IMPLEMENTASI DAN ANALISIS TINGKAT AKURASI *SOFTWARE* PENYELESAIAN PERSAMAAN NON LINIER DENGAN METODE *FIXED POINT ITERATION* DAN *METODE BISECTION*

Asminah [1)] **Vivi Sahfitri** [2)]
[1,2)] Jurusan Teknik Informatika Universitas Bina Darma Palembang
Jln. Ahmad Yani no. 12 Palembang 30264 Telp (0711) 515679
Email : ndung_haqi@yahoo.co.id & vsahvitri@yahoo.com

Abstract
This study aims to build the software solution of non-linear equations by the method of Bisection and Fixed Point Iteration, revealed that the software non-linear equations have been carried out through five stages, namely: (1) Engineering systems modeling (2) needs analysis software, (3) create a design (4) code generation, (5) conduct testing and implementation using the programming languages C # (C-Sharp). Validator research have 4 experts in the field of both matter and media. Data analysis using descriptive analysis techniques. Settlement of one case of non-linear equations assessed the average time it takes not too long because the turnaround time less than 1 second. Judging from the fault tolerance, the number of iterations and the average time it takes the Bisection method is better than the fixed point iteration method. Overall in terms of both content and media, that any assessment of the software is considered good and decent that can be used lecturers taught courses in numerical methods, specially methods Bisection and Fixed Point iteration.

***Keywords** : software, non-linear equation, C # (C-Sharp), bisection method, fixed point iteration method*

1. PENDAHULUAN

Tersedianya komputer secara luas saat ini, benar-benar telah memberikan banyak manfaat dalam pemakaiannya. Salah satu contoh dalam membantu melakukan komputasi atau perhitungan di bidang matematika. Dalam suatu perhitungan, ketepatan, ketelitian dan kecepatan hasil adalah suatu hal yang sangat penting dalam menyelesaikan masalah. Perhitungan yang cepat dan teliti akan menyebabkan sesuatu menjadi lebih diminati terutama dalam peramalan (*forecasting),* ekonomi, matematika, sains, teknologi dan sebagainya.

Salah satu proses komputasi yang dilakukan adalah persamaan-persamaan non-linier. Persamaan non-linier adalah suatu persamaan untuk mencari akar x sehingga F(x) = 0, fungsi ini tidak memiliki rumus tertentu sehingga untuk mendapatkan nilai akarnya digunakan beberapa metode pendekatan diantaranya metode *bisection* dan metode iterasi *fixed point*. Metode setengah interval atau metode *bisection* adalah cara menyelesaikan persamaan non-linier dengan membagi dua nilai x_1 dan x_2 dilakukan berulang-ulang sampai nilai x lebih kecil dari nilai tolerasi yang ditentukan. Metode *bisection* melakukan pengamatan terhadap nilai f(x) dengan berbagai nilai x, yang mempunyai perbedaan tanda. Taksiran akar diperhalus dengan cara membagi dua (2) pada interval x yang mempunyai beda tanda tersebut. Sedangkan metode iterasi *fixed point* dijalankan dengan cara membuat fungsi f(x) menjadi bentuk fungsi implisit f(x)=0 kemudian x=g(x), iterasi yang digunakan adalah dalam bentuk persamaan; $x_{n+1} = g(x_n)$.

2. TINJAUAN PUSTAKA

2.1. Persamaan Non-Linier (Tak Linier)

Persamanan non-linier adalah suatu persamaan untuk mencari akar x sehingga F(x) = 0, fungsi ini tidak mempunyai rumus tertentu sehingga untuk mendapatkn nilai akarnya digunakan beberapa metode pendekatan. Persamaan dengan sistem ini terdiri dari himpunan-himpunan nilai x yang secara simultan atau bersama-sama memberikan semua persamaan tersebut nilai yang sama dengan nol, serta penentuan akar-akar satu persamaan tunggal. Suatu masalah yang berkaitan dengan penyelesaian sistem ini adalah bagaimana melokasikan akar-akar himpunan persamaan non-linier.

2.2. Metode Penyelesaian persamaan non-Linier

2.2.1. Setengah interval (B*isection)*

Metode setengah interval atau metode *bisection* adalah cara menyelesaikan persamaan non-linier dengan membagi dua nilai x_1 dan x_2 dilakukan berulang-ulang sampai nilai x lebih kecil dari nilai tolerasi yang ditentukan. Algoritma metode setengah interval adalah sebagai berikut :

1) Pilih x_1 bawah dan x_2 puncak taksiran untuk akar, sehingga perubahan fungsi mencakup seluruh interval. Hal ini dapat diperiksa dengan memastikan : $f(x_1).f(x_2) < 0$
2) Taksiran akar x, ditentukan oleh :

$$x_r = \frac{x_1 + x_2}{2}$$

3) Buat evaluasi dengan memastikan pada bagian interval mana akar berbeda:
   a. Jika f(x1).f(x2) < 0 akan berada pada bagian interval bawah, maka x2 = xr ,dan kembali kelangkah 2
   b. Jika f(x1).f(x2) > 0 akan berada pada bagian interval atas , maka x1 = xr , dan kembali kelangkah 2
   c. Jika f(x1).f(x2) = 0, akar setara xr, perhitungan dihentikan, atau bisa juga :
   $|f(x_1).f(x_2)| < \varepsilon$ Dimana ε adalah harga toleransi yang dibuat (Munir : 2000).

2.2.2 Metode Iterasi *Fixed Point*

Teknik iterasi *fixed point* dijalankan dengan cara membuat fungsi f(x) menjadi bentuk fungsi implisit f(x)=0 kemudian x=g(x). Algortima dari metode iterasi Fixed point adalah sebagai berikut :

1. Tentukan nilai taksiran awal x_n
2. Lakukan perhitungan taksiran akar dengan mempergunakan persamaan; $X_{n+1} = g(x_n)$
3. Perhitungan dihentikan jika; $\left| x_{n+1} - x_n \right| \leq \varepsilon$

2.2.3 Galat (Kesalahan / ketidaksesuaian)

Analisis Galat (kesalahan) dalam suatu hasil komputasi merupakan dasar semua perhitungan yang baik, baik dikerjakan dengan tangan atau manual ataupun dengan komputer. Walaupun selalu berusaha untuk memperoleh jawaban yang eksak, jawaban demikian jarang diperoleh secara numerik. Dalam tiap langkah penyelesaian persoalan dan formulasi hingga komputasi numeriknya, galat (kesalahan) dan ketidakpastian dapat terjadi.

Menurut Djojodiharjo (2000) proses pemecahan persoalan, pada umumnya berlangsung tiga tahap yaitu:

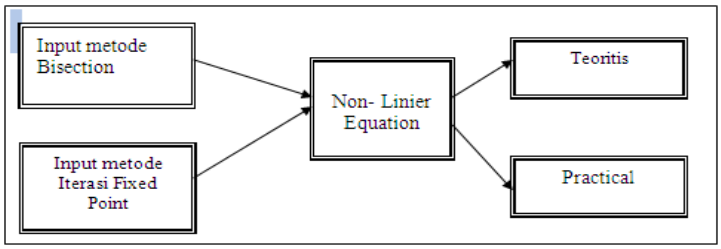
1. Perumusan secara tepat dari model matematika dan model numerik yang berkaitan
2. Penyusunan (konstruksi) metode untuk memecahkan persoalan numerik
3. Penerapan metode untuk menghitung jawaban yang di cari.

3. METODE PENELITIAN

3.1. Desain Penelitian

Penelitian ini adalah penelitian pengembangan menurut Borg & Gall (2003 : 772), penelitian pengembangan adalah penelitian yang berorientasi untuk mengembangkan dan memvalidasi produk-produk yang digunakan dalam pendidikan. Sedangkan menurut Gay (1981 : 10) penelitian pengembangan bukan untuk membuat teori atau menguji teori melainkan untuk mengembangkan produk-produk yang efektif untuk digunakan di sekolah.

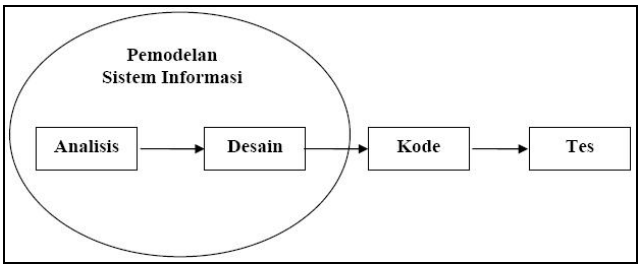
Penelitian ini akan dilaksanakan melalui dua tahap yaitu, yang pertama adalah membuat rancangan atau desain yang akan digunakan untuk membangun perangkat lunak (*software)* yang nantinya akan diimplementasikan. Yang kedua adalah analisis kualitatif yang di lakukan untuk mengetahui tingkat akurasi, ketelitian dan kecepatan hasil dari perhitungan non-linier yang menggunakan metode setengah interval (*bisection)* dan Iterasi *fixed point*. Kerangka pemikiran dalam penelitian ini dapat digambarkan pada gambar berikut ini :



Gambar 1. Kerangka pemikiran perancangan *Software*

3.2. Analisis dan Pengembangan Sistem

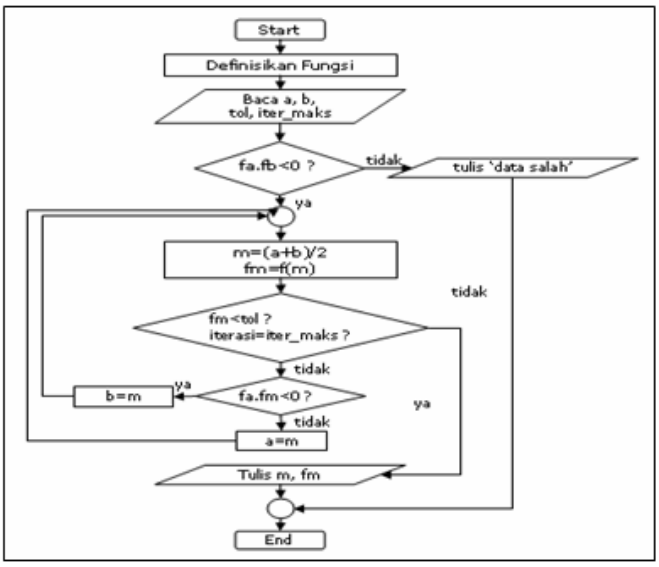
Pada penelitian ini, model yang menjadi acuan adalah model *waterfall* dimana salah satu strategi yang sering dipakai sebagai *model proses* atau *paradigma rekayasa perangkat lunak.* Model *waterfall* merupakan pendekatan perangkat lunak sistematik dan sekuensial yang dimulai dari tahap analisis, desain, kode, pengujian, dan pemeliharaan. Secara garis besar model pengembangan tersebut dapat dilihat pada gambar berikut ini (Pressman : 2002).



Gambar 2.Siklus model *waterfall*

3.2.1. Sub menu metode *bisection*

Algoritma dalam sub menu Metode *Bisection* dapat di lihat dalam diagram alir berikut ini:



Gambar 3. Diagram alir metode *Bisection*

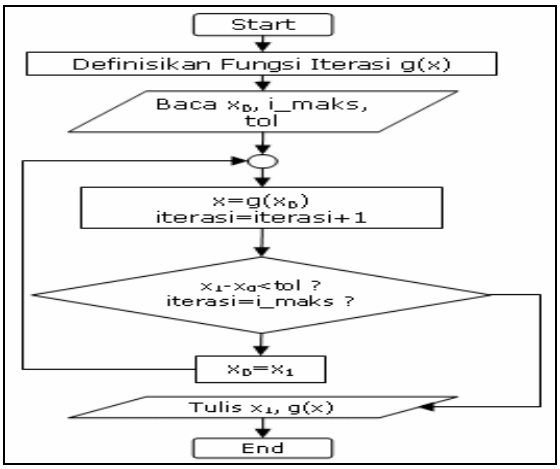
Pada sub menu ini dosen dapat menginputkan nilai variabel untuk membentuk suatu persamaan non-linier, menentukan nilai toleransi kesalahannya yaitu (e), nilai (X1) dan nilai (X2). Nilai-nilai yang sudah diinputkan tadi diproses secara otomatis oleh sistem. Hasil dari proses tadi akan menampilkan prosedur atau proses iterasinya. Iterasi-iterasi tersebut akan selesai prosesnya dengan sendirinya apabila nilai f(x) nya sudah mendekati nilai galatnya atau $|f(xi)| \geq e$. Setelah itu hasil iterasi-iterasi tadi dibuat bentuk tabelnya secara keseluruhan. Selain tabel sistem juga akan menggambarkan grafiknya sesuai dengan nilai dari setiap iterasi yang ada. Algoritma program sebagai berikut:

1. Tentukan fungsi f(x), batas bawah a, batas atas b, toleransi, dan jumlah iterasi maksimum.
2. Hitung f(a) dan f(b).
3. Periksa apakah f(a).f(b)> 0; jika ya, keluar dari progam karena pada interval yang diberikan tidak terdapat akar persamaan.
4. Hitung nilai m = (a+b)/2.
5. Jika nilai mutlak f(m) < toleransi, tuliskan m sebagai hasil perhitungan, dan akhiri program; jika tidak, lanjutkan ke langkah berikutnya.
6. Jika jumlah iterasi > iterasi maksimum, akhiri program.
7. Jika f(a).f(m)<0, maka b=m, jika tidak,
   a = m.
8. Kembali ke langkah (2).

3.2.2. Sub menu metode iterasi *fixed point*

Pada menu metode Iterasi *fixed point dosen* dapat menginputkan nilai variabel untuk membentuk suatu persamaan non- linier, menentukan nilai toleransi kesalahannya yaitu (e) dan nilai pendekatan awal (X0). Nilai-nilai yang sudah diinputkan tadi diproses secara otomatis oleh sistem. Hasil dari proses tadi akan menampilkan

prosedur atau proses iterasinya. Iterasi-iterasi tersebut akan selesai prosesnya dengan sendirinya apabila nilai f(x) nya sudah mendekati nilai galatnya atau |f(xi)|≥ *e*. Setelah itu hasil iterasi-iterasi tadi dibuat bentuk tabelnya secara keseluruhan. Selain tabel sistem juga akan menggambarkan grafiknya sesuai dengan nilai dari setiap iterasi yang ada. Berikut ini diagram alir metode iterasi *fixed point*



Gambar 4. Diagram alir metode iterasi *fixed point*

Berikut ini Algoritma program dengan metode Iterasi *fixed point.*

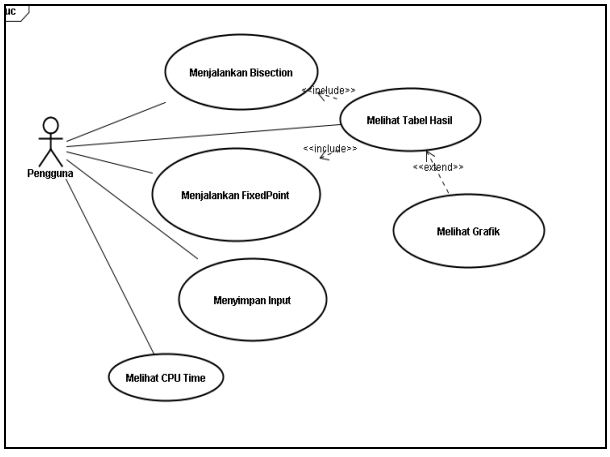
a). Tentukan x0, toleransi, dan jumlah iterasi maksimum.
b). Hitung xbaru= g(x0).
c). Jika nilai mutlak (xbaru – x0) < toleransi, diperoleh tulisan xbaru sebagai hasil perhitungan;jika tidak, lanjutkan ke langkah berikutnya.
d). Jika jumlah iterasi > iterasi maksimum, akhiri program.
e). x0 = xbaru, dan kembali ke langkah (b).

3.2.3 Desain Perangkat Lunak

Berdasarkan tahapan pengembangan sistem, maka tahapan selanjutnya adalah design. Pada tahap design ini menentukan struktur data, arsitektur perangkat lunak / sistemnya, antarmuka, dan algoritmanya. Dengan memahami permasalahan yang ada, maka perancangan pembuatanperangkat lunak penyelesaian persamaan non-linier dengan metode *bisection* dan iterasi *fixed point* terdiri dari *use case diagram*, dan *activity diagram.*

USE CASE DIARGRAM

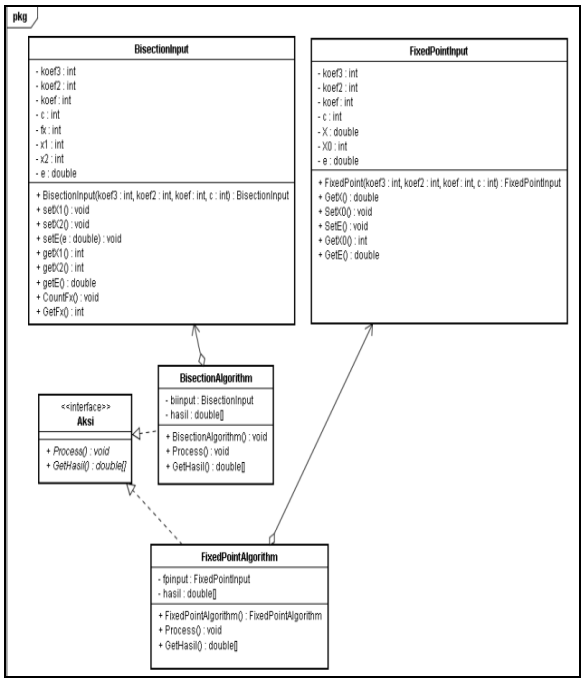
Use Case Diagram (UCD) menjelaskan apa yang akan dilakukan oleh perangkat lunak penyelesaian persamaan non-linier dengan metode *bisection* dan *iterasi fixed point.* UCD menjadi dokumen kerja dari user. Pada *Use Case Diagram* perangkat lunak ini terdiri dari 1 aktor yaitu dosen pengajar. Pada sistem yang dibuat terdapat 2 sub menu pilihan, yaitu sub menu metode *bisection* dan sub menu metode *iterasi fixed point.*



Gambar 5. *Use Case Diagram*

CLASS DIAGRAM

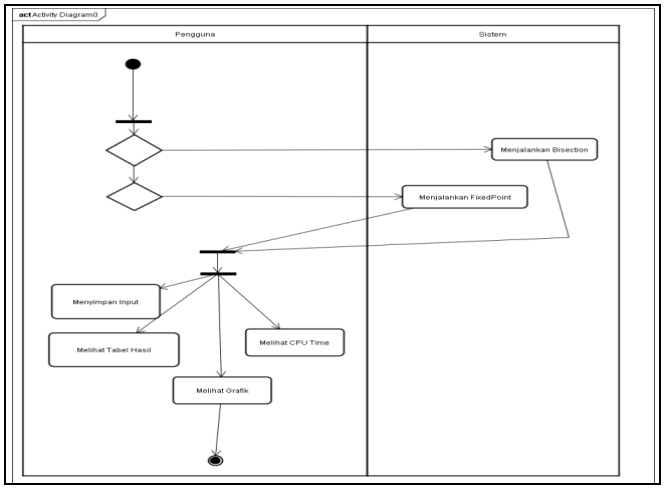
Class diagram merupakan bagian dari *structure diagram. Class diagram* menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Di dalam suatu kelas terdapat atribut dan metode/operasi. Atribut merupakan variabel/informasi yang dimiliki oleh suatu kelas, sedangkan operasi atau metode merupakan fungsi-fungsi yang dimiliki oleh suatu kelas. Untuk kelas, atribut dan metode/operasi sudah didefinisikan di tahap analisis. *Class diagram* pada penelitian ini dapat dilihat pada gambar berikut:



Gambar 6. *Class Diagram*

DIAGRAM ACTIVITY

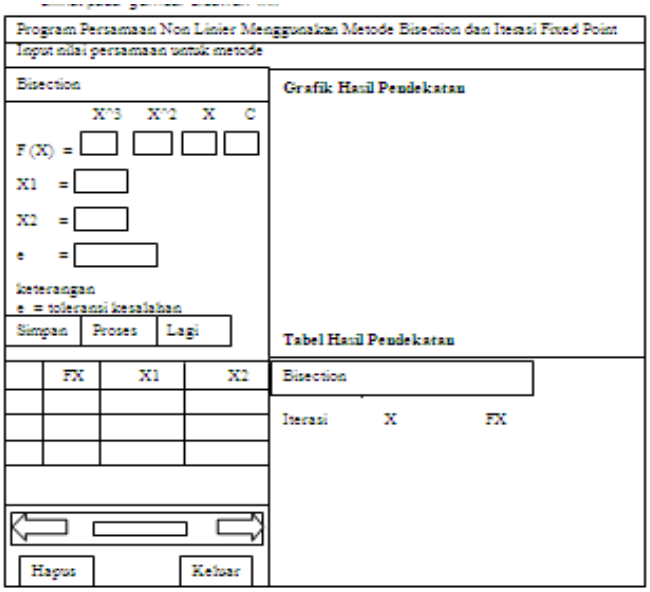
Activity diagram menggambarkan berbagai alir aktivitas (*workflows*) dalam sistem yang sedang dirancang, bagaimana masing-masing alir berawal, *decision* yang mungkin terjadi, dan bagaimana mereka berakhir. Activity diagram merupakan *state diagram* khusus, di mana sebagian besar state adalah action dan sebagian besar transisi di-*trigger* oleh selesainya *state* sebelumnya (*internal processing*). *Activity diagram* pada penelitian ini dapat dilihat pada gambar berikut:



Gambar 7. *Diagram Activity*

Desain Menu *Bisection*

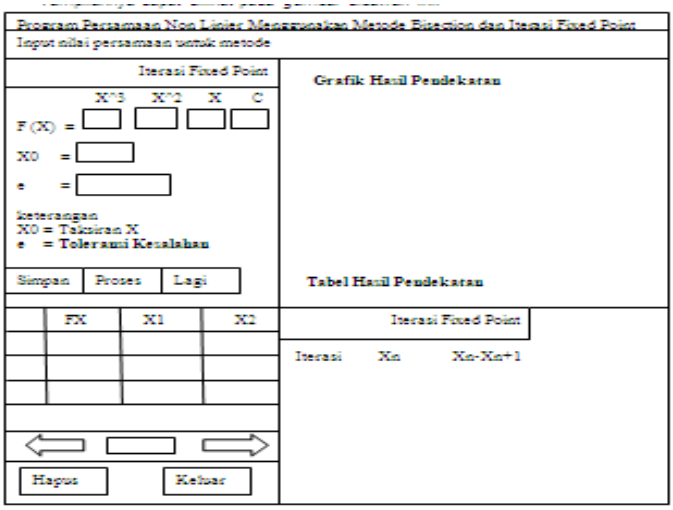
Rancangan form menu *Bisection* merupakan tampilan pertama ketika perangkat lunak penyelesaian persamaan non-linier dengan metode *Bisection* dan Iterasi *Fixed Point* ditampilkan. Tampilannya dapat dilihat pada gambar dibawah ini:



Gambar 8. Rancangan Menu *Bisection*

Desain Menu Iterasi *Fixed Point*

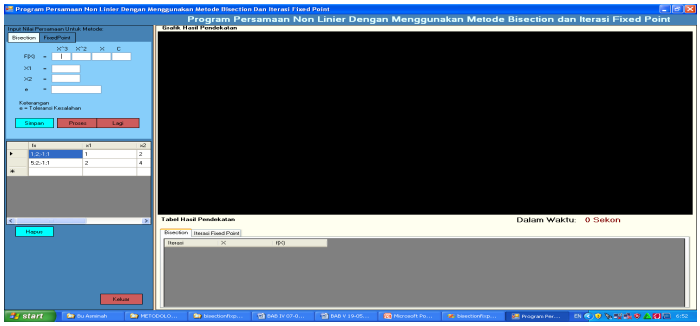
Rancangan form menu Iterasi *Fixed Point* merupakan tampilan pertama ketika perangkat lunak penyelesaian persamaan non-linier dengan metode *Bisection* dan Iterasi *Fixed Point* ditampilkan. Tampilannya dapat dilihat pada gambar dibawah ini:



Gambar 9. Rancangan Menu Iterasi *Fixed Point*

4. HASIL DAN PEMBAHASAN

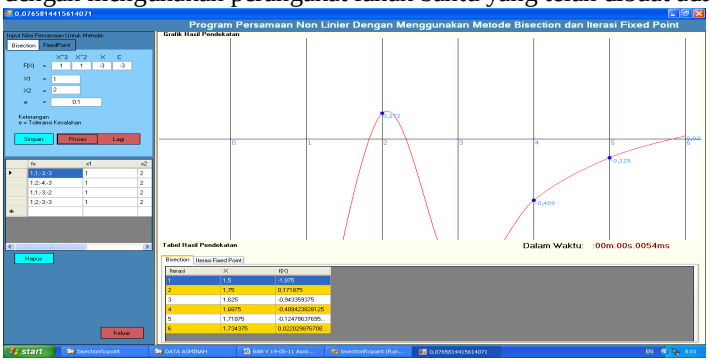
Spesifikasi dari Perangkat lunak ini yaitu terdapatnya Proses mengenai persamaan non-linier dengan metode *Bisection* dan Iterasi *Fixed Point.* Disini kita dapat melihat turunan iterasinya beserta bentuk grafik hasil pendekatan. Dibawah ini dalah tampilan utama dari perangkat lunak penyelesaian persamaan nonlinier dengan metode *bisection* dan Metode *fixed point iteration.*



Gambar 10. Tampilan utama perangkat lunak

3.1. SIMULASI PENYELESAIAN DENGAN METODE BISECTION

Jika di ketahui suatu persamaan non linier berikut :
$Fx = X^3+X^2-3X-3$
asumsi $Fx = 1^3+1^2-3x-x$
X1 = 1
X2 = 2
e = 0,1
Maka hasil pengolahan dengan mengunakan perangakat lunak bantu yang telah dibuat adalah sebagai berikut :

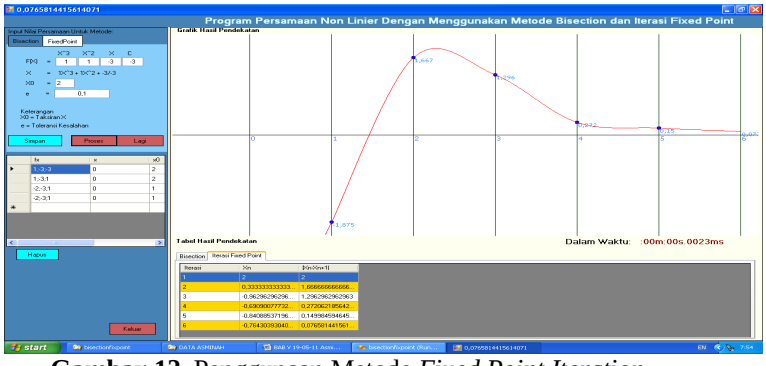


Gambar 11. Penggunaan Metode Bisection

Dari hasil diatas dapat dilihat bahwa untuk penyelesaian soal $F(x)=X^3+X^2-3X-3$, X1 = 1, X2 = 2 dan e = 0,1 berdasarkan perangkat lunak yang dibuat dengan menggunakan metode *bisection* menghasilkan jumlah iterasi sebanyak 2 kali dengan nilai 0,1718 dan dengan CPU time 0054 ms.

2.2. SIMULASI PENYELESAIAN DENGAN METODE FIXED POINT ITERATION

Sedangkan untuk soal yang sama penyelesaian dengan metode *fixed point iteration* yaitu $Fx = X^3+X^2-3X-3$ asumsi $Fx = 1^3+1^2-3x-x$, X0 = 2 e = 0,1dapat dilihat pada gambar berikut ini:



Gambar 12. Penggunaan Metode *Fixed Point Iteration*

Dari gambar perangkat lunak bantu yang dihasilkan diatas maka dapat dilihat bahwa untuk penyelesaian soal $F(x)=X^3+X^2-3X-3$, X0 = 2 dan e = 0,1 berdasarkan perangkat lunak yang dibuat dengan menggunakan

metode iterasi *Fixed point* menghasilkan jumlah iterasi sebanyak 5 kali dengan nilai 0,1499 dan dengan CPU time 0023 ms.

3.2. Analisis Kompleksitas

Berdasarkan dengan soal yang diberikan dan diselesaikan dengan 2 metode yang berbeda maka dapat disimpulkan dalam tabel berikut :

Tabel 1. Analisis Kompleksitas

Metode	Toleransi Kesalahan	Iterasi	F(X)	CPU Time
Bisection	0,1	2	0,1718	0054ms
Fixed Point iteration	0,1	5	0,1499	0023ms

Dari tabel analisis kompleksitas di atas dapat disimpulkan metode *Bisection* lebih baik dibanding metode Iterasi *Fixed Point* terlihat dari jumlah iterasi dan rata-rata waktu yang dibutuhkan tidaklah terlalu lama dalam menyelesaikan suatu kasus.

5. KESIMPULAN

Hasil penelitian pengembangan perangkat lunak persamaan non-linier dengan metode *Bisection* dan Iterasi *Fixed Point* ini dapat disimpulkan bahwa ditinjau dari aspek materi dan media dari perangkat lunak ini, kualitas materi dan media dinilai baik. Berdasarkan hasil penyelesaian satu kasus persamaan non linier dinilai rata-rata waktu yang dibtuhkan tidak terlalu lama karena waktu penyelesaian tidak sampai 1 detik dibandingkan dengan cara pengerjaan secara manual.

Perangkat lunak persamaan non-linier ini dibuat sebagai alternatif dalam membantu dosen memberikan penjelasan tentang persamaan non linier dengan metode *bisection* dan iterasi *fixed point.* Ditinjau dari toleransi kesalahan metode *Bisection* lebih baik dibanding metode iterasi *fixed point* terlihat dari jumlah iterasi dan rata-rata waktu yang dibutuhkan untuk penyelesaian satu kasus seperti terlihat pada tabel yang diberikan. Bahwa kompleksitas penyelesaian masalah non linier dipengaruhi oleh faktor derajat persamaan, dan nilai toleransi kesalahan.

DAFTAR PUSTAKA

Chapra, Steven C, and Canale, Raymond P. 1991 *Metode Numerik untuk Teknik,* Jakarta : Universitas Indonesia

Fairuz El Said, 2010, *Metode Numerik-Galat,* Online, (http://fairuzelsaid.wordpress.com/2010/10/18/metode-numerik-galat/, diakses tanggal 05-10-2010).

Fairuz El Said, 2010, Metode Numerik 01-Pengantar Metode Numerik, Online, (http://fairuzelsaid.wordpress.com/2010/10/13/metode-numerik-01-pengantar-metode-numerik/, diakses tanggal 20-09-2010).

Harijono, Djojodiharjo .2000. *Metode Numerik.* Jakarta : PT. Gramedia Pustaka umum http://sbypresidency.blogspot.com/2009_10_25_archive.html diakses 18-10-2010).

Jogiyanto, 1999. *Analisis dan Desain Sistem Informasi : Pendekatan Terstruktur Teori dan Aplikasi bisnis.* Yogyakarta: Andi offset.

Mahyuzir D. Tavri., 1989. *Analisis dan Perancangan Sistem Pengolahan Data.* Jakarta. Penerbit : PT. Elex Media Komputindo.

Munir R, 2000. *Algoritma dan Pemrograman,* Bandung Informatika.

Nasution, Amrinsyah dan Zakaria Hasbullah., 2001. *Metode Numerik dalam ilmu Rekayas sipil.* Bandung : Institit Teknologi Bandung (ITB).

New, 2010, *Pengertian Perangkat Lunak (Software) Komputer,* Online, (http://tentangkomputerkita.blogspot.com/2010/04/pengertian-perangkat-lunak-software.html, diakses tanggal 11-10-2010)

Pressman, Roger S. 2002. Rekayasa Perangkat Lunak Pendekatan Praktisi. Yogyakarta : Andi Offset.

SmithDev, 2009. *Cara mudah menguasai Microsoft C# 2008.* Yogyakarta : Andi Offset.

Tim Puslitjaknov, *Metode Penelitian Pengembangan,* online, (www.infokursus.net/.../0604091354Metode_Penel_Pengemb_Pembelajaran.pdf), diakses 11-11-2010)

Wilkinson & Cerullo, *Accounting Information* , New York, Jhon Willey And Sons Inc. 1997.