BAB I
PENDAHULUAN

1.1 LATAR BELAKANG

Peramalan adalah aktivitas menghitung atau memprediksi beberapa kejadian yang akan datang. Dalam penerapan, model *time series* dapat digunakan dengan mudah untuk meramal karena pendugaan masa depan dilakukan berdasarkan data masa lalu (Makridakis, Wheelwright, McGee.1994). Peramalan dapat membantu untuk mengurangi ketidakpastian dalam melakukan perencanaan. Oleh karena itu peramalan memegang peranan penting dalam perencanaan dan pengambilan keputusan diberbagai bidang.

Teknik peramalan harga valuta asing yang saat ini sering digunakan adalah *Artificial Neural Network* (ANN). ANN atau disebut juga jaringan syaraf tiruan (JST) adalah sistem pengelola informasi yang memiliki karakter seperti representasi buatan dari otak manusia yang selalu mencoba untuk mensimulasi proses pembelajaran pada otak manusia tersebut. Jaringan syaraf tiruan merupakan jaringan dari banyak unit pemroses kecil (*neuron*) yang masing-masing melakukan proses sederhana, ketika digabungkan akan menghasilkan perilaku yang kompleks yang dapat digunakan sebagai alat untuk memodelkan hubungan yang kompleks antara masukan (*input*) dan keluaran (*output*) pada sebuah sistem untuk menemukan pola-pola pada data. (Kusumadewi, 2003).

Dalam Implementasi Jaringan Syaraf Tiruan, terdapat beberapa kerumitan yang sering kali dijumpai antara lain cara input data, pemilihan inisialisasi, jumlah neuron pada lapisan hidden dan lain-lain (Buwana, 2006). Algoritma ini terkesan lama karena untuk menemukan model yang layak harus menentukan parameter jaringan dengan cara coba coba (*trial and error*).

Algoritma genetika adalah algoritma pencarian heuristik yang didasarkan atas mekanisme seleksi dan genetika alami. Konsep dasar yang mengilhami timbulnya algoritma genetika adalah teori evolusi, yang dikemukakan oleh Charles Darwin. Algoritma genetika dimulai dengan membentuk sejumlah alternatif solusi yang disebut sebagai populasi. Setiap solusi pada algoritma genetika

diwakili oleh satu individu atau satu kromosom (Kusumadewi, 2003).

Beberapa penelitian sebelumnya antara lain, (Cahyadi, 2010) tentang perbandingan algoritma pelatihan pada JST, memberikan saran yaitu untuk penelitian selanjutnya mencari suatu formula yang baik dalam menentukan parameter-parameter JST *backpropagation* akan sangat membantu agar jaringan mampu bekerja lebih cepat dan optimal. Darmawansyah (2009) tentang perbandingan jaringan syaraf tiruan algoritma *backpropagation* dan algoritma genetika untuk peramalan pada beberapa tipe data *time series* yang berbeda (trend, musiman, acak), menunjukan jika JST algoritma *backpropagation* memberikan hasil yang lebih baik dibandingkan algoritma genetika. Putri (2011) mampu menerapkan algoritma genetika sebagai algoritma pelatihan jaringan syaraf tiruan dengan hasil ramalan 98% akurat dibandingkan dengan harga saham aktual.

Mengacu pada penelitian di atas, peneliti tertarik untuk melakukan studi kasus berkenaan dengan nilai tukar mata uang asing. Pada penelitian ini akan diteliti mengenai inisialisasi pada jaringan syaraf tiruan. Algoritma genetika disini digunakan untuk mendapatkan bobot dan bias yang akan digunakan sebagai inisialisasi JST *backpropagation*, sehingga judul yang diambil adalah "Penerapan Inisialisasi Bobot Dan Bias Jaringan Syaraf Tiruan Dengan Algoritma Genetika Pada Data Nilai Tukar Mata Uang Asing". Metode aplikasi JST *backpropagation* dengan inisialisasi algoritma genetika dan JST *backpropagation* tanpa inisialisasi algoritma genetika diuji coba pada dua nilai tukar mata uang dunia. Data pertama adalah nilai tukar mata uang Euro terhadap US Dollar, dan data kedua adalah nilai tukar mata uang US Dollar terhadap Rupiah.

1.2 PERMASALAHAN

Permasalahan yang akan dikaji dalam penelitian ini adalah sebagai berikut :

1. Bagaimana cara menerapkan algoritma genetika untuk inisialisasi bobot dan bias pada JST *backpropagation* ?
2. Apakah model yang dihasilkan JST *backpropagation* layak dan bisa digunakan dalam meramalkan data *time series* yang diuji ?

3. Berapa tingkat kesalahan ramalan yang dihasilkan oleh jaringan syaraf tiruan dengan inisialisasi algoritma genetika?
4. Apakah penggunaan algoritma genetika untuk inisialisasi bobot mampu mempercepat proses *training* jaringan ?

1.3 BATASAN MASALAH

Penelitian pada skripsi ini dibatasi pada :

1. Arsitektur dari jaringan syaraf tiruan algoritma *backpropagation* yang digunakan adalah *Multi Layer Perceptron* dengan satu lapis masukan (input), satu lapisan tersembunyi (*hidden*), dan satu lapis keluaran (output).
2. Algoritma pelatihan yang digunakan adalah *gradient descent*.
3. Algoritma Genetika hanya digunakan untuk inisialisasi bobot JST *backpropagation*.

1.4 TUJUAN PENELITIAN

Tujuan penelitian pada skripsi ini adalah :

1. Menguji kelayakan model yang dihasilkan metode jaringan syaraf tiruan dan menerapkan algoritma genetika sebagai inisialisasi bobot dan bias jaringan syaraf tiruan.
2. Melihat tingkat kesalahan dari ramalan yang dihasilkan oleh jaringan syaraf tiruan dengan menerapkan algoritma genetik.
3. Membandingkan waktu untuk menyelesaikan proses *training* dari JST *backpropagation* dengan inisialisasi bobot dan bias algoritma genetika.

1.5 MANFAAT PENELITIAN

Manfaat dari tugas akhir ini diharapkan dapat dijadikan masukan dalam mengembangkan teori metode jaringan syaraf tiruan algoritma *backpropogation* dan algoritma genetika untuk peramalan data *time series*.


UNIVERSITAS BRAWIJAYA

BAB II
TINJAUAN PUSTAKA

2.1 Definisi Peramalan

Peramalan adalah menduga atau memperkirakan suatu keadaan dimasa yang akan datang berdasarkan keadaan masa lalu dan sekarang yang diperlukan untuk menetapkan kapan suatu peristiwa akan terjadi, sehingga tindakan yang tepat dapat dilakukan (Makridakis *et.al.*, 1999).

Fungsi peramalan adalah sebagai dasar bagi perencanaan kapasitas, anggaran, perencanaan penjualan, perencanaan produksi, dan inventori, perencanaan sumber daya, serta perencanaan pembelian bahan baku. Ada dua hal pokok yang harus diperhatikan dalam proses peramalan yang akurat dan bermanfaat:

1. Pengumpulan data yang relevan yang berupa informasi yang dapat menghasilkan peramalan yang akurat.
2. Pemilihan teknik peramalan yang tepat yang akan memanfaatkan informasi data yang diperoleh semaksimal mungkin.

Prosedur peramalan dapat pula dikelompokkan sesuai dengan sifatnya yaitu kualitatif dan kuantitatif. Metode kualitatif adalah metode subjektif bedarsarkan atas pendapat pribadi yang digunakan *forecaster* (peramal) dan tidak memerlukan manipulasi data. Sedangkan metode kuantitatif tidak memerlukan *input* pendapat pribadi, dan lebih banyak memerlukan manipulasi data. Menurut Hanke (1992) bahwa ilmu dan pengalaman harus digunakan secara bersama-sama, hanya dengan cara inilah peramalan yang cerdas dapat terjadi.

2.2 Kecerdasan Buatan (*Artificial Intellegence*)

Selama ini banyak peramalan dilakukan secara intuitif atau dengan menggunakan metode-metode statistik. Banyak metode untuk melakukan peramalan tersebut, misalnya metode *smoothing*, Box-Jenkins, ekonometri, regresi, dan sebagainya. Pemilihan metode-metode tersebut yang digunakan pada perhitungan untuk meramalkan suatu hal tertentu tergantung pada berbagai aspek yang mempengaruhi yaitu aspek waktu, pola data, tipe arsitektur sistem yang diamati, tingkat keakuratan ramalan yang diinginkan dan

sebagainya. Karena itulah akan muncul suatu masalah apabila pengamatan atau pengujian dilakukan pada suatu sistem dinamis yang memiliki sistem pola data dengan formulasi yang selalu berubah-ubah pada suatu kurun waktu tertentu. Disamping itu untuk menerapakan metode tersebut data harus memenuhi beberapa asumsi-asumsi yang digunakan.

Dengan menggunakan teknologi kecerdasan buatan (*artificial intelligence)* yaitu teknologi jaringan syaraf tiruan, maka identifikasi pola data dari nilai tukar mata uang asing dapat dilakukan. Kelebihan utama jaringan syaraf tiruan adalah kemampuan komputasi dengan cara menghafalkan pola-pola yang diajarkan (*training*). Berdasarkan kemampuan belajar yang dimilikinya, maka jaringan syaraf tiruan dapat dilatih untuk mempelajari dan menganalisa pola data masa lalu dan berusaha mencari suatu formula atau fungsi yang akan menghubungkan pola data masa lalu dengan keluaran (*output*) yang diinginkan pada saat ini (Muharram J. 2005).

2.3 Algoritma Genetika

Algoritma genetika atau GA (*Genetic Algorithm*) merupakan suatu konsep komputasi yang pertama kali diutarakan oleh John Holland dari Universitas Michigan pada tahun 1975. Algoritma genetika adalah algoritma komputasi yang diinspirasi teori evolusi yang kemudian diadopsi menjadi algoritma komputasi untuk mencari solusi suatu permasalahan dengan cara yang lebih alamiah. Algoritma genetika mencerminkan kemampuan individu untuk melakukan perkawinan dan menghasilkan keturunan yang memiliki karakteristik yang hampir sama dengan orang tuanya. Sedangkan prinsip seleksi alam menyatakan bahwa setiap makhluk hidup dapat mempertahankan dirinya jika mampu beradaptasi dengan lingkungannya. Dengan demikian, diharapkan keturunan yang dihasilkan memiliki kombinasi karakteristik yang terbaik dari orang tuanya, dan dapat menopang generasi-generasi selanjutnya.

Sebuah solusi yang dibuat dalam algoritma genetika disebut sebagai kromosom, sedangkan kumpulan kromosom-kromosom tersebut disebut sebagai populasi. Sebuah kromosom dibentuk dari komponen-komponen penyusun yang disebut sebagai gen. Kromosom-kromosom tersebut akan berevolusi secara berkelanjutan

yang disebut dengan generasi. Dalam tiap generasi kromosom tersebut dievaluasi tingkat keberhasilan bertahan hidupnya, solusinya permasalahan ini menggunakan ukuran yang disebut dengan *fitness*.

Untuk memilih kromosom yang tetap dipertahankan untuk generasi selanjutnya dilakukan proses yang disebut dengan seleksi. Proses seleksi kromosom yaitu kromosom yang mempunyai nilai *fitness* tinggi akan memiliki peluang lebih besar untuk terpilih pada generasi selanjutnya. Kromosom baru yang disebut dengan *offspring*, dibentuk dengan cara melakukan perkawinan silang antar kromosom dalam satu generasi (*crossover*). Jumlah kromosom dalam populasi yang mengalami *crossover* ditentukan oleh paramater yang disebut dengan *crossover rate*. Mekanisme perubahan susunan unsur penyusun mahkluk hidup akibat adanya faktor alam (*mutasi*) direpresentasikan sebagai proses berubahnya satu atau lebih nilai gen dalam kromosom. Jumlah gen dalam populasi yang mengalami mutasi ditentukan oleh parameter yang dinamakan *mutation rate*. Setelah beberapa generasi akan dihasilkan kromosom yang nilai gennya merupakan solusi terbaik (Hermawanto, 2003).

2.3.1 Arsitektur Algoritma Genetika Untuk Inisialisasi Bobot

Berdasarkan adanya kesamaan antara struktur jaringan syaraf dengan pendekatan umum metode peramalan Autoregresi, maka arsitektur yang digunakan pada algoritma genetika untuk inisisalisasi bobot adalah sebagai berikut :

$$x_{n+1}=a_0+a_1x_1+a_2x_2+\ldots+a_nx_n \qquad (2.1)$$

Dimana :

x_{n+1}	adalah data satu periode ke depan
x_i	adalah data pada periode ke-i, i = 1,2,3,...,n
a_i	adalah bobot pengaruh dari periode ke-i
a_0	adalah galat acak (Halim & Wibisono, 2000)

2.3.2 Minimisasi *Error* Menggunakan Algoritma Genetika

Untuk menemukan solusi yang paling minimum dari suatu fungsi menggunakan algoritma genetika, diperlukan langkah yang harus difahami terlebih dahulu. Berikut adalah langkah langkah minimisasi tingkat kesalahan menggunakan algoritma genetika :

1. Representasi kromosom
2. Fungsi evaluasi
3. Fungsi Seleksi
4. Operator Genetika
   a) *Cross over*
   b) Mutasi

2.3.2.1 Representasi Kromosom

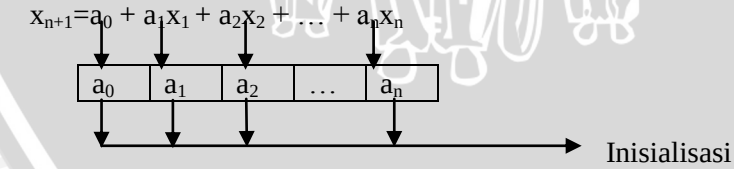
Pada algoritma genetika representasi kromosom diperlukan untuk mewakili setiap individu yang berada dalam suatu solusi. Skema representasi menentukan bagaimana permasalahan disusun dalam algoritma genetika dan operator genetika yang digunakan. Setiap individu atau kromosom dibuat dari serangkaian gen dengan urutan tertentu yang dapat terdiri dari *binary digits* (0 dan 1), *floating point, integer*, matriks, simbol, dan lain sebagainya.

Untuk persoalan yang memiliki kromosom bilangan riil, representasi ini cocok digunakan. Metode ini akan mengurangi beban komputasi tetapi akan menimbulkan masalah ketergantungan solusi pada persoalan dan munculnya solusi riil yang illegal. Hal ini dapat di atasi dengan memilih strategi *crossover* dan mutasi yang sesuai.

2.54	1.95	3.56	1.33	12.6	0.3	8.4

Gambar 2.1 Representasi kromosom secara riil

Berdasarkan hasil eksperimen Michalewiez dalam membandingkan algoritma genetika dengan susunan berupa angka riil dan biner didapatkan bahwa dengan angka riil algoritma genetika dapat menjadi lebih efisien, karena dapat memberikan hasil solusi yang lebih presisi. Sehingga setiap kromosom dari gen sebanyak n+1 akan direpresentasikan sebagai inisial bobot dari persamaan 2.2



Gambar 2.2. Representasi Kromosom

2.3.2.2 Fungsi Evaluasi

Fungsi evaluasi adalah suatu fungsi yang digunakan oleh algoritma genetika untuk menentukan nilai kecocokan (*fitness*) suatu kromosom. Fungsi evaluasi ini dijalankan ketika algoritma genetika sedang melakukan pencarian solusi terhadap suatu masalah yang dihadapi. Nilai *fitness* digunakan secara khusus pada tahap seleksi untuk menentukan besarnya probabilitas kromosom yang akan dipilih sebagai orang tua (*parent)* untuk menghasilkan keturunan *(offspring)* selanjutnya. Persamaan fungsi evaluasi dapat dituliskan sebagai berikut :

$$fitness = | k - b | \qquad (2.2)$$

Dimana :

Fitness adalah nilai minimum yang ingin dicapai
k adalah nilai prediksi periode x_{n+1}
b adalah nilai target periode x_{n+1}

evaluasi dari nilai *fitness* akan berhenti jika :

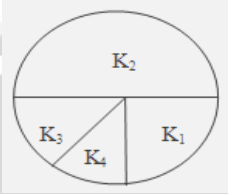
1. Diperoleh suatu individu memiliki nilai *fitness* yang sama dengan nilai batas (*threshold)* yang didefinisikan sebelumnya.
2. Algoritma genetika mencapai batas maksimum generasi.
3. Nilai *fitness* yang dihasilkan tidak berubah selama generasi tertentu *(stall generation)*.

2.3.2.3 Fungsi Seleksi

Seleksi individu untuk menghasilkan generasi berikutnya memainkan peranan yang sangat penting dalam algoritma genetika. Seleksi secara probabilistik dilakukan berdasarkan nilai *fitness* individu, semakin baik individu tersebut maka semakin besar kemungkinannya untuk terpilih. Setiap individu dalam populasi dapat dipilih lebih dari sekali.

Suatu metode seleksi yang umum digunakan adalah *roulette-wheel*. Metode ini seperti permainan *roulette-wheel* dimana masing-masing kromosom menempati lingkaran pada roda roulette secara proposional sesuai dengan nilai *fitness*nya.

Kromosom	Nilai *Fitness*
K_1	1
K_2	2
K_3	0,5
K_4	0,5
Junlah	4

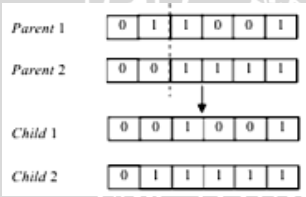


Gambar 2.3. Contoh Penggunaan *Roulette-Wheel*

2.3.2.4 Crossover

Perkawinan silang (*crossover*) mengkombinasikan dua kromosom untuk menghasilkan keturunan (*offspring*), pemilihan kromosom yang berlaku sebagai orang tua (*parent*) dipilih secara acak dari kromosom-kromosom yang berada pada populasi. Semakin banyak *crossover* yang dilakukan akan semakin besar nilai *crossover rate* (perbandingan keturunan yang dihasilkan secara *crossover* pada tiap generasi dengan jumlah populasi).

Metode untuk melakukan *crossover* pada percobaan ini adalah *one point crossover*. One-point Crossover adalah sebuah cara penentuan titik segmen pertukaran string bit kromosom dengan memilih satu titik potong pertukaran. Titik potong dipilih secara acak, kemudian bagian pertama dari parent 1 digabungkan dengan bagian kedua parent 2. Skema One-point crossover dapat dilihat pada Gambar 2.4



Gambar 2.4. *Simple crossover* dengan menggunakan representasi biner (sciencedirect, 2012)

Seperti terlihat pada Gambar 2.4 posisi titik potong yang terpilih secara acak adalah 2, maka bagian kiri dari titik potong kedua induk *(parent)* kromosom tersebut di tukarkan untuk menghasilkan kromososm anak *(child)*.

2.3.2.5 Mutasi

Mutasi (*mutation)* adalah aktifitas dimana mekanismenya menghasilkan perubahan acak secara spontan pada kromosom. Proses mutasi dalam algoritma genetika ditunjukkan pada Gambar 2.5.

Gambar 2.5. Mutasi dalam Algoritma Genetika

Proses mutasi dapat dilakukan dengan mengubah satu atau beberapa gen dalam kromosom. Peranan mutasi dalam algoritma genetika adalah sebagai berikut:

1) Menggantikan gen-gen yang hilang dari populasi dalam proses seleksi agar dapat dicoba pada konteks yang baru.
2) Menghasilkan gen-gen yang tidak terdapat pada populasi awal.

Mutation rate adalah perbandingan antara banyaknya keturunan yang dihasilkan melalui mutasi pada tiap generasi terhadap jumlah populasi yang ada. Jika nilai *mutation rate* terlalu rendah maka gen yang seharusnya berguna akan tidak terpilih, tetapi jika *mutation rate* ini terlalu besar maka akan muncul gen-gen baru yang menyebabkan keturunan mulai kehilangan kemiripan dengan orang tuanya.

(Suyanto, 2005)

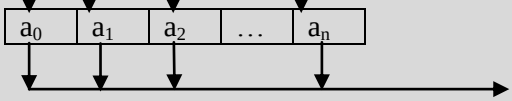
2.3.3. Penentuan Bobot Menggunakan Algoritma Genetika

Penggunaan algoritma genetika pada penelitian ini adalah untuk mencari fungsi *fitness* yang paling minimum. Berdasarkan persamaan (2.3), fungsi yang ingin diminimisasi adalah :

$$\min f(a) = | k - b | \qquad (2.3)$$

k adalah nilai prediksi dari x_{n+1}, dimana nilai k didapatkan dari persamaan (2.4) yaitu :

$$k = a_0 + a_1x_1 + a_2x_2 + \ldots + a_nx_n \qquad (2.4)$$



Fitness min f(a) akan tercapai disaat salah satu dari 3 kriteria fungsi evaluasi terpenuhi. Untuk mencapai kriteria *fitness* terbaik, akan dilakukan percobaan untuk menentukan parameter terbaik. Mengacu pada penelitian (Mufti, 2005) percobaan yang akan dilakukan antara lain menentukan parameter *crossover rate*, *mutation rate*, jumlah populasi, dan maksimum generasi terbaik. Percobaan ke. Secara garis besar proses dari algoritma genetika adalah sebagai berikut :

1. Inisialisasi atau penentuan populasi awal
   Inisialisasi dilakukan dengan cara memberikan nilai awal gen-gen dengan nilai acak sesuai batasan yang telah ditentukan.
2. Proses seleksi kromosom
   Seleksi secara probabilistik dilakukan berdasarkan nilai *fitness* individu, semakin baik individu tersebut maka semakin besar kemungkinannya untuk terpilih. Metode seleksi yang umum digunakan adalah *roulette-wheel*, dengan cara menghitung *cumulative* probabilitasnya maka proses seleksi menggunakan *roulete-wheel* dapat dilakukan. Prosesnya adalah dengan membangkitkan bilangan acak R dalam range 0 - 1. Jika R[k] < C[1] maka pilih kromosom 1 sebagai induk, selain itu pilih kromosom ke-k sebagai induk dengan syarat C[k-1] < R < C[k]. Kita putar *roulete wheel* sebanyak jumlah populasi, misalkan 10 kali (bangkitkan bilangan acak R) dan pada tiap putaran, kita pilih satu kromosom untuk populasi baru
3. Proses *crossover*
   Setelah proses seleksi maka proses selanjutnya adalah proses *crossover*. Metode yang digunakan salah satunya adalah *one-cut point*, yaitu memilih secara acak satu posisi dalam kromosom induk kemudian saling menukar gen. Kromosom yang dijadikan induk dipilih secara acak dan jumlah kromosom yang mengalami *crossover* dipengaruhi oleh parameter *crossover_rate* (ρc). Ini dilakukan dengan cara membangkitkan bilangan acak dengan batasan 1 sampai (panjang kromosom-1). Misalkan didapatkan posisi *crossover* adalah 1 maka kromosom induk akan dipotong mulai gen ke 1 kemudian potongan gen tersebut saling ditukarkan antar induk.
4. Proses Mutasi
   Proses mutasi dilakukan dengan cara mengganti satu gen yang terpilih secara acak dengan suatu nilai baru yang didapat secara

acak. Pertama dihitung dahulu panjang total gen yang ada dalam satu populasi.

Total_gen = (jumlah gen dalam kromosom) * jumlah populasi

Untuk memilih posisi gen yang mengalami mutasi dilakukan dengan cara membangkitkan bilangan integer acak antara 1 sampai total_gen. Jika bilangan acak yang kita bangkitkan lebih kecil daripada variabel *mutation_rate* (ρm) maka pilih posisi tersebut sebagai sub-kromosom yang mengalami mutasi. Misal ρm kita tentukan 10% , maka diharapkan ada 10% dari total_gen yang mengalami mutasi. Gen yang mengalami mutasi pada diganti dengan bilangan acak. Setelah proses mutasi maka kita telah menyelesaikan satu iterasi dalam algoritma genetika atau disebut dengan satu generasi.

5. Bentuk generasi baru, proses ini akan berulang sampai sejumlah generasi yang telah ditetapkan sebelumnya.

2.4 Konsep Dasar Jaringan Syaraf Tiruan (JST)

2.4.1 Sejarah Jaringan Syaraf Tiruan

Jaringan syaraf tiruan telah dikembangkan sejak tahun 1940. Pada tahun 1943 McCulloch dan W.H.Pitts memperkenalkan pearsitekturan matematis neuron. Tahun 1949, Hebb mencoba mengkaji proses belajar yang dilakukan oleh neuron. Teori ini dikenal sebagai Hebbian Law. Tahun 1958, Rosenblatt memperkenalkan konsep perseptron suatu jaringan yang terdiri dari beberapa lapisan yang saling berhubungan melalui umpan maju (*feed foward*). Konsep ini dimaksudkan untuk memberikan ilustrasi tentang dasar-dasar intelejensia secara umum. Hasil kerja Rosenblatt pada tahun 1962 yang sangat penting adalah *perceptron convergence theorem*, membuktikan bahwa bila setiap perseptron dapat memilah-milah dua buah pola yang berbeda maka siklus pelatihannya dapat dilakukan dalam jumlah yang terbatas.

Mengacu kronologis Penelitian Widrow dan Hoff, tahun 1960 peneliti menemukan ADALINE (*Adaptive Linear Neuruon*). Alat ini dapat beradaptasi dan beroperasi secara linier. Penemuan ini telah memperlebar aplikasi jaringan syaraf tiruan tidak hanya untuk pemilihan pola, tetapi juga untuk pengiriman sinyal khususnya dalam

bidang *adaptive filtering*. Tahun 1969, Minsky dan Papert melontarkan suatu kritikan tentang kelemahan perseptronnya Rosenblatt di dalam pemilihan pola yang tidak linier. Sejak saat itu penelitian di bidang jaringan syaraf tiruan telah mengalami masa vakum untuk kurang lebih satu dasawarsa.

Tahun 1982, Hopfield telah memperluas aplikasi JST untuk memecahkan masalah-masalah optimasi. Hopfield telah berhasil memperhitungkan fungsi energi ke dalam jaringan syaraf yaitu agar jaringan memiliki kemampuan untuk mengingat atau memperhitungkan suatu obyek dengan obyek yang pernah dikenal atau diingat sebelumnya (*associative memory*). Konfigurasi jaringan yang demikian dikenal sebagai *recurrent network*. Salah satu aplikasinya adalah *Travelling Salesman Problem* (TSP).

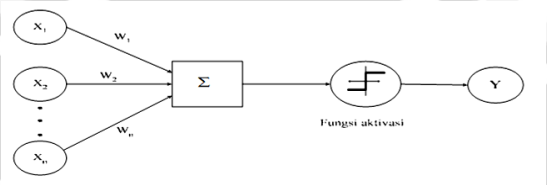
Pada tahun 1986 Rumelhart, Hinton dan William menciptakan suatu algoritma belajar yang dikenal sebagai propagasi balik (*backpropagation*). Bila algoritma ini diterapkan pada perseptron yang memiliki lapisan banyak (*multi layer perceptron*), maka dapat dibuktikan bahwa pemilahan pola-pola yang tidak linier dapat diselesaikan sehingga dapat mengatasi kritikan yang dilontarkan oleh Minsky dan Papert. (Basuki, 2003)

2.4.2 Definisi Jaringan Syaraf Tiruan

Jaringan syaraf tiruan merupakan sistem pengelola informasi yang memiliki karakter seperti jaringan biologis, yaitu representasi buatan dari otak manusia yang selalu mencoba untuk mensimulasi proses pembelajaran pada otak manusia tersebut. Istilah buatan digunakan karena jaringan syaraf ini diimplementasikan dengan menggunakan program komputer yang mampu menyelesaikan sejumlah proses perhitungan selama proses pembelajaran (Kusumadewi, 2003). Jaringan syaraf tiruan memiliki 3 karakteristik utama, yaitu:

1. Arsitektur Jaringan
   Merupakan pola keterhubungan antara neuron. Hubungan neuron-neuron inilah yang membentuk suatu jaringan.
2. Algoritma Pembelajaran
   Merupakan metode yang digunakan untuk menentukan bobot dari hubungan neuron.

3. Fungsi Aktivasi
   Merupakan fungsi untuk menentukan nilai *output* berdasarkan nilai *input* pada neuron.



Gambar 2.5 Fungsi aktivasi pada JST (Wing, 2012)

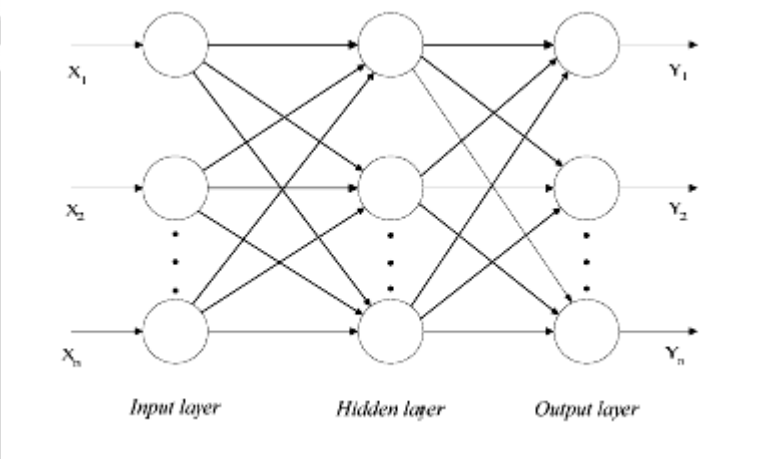
2.4.3 Arsitektur Umum Jaringan Syaraf Tiruan

Jaringan syaraf tiruan memiliki beberapa tipe namun hampir semuanya memiliki komponen yang sama. Komponen-komponen utama dari jaringan syaraf tiruan antara lain sebagai berikut (Puspitaningrum, 2006) :

1. *Neuron* atau *node* atau unit
   Setiap neuron menerima *input*, melakukan pengolahan *input* dan mengirimkan hasilnya berupa sebuah *output*.
2. *Input* atau masukan
   Sinyal input yang diterima akan diteruskan ke lapisan selanjutnya.
3. Bobot
   Merupakan nilai matematis dari koneksi, yang mentransfer data dari satu lapisan ke lapisan lainnya. Bobot digunakan untuk mengatur jaringan sehingga dapat melakukan pembelajaran (*training*) untuk menghasilkan *output* yang diinginkan.
4. Fungsi Aktivasi
   Fungsi aktivasi adalah fungsi yang menentukan hasil *output* sebuah neuron dari hasil penjumlahan bobot.
5. Output atau keluaran
   Output atau keluaran merupakan solusi atau hasil pemahaman jaringan terhadap data input..

2.4.4 Arsitektur Jaringan Syaraf Tiruan

Jaringan syaraf tiruan terdiri atas lapisan *input* dan lapisan *output*. Tetapi ada juga yang mempunyai lapisan tersembunyi (*hidden layer*) di antara lapisan *input* dan *output*. *Node* yang ada pada lapisan *input* disebut dengan *node input*. Pada *node input* tidak memproses suatu informasi tetapi hanya menyalurkan ke *node* didepannya. Sedangkan *node* yang ada pada *hidden layer* akan menghasilkan suatu keluaran (*output)*.



Gambar 2.6 JST dengan 3 lapisan (Javaneural, 2009)

Pada Gambar 2.2 jaringan syaraf tiruan terdiri dari tiga lapisan yaitu :

a. *Input layer* (X_1, X_2, X_n)
   Nodes yang terdapat dalam lapisan ini disebut sebagai *node input* yang dimasukkan ke dalam bentuk sinyal yang dapat dimengerti sistem dan diteruskan ke dalam *hidden layer* untuk diproses.
b. *Hidden layer*
   Nodes yang berada pada lapisan ini disebut sebagai *hidden node*, dalam lapisan ini terjadi proses komputasi.
c. *Output Layer* (Y_1, Y_2, Y_n)
   Node ini merupakan *node* yang berada di dalam lapisan *output*, sehingga dapat ditafsirkan sebagai hasil keluaran (*output*) dari kasus yang dikehendaki.

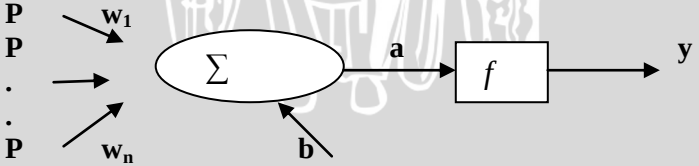
2.5 Jaringan Syaraf Tiruan *Backpropagation*

Jaringan saraf tiruan dengan arsitektur jaringan *backpropagation* dicirikan dengan adanya *looping* (koneksi balik atau rambat balik). Metode ini adalah metode yang paling sederhana dan mempunyai konsep belajar yang mudah dipahami. JST dengan arsitektur jaringan *backpropagation* pertama kali diperkenalkan oleh Rumelhart, Hinton dan William pada tahun 1986, kemudian Rumelhart dan Mc Clelland mengembangkannya metode ini pada tahun 1988. JST *backpropagation* akan mengubah bobot dan biasnya untuk mengurangi perbedaan antara *output* jaringan dan target *output*. Pelatihan akan terus dilakukan hingga bobot yang dicapai pada jaringan tersebut dianggap ideal atau telah mencapai *minimum error*. Setelah pelatihan selesai, dilakukan pengujian terhadap jaringan yang telah dilatih. *Backpropagation* merupakan algoritma pembelajaran yang terawasi. Secara umum konsep pelatihan dari JST *backpropagation* adalah :

1. Belajar dari kesalahan.
2. Memasukan secara umpan maju (*feed forward*) pola-pola masukan.
3. Menghitung kesalahan (*error*) yang besangkutan.
4. Mengatur bobot-bobot koneksi

2.5.1 Arsitektur JST *Backpropagation*

Sebuah *neuron* dengan jumlah *input* P dan bobot w pada tiap koneksi ditunjukkan pada gambar 2.3. Jumlah i*nput*, bobot dan bias menghasilkan masukan ke fungsi aktivasi *f*. Neuron-neuron dapat menggunakan sembarang fungsi aktivasi *f* yang ada diferensialnya untuk menghasilkan suatu *output* (Siang, 2005).



Gambar 2.7 Satu *layer* jaringan sebagai penyusun *multilayer*.

Arsitektur yang didapatkan dari arsitektur JST *backpropagation* bekerja dengan menerima *input*an **x,** kemudian menghitung suatu respon ŷ(**x**) dengan runtutan (*propagating*) **x** yang saling terkait. Proses ini tersusun dalam beberapa lapisan (*layer*) dan data *input*, **x** , mengalir dari satu lapisan ke lapisan berikutnya secara berurutan. Dalam tiap lapisan, *input* ditransformasi ke dalam lapisan selanjutnya sesuai fungsi aktivasi yang digunakan kemudian diproses maju ke lapisan berikutnya. Nilai *output* ŷ, dihitung pada lapisan *output* dengan persamaan sebagai berikut :

$$\hat{y}_{(k)} = f^{o}\left[\left[\sum_{j=1}^{q} v_j^{o} f_j^{h}\left(\sum_{j=1}^{p} w_{j,i}^{h} x_{i(k)} + b_j^{h}\right) + b^{o}\right]\right] \qquad (2.4)$$

Dimana :

$x_{i(k)}$ = Variabel *input* ke- *i* , (*i* =1,2,..., *p*) dimana *p* merupakan jumlah *input.*

$\hat{y}_{(k)}$ = Nilai dugaan dari variabel *output*

k = Indeks pasangan data *input*-target ($x_{i(k)}$, $\hat{y}_{(k)}$) , *k* =1,2,..., *n*, dimana *n* merupakan jumlah pola.

$w_{j,i}^{h}$ = Bobot dari *input* ke- *i* yang menuju *neuron* ke- *j* pada lapisan tersembunyi, (*j* =1,2,...,*q*)

b_j^{h} = Bias pada *neuron* ke- *j* pada lapisan tersembunyi, (*j* = 1,2,…,*q*) dimana *q* merupakan jumlah node pada lapisan tersembunyi.

f_j^{h} = Fungsi aktivasi di *neuron* ke- *j* pada lapisan tersembunyi

v_j^{o} = Bobot dari *neuron* ke- *j* di lapisan tersembunyi yang menuju *neuron* pada lapisan *output*

b^{o} = Bias pada *neuron* di lapisan *output*

f^{o} = Fungsi aktivasi pada *neuron* di lapisan *output.*

(Suhartono, 2007)

2.5.2 Fungsi Aktivasi JST *Backpropagation*

Fungsi aktivasi pada jaringan syaraf tiruan digunakan untuk memformulasikan *output* dari setiap neuron. Fungsi aktivasi yang umum digunakan yaitu fungsi aktivasi sigmoid biner (logisig), fungsi aktivasi sigmoid bipolar (*tansig*), dan fungsi aktivasi linier (*purelin*). Fungsi aktivasi yang akan digunakan antara lain :

- Fungsi Sigmoid Biner

  Fungsi aktivasi sigmoid biner mempunyai nilai keluaran berkisar antara 0 dan 1

$$f(x) = \frac{1}{1+e^{-x}} \qquad (2.5)$$

- Fungsi *Sigmoid Bipolar (Tansig)*

  Fungsi ini merupakan fungsi *sigmoid* yang sangat menyerupai fungsi *bipolar*, yang juga sering digunakan sebagai fungsi aktivasi ketika rentang yang diinginkan adalah antara -1 dan 1.

$$f(x) = \frac{1-e^{-2x}}{1+e^{-2x}} \qquad (2.6)$$

- Fungsi Linier *(Pureline)*

  Fungsi linier digunakan pada permasalahan di mana ouput dihasilkan adalah penjumlahan dari *input* terboboti

$$f(x) = \sum x_i v_{ij} \qquad (2.7)$$

Dimana :

$f(x)$ = fungsi aktivasi

x = *input* ke *i*

v_{ij} = bobot *input* ke *i* pada node ke *j*

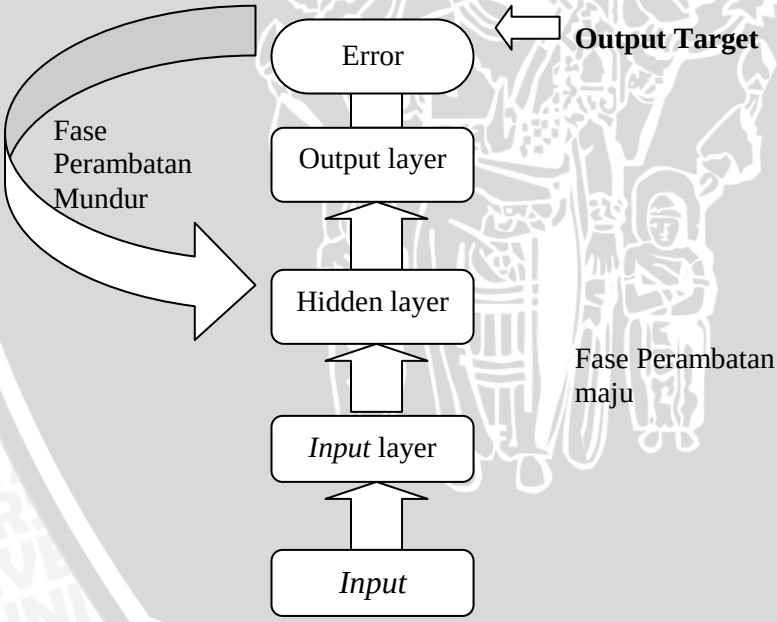
(Puspita, 2006)

2.5.3 Algoritma Pelatihan JST *Backpropagation*

Untuk pelatihan Algoritma dengan arsitektur jaringan *backpropagation* terbagi atas 3 fase, antara lain :

1. Fase Perambatan Maju (*forward)*, operasi pada fase maju dihitung dari *input layer* hingga *output layer* menggunakan fungsi aktivasi yang digunakan.
2. Dilanjutkan dengan Fase perambatan mundur (*backward)*. Selisih antara hasil keluaran (*output*) dengan target yang ditentukan merupakan *error*. *Error* ini dihitung ulang atau dipropagasikan mundur.
3. Fase ketiga adalah modifikasi bobot untuk memperkecil *error* agar mendekati target yang ditentukan.

Ketiga fase tersebut diulang terus hingga kondisi penghentian dipenuhi. Kondisi penghentian yang bisa digunakan adalah jumlah iterasi *(epoch)* atau toleransi kesalahan *(error)*. Operasi *backpropagation* akan berhenti saat salah satu dari kedua kondisi tersebut telah tercapai.



Gambar 2.8 Alur kerja *Backpropagation*

Dengan menggunakan satu *hidden layer*, algoritma pelatihan *backpropagation* akan dijelaskan sebagai berikut :

Langkah 0 Inisialisasi bobot. (diatur pada nilai acak yang kecil),

Langkah 1 Jika kondisi tidak tercapai, lakukan langkah 2-9,

Langkah 2 Untuk setiap pasangan pelatihan, lakukan langkah 3-8,

Fase I : Perambatan Maju

Langkah 3 Tiap unit masukan (x_i, $i = 1,\ldots, n$) menerima signal x_i dan menghantarkan signal ini ke semua unit lapisan di atasnya (*hidden layer*),

Langkah 4 Setiap *hidden layer* (x_i, $i = 1,\ldots, p$) jumlahkan bobot signal masukannya,

$$z\_in_j = v_{oj} + \Sigma\, x_i\, v_{ij} \qquad (2.8)$$

v_{0j} = bias pada *hidden layer j* dan aplikasikan fungsi aktivasinya untuk menghitung sinyal *outputan* $z_j = f(z\_in_j)$ (2.9)

lalu kirimkan signal ini keseluruh unit pada lapisan di atasnya (unit *output*).

Langkah 5 Tiap unit *output* (y_k, $k = 1,\ldots, m$) jumlahkan bobot signal masukannya,

$$y\_in_k = w_{0k} + \Sigma\, z_j\, w_{jk} \qquad (2.10)$$

w_{0k} = bias pada unit keluaran k dan aplikasikan fungsi aktivasinya untuk menghitung signal keluarannya,

$$y_k = f(y\_in_k). \qquad (2.11)$$

Fase II : Perambatan Mundur

Langkah 6 Tiap unit keluaran (y_k , $k = 1,\ldots, m$) menerima pola target yang saling berhubungan pada masukan pola pelatihan, hitung kesalahan informasinya,

$$\delta_k = (t_k - y_k)\, f'(y\_in_k) \qquad (2.12)$$

hitung koreksi bobotnya (digunakan untuk memperbaharui w_{jk} nantinya),

$$\Delta w_{jk} = \alpha\, \delta_k\, z_j \qquad (2.13)$$

hitung koreksi biasnya (digunakan untuk memperbaharui w_{0k} nantinya), dan kirimkan δ_k ke unit-unit pada lapisan di bawahnya,

Langkah 7 Setiap unit lapisan tersembunyi (z_j, $j = 1,\ldots, p$) jumlahkan hasil perubahan masukannya (dari unit-unit lapisan di atasnya),

$\delta\_in_j = \Sigma\ \delta_k\ w_{jk}$ (2.14)

kalikan dengan turunan fungsi aktivasinya untuk menghitung informasi kesalahannya,

$\delta j = \delta\_in_j\ f'\ (z\_in_j)$ (2.15)

hitung koreksi bobotnya (digunakan untuk memperbaharui v_{ij} nanti),

$\Delta v_{ij=}\ \alpha\ \delta_j\ z_j$ (2.16)

Fase III : Perubahan Bobot

Langkah 8 Tiap unit keluaran (y_k, $k = 1,\ldots, m$) update bias dan bobotnya ($j = 0,\ldots, p$) :

w_{jk} (baru) = w_{jk} (lama) + $\Delta\ w_{jk}$ (2.17)

Tiap unit lapisan tersembunyi ($zj, j = 1,\ldots, p$) update bias dan bobotnya ($i = 0,\ldots,n$) :

v_{ij} (baru) = vij (lama)+ Δv_{ij} (2.18)

Langkah 9 : Test kondisi penghentian (proses dihentikan jika error mendekati atau bernilai 0).

2.5 Permasalahan Pada Jaringan Syaraf Tiruan

Jaringan syaraf tiruan (JST) adalah suatu sistem pemroses informasi yang mempunyai karakteristik seperti halnya jaringan syaraf biologi. Karakteristik yang menarik dari jaringan syaraf tiruan adalah kemampunnya untuk belajar (*learning/training*). Proses *training* pada JST betujuan untuk mencari bobot-bobot yang konvergen antar lapisan sedemikian hingga bobot-bobot yang diperoleh menghasilkan output yang diinginkan (Halim, 2011).

Permasalahan pada JST adalah upaya membentuk jaringan yang mampu mengenali pola data yang diberikan, merupakan permasalahan tersendiri bagi peneliti. Hal ini disebabkan belum adanya metode baku yang bisa dijadikan pedoman oleh pengguna jaringan syaraf tiruan dalam menerapkan struktur jaringan terbaik. Selama ini jika ingin metode peramalan JST memberikan hasil ramalan yang akurat harus mencari parameter yang optimal dengan cara *trial and error*.

Jaringan syaraf tiruan dengan arsitektur *backpropagation* merupakan salah satu arsitektur yang banyak digunakan dalam peramalan. Proses *training* jaringan ini mempunyai kelemahan dalam hal mencari bobot yang optimal. Pada algoritma *backpropagation*, penentuan bobot dapat berjalan dengan lambat dan membutuhkan parameter yang sesuai. Teknik yang disarankan untuk mengembangkan kecepatan proses *training* salah satunya adalah teknik optimasi algoritma genetika (Kusumadewi, 2004).

2.6. Kriteria Pemilihan Algoritma Terbaik untuk Peramalan

Metode peramalan yang paling sesuai umumnya adalah metode yang memiliki Root Mean Squared Error *(RMSE)* dan kesalahan persentase absolut *(MAPE)* yang paling kecil. Seperti ditunjukkan persamaan berikut :

- $\text{RMSE} = \dfrac{\sqrt{\sum_{i=1}^{n}(x_t - \hat{x}_t)^2}}{n}$ (2.20)

- $MAPE = \dfrac{1}{n}\sum_{t=1}^{n}\left|\dfrac{x_t - \hat{x}_t}{x_t}\right|.100\%$ (2.21)

dengan, x_t adalah target dan $\hat{x}_t$ adalah nilai prediksi pada data *training/testing*. Kegunaan dari kedua ukuran ketepatan peramalan tersebut adalah untuk membandingkan ketepatan peramalan yang dilakukan dengan dua metode yang berbeda dan untuk mencari teknik yang optimal.

Karakteristik yang diinginkan pada sebuah pendekatan baru biasanya tampak sebagai suatu kontradiksi. Misalnya, setiap metode *time series* harus didasari oleh data masa lalu, sedangkan dalam saat yang bersamaan kondisi data di masa mendatang belum tentu sama dengan masa lalu. Oleh karena itu akurasi peramalan tidak hanya diukur sampai sejauh mana metode yang digunakan sesuai dengan data historis, tetapi juga diukur dari sampai sejauh mana metode yang digunakan tersebut mampu untuk memprediksi kondisi 1,2,3,...,m periode ke depan (Makridakis dan Wheelwright, 1994).


UNIVERSITAS BRAWIJAYA

BAB III
METODELOGI PENELITIAN

3.1 Sumber Data

Data yang digunakan dalam skripsi ini adalah data forex nilai tukar EUR/USD dari http://www.forexpros.com/currencies/eur-usd-historical-data. Data yang diamati adalah harga penutupan (*close position)*, sehingga yang diramalkan adalah posisi *close* yang akan datang. Pengamatan pada data ini terbagi atas 2 amatan, data pertama adalah nilai tukar mata uang Euro terhadap US dollar dan data kedua adalah nilai tukar mata uang US dollar terhadap Rupiah. Data pertama dan data kedua adalah data harian sebanyak 347 data (1 januari 2011 sampai 8 februari 2012). Data selengkapnya dapat dilihat pada Lampiran 1 dan 2.

3.2 Metode Analisis Data

Aplikasi jaringan syaraf tiruan dan algoritma genetika dalam mencari bobot dan bias dilakukan secara berurutan. Kedua data diberi perlakuan yang sama untuk melihat apakah metode JST mampu dan bisa digunakan pada semua jenis nilai tukar mata uang asing. Berikut metodelogi penelitian untuk kedua data yang diuji :

1. Diketahui sejumlah data *time series* $x_1, x_2,\ldots,x_n$.
2. Pada proses Algoritma Genetika :
    1) Mencari parameter terbaik dari Algoritma Genetika
    2) Mencari bobot dan bias, selanjutnya bobot dan bias ini disimpan.
3. Pada proses Jaringan Syaraf Tiruan :
    1) Menentukan jumlah *hidden layer* dan note (*neuron*) pada masing-masing *hidden layer*.
    2) Proses *Training* dan *testing* untuk menentukan arsitektur layak.
    3) Pemeriksaan kelayakan arsitektur dilakukan dengan pengujian Autokorelasi sisaan dengan plot ACF.
    4) *Testing* model terbaik untuk melihat hasil prediksi.
4. Bobot dan bias yang disimpan dari Algoritma Genetika digunakan untuk menggantikan bobot dan bias model terbaik JST.

5. Melihat hasil prediksi JST dengan inisialiasi AG dibandingkan JST tanpa inisialisasi AG.

Secara lebih detail, tahapan dalam penelitian ini adalah sebagai berikut :

3.2.1 Langkah langkah Algoritma Genetika

1. Menentukan parameter *crossover rate, mutation rate,* jumlah populasi, dan maksimum generasi terbaik.
2. Setelah parameter Algoritma Genetika ditemukan, parameter ini akan digunakan acuan untuk menentukan bobot dan bias. Secara garis besar untuk menentukan *fitness* minimum adalah sebagai berikut :
    a) Evaluasi populasi kromosom untuk menentukan nilai kecocokan (*fitness*) suatu kromosom (solusi).
    b) Seleksi kromosom sesuai dengan nilai *fitness*nya menggunakan metode *roulette-wheel*.
    c) *Cross over* kromosom induk untuk meningkatkan populasi menggunakan metode *one-cut point.*
    d) Mutasi gen untuk mendapatkan kromosom baru.

Jika fitness terbaik sudah tercapai,maka proses berhenti. Bobot dan bias yang didapat dari perhitungan minimisasi dengan Algoritma Genetika, digunakan sebagai inisialisasi bobot dan bias pada jaringan terbaik yang ditemukan JST.

3.2.1 Langkah langkah Jaringan Syaraf Tiruan

1. Penentuan fungsi aktivasi terbaik *(logsig, tansig, dan purelin).*
2. Penentuan jaringan yang layak digunakan di antara 5 kemungkinan.
3. Secara garis besar langkah langkah JST dengan algoritma *backpropagation* sebagai berikut :
    a) Inisialisasi bobot [-1 , 1]
    b) Perambatan maju (*Feedforward*) pola input hingga respon mencapai lapisan output (proses ini ada pada sub bab 2.5.3 langkah 3 sampai 5)
    c) Respon yang dihasilkan pada lapisan output akan dibandingkan dengan nilai target, dan dihitung MSE. Jika sudah terpenuhi sesuai dengan dengan toleran kesalahan sebesar 0,01 maka proses penghitungan dihentikan. Namun sebaliknya jika kriteria penghentian belum

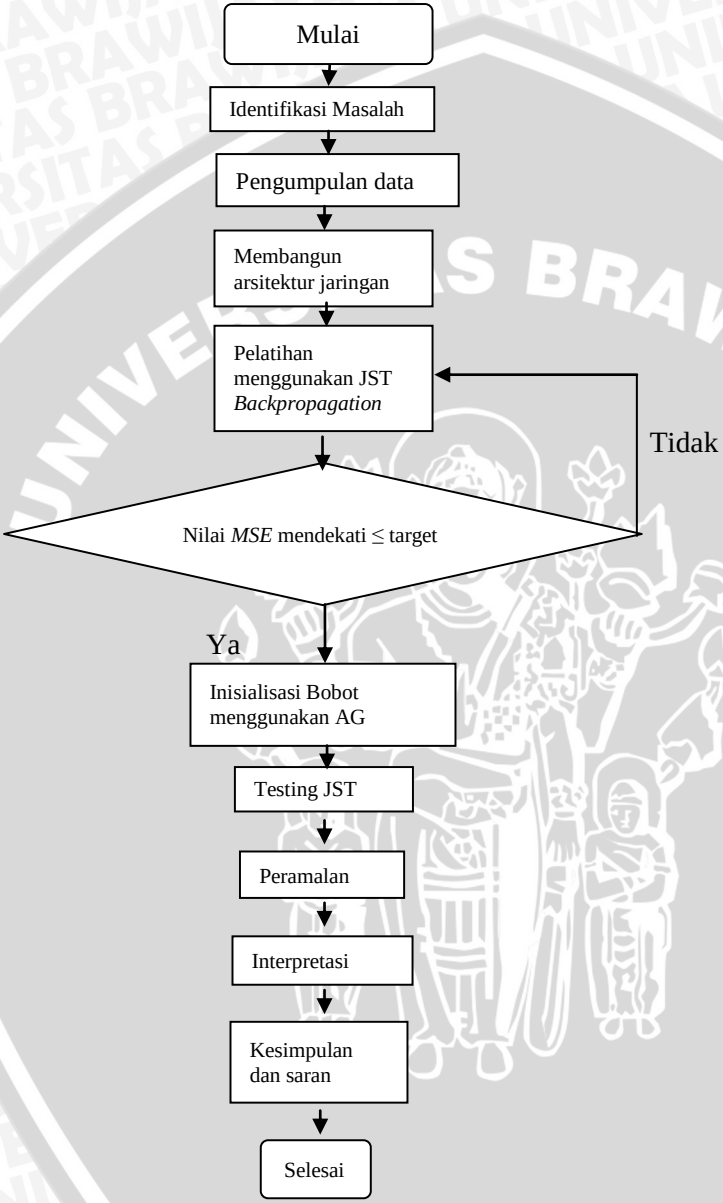
terpenuhi maka, dilanjutkan ke langkah d.

d) Perambatan mundur dari lapisan output kembali ke lapisan input dan melakukan penyesuaian bobot serta mengulangi proses (sub bab 2.5.3 dan ikuti langkah 6 sampai 7).
e) Menghitung nilai *Mean Square Error* (MSE), RMSE, dan MAPE arsitektur yang dihasilkan JST *Backpropogation.*

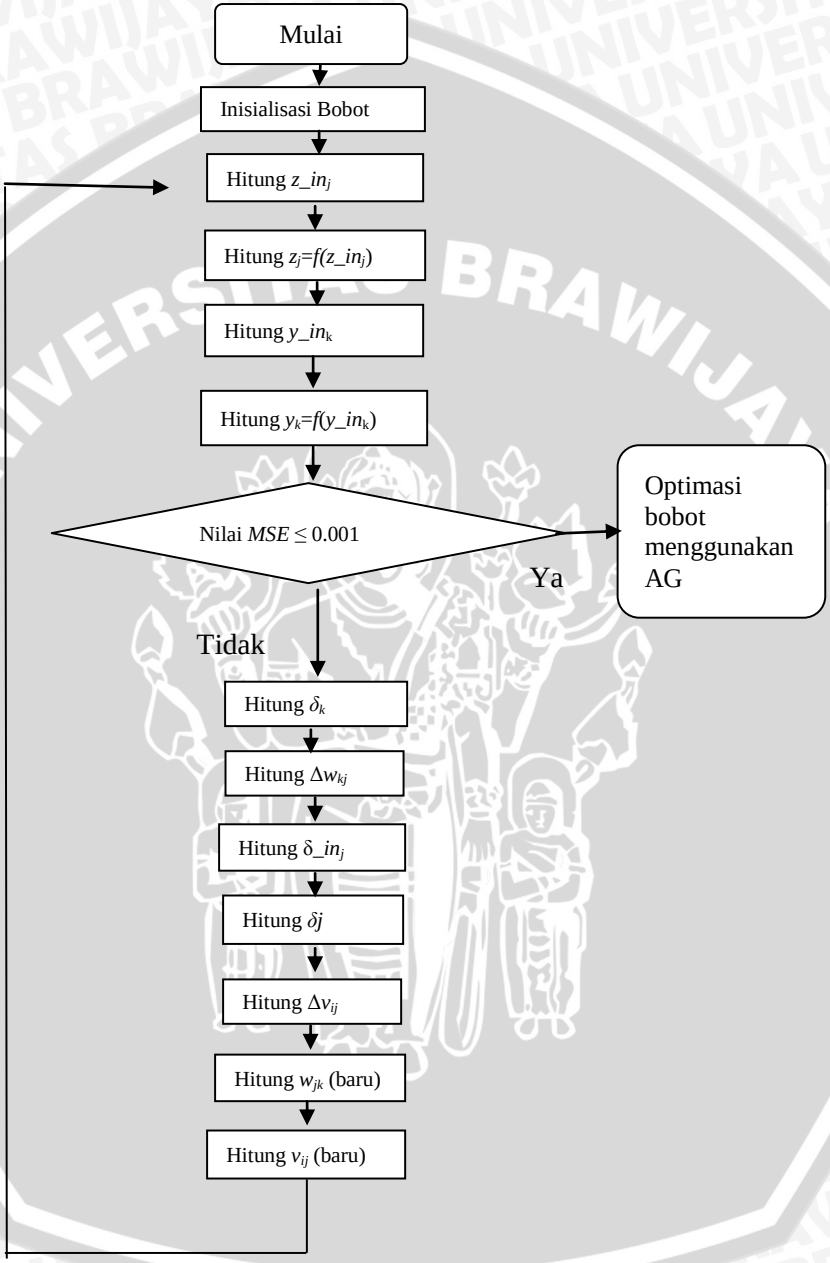
4. Setelah melakukan training dari 5 kemungkinan arsitektur yang diuji, dilihat kelayakan arsitektur dari plot ACF sisaan masing masing arsitektur.
5. Arsitektur terbaik yang dipilih adalah arsitektur dengan MSE training terkecil dan memiliki plot ACF sisaan yang layak.

3.2.3 Penggabungan AG dengan JST *Backpropagation*

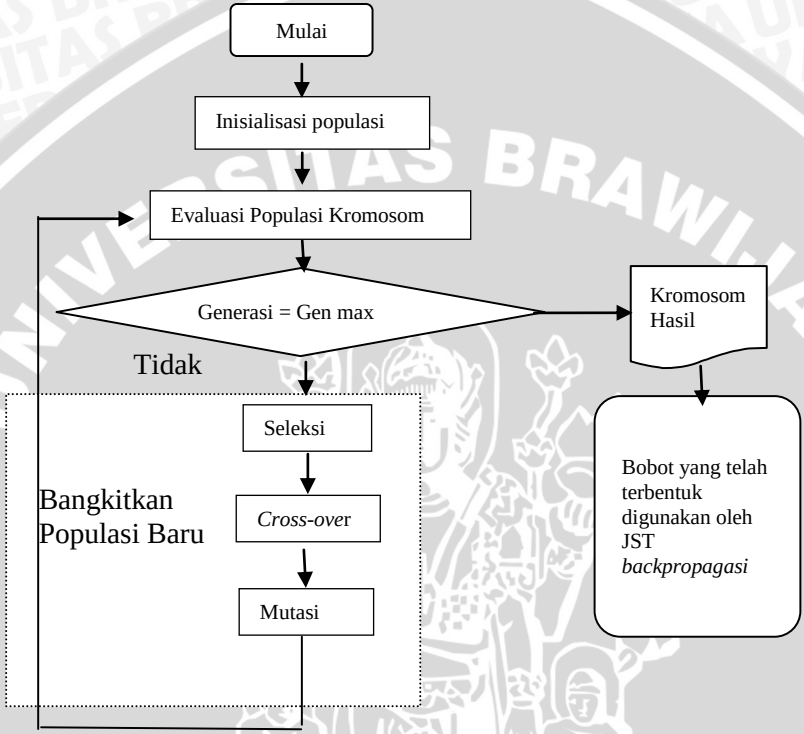
1) Bobot dan bias dari perhitungan minimisasi dengan Algoritma Genetika selanjutnya akan disimpan dalam bentuk matrix dalam matlab. Bobot ini akan diberi nama bias1, bias2, bobot_IW, dan bobot_LW.
2) Jaringan yang telah diinisialisasi digunakan untuk peramalan, menggunakan parameter JST yang sama dengan sebelumnya.
3) Dilihat kelayakan arsitektur dari plot ACF sisaan masing masing arsitektur inisialisasi Algoritma Genetik
4) Arsitektur terbaik yang dipilih adalah arsitektur dengan MSE training terkecil dan memiliki plot ACF sisaan yang layak.
5) Jaringan dengan inisialisasi AG dibandingkan dengan jaringan tanpa inisialisasi AG, dilihat dari waktu *training* dan kebaikan arsitektur melalui MAPE.
6) Kesimpulan.



Gambar 3.1. Diagram Alir Penelitian Inisialisasi Algoritma Genetika Pada JST *Backpropagation*



Gambar 3.2. Diagram Alir Algoritma *Backpropagation*



Gambar 3.3. Diagram Alir Algoritma Genetika

BAB IV
HASIL DAN PEMBAHASAN

4.1 Algoritma *Backpropagation*

Langkah pertama yang harus dilakukan adalah menyusun data berdasarkan arsitektur dari JST algoritma *backpropagation* yaitu data *input* dan *target*. Data dibagi menjadi dua bagian yaitu data *training* dan data *testing*. Peramalan nilai tukar mata uang asing hari berikutnya ditentukan oleh nilai tukar pada hari yang sama minggu sebelumnya.

Input sistem adalah nilai tukar dengan hari awal ke P1 (minggu), P2 (senin), P3 (selasa), P4 (rabu), P5 (kamis), dan P6 (jumat), sedangkan target sistem adalah nilai tukar ke T (hari minggu pada minggu berikutnya). Tabulasi untuk inputan data dapat dilihat pada *Lampiran 2*.

4.1.1 Proses Pelatihan (*Training*) Menggunakan Algoritma *Backpropagation*

Penggunaan jaringan syaraf tiruan *backpropagation* bertujuan mengenal pola dari data *input* secara benar dan mampu memberikan respon yang baik. Semakin lama jaringan dilatih (*train*), tidak menjamin jaringan akan mampu mengenal pola pengujian dengan tepat. Jadi tidaklah bermanfaat untuk meneruskan iterasi hingga semua kesalahan pola pelatihan sama dengan nol (Siang, 2005). Proses *training* menggunakan iterasi (*epoch*) sebanyak 10000. Pertimbangan ini berdasarkan peneltian dan hasil coba-coba sebelumnya.

Fungsi aktivasi dari input layer ke *hidden* layer juga akan dicoba coba menggunakan *tansig* dan *logsig* dan *hidden* layer ke output digunakan fungsi *linier*. Besarnya *learaning rate* yang digunakan adalah 0,06. Teknik optimasi yang digunakan pada jaringan syaraf *backpropagation* adalah *Gradient Descent* (*traingd*).

4.1.2 Penentuan Fungsi Aktivasi terbaik

Sebelum mencari fungsi aktivasi dalam matlab, terlebuh dahulu kita harus menyusun *perintah* yang akan dimasukan dalam matlab. Untuk penulisan *perintah* JST backpropagaton secara jelas dapat dilihat pada *Lampiran 5.*

Fungsi aktivasi adalah fungsi matematis di dalam neuron yang mengolah *input* yang masuk di dalam neuron. *backpropagation* melatih jaringan untuk mendapatkan keseimbangan antara kemampuan jaringan untuk mengenali pola yang digunakan selama pelatihan *(training)* serta kemampuan jaringan untuk memberikan respon yang benar terhadap pola masukan (*input*) yang serupa (tapi tidak sama) dengan pola yang dipakai saat *training*.

Penentuan fungsi aktivasi terbaik dilihat dari jaringan yang ditentukan sebelumnya. Setting awal JST yang ditetapkan sebelumnya antara lain batas iterasi yang digunakan adalah 10000, metode pembelajaran (*training*) untuk kedua data dengan gradient descent (*traingd*). *Performe goal met* data nonlinier kubik : MSE < 0.035, dan jaringan yang akan digunakan adalah 6 – 25 – 1.

Fungsi aktivasi berperan saat proses *training* data, fungsi aktivasi yang baik adalah fungsi yang mampu menyelesaikan *training* data (*performance goal met).* Berikut hasil pengujian jaringan dalam menentukan fungsi aktivasi terbaik :

Tabel 4.1 Fungsi aktivasi terbaik

Fungsi aktivasi	Iterasi	Keterangan
tansig – logsig	10000	*Maximum Epoch reached*
tansig - purelin	**5228**	**Performance Goal Met**
logsig – tansig	10000	*Maximum Epoch reached*
logsig – purelin	10000	*Maximum Epoch reached*
purelin – logsig	10000	*Maximum Epoch reached*
purelin – tansig	10000	*Maximum Epoch reached*

Fungsi aktivasi yang mampu menyelesaikan *training* jaringan adalah fungsi aktivasi *tansig - purelin*. Untuk fungsi aktivasi yang lain tidak mampu menyelesaikan proses *training*, sehingga fungsi tersebut tidak diperhitungkan.

4.1.3 Pemilihan Arsitektur Terbaik

Pada penelitian ini arsitektur terbaik dari jaringan syaraf tiruan algoritma *backpropagation* harus layak. Arsitektur yang tebentuk dikatakan layak digunakan jika sisaan *(error)* dari plot ACF nya berada dalam batas $2/\sqrt{n}$. Jumlah node (*neuron*) pada masing-masing *hidden layer* sangat menentukan suatu arsitektur dikatakan baik. Pemilihan arsitektur pada penelitian ini melanjutkan parameter terbaik sebelumnya. Batas iterasi yang digunakan adalah 10000, metode pembelajaran (*training*) untuk kedua data adalah gradient descent (*traingd*). *Performe goal met* menggunakan MSE < 0,035.

Tabel 4.2 Jaringan Terbaik Nilai Tukar Euro Terhadap US Dollar

Arsitektur (I - H - O)	Kelayakan	RMSE	MAPE
(6 - 5 - 1)	Tidak layak	0.008325711	0.502431916
(6 - 10 - 1)	Tidak layak	0.008325182	0.492786414
(6 - 15 - 1)	Tidak layak	0.008182903	0.328162857
(6 - 20 - 1)	Tidak layak	0.007248376	0.409000913
(6 - 25 - 1)	**Layak**	**0.0044412**	**0.1284641**

Terlihat pada Tabel 4.2 dari 5 jaringan yang diuji cobakan, hanya 1 jaringan yang mampu menyelesaikan proses *training*. Arsitektur 6 - 25 - 1 mampu menyelesaikan proses *training* jaringan dan menghasilkan arsitektur peramalan yang layak. Uji kelayakan arsitektur menggunakan ACF sisaan dapat dilihat pada *Lampiran 10*. Dengan demikian untuk data petama, arsitektur terbaik untuk jaringan syaraf tiruan dengan algoritma pembelajaran *backpropagation* untuk kasus nilai tukar Euro terhadap US Dollar periode 1 Januari 2011 – 8 Februari 2012 adalah arsitektur 6 - 25 – 1.

Tabel 4.3 Jaringan Terbaik Nilai Tukar US Dollar Terhadap Rupiah

Arsitektur (I - H - O)	Kelayakan	MAPE	RMSE
(6 - 5 - 1)	Tidak layak	0.356670594	41.79649222
(6 - 10 - 1)	Tidak layak	0.366973968	42.8454446
(6 - 15 - 1)	Tidak layak	0.378496313	43.25537647
(6 - 20 - 1)	Tidak layak	0.402503222	47.02715413
(6 - 25 - 1)	Tidak layak	0.37085806	43.7328759

Terlihat pada Tabel 4.3 dari 5 jaringan yang dicoba, seluruh jaringan mampu menyelesaikan proses *training* JST *backpropagation.* Namun kelima jaringan ini tidak layak digunakan, karena jika dilihat dari plot ACF sisaan ada beberapa lag yang melebihi batas $2/\sqrt{n}$. Untuk lebih jelas hasil uji kelayakan arsitektur dapat dilihat pada *Lampiran 11*.

Pada kasus data kedua, Data ini tidak mampu memenuhi kelayakan arsitektur dari semua jaringan yang dicobakan. Walaupun telah dilakukan percobaan pemilihan fungsi aktivasi dan arsitektur jaringan, masih belum bisa menjawab kelemahan jaringan syaraf tiruan untuk mendapatkan arsitektur yang layak. Arsitektur yang dihasilkan dari data ini tidak baik jika dipaksakan.

Analisa dilanjutkan hanya pada data pertama, data ini memiliki arsitektur yang layak digunakan untuk peramalan. Data yang digunakan adalah nilai tukar mata uang Euro terhadap US Dollar periode 1 Januari 2011 – 8 Februari 2012 dengan arsitektur jaringan 6 – 25 – 1. Selanjutnya Bobot dari jaringan ini akan diganti oleh hasil inisialisasi dari algoritma genetika.

4.2 Algoritma Genetika

4.2.1 Penentuan Parameter Algoritma Genetika

Pada proses menemukan jaringan terbaik pada JST dengan Algoritma *backpropagation*, langkah selanjutnya yang dilakukan adalah menemukan inisialisasi bobot dan bias untuk jaringan tersebut. Jaringan terbaik yang dapat terbentuk adalah arsitektur jaringan (6-25-1) sehingga jumlah bobot dan bias yang akan dibangkitkan sebanyak :

$(6_{\text{node input}} \text{ x } 25_{\text{node hidden}}) + (25_{\text{node hidden}} \text{ x } 1_{\text{ node output}}) + (25_{\text{bias input ke hidden}}) + (1_{\text{ bias hidden ke output}}) = 200$ bobot

Arsitektur dari algoritma genetika mengacu pada persamaan 2.16 adalah sebagai berikut :

$$x_{n+1}=a_0+a_1x_1+a_2x_2+\ldots+a_nx_n$$

Sehingga data ke X_{175} dipengaruhi a_0+a_1(data X_1) + a_2(data X_2)+…+a_{199}(data X_{199}). Yang ingin dicari adalah nilai bobot (a_0, a_1,.., a_{199}) yang paling optimal dalam mendekati nilai X_{200}.

Untuk mencari bobot yang paling optimal, dibentuk suatu fungsi yang ingin diminimalisasikan (*fitness)*. Fungsi atau *fitness* yang ingin diminimalisasikan mengacu pada persamaan 2.17, dimana

Min (fitness) = | k - b |

Fitness	adalah nilai minimum yang ingin dicapai
k	adalah nilai prediksi periode x_{n+1}
b	adalah nilai target periode x_{n+1}

Untuk mencapai kriteria *fitness* terbaik (minimum), akan dilakukan percobaan untuk menentukan parameter algoritma genetika terbaik. Parameter yang akan dioptimumkan antara lain parameter *crossover rate*, *mutation rate*, dan jumlah populasi terbaik. Penentuan parameter ini dilakukan dengan mencoba berbagai kemungkinan seperti penentuan jaringan terbaik pada JST *backpropagation* sebelumnya. Percobaan dilakukan menggunakan *software* Matlab R2008b memanfaatkan tool yang tersedia (gatool), penggunaan tool dapat dilihat pada *Lampiran 7*.

4.2.2 Parameter Algoritma Genetika terbaik

Percobaan pertama dilakukan untuk mengetahui kombinasi nilai parameter terbaik sehingga diperoleh nilai *fitness* yang paling minimum. Percobaan dilakukan pengulangan masing-masing sebanyak 5 kali karena inisialisasi dari nilai *fitness* dilakukan secara acak. Percobaan untuk mengetahui nilai parameter *crossover* terbaik

dilakukan dengan mengubah nilai parameter *crossover*, sedangkan nilai parameter yang lain dibuat konstan. Percobaan ini menggunakan nilai parameter mutasi (*mutation rate*), maksimum generasi, jumlah populasi, batas (*treshold*), dan *stall generation* secara berurutan 0.1, 500, 50, 0, dan 100.

Tabel 4.4 Parameter *crossover* optimal

parameter crossover	**0.6**	0.7	0.8	0.9
Rata rata *fitness*	**0.000678**	0.001824	0.001061	0.001147

Nilai parameter crossover yang diuji adalah 0.6, 0.7, 0.8, dan 0.9. Hasil pengujian dapat dilihat pada *Lampiran 7*. Pada Tabel 4.4 menunjukkan nilai parameter crossover yang paling optimal memiliki nilai rata rata *fitness* paling kecil. Nilai *crossover* dengan rata-rata *fitness* optimal yang dihasilkan adalah 0.6.

Tabel 4.5 Parameter mutasi optimal

parameter mutasi	0.1	0.2	0.3	0.4	0.5	**0.6**
rata - rata fitness	0.000406	0.000394	0.000324	0.000297	0.000381	**0.0002**

Nilai parameter mutasi yang diuji adalah 0.1, 0.2, 0.3, 0.4, 0.5, dan 0.6. Setiap pengujian dilakukan pengulangan sebanyak 5 kali dengan menggunakan nilai parameter *crossover* terbaik yang diperoleh pada percobaan sebelumnya, yaitu 0.6. Hasil pengujian dapat dilihat pada *Lampiran 7*. Hasil pengujian pada tabel 4.5 menunjukkan bahwa nilai mutasi yang paling optimal adalah 0.6.

Tabel 4.6 Parameter jumlah populasi optimal

Parameter jumlah populasi	40	50	60
rata - rata *fitness*	0.000252	0.000272	0.000389

Jumlah populasi yang diuji adalah 40, 50, dan 60. Setiap pengujian dilakukan pengulangan sebanyak 5 kali. Nilai parameter *crossover* dan mutasi yang digunakan adalah 0.6 dan 0.6. Hasil pengujian dapat dilihat pada *Lampiran 7*. Hasil pengujian Tabel 4.6 menunjukkan bahwa jumlah populasi yang paling optimal adalah 40.

Penentuan maksimum generasi dilakukan tanpa pengujian karena dari percobaan sebelumnya dapat dilihat bahwa rata-rata nilai optimal dicapai di bawah 150 generasi, sehingga pemberian nilai 500 pada maksimum generasi tidak terlalu bermasalah. Untuk penentuan *Stall generation* yang digunakan adalah 100, penentuan nilai ini diambil berdasarkan nilai rata-rata selisih generasi pada satu pengulangan disaat terjadi perbaikan nilai fitness. Penentuan nilai stall generation yang terlalu kecil mengakibatkan algoritma genetika berhenti terlalu cepat sehingga hasil yang diperoleh tidak optimal,

sebaliknya nilai *stall generation* yang terlalu besar mengakibatkan proses semakin lama. Waktu yang digunakan dalam proses algoritma genetika dibatasi selama 100 detik.

Kesimpulan dari penentuan parameter algoritma genetika terbaik adalah:

- *Crossover rate* : 0.6
- *Mutation rate* : 0.6
- Jumlah Populasi : 40
- Maksimum generasi : 500
- *Stall generation* : 100
- *Time limit* : 100

Hasil running program algoritma genetika dapat dilihat pada *lampiran* 7. Dengan *stall generation* dibatasi 100 nilai fitness terbaik yang dapat dicapai adalah 0.000053004. Waktu yang diperlukan untuk menyelesaikan proses running algoritma genetika adalah 16 detik. Bobot optimal yang dihasilkan algortima genetika dapat dilihat pada *Lampiran 8*. Untuk selanjutnya bobot ini kan disimpan dan digunakan untuk inisialisasi bobot pada JST *backpropagation*.

4.3 Inisialisasi Algoritma Genetika Pada Bobot Jaringan Syaraf Tiruan Algoritma *Backpropagation*.

Langkah selanjutnya, ke - 200 bobot yang telah dihasilkan akan dimasukan sebagai inisialisasi dari jaringan 6-25-1 pada JST algoritma *backpropagation* yang telah ditemukan sebelumnya. Hasil *training* yang telah diinisialisasi bobotnya oleh algoritma genetiika dapat dilihat pada *Lampiran 9*.

Dari *Lampiran 9*, terlihat jika hasil inisialisasi bobot menggunakan algoritma genetika memberikan hasil *training* yang memuaskan. Jumlah iterasi selama proses *training* jaringan sebanyak 3598 iterasi terselesaikan dalam waktu 2 menit 29 detik dan arsitektur yang dihasilkan jaringan ini layak untuk digunakan. Jika dibandingkan dari jumlah iterasi pada JST *backpropagation* 6-25-1 tanpa inisialisasi *(Lampiran 10)*, jumlah iterasi jaringan ini sebanyak 6799 iterasi dan memerlukan waktu 5 menit 9 detik untuk menyelesaikan proses *training*.

Jika dibandingkan berdasarkan jumlah iterasi dan waktu dalam menyelesaikan *training* jaringan, inisialisasi dengan algoritma genetika memberikan keuntungan yang banyak dalam menyingkat waktu. Untuk lebih jelas, waktu dan proses iterasi dapat dilihat pada Tabel 4.7 berikut :

Tabel 4.7 Waktu *training* dan jumlah iterasi jaringan Tanpa Inisialisasi Dan Dengan Inisialisasi Algoritma Genetika.

	Jaringan Tanpa Inisialisasi Algoritma Genetika		Jaringan Dengan Inisialisasi Algoritma Genetika	
Jaringan Terbaik	Jumlah iterasi	Waktu Training	Jumlah iterasi	Waktu Algen + Training JST
6 – 25 – 1	6799	5 menit 9 detik	3598	16 detik + 2 menit 29 detik = 2 menit 45 detik

Setelah didapatkan arsitektur optimal, penggunaan metode jaringan syaraf tiruan bisa dimanfaatkan untuk peramalan beberapa periode ke depan. Data yang akan diprediksi langsung dibandingkan dengan data aktual yang telah muncul. Untuk melihat hasil prediksi dan data aktual nilai tukar Euro terhadap US Dollar dapat dilihat pada *Lampiran 11*. Kemampuan jaringan dalam memprediksi 1,2,3,..., m periode ke depan juga merupakan suatu indikator dalam menetukan suatu arsitektur peramalan dikatakan baik.

Arsitektur yang terbentuk dari kedua metode yang diuji adalah sama, arsitektur jaringan ini adalah 6-25-1 atau jaringan yang memiliki 6 unit *input*, 25 *hidden layer* dan 1 unit *output*. Model dari arsitektur tersebut, adalah sebagai berikut :

$$\hat{y}_{(k)} = f^{o}\left[\left[\sum_{j=1}^{25} v_j^{o} f_j^{h}\left(\sum_{j=1}^{6} w_{j,i}^{h} x_{i(k)} + b_j^{h}\right) + b^{o}\right]\right]$$

Arsitektur JST algoritma *backpropagation* dengan inisialisasi algoritma genetika adalah sebagai berikut :

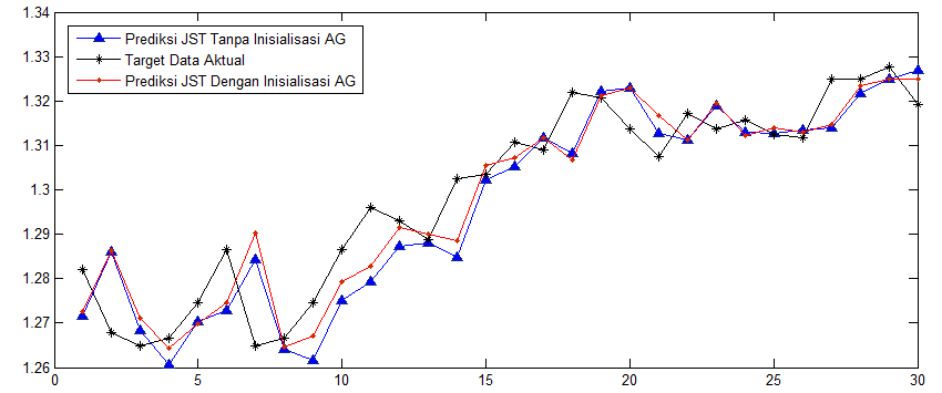
$$\hat{y}_{(k)} = f^{o}[-0.2736\, f_j^{h}((0.5566.\, X_{1,1} - 2.2628) + (0.0071.X_{2,1} - 2.2628) + ... + (-0.0056.X_{6,25} - 2.5571)) - 0.4701] +$$
$$[0.355\, f_j^{h}((0.5566.\, X_{1,1} - 2.2628) + (0.0071.X_{2,1} - 2.2628) + ... + (-0.0056.X_{6,25} - 2.5571)) - 0.4701]$$
$$+ .... + [0.2165\, f_j^{h}((0.5566.\, X_{1,1} - 2.2628) + (0.0071.X_{2,1} - 2.2628) + ... + (-0.0056.X_{6,25} - 2.5571)) - 0.4701]$$

Arsitektur JST tanpa inisialisasi Algoritma genetika adalah sebagai berikut :

$$\hat{y}_{(k)} = f^{o}[-0.7323\, f_j^{h}((-0.3733X_{1,1} + 2.2989) + (-0.3780X_{2,1} + 2.2989) + ... + (0.3440.X_{6,25} + 2.4487)) - 0.3228] +$$
$$[-0.1034\, f_j^{h}((-0.3733X_{1,1} + 2.2989) + (-0.3780X_{2,1} + 2.2989) + ... + (0.3440.X_{6,25} + 2.4487)) - 0.3228]$$
$$+ ... + [0.5109\, f_j^{h}((-0.3733X_{1,1} + 2.2989) + (-0.3780X_{2,1} + 2.2989) + ... + (0.3440.X_{6,25} + 2.4487)) - 0.3228]$$

Kedua arsitektur di atas memiliki arsitektur jaringan yaitu 6 unit pada lapisan *input* (x_1, x_2, x_3, x_4, x_5, x_6), 25 unit pada lapisan tersembunyi, 25 bias dari input ke hidden, 1 bias dari hidden ke output, dan 1 unit lapisan *output*. Arsitektur ini memiliki total jumlah bobot dan bias hasil proses training sebanyak 200. Bobot sendiri adalah suatu nilai yang mendefinisikan tingkat hubungan antar suatu unit dengan unit lainnya. Semakin besar bobot menandakan semakin pentingya hubungan antara kedua unit. Tujuan dari penentuan bobot-bobot ini adalah agar jaringan mampu mengenali pola data dengan baik agar saat arsitektur digunakan untuk peramalan arsitektur mampu meramalkan dengan tepat.

Hasil peramalan antara JST *backpropagation* memanfaatkan inisialisasi AG dan JST *backpropagation* tanpa inisialisasi AG dapat dilihat pada gambar berikut :



Gambar 4.1 Grafik peramalan JST *backpropagation* dengan inisialisasi AG dan JST *backpropagation* tanpa inisisalisasi AG

Gambar 4.1 menunjukan hasil prediksi untuk JST *backpropagation* dengan inisialisasi AG dan JST *backpropagation* tanpa inisialisasi AG. Pada awal mengerjakan tugas akhir, hasil prediksi untuk ke n+1 (8 januari 2012) berada pada posisi 1.2731, Prediksi ke n+2 (9 januari 2012) berada pada posisi 1.2836, dan untuk lebih lengkap dapat dilihat pada *Lampiran 12*. Seiring berjalannya hari, nilai aktual (target) dibandingkan dengan hasil prediksi. Prediksi yang dihasilkan kedua metode jaringan syaraf tiruan mampu mengikuti pola data target, namun untuk keakuratan hasil ramalannya tampak terlambat dari waktu ke waktu. Untuk JST *backpropagation* dengan inisialisasi AG, prediksi untuk ke n+1 (8 januari 2012) berada pada posisi 1.2731 sedangkan poin penutupan nilai tukar untuk tanggal tersebut adalah 1.2821. Prediksi ke n+2 (9 januari 2012) berada pada posisi 1.2836 sedangkan poin penutupan nilai tukar untuk tanggal tersebut adalah 1.2678.

Sedangkan untuk JST *backrpopagation* tanpa inisisalisasi Algoritma genetika, prediksi ke n+1 (8 januari 2012) berada pada posisi 1.2719 sedangkan poin penutupan nilai tukar untuk tanggal tersebut adalah 1.2821. Prediksi ke n+2 (9 januari 2012) berada pada posisi 1.2855 sedangkan poin penutupan nilai tukar untuk tanggal tersebut adalah 1.2678. Kedua hasil prediksi secara lengkap hasil prediksi dapat dilihat pada *Lampiran 12*.

Tabel 4.8 Nilai RMSE dan MAPE JST *Backpropagation* tanpa inisialisasi algoritma genetika

Jaringan Tanpa Inisialisasi Algoritma Genetika			
Waktu Training	Jumlah iterasi	MAPE	RMSE
5 menit 9 detik	6799	12.84641 %	0.0044412

Tabel 4.9 Nilai RMSE dan MAPE JST *Backpropagation* dengan inisialisasi algoritma genetika

Jaringan Dengan Inisialisasi Algoritma Genetika			
Waktu Training	Jumlah iterasi	MAPE	RMSE
2 menit 29 detik	3598	47.5417 %	0.008506

Melihat kedua tabel di atas, nilai MAPE untuk JST memanfaatkan inisialisasi AG lebih besar jika dibandingkan JST tanpa inisialisasi AG. Berdasarkan tabel 4.9 nilai RMSE sebesar 0.008506 dan nilai MAPE sebesar 47,5417%. Sedangkan untuk JST tanpa inisialisasi AG, pada tabel 4.8 nilai RMSE sebesar 0.0044412 dan nilai MAPE sebesar 12,84641 %. Jika dibandingkan kedua algoritma ini, dapat dilihat jika JST *backpropagation* tanpa inisialisasi AG memiliki hasil prediksi yang lebih akurat jika dibandingkan JST *backpropagation* dengan inisialisasi AG.

Namun jika dibandingkan waktu *training* kedua algoritma ini, JST *backpropagation* dengan inisialisasi AG lebih cepat menyelesaikan proses training dengan iterasi lebih sedikit. Waktu training JST *backpropagation* dengan inisialisasi AG adalah 2 menit 29 detik dengan 3598 iterasi, sedangkan JST *backpropagation* tanpa inisialisasi AG memerlukan waktu 5 menit 9 detik dengan 6799

iterasi.Dalam kasus ini inisialisasi dengan algoritma genetik mampu mempercepat proses training dari jaringan syaraf tiruan.

Permasalahan muncul pada nilai tukar US Dollar terhadap Rupiah. Setelah dilakukan percobaan pemilihan fungsi aktifasi dan percobaan pada 5 arsitektur jaringan, data ini tidak bisa memenuhi kelayakan model. Kelemahan jaringan syaraf tiruan belum sepenuhnya terjawab pada data ini. Karena tidak ada aturan baku dalam menentukan parameter jaringan syaraf tiruan, permasalahan pada data ini memiliki beberapa dugaan antara lain cara input data yang salah, jumlah hidden layer yang kurang variasi, teknik optimasi gradient desent tidak mampu menyidik arsitektur yang diuji, atau penggunaan plot ACF membatasi pemilihan kelayakan model dari metode jaringan syaraf tiruan.

Kemampuan jaringan syaraf tiruan algoritma *backpropagation* tanpa inisisalisasi AG dan dengan inisialisasi AG dalam memPrediksi nilai tukar mata uang Euro terhadap US Dollar mempunyai kelebihan dan kerumitan masing-masing. Jaringan saraf tiruan mampu menjawab permasalahan kelayakan model dan arsitektur yang dihasilkan bisa digunakan untuk menguji masalah inisialisasi bobot. Nilai RMSE dan MAPE dari JST *backpropagation* tanpa inisialisasi AG menunjukan jika keakuratan hasil peramalan jaringan ini lebih baik dibandingkan JST dengan inisialisasi AG. Namun kelebihan dari JST *backpropagation* dengan inisialisasi algoritma genetika adalah waktu penyelesaian proses *training* yang lebih cepat.

BAB V
KESIMPULAN DAN SARAN

5.1. Kesimpulan

Berdasarkan hasil pembahasan dapat disimpulkan bahwa :

1. Pemanfaatan algoritma genetika untuk inisialisasi bobot dan bias jaringan syaraf tiruan algoritma *backpropagation* dapat diterapkan pada studi kasus nilai tukar mata uang Euro terhadap US Dollar. Arsitektur jaringan terbaik adalah (6 – 25 – 1), jika diinisialisasi dengan algoritma genetika mampu menghasilkan arsitektur peramalan yang layak digunakan.
2. JST *backpropagation* tanpa inisialisasi AG memiliki hasil prediksi yang lebih akurat jika dibandingkan JST *backpropagation* dengan inisialisasi AG. untuk JST dengan inisialisasi AG nilai RMSE sebesar 0.008506 dan nilai MAPE sebesar 47,5417%. Sedangkan untuk JST tanpa inisialisasi AG nilai RMSE sebesar 0.0044412 dan nilai MAPE sebesar 12,84641 %.
3. Inisialisasi dengan algoritma genetika jika diaplikasikan pada JST algoritma *backpropagation* mampu mempercepat proses *training* namun memberikan hasil ramalan yang kurang akurat. Waktu *training* JST *backpropagation* dengan inisialisasi algoritma genetika adalah 2 menit 29, sedangkan waktu *training* JST *backpropagation* tanpa inisialisasi algoritma genetika selama 5 menit 9 detik.

5.2. Saran

Dalam Implementasi Jaringan Syaraf Tiruan, Algoritma Genetika tidak disarankan dalam inisialisasi bobot awal. Diharapkan menggunakan metode lain untuk menjawab permasalahan lainnya (cara input data, inisialisasi bobot, jumlah node hidden, dan teknik optimasi).

DAFTAR PUSTAKA

Basuki, A. 2003. Algoritma Genetika. http://ilmukomputer.com/2006/09/01/pengantar-algoritma-dan-pemrograman-2/. Tanggal akses: Desember 2011.

Cahyadi , B. 2011. *Jaringan Syaraf Tiruan Backpropagasi Dengan Algoritma Gradient Descent Dan Algoritma Levenberg Marquadt Untuk Peramalan Data Time Series Financial*. Skripsi Program Studi Statistika Jurusan Matematika Fakultas MIPA Universitas Brawijaya Malang. (Tidak dipublikasikan)

Darmawansyah, R. 2009. *Perbandingan Jaringan Syaraf Tiruan Algoritma Backpropagation Dan Algoritma Genetika Untuk Peramala Data Time Series*. Skripsi Program Studi Statistika Jurusan Matematika Fakultas MIPA Universitas Brawijaya Malang. (Tidak dipublikasikan)

Halim, S. dan A.M. Wibisono. *Penerapan Jaringan Syaraf Tiruan Untuk Peramalan*. http://www.petra.ac.id/~puslit/journals/pdf.php?PublishedID=IND00020205. Tanggal akses: 6 Januari 2011.

Hanke, J. E; A. G. Reitsch dan D. W. Winchern. 1992. *Peramalan Bisnis*. Edisi Ketujuh. Alih Bahasa Dewi Anantanur. PT. Prenhallindo. Jakarta.

Hermawanto, D. 2003. *Algoritma Genetika dan Contoh Aplikasinya*. http://ilmukomputer.com/2007/03/29/algoritma-genetika-dan-contoh-aplikasinya/. Tanggal akses: 15 Desember 2011.

Ishikawa M. (2002). *Special Issue of Meteorological Society of Japan on Mathematical Perspective of Neural Network and its Application to Meteorological Problem*. Meteorological Research Note, No.203, pp.29-70.

Kusumadewi, S. 2004. Membangun Jaringan Syaraf Tiruan (Menggunakan Matlab dan Exel Link). Graha Ilmu, Yogyakarta.

Kumar, P; Murthy, R; Eashwar, D. 2008. *Time Series Modeling Using Artificial Neural Network*. Journal of Theoretical and Applied Information Technology.

Makridakis, S. dan S.C. Wheelwright. 1994. *Metode-Metode Peramalan untuk Manajemen.* Edisi Kelima. Alih Bahasa Drs. Daniel Wirajaya. Binarupa Akasara.

Makridakis, S; S.C. Wheelwright dan V.E. McGee. 1999. *Metode dan Aplikasi Peramalan*. Edisi Kedua Jilid Satu. Alih Bahasa Hari Suminto. Binarupa Aksara. Jakarta.

Mufti, A. 2005. *Penerapan Algoritma Genetika Pada Penjejakan Lintasan Serta Pengendalian Gerak Robot Otonom.* http://ft-elektro.usk.ac.id/rekayasa/2005/422_2005.pdf. Tanggal akses: 15 Desember 2011.

Sheng, L. - . A Fuzzy Neural Network Model For Forecasting Stock Price. Zhejiang : Zhejiang University.

Siang, J. J. 2005. *Jaringan Syaraf Tiruan & Programnya menggunakan MATLAB*. Andi . Yogyakarta.

Suhartono. 2007. *Feedforward Neural Networks Untuk Pemodelan Runtun Waktu*. Disertasi Universitas Gajah Mada. Yogyakarta.

Suyanto. 2005. *Algoritma Genetika dalam MATLAB*. Andi . Yogyakarta.

Syamsuddin, A. 2004. *Pengenalan Algoritma Genetika*. http://ilmukomputer.com/2007/03/29/pengantar algoritma-genetika-dan-pemprograman /. Tanggal akses: 21 Desember 2011.

Lampiran 1.

Data Nilai Tukar Mata Uang Euro terhadap US Dollar
periode 2 Januari 2011 sampai 8 Februari 2012

No	Periode	Close Position
1	**Sunday, January 02, 2011**	1.3357
2	**Monday, January 03, 2011**	1.3356
3	**Tuesday, January 04, 2011**	1.3319
4	**Wednesday, January 05, 2011**	1.3156
5	**Thursday, January 06, 2011**	1.2979
6	**Friday, January 07, 2011**	1.2907
7	**Sunday, January 09, 2011**	1.2896
8	**Monday, January 10, 2011**	1.2965
9	**Tuesday, January 11, 2011**	1.2983
10	**Wednesday, January 12, 2011**	1.3127
11	**Thursday, January 13, 2011**	1.3351
12	**Friday, January 14, 2011**	1.3388
13	**Sunday, January 16, 2011**	1.3379
14	**Monday, January 17, 2011**	1.3283
15	**Tuesday, January 18, 2011**	1.3378
.	.	.
.	.	.
.	.	.
.	.	.
.	.	.
337	**Wednesday, February 01, 2012**	1.3171
338	**Thursday, February 02, 2012**	1.3136
339	**Friday, February 03, 2012**	1.3158
340	**Sunday, February 05, 2012**	1.3124
341	**Monday, February 06, 2012**	1.3118
342	**Tuesday, February 07, 2012**	1.3248
343	**Wednesday, February 08, 2012**	1.3248
344	**Thursday, February 09, 2012**	1.3276
345	**Friday, February 10, 2012**	1.3192

Sumber dapat diakses di :
http://www.forexpros.com/currencies/eur-usd-historical-data

Lampiran 2.

Data Nilai Tukar Mata Uang Euro terhadap US Dollar periode 2 Januari 2011 sampai 8 Februari 2012

No	Periode	Close Position
1	Sunday, January 02, 2011	9001.5
2	Monday, January 03, 2011	8965.8
3	Tuesday, January 04, 2011	8985
4	Wednesday, January 05, 2011	8985.1
5	Thursday, January 06, 2011	9065.8
6	Friday, January 07, 2011	9015.3
7	Sunday, January 09, 2011	9015.6
8	Monday, January 10, 2011	9027.3
9	Tuesday, January 11, 2011	9067
10	Wednesday, January 12, 2011	9048
11	Thursday, January 13, 2011	9055.4
12	Friday, January 14, 2011	9085
13	Sunday, January 16, 2011	9074.4
14	Monday, January 17, 2011	9065
15	Tuesday, January 18, 2011	9058
.	.	.
.	.	.
.	.	.
.	.	.
.	.	.
337	Wednesday, February 01, 2012	8997
338	Thursday, February 02, 2012	8987.7
339	Friday, February 03, 2012	8987.5
340	Sunday, February 05, 2012	8966.4
341	Monday, February 06, 2012	8962.7
342	Tuesday, February 07, 2012	8956.8
343	Wednesday, February 08, 2012	8985
344	Thursday, February 09, 2012	8955
345	Friday, February 10, 2012	8879.8

Sumber dapat diakses di :
http://www.forexpros.com/currencies/eur-usd-historical-data


UNIVERSITAS BRAWIJAYA

Lampiran 3 Tabulasi Data Training dan Testing

1. Data Nilai Tukar Mata Uang Euro terhadap US Dollar (2 Januari 2011 - 8 Februari 2012)

Data training

No	t1	t2	t3	t4	t5	t6	target
1	1.3357	1.3356	1.3319	1.3156	1.2979	1.2907	1.2896
2	1.3356	1.3319	1.3156	1.2979	1.2907	1.2896	1.2965
3	1.3319	1.3156	1.2979	1.2907	1.2896	1.2965	1.2983
4	1.3156	1.2979	1.2907	1.2896	1.2965	1.2983	1.3127
5	1.2979	1.2907	1.2896	1.2965	1.2983	1.3127	1.3351
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
467	1.3049	1.3068	1.2926	1.2952	1.2956	1.2958	1.2936
468	1.3068	1.2926	1.2952	1.2956	1.2958	1.2936	1.3054
469	1.2926	1.2952	1.2956	1.2958	1.2936	1.3054	1.2931

Data Testing

No	t1	t2	t3	t4	t5	t6	target
1	1.3357	1.3356	1.3319	1.3156	1.2979	1.2907	1.2896
2	1.3356	1.3319	1.3156	1.2979	1.2907	1.2896	1.2965
3	1.3319	1.3156	1.2979	1.2907	1.2896	1.2965	1.2983
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
28	1.3171	1.3136	1.3158	1.3124	1.3118	1.3248	1.3248
29	1.3136	1.3158	1.3124	1.3118	1.3248	1.3248	1.3276
30	1.3158	1.3124	1.3118	1.3248	1.3248	1.3276	1.3192

Lampiran 3 (lanjutan)

2. Data Nilai Tukar Mata Uang US Dollar terhadap Rupiah
(2 Januari 2011 - 8 Februari 2012)

Data training

No	t1	t2	t3	t4	t5	t6	target
1	9001.5	8965.8	8985	8985.1	9065.8	9015.3	9015.6
2	8965.8	8985	8985.1	9065.8	9015.3	9015.6	9027.3
3	8985	8985.1	9065.8	9015.3	9015.6	9027.3	9067
4	8985.1	9065.8	9015.3	9015.6	9027.3	9067	9048
5	9065.8	9015.3	9015.6	9027.3	9067	9048	9055.4
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
515	8969.5	8962.1	8971.8	8968.6	8994.7	8992.8	9000.4
516	8962.1	8971.8	8968.6	8994.7	8992.8	9000.4	9019.2
517	8971.8	8968.6	8994.7	8992.8	9000.4	9019.2	9025

Data Testing

No	t1	t2	t3	t4	t5	t6	target
1	9024.6	9044.8	9045	8990.5	9054.9	9055	9146.8
2	9044.8	9045	8990.5	9054.9	9055	9146.8	9165
3	9045	8990.5	9054.9	9055	9146.8	9165	9082.3
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
123	8997	8987.7	8987.5	8966.4	8962.7	8956.8	8985
124	8987.7	8987.5	8966.4	8962.7	8956.8	8985	8955
125	8987.5	8966.4	8962.7	8956.8	8985	8955	8879.8

Lampiran 4 Fungsi JST *Backpropagation*

```
load belajar.mat;
[pn,meanp,stdp,tn,meant,stdt]=prestd(p100,t100);
[p1n,meanp1,stdp1,t1n,meant1,stdt1]=prestd(p101,t101);
          function [g,g1,gt]= JST1(pn,tn,p1n,t1n)
          end
```

Langkah – Langkah pada tabel diata untuk memanggil menggunakan Function :

1. Panggil Data terlebih dahulu, disini cara memanggil data adalah `load belajar.mat`
2. Normalisasikan data yang telah di panggil sebelumnya
3. Untuk menjalankan Function, hanya perlu mengetikan

```
[g,g1,gt]= JST1(pn,tn,p1n,t1n)
```

Lampiran 5 Syntax JST *Backpropagation* secara lengkap

1. Buka software matlab (R2008b)
2. Masukan data time series sesuai pada Lampiran 1. Simpan data tersebut dalam matlab (belajar.mat).
3. Pada Command Window ketik perintah berikut :

1	load belajar.mat;
2	[pn,meanp,stdp,tn]=prestd(p100,t100);
3	[p1n,meanp1,stdp1,t1n,meant1,stdt1]=prestd(p101,t101);
4	net= newff(minmax(pn), [25 1], {'tansig' 'purelin'}, 'traingd');
5	net.trainParam.epochs = 10000;
6	net.trainParam.goal = 0.035;
7	net.trainParam.lr = 0.06;
8	net=train(net,pn,tn);
9	g=sim(net,pn);
10	gt=poststd(g,meant,stdt);
11	g1=sim(net,p1n);
12	gt1=poststd(g1,meant1,stdt1);

- Poin 1 adalah *perintah* untuk memanggil data
- Poin 2 adalah normalisasi data training
- Poin 3 adalah normalisasi data input yang akan d testing.
- Poin 4 adalah *perintah* untuk membentuk jaringan (net). format penulisan dalam matlab adalah :

  minmax(jumlah input training), [jumlah node hidden jumlah node outoput], {'fungsi aktvasi hidden layer' 'fungsi aktivasi output layer'}, 'pembelajaran');
- Poin 5 dan poin 6 adalah batasan untuk menghentikan proses *training*. Jika proses *training* melebihi *setting* awal, maka jaringan yang dicoba dinyatakan buruk.
- Poin 8, *train* adalah proses *training* jaringan
- Poin 9, *g* adalah simulasi dari jaringan *(net)* yang telah d *trainining*
- Poin 10 adalah denormalisasi data training
- Poin 11, *g1* adalah simulasi jaringan *(net)* untuk ditest dengan pola data baru *(testing)*.
- Poin 12 adalah denormaliasi data *testing*.

Lampiran 6. Fungsi Fitness Sebagai Input "gatool" Di Matlab

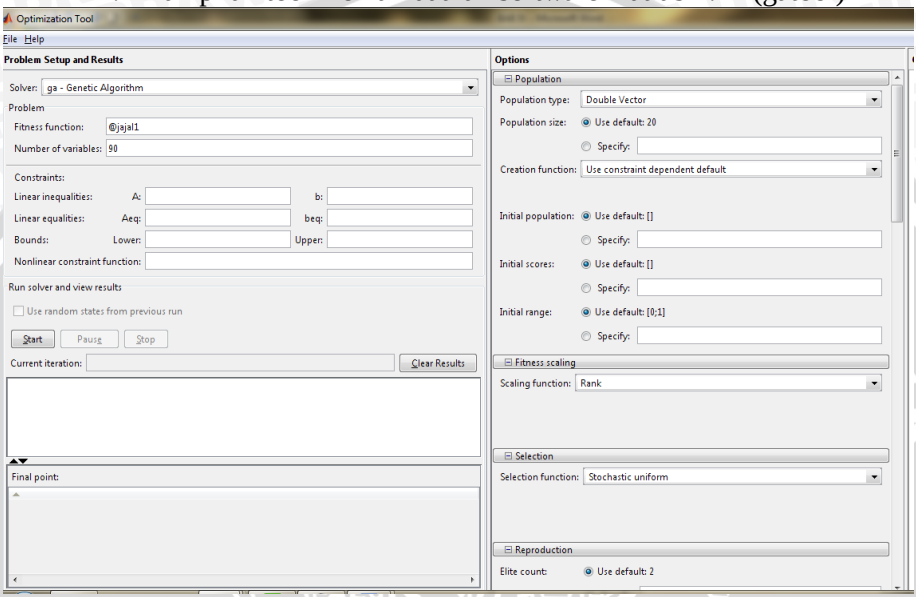
```
function y = jajal1(a)
n=90;
x=    [ 1.368 1.3644   1.3654  1.3781  1.3747  1.3691  1.3631
        1.3622  1.3598  1.3762  1.3608  1.3467  1.3615  1.363
        1.3606  1.3525  1.3543  1.3537  1.3627  1.3639  1.3563
        1.362   1.37    1.3582  1.3625  1.3637  1.362   1.3606
        1.3643  1.3684  1.3768  1.3773  1.3674  1.3768  1.3737
        1.3612  1.3531  1.3513  1.3567  1.3469  1.3331  1.3305
        1.341   1.3445  1.349   1.3435  1.3587  1.3504  1.3494
        1.3467  1.3381  1.3331  1.3376  1.35    1.3657  1.3586
        1.3615  1.3658  1.3502  1.3476  1.3487  1.3423  1.3388
        1.3223  1.3385  1.3363  1.3395  1.318   1.3197  1.3251
        1.3294  1.3313  1.3196  1.2977  1.2817  1.2655  1.275
        1.2868  1.2738  1.2647  1.2533  1.2358  1.2358  1.2383
        1.2145  1.2385  1.254   1.257   1.251   1.234];
b=1.2332;
s=a(1);
      for j=1:89
      k=s+x(j)*a(j+1);
end;
y = abs |k-b|
```

Dalam merancang fungsi *fitness* di atas, model yang ingin dibentuk adalah $x_{n+1}=a_0+a_1x_1+a_2x_2+\ldots+a_nx_n$ dengan keterangan :

- x di dalam fungsi adalah input data.
  Pada persamaan untuk variabel $x = x_1, x_2, x_3, \ldots, x_n$
- b di dalam fungsi adalah target (nilai yang ingin dicapai).
  Pada persamaan untuk variabel $b = x_{n+1}$
- k di dalam *function* adalah prediksi (nilai duga).
- y adalah fungsi *fitness.*Fungsi *fitness* dianalogikan sebagai *error*. *Error* yang ingin dicari adalah nilai *error* yang paling minimum.
- a adalah bobot yang akan dihitung dengan *gatool* (lampiran 6). Nantinya nilai bobot yang mampu menghasilkan *fitness* paling minumum akan disimpan dan dijadikan input sebagai pengganti bobot jaringan syaraf tiruan.

Lampiran 7 Penentuan Parameter Algoritma Genetika.

1. Tampilan tool memanfaatkan software matlab 7.71 (gatool)



- Langkah langkah penggunaan tool algoritma genetika pada matlab terbagi menjadi 2 tahapan, tahapan pertama adalah mengisi layer sebelah kanan (*Problem Setup and Result)* . Layer ini berfungsi sebagai pemanggil fungsi yang telah dibuat sebelumnya. Langkah langkahnya adalah sebagai berikut :

    1. Buka matlab R2008b.
    2. Buka M-file baru dan ketikan fungsi pada Lampiran 5, dan simpan dengan nama *jajal1.m*
    3. Pada Command Window matlab ketikan "*gatool"*
    4. Setelah muncul GUI seperti gambar di atas, pada kolom *fitness function* isi dengan *@jajal1.m*
    5. Pada kolom *Number of Variabel* isikan jumlah bobot yang akan dibangkitkan (90 bobot).

- Tahapan selanjutnya adalah mengisikan parameter dari algoritma genetika pada layer sebelah kiri *(option)*. Parameter yang akan diisikan antara lain :

    1. Pada tab *population* untuk masukan parameter jumlah populasi yang ingin diuji.
    2. Pada tab *Fitness Scaling* untuk memilih fungsi evaluasi dalam pembentukan generasi baru.
    3. Pada tab Selection untuk masukan parameter *crossover rate (cr)* yang ingin diuji.
    4. Pada tab mutasi untuk masukan parameter *mutation rate (mr)* yang ingin diuji.
    5. Untuk Stoping criteria, masukan 3 dari semua table yang tersedia :
        a) Stall Generation : 100 generasi
        b) Generation : 500 generasi bangkitan
        c) Time : 100 detik

Selanjutnya output dari uji coba pencarian parameter terbaik dari algoritma genetika adalah sebagai berikut :

2. Pengujian Parameter *crossover*

	Parameter Crossover			
Fitness	**0.6**	0.7	0.8	0.9
percobaan 1	**4.78E-04**	0.004317	0.002742	9.33E-04
percobaan 2	**1.33E-04**	4.79E-04	2.00E-04	2.00E-04
percobaan 3	**0.001176**	7.27E-05	6.14E-04	0.001148
percobaan 4	**9.40E-04**	1.84E-04	4.71E-05	0.001331
percobaan 5	**6.61E-04**	0.004067	0.001702	0.002122
rata - rata *fitness*	**0.000678**	0.001824	0.001061	0.001147

Lampiran 7 (lanjutan)

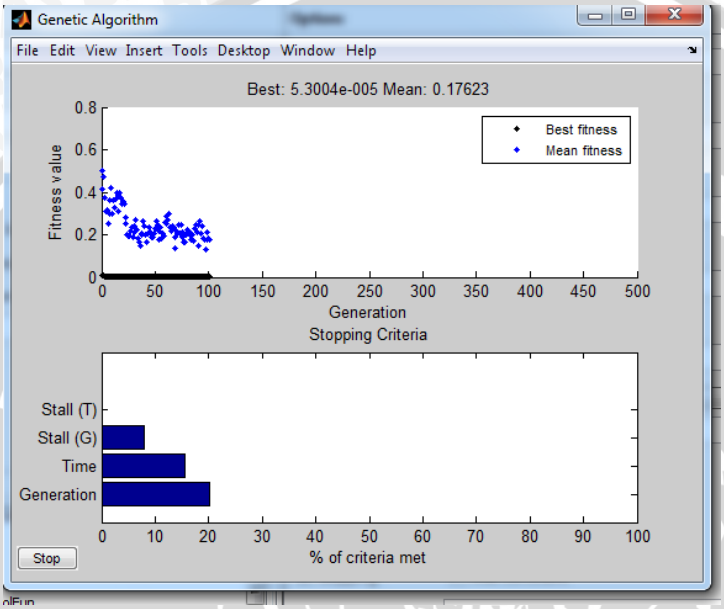
3. Pengujian Parameter mutasi

	Parameter Mutasi					
Fitness	0.1	0.2	0.3	0.4	0.5	**0.6**
percobaan 1	4.36E-04	3.21E-04	3.34E-04	8.15E-06	4.66E-04	**4.85E-04**
percobaan 2	1.35E-04	0.001096	3.33E-04	2.55E-04	2.07E-04	**4.41E-06**
percobaan 3	2.09E-04	6.48E-05	7.61E-05	1.09E-04	1.10E-03	**1.75E-04**
percobaan 4	0.001178	3.42E-04	3.19E-04	7.19E-04	1.18E-04	**3.10E-05**
percobaan 5	7.18E-05	1.49E-04	5.57E-04	3.92E-04	1.34E-05	**3.07E-04**
rata - rata *fitness*	4.06E-04	3.94E-04	3.24E-04	2.97E-04	3.81E-04	**2.00E-04**

4. Pengujian Parameter Jumlah Populasi

	Parameter Jumlah populasi		
Fitness	40	50	60
percobaan 1	4.25E-05	3.15E-04	3.55E-04
percobaan 2	1.84E-04	1.18E-04	1.97E-04
percobaan 3	5.67E-04	2.32E-04	1.56E-04
percobaan 4	2.34E-04	1.13E-04	7.70E-04
percobaan 5	2.31E-04	5.81E-04	4.65E-04
rata - rata *fitness*	2.52E-04	2.72E-04	3.89E-04

5. Hasil *running* progam Algoritma Genetika



- *Fitness* minimum (Best) = 0.000053004
- Waktu yang diperlukan running program AG = 16 detik

Lampiran 8. Bobot Optimal Yang Dihasilkan Algoritma Genetika.

1. Bobot dari layer *input* ke layer *hidden*

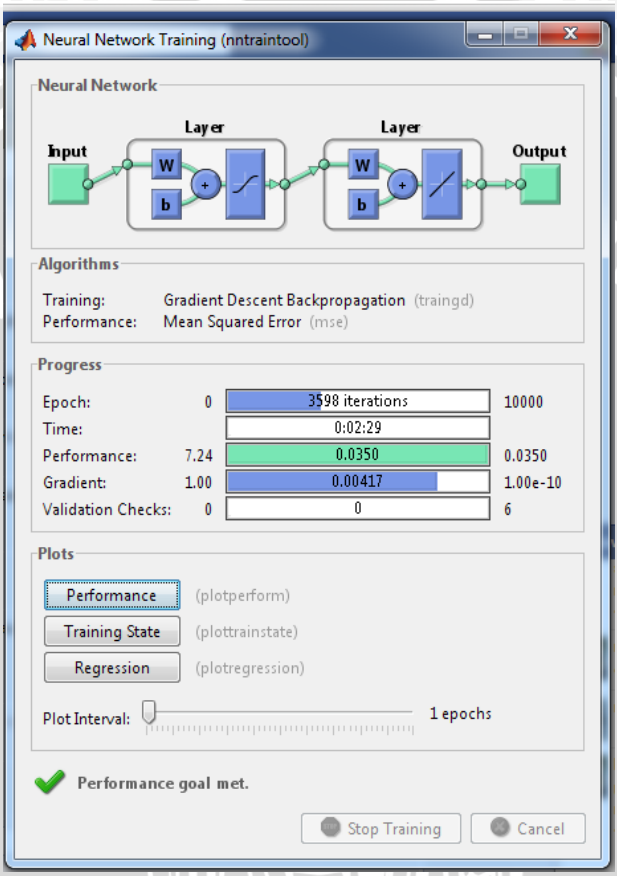
Bobot IW					
0.422798	-0.0337	0.909294	0.487149	0.133432	0.649817
0.456544	0.416257	-0.70203	-0.50219	0.417678	-0.00144
0.527843	0.262166	-0.50489	-0.53911	0.416271	0.550362
-0.92461	-0.09397	0.19281	-0.88658	-0.5439	-0.36339
-0.10569	0.807755	-0.35214	-0.34217	-0.38679	-0.38348
-0.59313	0.097895	-0.72803	0.201409	-0.17984	-0.81514
0.676477	0.012051	-0.18666	0.958725	-0.13912	0.398716
0.747876	0.026322	-0.53001	0.634619	0.479211	-0.1254
0.52337	-0.64237	-0.02937	-0.43702	0.072516	0.692128
-0.81659	0.471724	0.636132	-0.49238	-0.21996	0.167316
-0.50071	0.277739	-0.61641	0.098161	-0.5711	0.627259
-0.26725	-0.00422	0.294476	0.539739	-0.36172	-0.49656
-0.15873	-0.6254	0.471663	0.674597	-0.44378	0.486923
0.520079	-0.36773	-0.49561	-0.13241	0.576251	0.748307
0.927221	0.406277	0.0485	-0.58857	-0.31641	0.32475
0.257455	-0.49175	0.143209	-0.13209	-0.94295	0.301038
-0.33739	-0.57116	-0.62547	0.236411	-0.74275	-0.22854
0.043792	-0.65882	-0.55764	0.103515	-0.61678	0.857357
0.05109	0.655127	0.55272	0.47099	-0.51224	0.604449
0.583834	-0.258	-0.60879	-0.46096	0.171777	0.227623
0.121531	-0.33046	-0.95916	0.200638	-0.59739	0.091081
0.082917	0.717686	0.52791	0.745935	-0.45972	-0.07985
0.294929	0.276172	-0.32433	-0.84078	-0.49139	-0.11853
-0.55041	0.477491	-0.58708	0.118803	0.80716	-0.50445
-0.16063	0.845199	0.244961	-0.30149	0.836836	0.913555

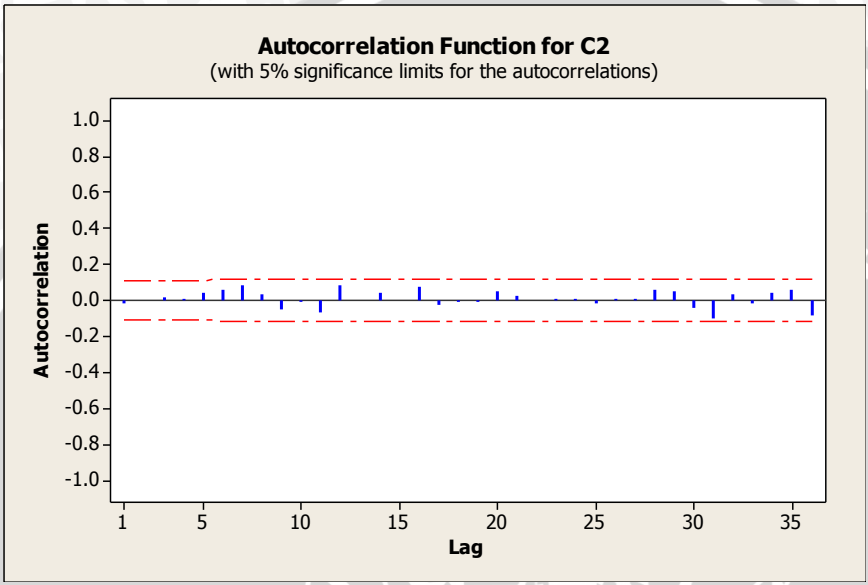
Lampiran 8 (lanjutan)

2. Bobot dari layer *hidden* ke layer *output*

bobot LW				
-0.38723	0.112852	0.644126	0.284377	-0.99828
-0.42489	-0.10705	0.459031	0.168846	-0.4267
0.148748	-0.66685	0.559193	-0.40318	0.38625
0.035998	-0.1012	0.399316	0.478458	0.634294
-0.65067	0.103646	0.280715	-0.52987	-0.19139

Lampiran 9 Hasil training jaringan terbaik dengan inisialisasi bobot Algoritma genetika.

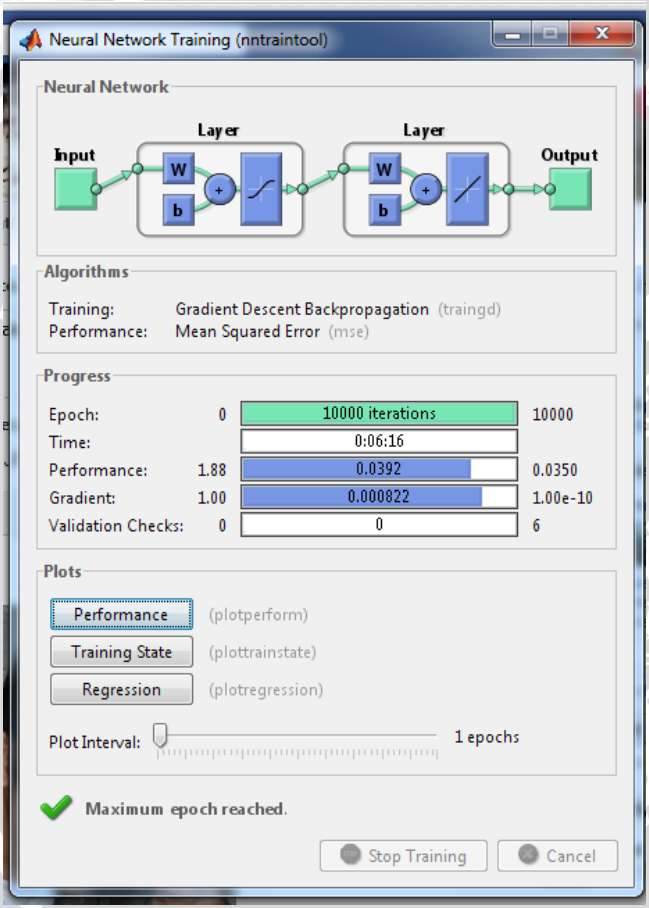




Plot ACF sisaan dari jaringan dengan inisialisasi algoritma genetika memberi informasi tidak ada lag yang keluar dari batas. Model dari jaringan ini layak digunakan.

Lampiran 10 Penentuan Jumlah Hiden Layer Terbaik Dalam Menentukan Arsitektur JST Algoritma *Backpropagation* (Data Nilai Tukar Mata Uang Euro terhadap US Dollar)

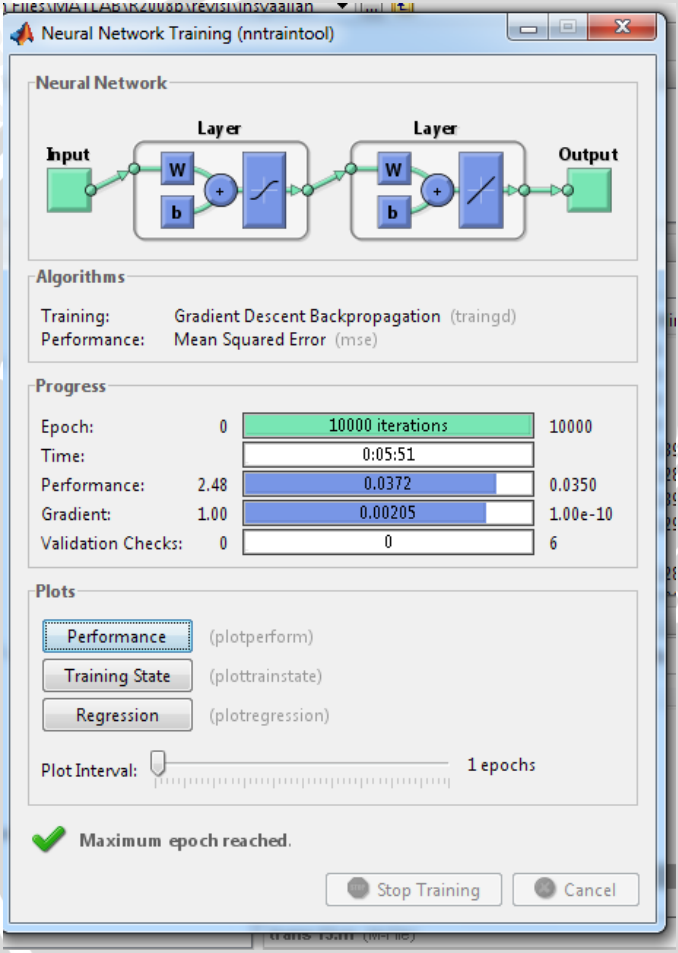
1. 6 - 5 – 1



- Maximum epoch reached :
  Jaringan ini tidak dapat digunakan

Lampiran 10 (lanjutan)

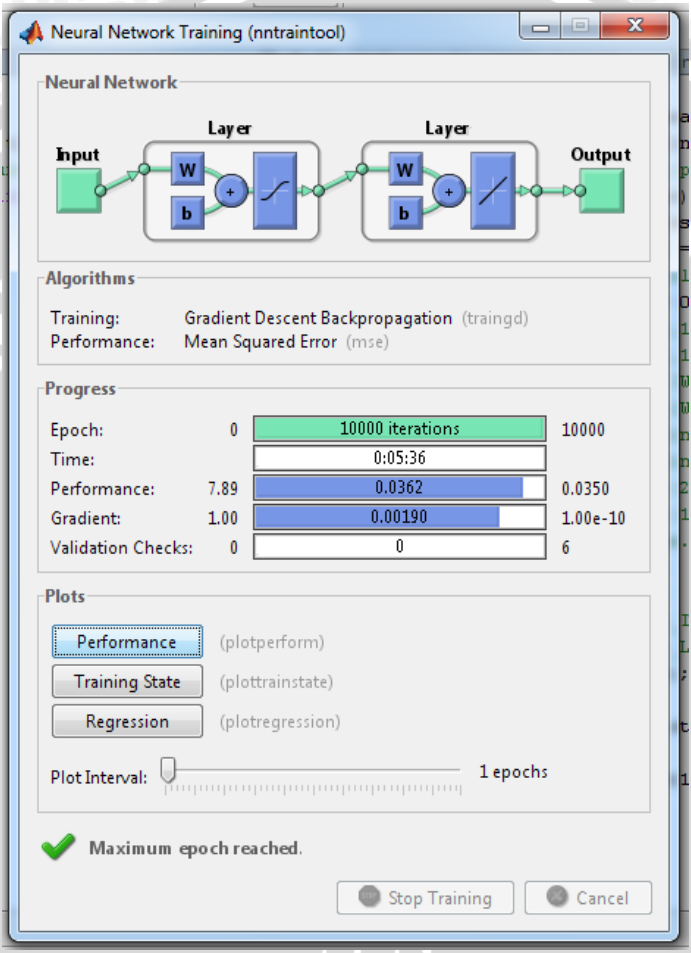
2. 6 – 10 – 1



- Maximum epoch reached :
  Jaringan ini tidak dapat digunakan

Lampiran 10 (lanjutan)

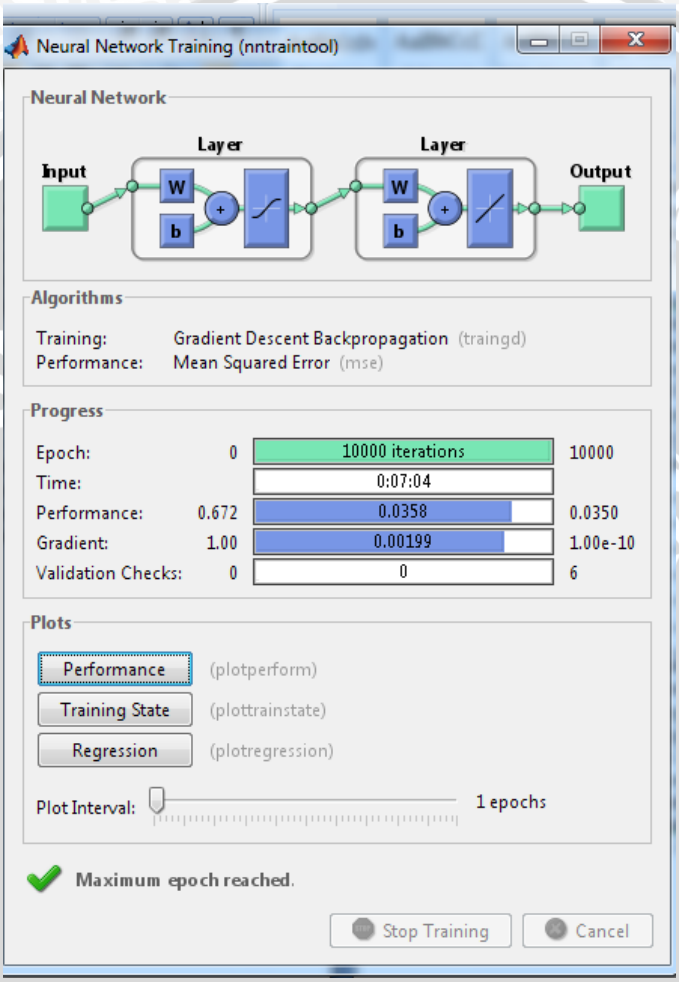
3. 5 – 15 – 1



- Maximum epoch reached :
  Jaringan ini tidak dapat digunakan

Lampiran 10 (lanjutan)

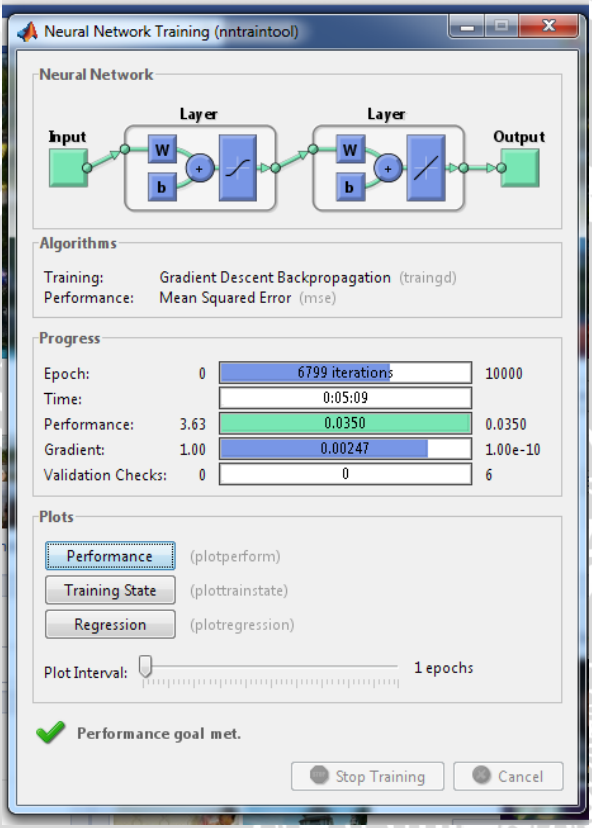
4. 6 – 20 – 1



- Maximum epoch reached :
  Jaringan ini tidak dapat digunakan.

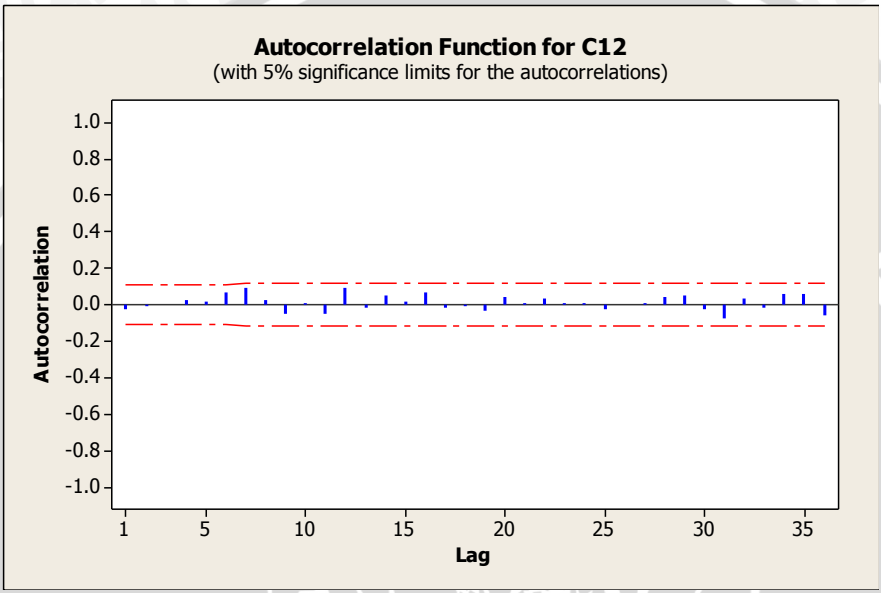
Lampiran 10 (lanjutan)

5. 6 – 25 – 1



Lampiran 10 (lanjutan)

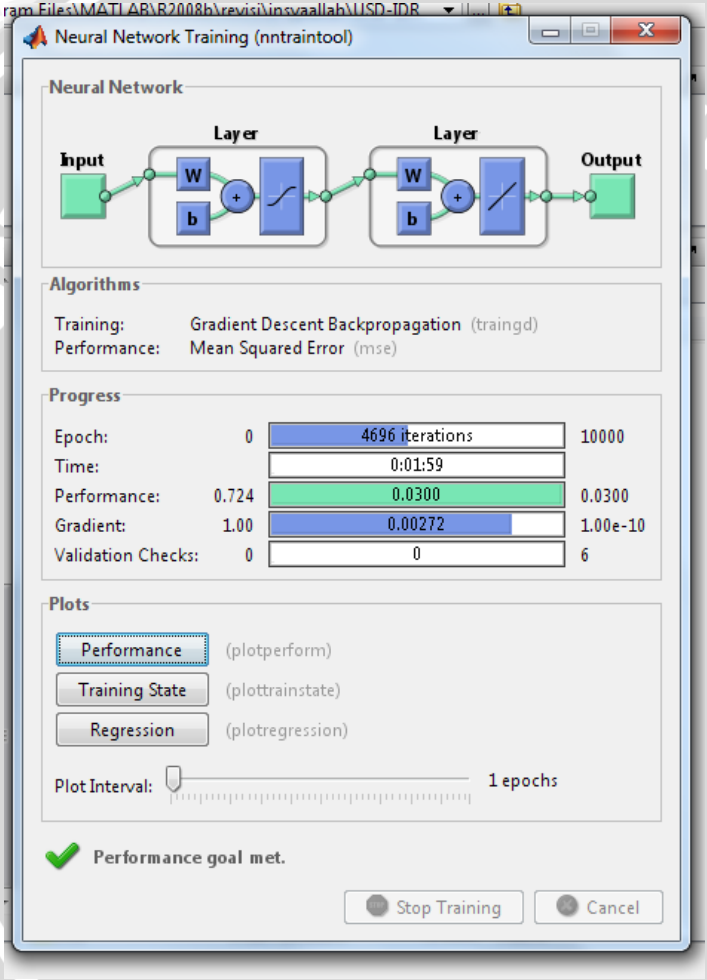
Plot ACF sisaan dari jaringan 6 – 25 - 1

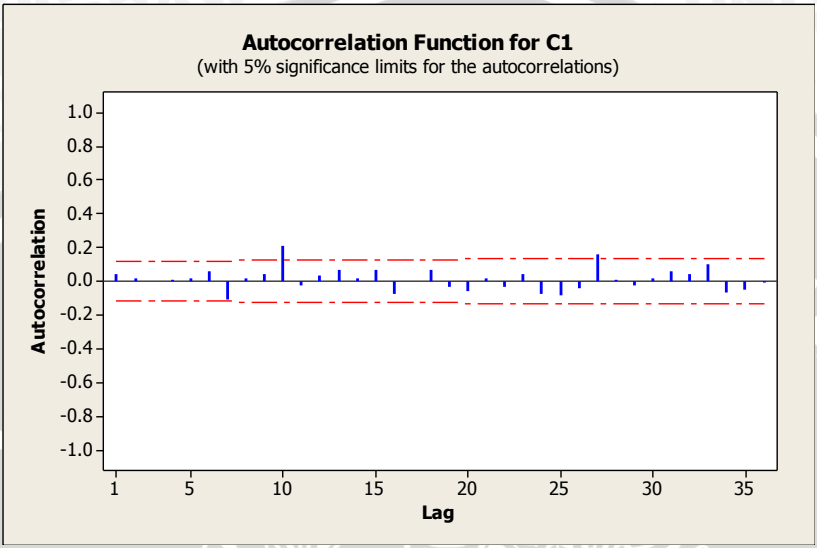


- Plot ACF sisaan dari jaringan 6 – 25 – 1 memberi informasi tidak ada lag yang keluar dari batas. Arsitektur dari jaringan ini layak digunakan

Lampiran 11 Penentuan Jumlah Hiden Layer Terbaik Dalam Menentukan Arsitektur JST Algoritma *Backpropagation* (Data Nilai Tukar Mata Uang US Dollar terhadap Rupiah)

6. 6 - 5 – 1

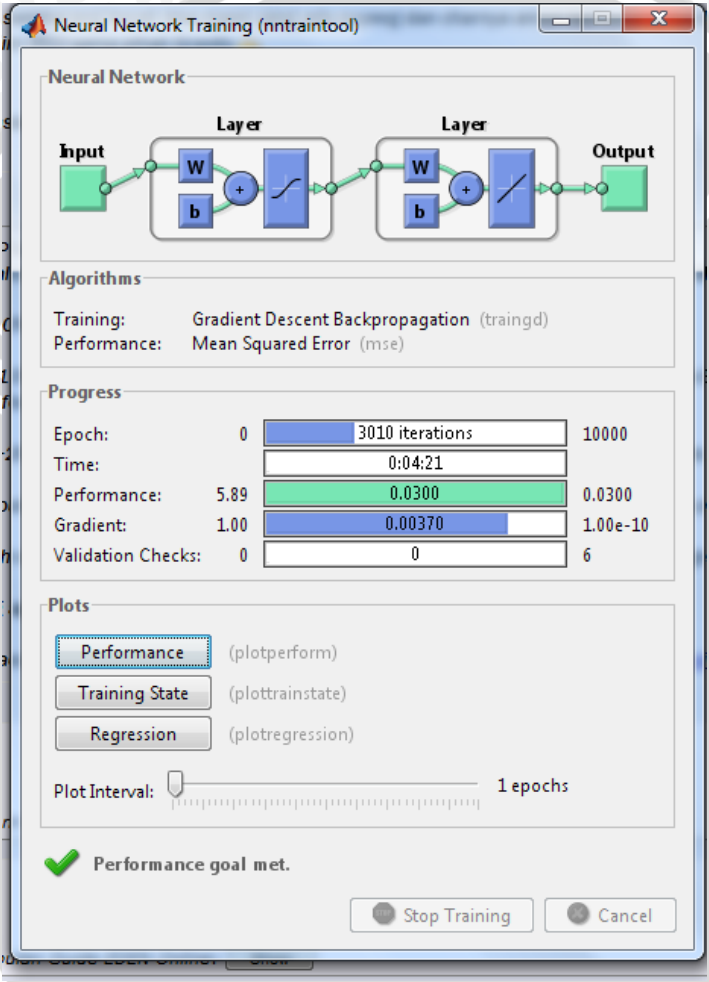




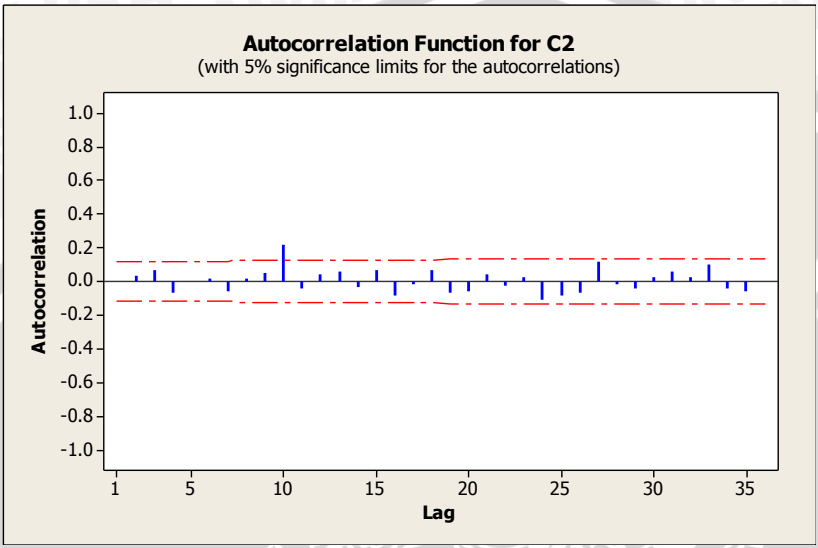
- Dilihat dari ACF sisaan, model yang dihasilkan arsitektur 6-5-1 tidak layak digunakan.

Lampiran 11 (lanjutan)

7. 6 – 10 – 1



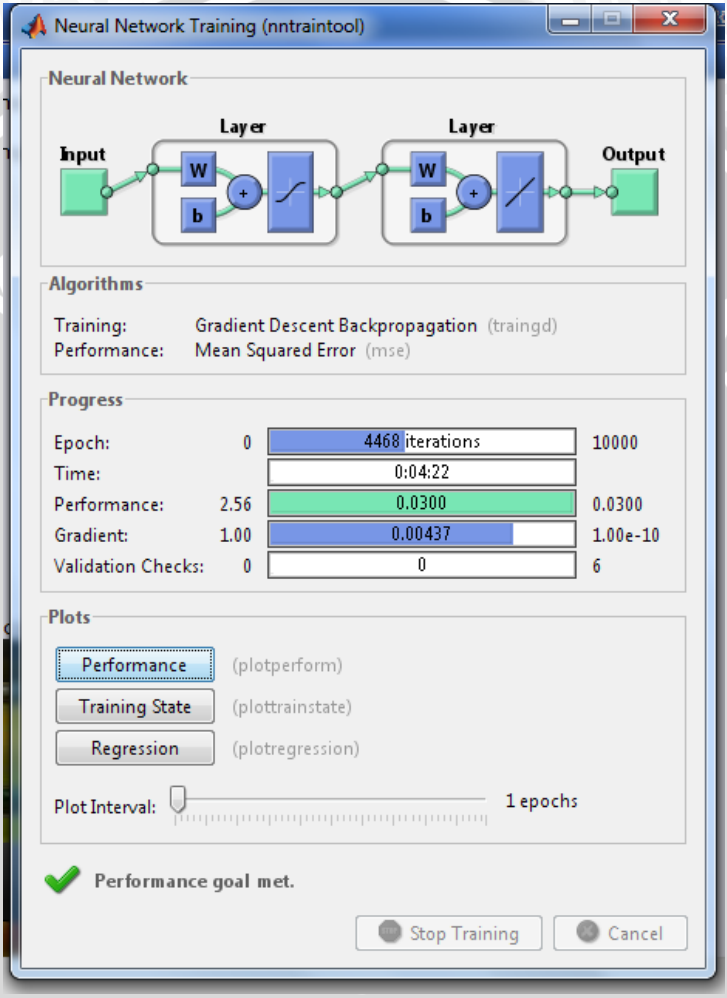
Lampiran 11 (lanjutan)

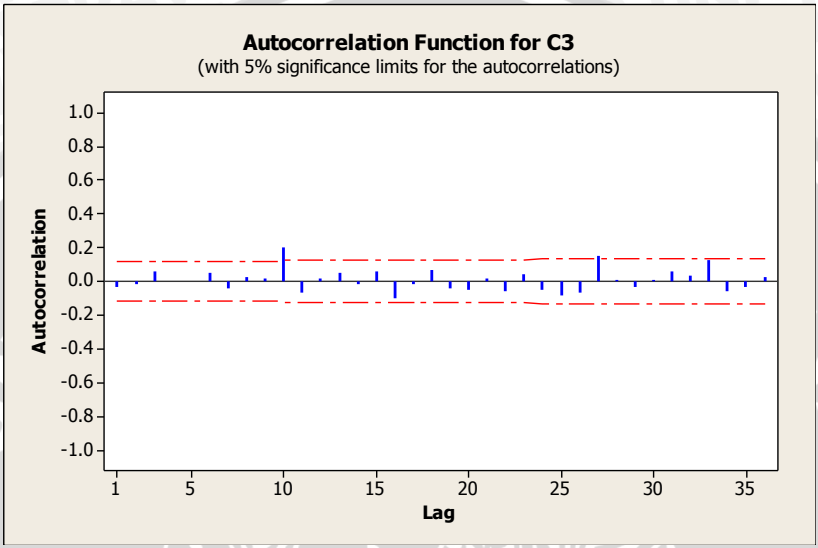


- Dilihat dari ACF sisaan, model yang dihasilkan arsitektur 6-10-1 tidak layak digunakan.

Lampiran 11 (lanjutan)

8. 6 – 15 – 1

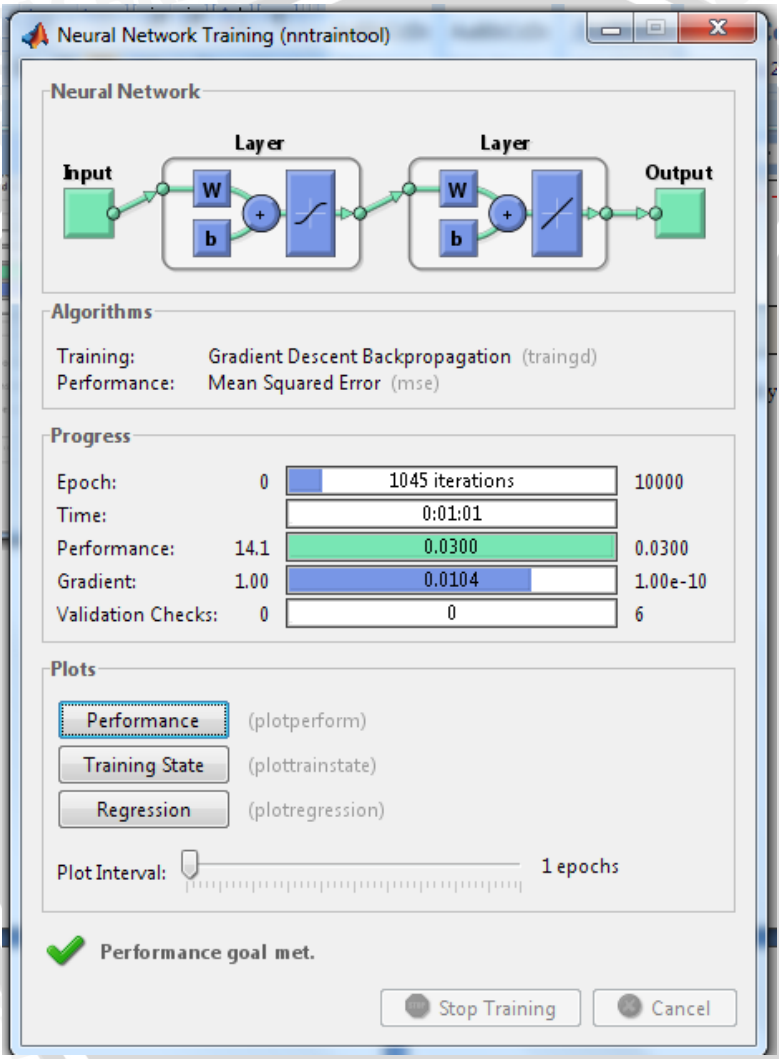




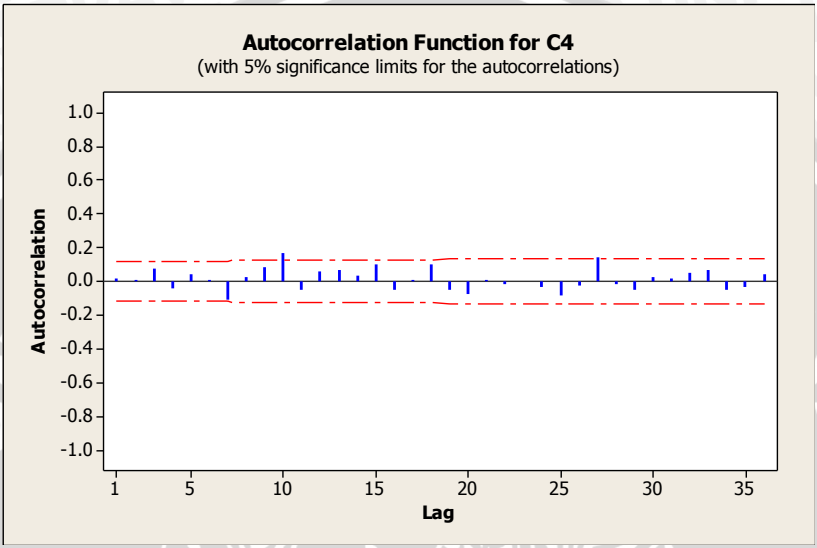
- Dilihat dari ACF sisaan, Arsitektur yang dihasilkan arsitektur 6-15-1 tidak layak digunakan.

Lampiran 11 (lanjutan)

9. 6 – 20 – 1



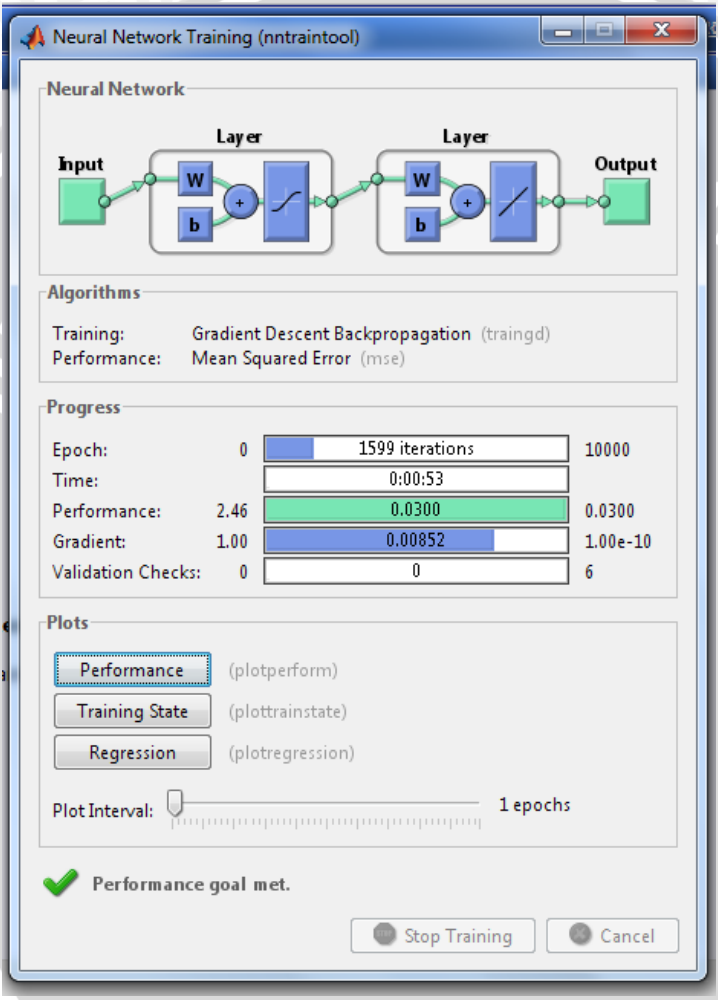
Lampiran 11 (lanjutan)

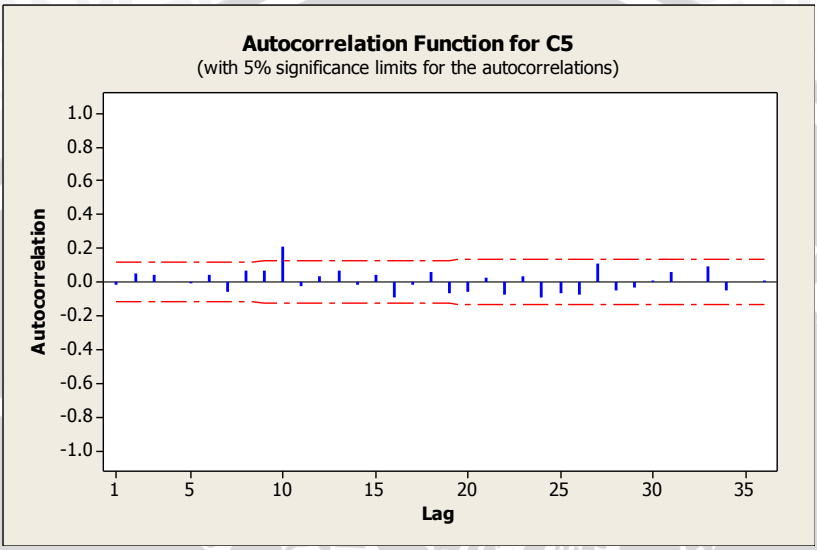


- Dilihat dari ACF sisaan, model yang dihasilkan arsitektur 6-20-1 tidak layak digunakan.

Lampiran 11 (lanjutan)

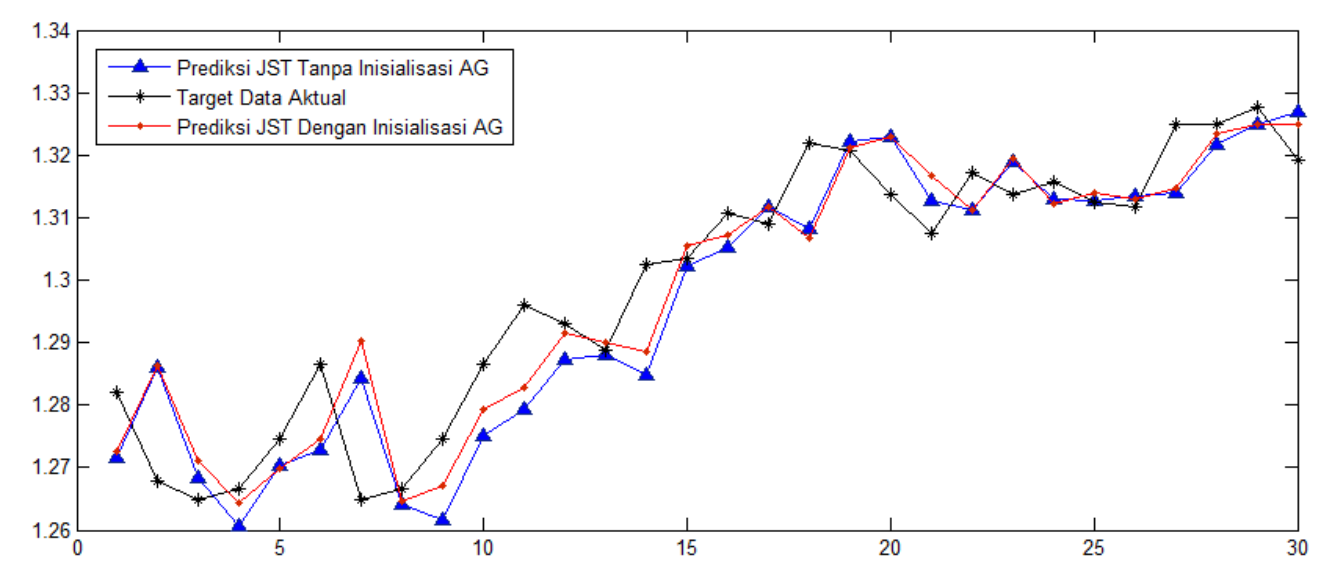
10. 6 – 25 – 1





- Dilihat dari ACF sisaan, model yang dihasilkan arsitektur 6-25-1 tidak layak digunakan.

Lampiran 12 Data hasil peramalan JST *backpropagation* tanpa inisialisasi dan JST *backpropagation* dengan inisialisasi algoritma genetika.



Lampiran 12 (lanjutan)

1. Perbandingan Nilai Tukar Aktual *(target)* Dengan Output Jaringan *(prediksi)* jaringan tanpa inisialisasi algoritma genetika

DATE	prediksi	target	error
Sunday, January 08, 2012	1.2719	1.2821	-0.0102
Monday, January 09, 2012	1.2855	1.2678	0.0177
Tuesday, January 10, 2012	1.271	1.2648	0.0062
Wednesday, January 11, 2012	1.2656	1.2666	-0.0010
Thursday, January 12, 2012	1.2713	1.2747	-0.0034
Friday, January 13, 2012	1.2756	1.2865	-0.0109
Sunday, January 15, 2012	1.2888	1.2648	0.0240
\Monday, January 16, 2012	1.2661	1.2666	-0.0005
Tuesday, January 17, 2012	1.2627	1.2747	-0.0120
Wednesday, January 18, 2012	1.2784	1.2865	-0.0081
Thursday, January 19, 2012	1.2835	1.2961	-0.0126
Friday, January 20, 2012	1.2959	1.2931	0.0028
Sunday, January 22, 2012	1.2912	1.2888	0.0024
Monday, January 23, 2012	1.2872	1.3024	-0.0152
Tuesday, January 24, 2012	1.302	1.3035	-0.0015
Wednesday, January 25, 2012	1.3059	1.3107	-0.0048
Thursday, January 26, 2012	1.313	1.3089	0.0041
Friday, January 27, 2012	1.3076	1.322	-0.0144
Sunday, January 29, 2012	1.3209	1.3207	0.0002
Monday, January 30, 2012	1.3231	1.3138	0.0093
Tuesday, January 31, 2012	1.3174	1.3076	0.0098
Wednesday, February 01, 2012	1.3094	1.3171	-0.0077
Thursday, February 02, 2012	1.3167	1.3136	0.0031
Friday, February 03, 2012	1.3132	1.3158	-0.0026
Sunday, February 05, 2012	1.3131	1.3124	0.0007

Monday, February 06, 2012	1.3135	1.3118	0.0017
Tuesday, February 07, 2012	1.3135	1.3248	-0.0113
Wednesday, February 08, 2012	1.3251	1.3248	0.0003
Thursday, February 09, 2012	1.3251	1.3276	-0.0025
Friday, February 10, 2012	1.3243	1.3192	0.0051

2. Perbandingan Nilai Tukar Aktual *(target)* Dengan Output Jaringan *(prediksi)* jaringan dengan inisialisasi algoritma genetika

DATE	prediksi	target	error
Sunday, January 08, 2012	1.2731	1.2821	-0.0090
Monday, January 09, 2012	1.2836	1.2678	0.0158
Tuesday, January 10, 2012	1.2717	1.2648	0.0069
Wednesday, January 11, 2012	1.2656	1.2666	-0.0010
Thursday, January 12, 2012	1.2731	1.2747	-0.0016
Friday, January 13, 2012	1.2779	1.2865	-0.0086
Sunday, January 15, 2012	1.2916	1.2648	0.0268
Monday, January 16, 2012	1.2669	1.2666	0.0003
Tuesday, January 17, 2012	1.2663	1.2747	-0.0084
Wednesday, January 18, 2012	1.2821	1.2865	-0.0044
Thursday, January 19, 2012	1.2916	1.2961	-0.0045
Friday, January 20, 2012	1.2963	1.2931	0.0032
Sunday, January 22, 2012	1.2893	1.2888	0.0005
Monday, January 23, 2012	1.2865	1.3024	-0.0159
Tuesday, January 24, 2012	1.3068	1.3035	0.0033
Wednesday, January 25, 2012	1.3065	1.3107	-0.0042
Thursday, January 26, 2012	1.3096	1.3089	0.0007
Friday, January 27, 2012	1.3089	1.322	-0.0131

Sunday, January 29, 2012	1.3224	1.3207	0.0017
Monday, January 30, 2012	1.3244	1.3138	0.0106
Tuesday, January 31, 2012	1.3153	1.3076	0.0077
Wednesday, February 01, 2012	1.3103	1.3171	-0.0068
Thursday, February 02, 2012	1.3169	1.3136	0.0033
Friday, February 03, 2012	1.3142	1.3158	-0.0016
Sunday, February 05, 2012	1.3133	1.3124	0.0009
Monday, February 06, 2012	1.3129	1.3118	0.0011
Tuesday, February 07, 2012	1.3141	1.3248	-0.0107
Wednesday, February 08, 2012	1.3216	1.3248	-0.0032
Thursday, February 09, 2012	1.3269	1.3276	-0.0007
Friday, February 10, 2012	1.327	1.3192	0.0078