuran dan tidak berlaku untuk hal yang sifatnya analitis/eksak (seperti hasil dari membilang atau menghitung).
[2] Diskritisasi merupakan proses kuantisasi sifat-sifat kontinu.



(a) (b) (c)

Gambar 3. a) Interval -2 < x < 0, b) Interval 0 < x < 1.5, c) Interval 1.5 < x < 3

Algoritma Metode Bisection

Pada *book section* [25] menjelaskan bahwa metode bisection dapat diterapkan dengan mengikuti langkah-langkah atau algoritma dibawah ini

1. Tentukan dan masukkan rumus persamaan dengan koefisien polinomial menurun
2. Jika *f(i) > 0* dan *f(i+1) < 0* maka *aa(k) = x(i)* dan *bb(k) = x(i+1)*
3. Jika *f(i) < 0* dan *f(i+1) > 0* maka *aa(k) = x(i)* dan *bb(k) = x(i+1)*
4. Hitung $xx = \frac{aa(k)+bb(k)}{2}$
5. Hitung *fx*
6.
```
IF f(x) * f(a) < 0 THEN
   SET bb(k) = xx
ELSE
   SET aa(k) = xx
IF ABS(f(x)) < 0.0001 THEN
   root = xx
   WRITE root
   END
ELSE
   Kembali ke langkah 4
```
7. Selesai

METODE

Gambaran Umum

Dalam penelitian ini akan menerapkan studi *pre-test* dan *post-test* untuk menguji keefektifan algoritma bisection dari persamaan (1) pada program yang dibangun dalam mencari akar persamaan pada suatu permasalahan nantinya. Rumus dari metode ini menerapkan konsep nilai rata-rata dari dua nilai, karena nama dari metode ini adalah Bisection atau dapat diartikan menjadi metode bagi dua, maka cara penerapannya akan sangat mirip dengan mencari nilai rata-rata dua angka. Dalam mencari nilai error dapat dilakukan dengan dua cara yaitu persamaan (2) dann (3) karena dalam masukan permasalahan akan terjadi dua kondisi yakni kondisi masukan dengan nilai eksak atau nilai akar sebenarnya (jika diketahui) tetapi jika nilai eksak tidak diketahui (masukan tanpa nilai eksak) maka akan ada toleransi error yang berlaku, toleransi error yang ditetapkan oleh penulis adalah 0.00001, dengan toleransi itu nilai error yang baru (nilai $f(x_2)$) akan dapat dibandingkan dengan nilai dari toleransi error tersebut.

$$x_3 = \frac{x_1 + x_2}{2} \quad \text{….. (1)}$$

$$Error = \frac{|x_3 - eksak|}{eksak} * 100 \quad \text{….. (2)}$$

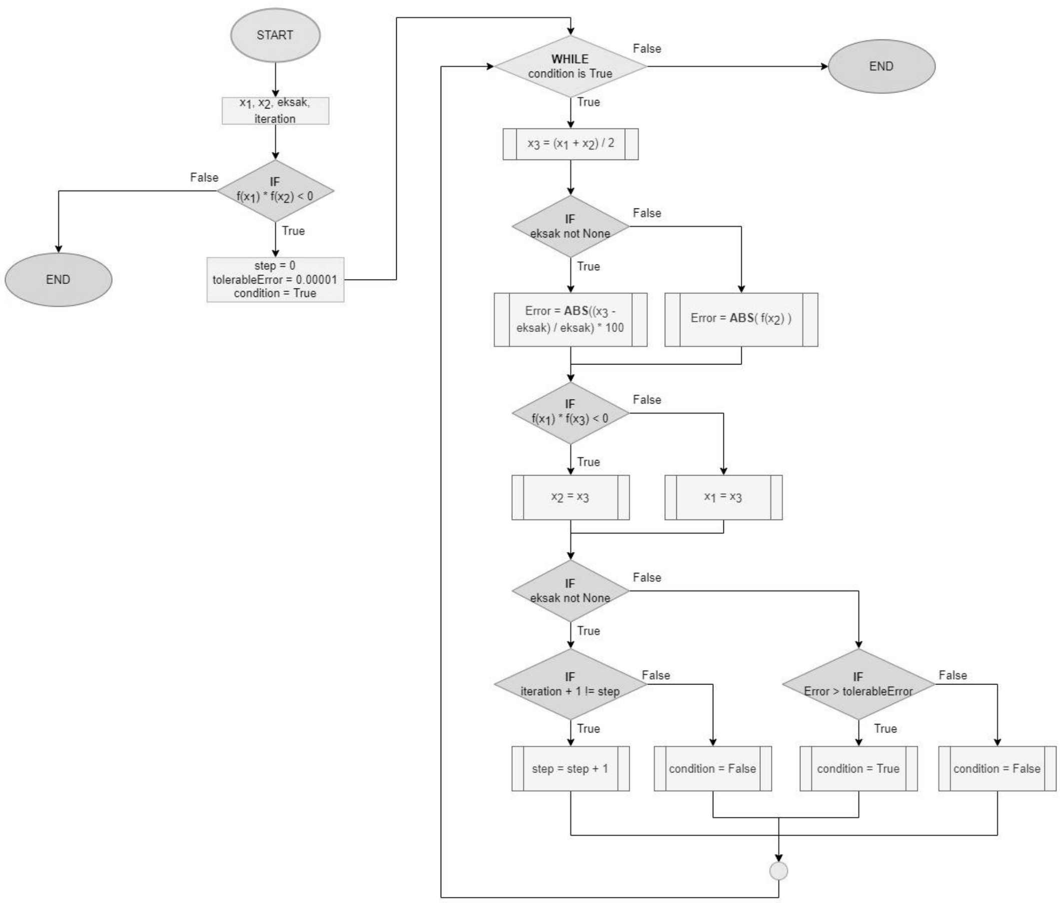
$$Error = |f(x_2)| \quad \text{….. (3)}$$

Perancangan dan Pembangunan

Tahap ini akan dijelaskan tentang alur proses dalam bentuk *flowchart*, perhitungan manual pada Excel, prosedur atau proses yang akan dilakukan oleh program serta tampilan dasar program. Diagram Alir dari program dapat dilihat pada Gambar 4. Peneliti mengambil satu permasalahan sebagai percobaan awal dalam menguji algoritma dari flowchart diatas, untuk mempermudah dalam mengevaluasi permasalahan yang dipilih berupa persamaan berpangkat seperti berikut.

$$f(x) = x^3 - 5x + 4 = 0 \qquad \text{….. (4)}$$

Sebelum membuat atau mengimplementasikan flowchart diatas kedalam program, sebaiknya membuat sebuah peritungan manual menggunakan software Ms. Excel yang nantinya dapat digunakan sebagai tolak ukur dari perhitungan pada program. Perhitungan manual dengan excel menggunakan formula – formula khusus yang ada pada excel untuk mempermudah dalam perhitungan nantinya. Hasil dari perhitungan manual dapat dilihat pada gambar 5. Dengan menggunakan permasalahan yang telah ditentukan, peneliti menetapkan bahwa $x_1 = 1$; $x_2 = 2$; karena di interval itu terdapat nilai akar yang dapat dicari, kemudian peneliti juga memberikan nilai eksak = 1.55938 yang artinya nilai akar sebenarnya dari persamaan ini adalah nilai yang ada pada eksak; dan peneliti juga membatasi banyaknya iterasi atau perulangan perhitungan adalah sebanyak 17 kali (di excel tertulis 0 sampai 16 yang artinya ada 17 baris). Pada Gambar 5, bagian yang berwarna kuning merupakan hasil akar persamaan jika diberikan masukan nilai eksak, hasil dari akar tersebut bernilai 1.55859 (lihat kolom x_3); mengapa yang diambil adalah baris ***i*-7**? karena nilai *Error Relative* nya yang paling kecil dari semua perulangan lain. Sedangkan bagian yang diwarnai dengan warna jingga (oranye) adalah hasil akar persamaan dengan menggunakan Toleransi error (tidak ada campur tangan eksak), nilai akar diambil pada ***i*-14** karena nilai Error Tolerable nya masih ada diatas bilangan nol dan tidak melewati (dibawah) nilai Toleransi Error 0.00001.



Gambar 4. Flowchart program perhitungan solusi numerik

formula	x^3 - 5*x + 4		
x1	1		
x2	2		
eksak	1,55938	Tolerable error	0,00001

i	x1	x2	x3	f(x1)	f(x2)	f(x3)	|Error Tolerable|	|Error Relative|
0	1	2	1,5	0	2	-0,125	2	3,80761523
1	1,5	2	1,75	-0,125	2	0,609375	2	12,2244489
2	1,5	1,75	1,625	-0,125	0,60938	0,166015625	0,609375	4,208416834
3	1,5	1,625	1,5625	-0,125	0,16602	0,002197266	0,166015625	0,200400802
4	1,5	1,5625	1,53125	-0,125	0,0022	-0,065887451	0,002197266	1,803607214
5	1,53125	1,5625	1,54688	-0,06589	0,0022	-0,032978058	0,002197266	0,801603206
6	1,54688	1,5625	1,55469	-0,03298	0,0022	-0,015675068	0,002197266	0,300601202
7	1,55469	1,5625	1,55859	-0,01568	0,0022	-0,006810248	0,002197266	0,0501002
8	1,55859	1,5625	1,56055	-0,00681	0,0022	-0,00232435	0,002197266	0,075150301
9	1,56055	1,5625	1,56152	-0,00232	0,0022	-6,80098E-05	0,002197266	0,137775551
10	1,56152	1,5625	1,56201	-6,8E-05	0,0022	0,001063511	0,002197266	0,169088176
11	1,56152	1,56201	1,56177	-6,8E-05	0,00106	0,000497471	0,001063511	0,153431864
12	1,56152	1,56177	1,56165	-6,8E-05	0,0005	0,000214661	0,000497471	0,145603707
13	1,56152	1,56165	1,56158	-6,8E-05	0,00021	7,33081E-05	0,000214661	0,141689629
14	1,56152	1,56158	1,56155	-6,8E-05	7,3E-05	0,0000026	0,0000733	0,13973259
15	1,56152	1,56155	1,56154	-6,8E-05	2,6E-06	-0,0000327	0,0000026	0,138754071
16	1,56154	1,56155	1,56155	-3,3E-05	2,6E-06	-0,0000150	0,0000026	0,13924333

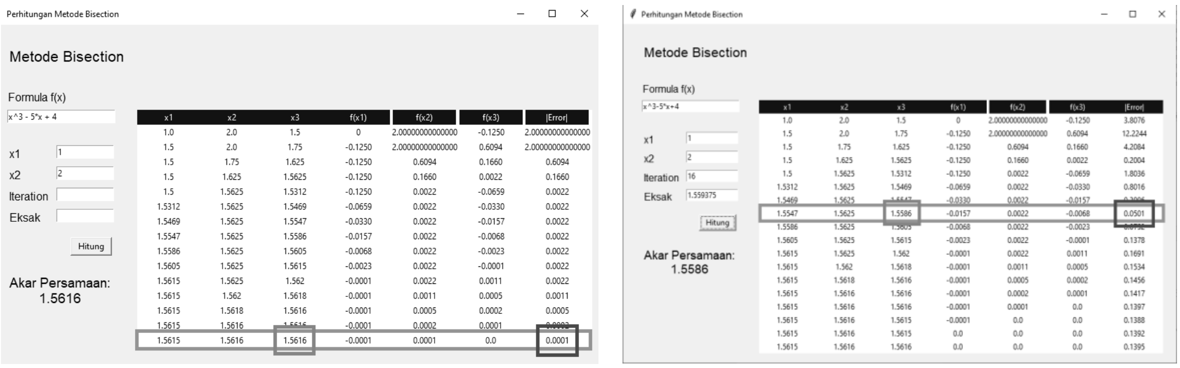
Hasil akar persamaan dengan inputan Eksak
Hasil akar persamaan dengan Toleransi Error

Gambar 5. Perhitungan manual dengan Ms.Excel

HASIL DAN PEMBAHASAN

Proses ini akan menjadi test terakhir dari program untuk nantinya dapat digunakan secara luas, pada bagian ini program yang ditest adalah program yang kompleks dengan tampilan layaknya software pada umumnya (GUI). Pengujian kebeberapa rumus persamaan dibawah ini semuanya akan mengambil nilai awal kecil dan akan dilakukan pengulangan perhitungan (iterasi) sebanyak 18 kali untuk yang memiliki eksak dan yang tidak memiliki eksak akan dilakukan perhitungan hingga mencapat toleransi error senilai 0.00001. Hasil dari dari pre-test menggunakan CLI dan post-test yang telah menggunakan tampilan GUI menghasilkan keluaran yang mirip bahkan ada beberapa yang sama satu sama lain. Perubahan tampilan dari CLI ke GUI tidak berpengaruh besar kepada keluaran programnya, hanya saja kemungkinan akan ada kesalahan pembulatan (*Roundoff Error*) atau kesalahan pemotongan (*Truncation error*) yang membuat hasil keluarannya berbeda.

Selanjutnya, peneliti mencoba untuk membagikan program ini ke beberapa responden yang bersedia menggunakannya. Dari data survei yang telah masuk, tercatat bahwa responden yang masuk sebanyak 35 responden dengan 28 responden laki-laki dan 7 responden perempuan. Dari ke-35 responden tersebut telah menjawab 5 pertanyaan mengenai kegunaan dari program ini dan hasilnya dapat dilihat pada Tabel 2. Rata-rata skor jawaban para responden dari setiap pertanyaan adalah sebesar 74.51%. Angka ini jika lihat pada skala likert akan masuk kedalam skala skor 3 sampai 4 yang artinya program ini bersifat *usability* atau “Berguna” bagi para responden.



(a) (b)

Gambar 6. a) Masukan tanpa nilai eksak, b) Masukan dengan nilai eksak

Tabel 1. Data Pengujian

X1	X2	Solusi yang didapat	Eksak	Error
$x^3 - 5x + 4$				
1	2	1.5616	-	0.0001
1	2	1.5586	1.5593	0.0504
$x^3 + x^2 - 3x - 3$				
1	2	1.732	-	0.000…
1	2	1.732	1.73205	0.000…
$\cos(x) - 3x$				
0.3	0.4	0.3168	-	0.0004
0.3	0.4	0.3164	0.31641	0.0012
$10^x - 100 + 2x$				
1	2	1.9824	-	0.000…
1	2	1.985	1.98	0.1346

Tabel 2. Pertanyaan untuk responden dan skornya

Q	Pertanyaan	Skor
Q1	Kemudahan Aplikasi Dipahami	75,43%
Q2	Kemudahan Aplikasi Dipelajari	73,71%
Q3	Kemudahan Pengoperasian Aplikasi	77,71%
Q4	Ketertarikan untuk menggunakan Aplikasi	74,29%
Q5	Kesesuaian Aplikasi	71,43%

KESIMPULAN

Semua orang dapat menerapkan perhitungan manual menggunakan tulis tangan tetapi itu sangat membosankan karena perhitungan dilakukan secara berulang-ulang, cara lain dapat menggunakan software Ms.Excel tetapi hanya orang-orang yang sudah paham dalam pengoperasian software Excel ini seperti halnya menulis formula, manipulasi *row* dan *columns*, dan lain sebagainya untuk dapat mendapatkan nilai solusinya, dengan adanya penelitian ini diharapkan dapat membantu orang-orang baik itu masyarakat umum maupun mahasiswa yang sedang berhadapan dengan pencarian nilai solusi dari persamaan numerik, dikarenakan dengan adanya program perhitungan solusi numerik ini para pengguna tidak perlu repot-repot untuk menuliskan *formula – formula* yang rumit, cukup memasukan rumus permasalahan numerik yang ingin dicari solusinya dan data-data pendukung lainnya, program ini sudah mampu mendapatkan solusi yang dicari secara cepat. Dari 35 responden yang telah mencoba program ini didapatkan skor rata-rata survei tentang kebergunaan aplikasi bagi mereka sebesar 74.51% yang berarti program ini "Berguna" bagi para responden. Program ini tidak dibatasi hanya berupa script maupun software aplikasi tetapi masih dapat dikembangkan lagi misalnya menjadi sebuah website supaya semua orang dapat menggunakannya secara bebas dan terbuka.

DAFTAR PUSTAKA

[1] Suarga, *Komputasi Numerik : Pemrograman MATLAB untuk Metode Numerik*, 1st ed. Yogyakarta: Andi offset, 2014. Accessed: Dec. 07, 2021. [Online]. Available: http://opac.lib.ugm.ac.id/index.php?mod=book_detail&sub=BookDetail&act=view&typ=htmlext&buku_id=744392&unit_id=1

[2] S. Nurhabibah Hutagalung, "Pemahaman Metode Numerik (Studi Kasus Metode New-Rhapson) Menggunakan Pemprogrman Matlab," *J. Teknol. Inf. JurTI*, vol. 1, no. 1, 2017, Accessed: Dec. 07, 2021. [Online]. Available: https://media.neliti.com/media/publications/281897-emahaman-metode-numerik-studi-kasus-meto-3465fedf.pdf

[3] F. Agus, O. Ihza Gifari, and Z. Alfandi Kamil, “Komputasi Numerik pada Kasus Penentuan Penyakit Tanaman Hias | Agus | Informatika Mulawarman : Jurnal Ilmiah Ilmu Komputer,” *Inform. Mulawarman J. Ilm. Ilmu Komput.*, vol. 16, no. 1, pp. 42–48, 2021, doi: http://dx.doi.org/10.30872/jim.v16i1.526.

[4] I. K. Adi Atmika, *Metode Numerik*. JURUSAN TEKNIK MESIN FAKULTAS TEKNIK UNIVERSITAS UDAYANA, 2016. Accessed: Dec. 07, 2021. [Online]. Available: https://simdos.unud.ac.id/uploads/file_pendidikan_1_dir/11ed6009feeb148225064cd0c4989964.pdf

[5] I. W. Santiyasa, “Algoritma Newton Raphson Dengan Fungsi Non-Linier,” *J. Ilmu Komput.*, 2009, Accessed: Dec. 07, 2021. [Online]. Available: https://ojs.unud.ac.id/index.php/jik/article/view/2679

[6] M. Panjaitan, “Pemahaman Metode Numerik Menggunakan Pemprogrman Matlab (Studi Kasus : Metode Secant),” *JurTI J. Teknol. Inf.*, vol. 1, no. 1, Art. no. 1, Jun. 2017.

[7] P. Batarius, “Nilai Awal Pada Metode Newton-Raphson Yang Dimodifikasi Dalam Penentuan Akar Persamaan | Pi: Mathematics Education Journal,” May 2020, Accessed: Dec. 07, 2021. [Online]. Available: https://ejournal.unikama.ac.id/index.php/pmej/article/view/2784

[8] I. Nur, “Penerapan Metode Bagi-Dua (Bisection) pada Analisis Pulang-Pokok (Break Even),” *Semin. Nas. MIPA 2006 Penelit. Pendidik. Dan Penerapan MIPA Serta Peranannya Dalam Peningkatan Keprofesionalan Pendidik Dan Tenaga Kependidikan*, Aug. 2006, Accessed: Dec. 07, 2021. [Online]. Available: http://fmipa.uny.ac.id

[9] J. Wigati, “Solusi Numerik Persamaan Non-Linier Dengan Metode Bisection Dan Regula Falsi,” *J. Teknol. Terap. G-Tech*, vol. 1, no. 1, pp. 5–17, 2017, doi: 10.33379/gtech.v1i1.262.

[10] T. P. Silaban and F. Ahyaningsih, “Pengaruh Perubahan Nilai Parameter Terhadap Nilai Error Pada Metode Runge-Kutta Orde 3,” *KARISMATIKA Kumpul. Artik. Ilm. Inform. Stat. Mat. Dan Apl.*, vol. 3, no. 2, Art. no. 2, Aug. 2017, doi: 10.24114/jmk.v3i2.8807.

[11] M. D. Nasution, E. Nasution, and F. Haryati, “Pengembangan Bahan Ajar Metode Numerik Dengan Pendekatan Metakognitif Berbantuan Matlab,” *Mosharafa J. Pendidik. Mat.*, vol. 6, no. 1, Art. no. 1, 2017, doi: 10.31980/mosharafa.v6i1.295.

[12] A. M. Retta, A. Isroqmi, and T. D. Nopriyanti, “Pengaruh Penerapan Algoritma Terhadap Pembelajaran Pemrograman Komputer,” *Indiktika J. Inov. Pendidik. Mat.*, vol. 2, no. 2, Art. no. 2, May 2020, doi: 10.31851/indiktika.v2i2.4125.

[13] G. G. Maulana, “Pembelajaran Dasar Algoritma Dan Pemrograman Menggunakan El-Goritma Berbasis Web,” *J. Tek. Mesin*, vol. 6, no. 2, pp. 69–73, Mar. 2017, doi: 10.22441/jtm.v6i2.1183.

[14] F. Mutia Putri, “Pembelajaran Dasar Algoritma Dan Pemrograman Menggunakan El-Goritma Berbasis Web,” 2021.

[15] E. Sunandar, “Penyelesaian Sistem Persamaan Non-Linier Dengan Metode Bisection & Metode Regula Falsi Menggunakan Bahasa Program Java,” *PETIR*, vol. 12, no. 2, Art. no. 2, Sep. 2019, doi: 10.33322/petir.v12i2.490.

[16] J. Wigati, “SOLUSI NUMERIK PERSAMAAN NON-LINIER DENGAN METODE BISECTION DAN REGULA FALSI,” *G-Tech J. Teknol. Terap.*, vol. 1, no. 1, pp. 5–17, 2017, doi: 10.33379/gtech.v1i1.262.

[17] “Error Analysis.” 2016. Accessed: Dec. 08, 2021. [Online]. Available: https://lms.su.edu.pk/download?filename=1588465655-error-analysis.pdf&lesson=17002

[18] B. D. Storey, “Error in Numerical Methods,” p. 6, 2015.

[19] U. M. Ascher and C. Greif, “Numerical Algorithms and Roundoff Errors.pdf,” in *Numerical Algorithms and Roundoff Errors*, 2011, p. 36. Accessed: Dec. 08, 2021. [Online]. Available: http://www.sml.ece.upatras.gr/images/UploadedFiles/efarmosmenes-ypologistikes-methodoi/RoundoffError.pdf

[20] B. Chen, “Roundoff and Truncation Errors.” 2012.

[21] P. Sandipan, “NP 2.pdf,” in *Bisection Method*, 2020. Accessed: Dec. 23, 2021.