Perbandingan Penyelesaian Persamaan Diferensial Biasa Menggunakan Metode *Backpropagation*, Euler, Heun, dan Runge-Kutta Orde 4

Jayme Yeremia Wijaya[#1], The Houw Liong[*2], Ken Ratri Retno Wardani[*3]

[#]*Magister Manajemen, Universitas Katolik Parahyangan*
Jl. Merdeka no. 30 Bandung

[*]*Departemen Teknik Informatika, Institut Teknologi Harapan Bangsa*
Jl. Dipatiukur no. 83-84 Bandung

[1]jaymeyeremia@gmail.com
[2]thehl007@gmail.com
[3]ratri.ken@gmail.com

Abstract* — *Differential equation are widely used as a model in the mathematics model or other science. In this equation takes a very high level of accuracy that was created several methods to solve the differential equations. One of the method used is Numerical Method and Artificial Neural Network (ANN). There are four methods involved in this study, Euler Method, Heun, and Runge-Kutta Order 4 are included in Numerical Methods, and Backpropagation Neural Network (BPNN) which included in ANN Method. This research is to prove that in solving differential equations using BPNN Method is better than Numerical Method. This is evidenced by the result of Euclidean Distance from BPNN is better than other methods. The result of the solving will be seen more clearly when the differential equation contains elements of chaos. If seen from the graph, BPNN have a graph similar to the graph of the Analitic Solution. Contrast to the solving using Numerical Methods, the line graph has no resemblance to the Analitic Solution.

Keywords* — *diferential equation, artificial neural network, backpropagation, numerical method , chaos, Euclidean , line graph

Abstrak— Persamaan diferensial banyak digunakan sebagai model matematika atau dalam bidang sains lainnya. Dalam persamaan tersebut dibutuhkan tingkat akurasi yang sangat tinggi sehingga diciptakan beberapa metode untuk menyelesaikan persamaan diferensial itu. Salah satu metode yang digunakan adalah Metode Numerik dan Metode *Artificial Neural Network* (ANN). Ada 4 metode yang terlibat dalam penelitian ini, yaitu Metode Euler, Heun, Runge-Kutta Orde 4 yang termasuk pada metode Numerik, dan *Backpropagation Neural Network* (BPNN) yang termasuk dalam Metode ANN. Penelitian ini untuk membuktikan bahwa dalam menyelesaikan persamaan diferensial penggunaan Metode BPNN lebih baik daripada Metode Numerik. Hal ini dibuktikan dengan hasil *Euclidean Distance* dari BPNN lebih baik dibandingkan metode yang lain. Hasil penyelesaian akan terlihat lebih jelas ketika persamaan diferensial tersebut mengandung unsur *chaos.* Jika dilihat dari grafik penyelesaiannya, BPNN memiliki grafik yang mirip dengan grafik dari solusi sejatinya. Berbeda dengan penyelesaian yang menggunakan Metode Numerik, hasil grafik garis yang diperoleh tidak memiliki kemiripan dengan solusi sejatinya.

Kata kunci — persamaan diferensial*, artificial neural network, backpropagation,* metode numerik*, chaos, Euclidean,* grafik garis

I. PENDAHULUAN

Persamaan diferensial dapat dibagi menjadi dua kelompok besar, yaitu persamaan diferensial biasa (PDB) dan persamaan diferensial parsial (PDP). Didalam jurnal ini akan membahas kasus dari PDB saja. PDB adalah persamaan diferensial yang hanya mempunyai satu peubah bebas yang biasanya disimbolkan dengan *x*. Contohnya $y' = x^2 + y^2$ peubah bebasnya adalah *x*, sedangkan peubah terkaitnya adalah y yang merupakan fungsi dari *x* atau ditulis sebagai $y = g(x)$ [2].

Terdapat beberapa metode numerik yang sering digunakan untuk menghitung solusi PDB, mulai dari metode yang paling dasar sampai dengan metode yang lebih teliti, yaitu metode Euler, Heun, Deret Taylor, dan Runge-Kutta bertingkat.[2], maka dari hasil yang didapat tiap metode, seharusnya *error* yang diperoleh metode Euler lebih besar disbanding Runge-Kutta. Terdapat satu metode lagi, yaitu *Artificial Neural Network* (ANN) yang digunakan untuk memprediksi hasil sebuah solusi. ANN terbagi menjadi 2 macam, yaitu *supervised learning* dan *unsupervised learning* [5]. Untuk memprediksi suatu *input* adalah metode *supervised learning* sehingga ANN ini yang digunakan untuk memprediksi hasil dari sebuah solusi. Contoh algoritma dari *supervised learning* adalah: *Backpropagation*, ADALINE, Hebbian, Boltzman, dan Perceptron. *Unsupervised learning* lebih cocok untuk pengklasifikasian suatu pola [3]. Solusi dari persamaan tersebut ditampilkan ke dalam bentuk grafik agar dapat mempermudah dalam membandingkan hasil dari setiap metode, sehingga dapat terlihat metode yang paling baik dalam menyelesaikan PDB.

II. ALGORITMA *BACKPROPAGATION (BPNN)*

Algoritma pelatihan BPNN terdiri dari dua tahap, yaitu *Feed Forward Propagation* dan *Feed Backward Propagation.* Secara umum langkah pelatihan ANN menggunakan BPNN yang dilengkapi bias dan momentum adalah sebagai berikut [1]:

1. Jumlah *input* (pola masukan), *hidden layer*, dan *output* (target pelatihan) ditentukan.

2. Nilai awal diberikan secara random bagi seluruh *weight* antara *input-hidden layer* dan *hidden layer-output.*
3. Langkah 3-11 dilakukan secara berulang, sehingga diperoleh nilai *error* minimal yang memungkinkan bagi ANN untuk belajar dengan baik (*feed forward propagation*).
4. Tiap unit *input* diterima sinyal *input* lalu sinyal tersebut dikirimkan pada seluruh unit *hidden layer.*
5. Tiap unit *hidden layer* ditambah dengan *input* yang dikali dengan *weight* dan dijumlah dengan bias bagian *input.*
6. Tiap unit *output* ditambah dengan nilai keluaran *hidden layer* yang dikali dengan *weight* dan dijumlah dengan bias bagian *hidden layer* (*feed backward propagation*).
7. Tiap *output* dibandingkan dengan target yang diinginkan untuk memperoleh *error* atau disebut dengan MSE(*Mean Squared Error)* dengan rumus:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}\left(\hat{Y}_i - Y_i\right)^2 \quad (1)$$

   Nilai MSE ini adalah sebuah nilai yang diperoleh untuk mengetahui *error* dari metode BPNN yang akan dijadikan patokan sebagai nilai henti.
8. Pola target diterima oleh tiap unit *output* sesuai dengan pola masukan saat pelatihan lalu nilai *error*-nya dihitung dan nilai *weight*-nya diperbaiki.
9. Tiap *weight* yang menghubungkan unit *output* dengan unit *hidden layer* dikali selisi *error* dan dijumlahkan sebagai masukan unit berikutnya.
10. Tiap *weight* dan bias yang ada pada ANN diperbaiki.
11. Pelatihan dihentikan dalam uji kondisi.

Setelah memperoleh hasilnya, pencarian hasil titik dari grafik persamaan diferensial dapat dicari dengan *trial solution* menggunakan rumus umum:

$$\psi t(x) = A + xN(x, p) \quad (2)$$

A = Nilai yang sudah diberikan terlebih dahulu.
x = Nilai *input*
N(x,p) = Nilai ANN, dengan *p* adalah bobot.

III. EULER

Metode Euler disebut juga metode orde pertama karena pencarian dalam persamaannya hanya mengambil sampai orde pertama saja. Metode Euler juga menggunakan bantuan dari deret Taylor [4].

Algoritma metode Euler:

```
function y_Euler (x0, y0, b, h:real) :real;
//menghitung nilai y(b) pada PBD y' = f(x,y); y(x0) = y0;
dengan metode Euler


var
        r, n: integer;
        x, y: real;
        begin
                n←(b0-x0)/h;// langkah
                y← y0;   //nilai awal
                x←  x0;
                for r← 1 to n do
                begin
                y←  y + h*f(x, y);              //hitung solusi
y[x,r]
                x←  x + h                       //hitung titik
berikutnya
                end;      //endfor
                y_Euler←y//y(b)
        end;
```

IV. HEUN

Metode Heun mempunyai ketelitian yang rendah karena memiliki nilai *error* yang besar (sebanding dengan *h*). Nilai *error* dapat dikurangi dengan menggunakan metode Heun yang merupakan perbaikan dari metode Euler [4]. Pada metode Heun, solusi metode Euler dijadikan solusi perkiraan awal (*predictor*). Selanjutnya, solusi perkiraan awal ini diperbaiki dengan metode Heun (*corrector*).

Algoritma metode Heun:

```
function y_Heun (x0, y0, b, h:real) :real;
//menghitung nilai y(b) dengan metode Heun pada PBD y' =
f(x,y); y(x0) = y0
var
        r, n : integer;
        x,y, y_s : real;
begin
        n ← (b0-x0) /h;      //jumlah langkah
        y ← y0;              //nilai awal
        x ←  x0;
        for r:=1 to n do
        begin
                y_s ← y                                  //y
dari langkah r-1
                y ← y + h*f(x,y);
        //y(xr) dengan Euler
                  y ←  y_s +h/2 * ((f(x,y_s) +
f(x+h,y))
//y(xr) dengan Heun
                  x ← x+1;//titik berikutnya
        end;
        y_Heun ←  y;
end;
```

V. RUNGE KUTTA ORDE 4

Metode Runge-Kutta mencapai keakuratan dari suatu pendekatan Taylor tanpa memerlukan turunan-turunan tingkat tinggi. Jadi, keakuratan metode Runge-Kutta adalah metode numerik yang memiliki keakuratan di atas metode numerik seperti Euler, Heun, dan Titik Tengah [4].

Algoritma metode Runge-Kutta Orde 4:

```
function y_RK4 (x0, y0, b, h:real) :real;
```

```
//menghitung y(b) dengan metode Runge-Kutta orde empat pada
PDB y'=f(x,y); y(x0) = y0

var
        r, n : integer;
        x, y, k1, k2, k3, k4 : real;

begin
        n←  (b-x0)/h;       //jumlah langkah
        y←  y0;             //nilai awal
        x← x0;
        for r:=1 to n do
        begin
        k1 ←  h*f(x, y);
        k2 ←  h*f(x + h/2, y +k1/2);
        k3 ←  h*f(x +h/2, y+k2/2);
        k4 ←  h*f(x + h, y +k3);
        y←  y + (k1 + 2*k2 + 2*k3 + k4) / 6     //nilai y(xr)
        x ← x + h;//titik berikutnya
        end;
        y_RK4 ← y;
end;
```

VI. *Euclidean Distance*

Euclidean Distance adalah rumus untuk menghitung jarak antara dua buah titik, dengan rumus umum:

$$d(p,q) = \sqrt{(p_1 - q_1)^2 + \ldots + (p_n - q_n)^2} \qquad (3)$$

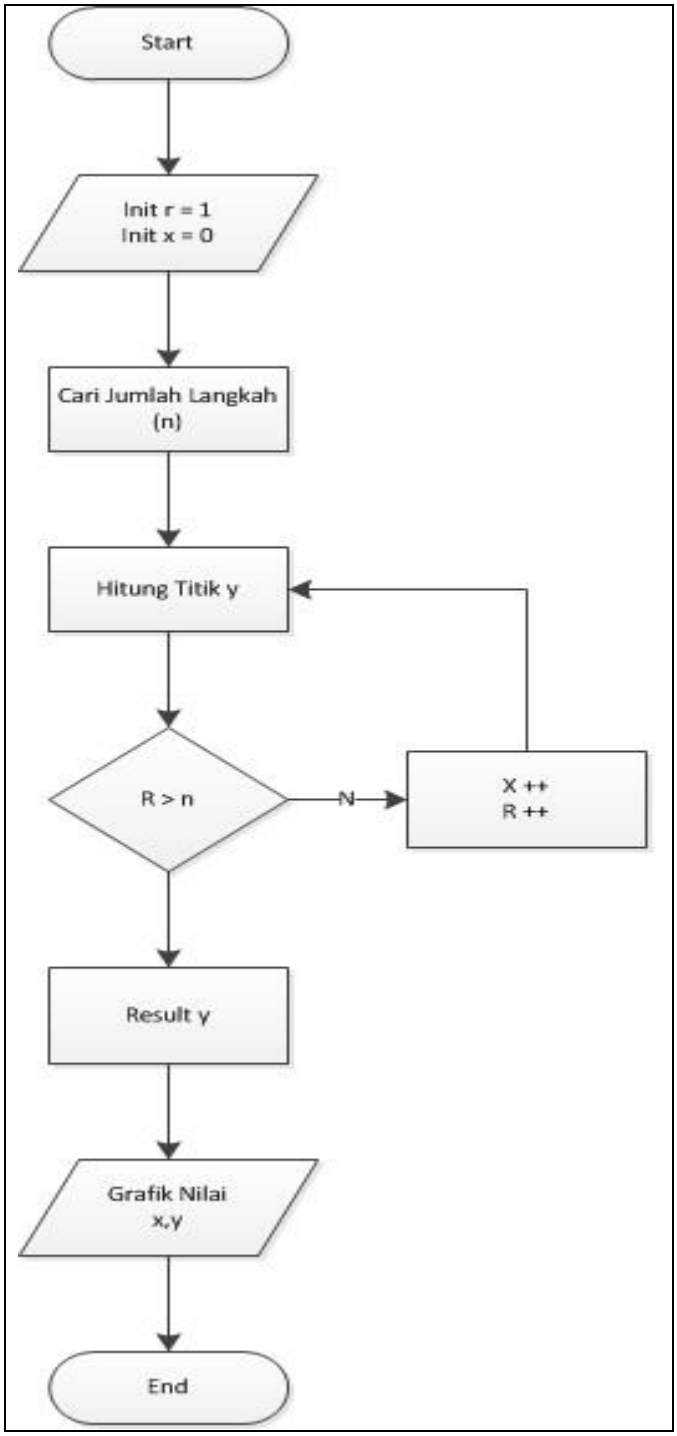
Pada perbandingan ini akan dibandingkan *Euclidean Distance* dari metode numerik, yaitu Metode Euler, Heun, dan Runge Kutta orde 4, serta metode BPNN. Hal tersebut dilakukan dengan cara membandingkan setiap titik dari solusi sejati dengan metode tersebut, sehingga diperoleh 4 nilai. Nilai yang paling kecil berarti memiliki jarak paling dekat, sehingga bisa dikatakan metode tersebut paling bagus.

VII. Penyelesaian PDB dengan Metode Numerik

Dalam menyelesaikan PDB dari metode numerik, uji coba dilakukan dengan membuat perangkat lunak untuk menyelesaikan PDB. Dalam perangkat lunak ini digunakan 3 metode numerik saja, yaitu Euler, Heun, dan Runge-Kutta orde 4. Tiga metode tersebut digunakan karena dari metode numerik yang ada, Euler memiliki tingkat akurasi yang paling rendah, Runge-Kutta memiliki tingkat akurasi tertinggi, dan Heun berada diantara kedua metode tersebut [2].

Pada proses *flowchart* Gambar 1, penyelesaian PDB dengan metode numerik adalah sebagai berikut:

1. Masukan inisialisasi $r = 1$ yang merupakan tetapan pada metode ini dan nilai x yang merupakan *input* untuk mencari jumlah langkah dan mencari y.
2. Hitung jumlah langkah yang dilakukan dengan menggunakan rumus yang terdapat pada algoritma (2,9), (2,10), dan (2,11), yaitu $n = (b\text{-}x_0)/h$.
3. Hitung nilai y sebanyak langkah yang tadi sudah dihitung. Rumus untuk menghitung nilai y terdapat pada algoritma (2,9), (2,10), dan (2,11) dengan rumus yang berbeda-beda. Rumus untuk mencari nilai y terdapat di dalam *for*.
4. Setelah mendapatkan semua nilai y dan x, nilai tersebut ditampung dan ditampilkan dalam bentuk grafik.



Gambar 1 Menyelesaikan PDB dengan metode numerik

VIII. Penyelesaian PDB dengan ANN

Menurut teori, *error* ANN dalam menyelesaikan PDB lebih kecil daripada menggunakan metode numerik, tetapi tidak semua ANN dapat digunakan untuk menyelesaikan PDB. ANN dengan tipe *unsupervised learning* tidak cocok untuk menyelesaikan metode ini karena *unsupervised learning* digunakan untuk mengelompokan jenis-jenis data yang hampir serupa. Contoh dari *unsupervised learning* adalah LVQ, sehingga metode ANN yang digunakan adalah *supervised learning*. Metode *supervised learning* adalah metode yang dapat menentukan *output* sendiri sedekat mungkin dengan data pelatihan.

Pencarian solusi dengan mengunakan ANN ini berbeda dengan pencarian solusi menggunakan metode numerik. Jumlah parameter model yang dibutuhkan jauh lebih sedikit daripada teknik solusi yang lain. Dalam kasus perangkat lunak ini, parameter yang dibutuhkan untuk menyelesaikan PDB hanya satu.

Metode ANN yang digunakan untuk menyelesaikan PDB adalah *Backpropagation* (BP). BP adalah gabungan dari algoritma *feedforward* dan *backforward*, sehingga algoritma hasil dari algoritma BP memiliki hasil yang lebih mendekati *learning rate* dibandingkan dengan *feedforward* maupun *backward.* Untuk data *training*, digunakan data yang berasal dari solusi sejatinya. Semakin banyak data *training*, semakin baik dalam perhitungan PDB. Namun, waktu yang dibutuhkan lebih banyak karena setiap *input* membutuhkan bobot yang harus dihitung. Otomatis waktu yang dibutuhkan untuk mencapai *error* yang baik akan lebih lama.

Untuk lebih jelasnya, cara penyelesaian PDB dengan ANN adalah membuat simulasi ANN dengan perangkat lunak. Uraian proses tersebut dapat dilihat pada Gambar 2.

Uraian proses *flowchart* pada Gambar 2:

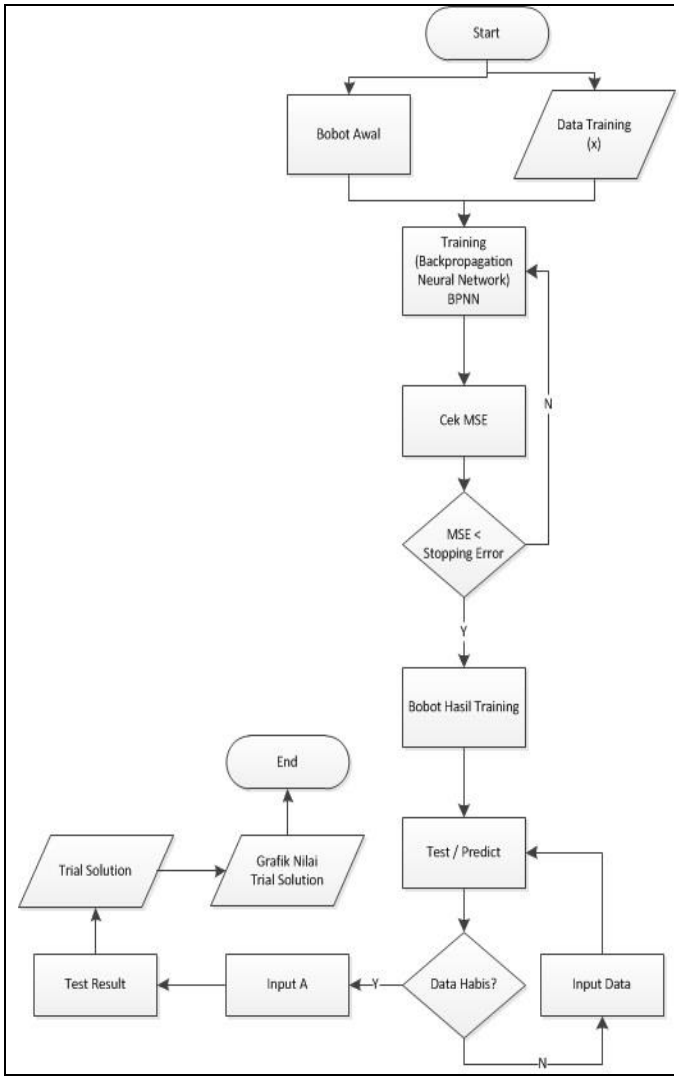
1. Masukan data *training* berupa nilai *x* dan bobot. Nilai *x* sesuai dengan nilai *input* pada Persamaan Diferensial Biasa.
2. Mencari nilai *MSE* (1) dan data *training*.
3. Hasil terbaik dari data *training* ketika *MSE* <*error* yang diinginkan (dalam perangkat lunak ini, *error* maksimum di 0,05)
4. Setelah mendapatkan nilai data *training*, memprediksi nilai yang ingin dihitung dengan memasukan semua data *x*.
5. Mendapatkan hasil tes menggunakan rumus (2).
6. Nilai ditampung dalam *Trial Solution* untuk dibentuk menjadi grafik nilai *trial solution.*

IX. PENGUJIAN

PDB yang diuji adalah persamaan yang memiliki gejala *chaos* dan yang tidak memiliki gejala *chaos.* Untuk yang tidak memiliki unsur *chaos,* contoh persamaan yang digunakan adalah:

$$\frac{dy}{dx}+\frac{1}{5}=e^{-\left(\frac{x}{5}\right)}\cos(x) \qquad (4)$$

Diketahui bahwa $y(0) = 0$, dengan *x* dimulai dari 0 sampai batas *x* yang diinginkan. Persamaan diferensial yang mengandung gejala *chaos* menggunakan persamaan:



Gambar 2 *Flowchart* menyelesaikan PDB dengan ANN

$$\frac{d\omega}{dt}+C\omega+\frac{g}{l}\sin(\omega * t)-F\sin(\omega t)=0 \qquad (5)$$

Di mana:
t (waktu) = dimulai dari 0s sampai batas yang ditentukan
C (konstanta) = 0,1
ω (radian) = 0,67 rad/s
g (gravitasi) = 9,8 m/s^2
l (panjang tali) = 10 m
F (gaya) = 2 N

Dari persamaan diatas, dapat dicari solusi sejatinya yang akan dibandingkan dengan keempat metode dengan persamaan yang sama. Solusi sejati dicari dengan cara memasukan nilai yang berada di sumbu *x*. Untuk persamaan yang tidak mengandung gejala *chaos*, nilai masukannya berupa nilai *x* untuk mencari nilai *y*, sedangkan metode yang lain digunakan dengan algoritma yang berbeda seperti yang sudah dituliskan sebelumnya. Setelah hasil dari solusi sejati dan keempat

metode itu diperoleh, maka dimasukkan ke dalam grafik garis yang terlihat pada Gambar 3 dan Gambar 4.

Dari Gambar 3, dapat dilihat bahwa semua metode memiliki hasil yang hampir sama. Warna merah merupakan solusi analitik dari persamaan, biru Euler, kuning Heun, dan Runge Kutta berwarna hijau dan BPNN berwarna merah muda, sedangkan pada Gambar 4, Runge Kutta berwarna biru, BPNN berwarna hijau, dan solusi sejati berwarna merah.

Perbandingan antara Gambar 3 dan Gambar 4 terlihat jelas. Ketika sebuah persamaan mengandung unsur *chaos*, maka hasil dari perhitungan akan berubah. Dapat dilihat pada Gambar 4, garis dari BPNN hampir mendekati dengan solusi sejatinya. Ini membuktikan bahwa metode ANN BPNN tersebut memiliki hasil yang lebih akurat dibandingkan dengan metode numerik.

Untuk lebih jelasnya, angka dari *Euclidean Distance* (3) diperlihatkan pada Gambar 5 dan Gambar 6.

Dari Gambar 5 dapat dilihat *Euclidean Distance* tiap metode yang telah dihitung berdasarkan Gambar 3:

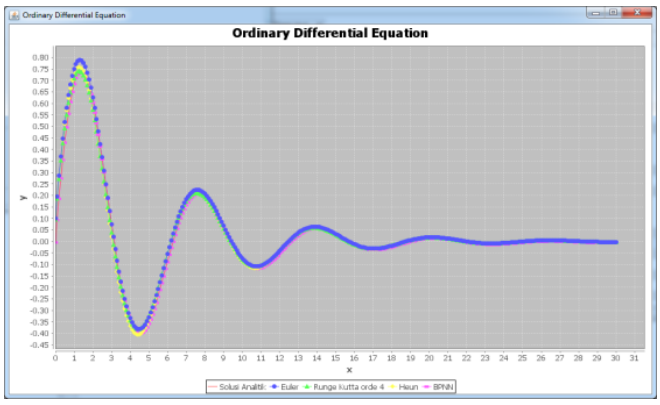
- Metode Euler = 3,63
- Metode Heun = 3,49
- Metode Runge-Kutta orde 4 = 0,35
- Metode BPNN = 0,0028

Dari angka tersebut dapat diurutkan nilai *Euclidean* dari tiap metode yang diurutkan dari yang paling mendekati solusi sejatinya, yaitu metode BPNN, Runge-Kutta orde 4, Heun, dan terakhir Euler.

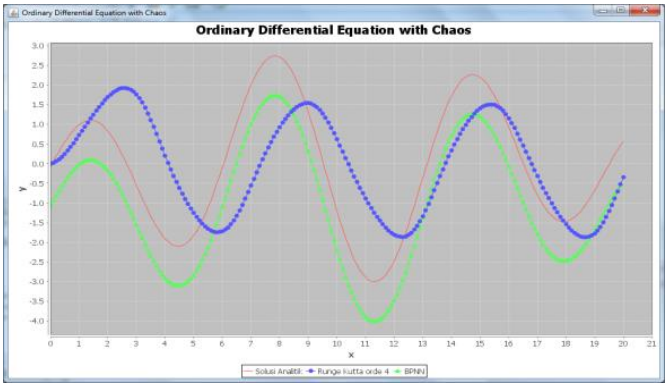
Dari Gambar 6 dapat dilihat *Euclidean Distance* dari 2 metode yang paling mendekati dengan solusi sejatinya, yaitu:

- Metode Runge-Kutta orde 4 = 24,85
- Metode BPNN = 3,73

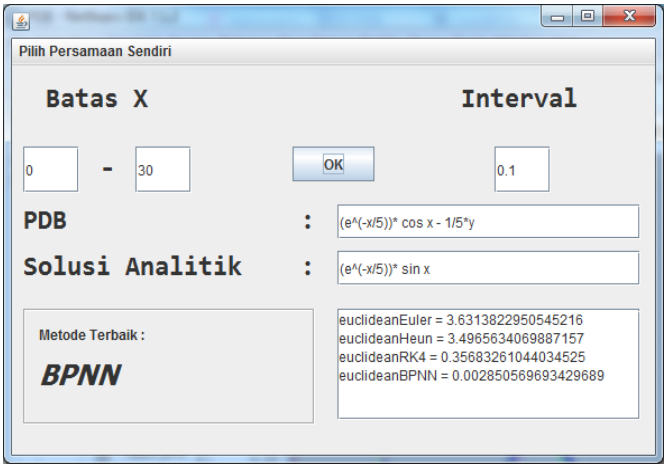
Dapat disimpulkan bahwa metode yang paling mendekati dengan solusi sejatinya adalah metode BPNN.



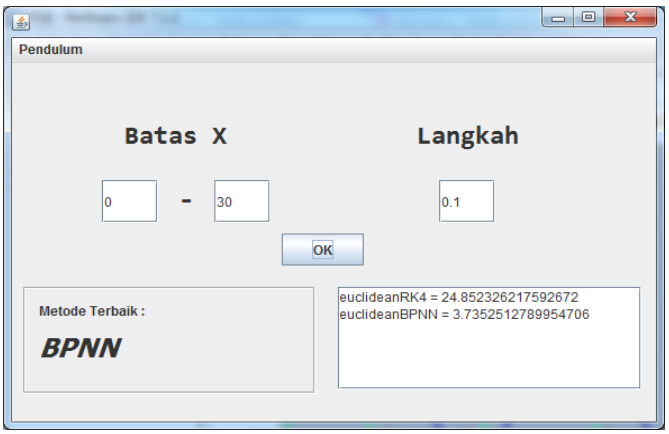
Gambar 3 PDB tanpa gejala *chaos*



Gambar 4 PDB dengan gejala *chaos*



Gambar 5 Hasil Euclidean Gambar 3



Gambar 6 Nilai *Euclidean* Gambar 4

X. KESIMPULAN

Dari pengujian diatas dapat dilihat bahwa Metode *Backpropagation Neural Network* (BPNN) memiliki tingkat akurasi yang lebih tinggi dibandingkan dengan metode lainnya. BPNN bisa memiliki akurasi yang lebih baik daripada metode lain karena BPNN memiliki data *training* dan jumlah data *training* berpengaruh pada prediksi nilai dari BPNN. Selain itu, data *training* yang didapat adalah data *training* dari solusi sejatinya, sehingga nilai dari BPNN tidak akan jauh berbeda dari nilai solusi sejati.

Selain itu, *error* dari ANN dapat diperkecil dengan cara memperbanyak data *training*. Dengan memiliki data *training* yang lebih banyak, maka pencarian hasil akan lebih akurat, sehingga *error* yang didapatkan akan lebih kecil. Namun, memperbanyak data *training* akan memperlambat daya kerja komputer dalam mendapatkan hasilnya, sehingga dibutuhkan spesifikasi komputer yang lebih canggih dan tidak akan memakan waktu yang lebih lama.

Terakhir, dari hasil pengujian diatas, tidak semua metode berhasil untuk menyelesaikan persamaan diferensial dengan tingkat akurasi yang baik. Contohnya, ketika menghitung PDB yang mengandung *chaos*, metode Runge-Kutta tidak dapat menyesuaikan grafik dengan solusi sejatinya, sehingga tidak semua metode menghasilkan *output* yang memuaskan. Dari persamaan diferensial yang tidak mengandung *chaos*, metode Euler memiliki solusi yang jauh dari solusi sejatinya. Ini menunjukkan bahwa hasil dari metode Euler kurang memuaskan.

DAFTAR REFERENSI

[1] *Lagaris, Isaac Elias, Aristidis Likas, Dimitrios I. Fotiadis. "Artificial Neural Network for Solving Ordinary and Partial Differential Equation," 1998.*

[2] R. Munir. *Metode Numerik Revisi ke Tiga*. Informatika. 2010.

[3] SBU Website Artificial Neural Network. [Online]. Available: http://www.cs.sunysb.edu [2012].

[4] S. C. Chapra, *Numerical Methods for Engineers With Personal Computer Application – 2nd edition*, Mcgraw Hill College, 1998.

[5] USU Insitutional Repository Analisis Simulasi Gelaja Chaos Pada Gerak Pendulum Nonlinier. [Online]. Available: http://repository.usu.ac.id [2013].

Jayme Yeremia Wijaya, lahir pada tahun 1990 di Bandung, menerima gelar Sarjana Teknik dari Institut Teknologi Harapan Bangsa. Saat makalah ini ditulis, sedang melanjutkan studi bidang manajemen bisnis di Universitas Katolik Parahyangan. Saat ini aktif sebagai IT System Analyst di suatu perusahaan swasta di Bandung.