

PERANCANGAN SISTEM PENGENDALIAN *LEVEL* DENGAN JARINGAN SYARAF TIRUAN PADA *STEAM DRUM BOILER 1102* DI PT. PETROKIMIA GRESIK

(Sofidul Aris, Ya'umar)

Jurusan Teknik Fisika
Fakultas Teknologi Industri
Institut Teknologi Sepuluh Nopember
Kampus ITS, Keputih – Sukolilo, Surabaya 60111

ABSTRAK

Pengendalian level yang selama ini digunakan pada steam drum adalah dengan pengendali PID. Pengendali konvensional ini dirasa kurang mampu memberikan performansi yang baik karena sifatnya yang linier padahal proses-proses pada industri biasanya nonlinear, multivariabel dan komplek. Diperlukan suatu pengendalian yang mampu menangani sifat kenonlinearan steam drum tersebut. Dengan kemampuan belajar terhadap berbagai perubahan kondisi yang tak terduga maka jaringan syaraf tiruan digunakan untuk merancang sistem pengendalian level dengan struktur model NNARX (Neural Network AutoRegresive with eXternal input), algoritma pembelajaran Levenberg Marquardt dan Direct Inverse Control berbasis jaringan syaraf tiruan. Penelitian dilakukan dengan memodelkan plant serta pengendali sehingga dihasilkan RMSE = 0.0019 untuk model plant dan RMSE = 0.0014 untuk model pengendali. Didapat performansi pengendalian DIC dan PID berbasis JST, level steam drum yakni; DIC menghasilkan $M_p = 2.24\%$, $t_s = 30$ sekon, dan pengendali PID menghasilkan $M_p = 2.26\%$, $t_s = 31$ sekon, pada set point 75 % (4.533meter) level steam drum.

Kata kunci: *Steam Drum, Pengendalian level, Jaringan Syaraf Tiruan, Direct Inverse Control*

BAB I. PENDAHULUAN

1.1 Latar Belakang

Steam drum merupakan salah satu komponen penting pada proses pembangkitan yang berfungsi untuk memisahkan uap antara fase gas dan fase cair kemudian melewati uap fase gas dengan tekanan yang tinggi. Uap bertekanan tinggi tersebut nantinya akan dipanaskan kembali sampai suhu tertentu agar menjadi uap kering. Variabel yang dikendalikan di dalam adalah *level* fluida

cair. *Level* fluida tersebut dijaga pada ketinggian yang telah ditentukan. Bila *level* fluida cair melebihi dari *level* yang ditentukan, maka fluida gas yang dihasilkan akan mengandung uap air sehingga akan membahayakan proses berikutnya selain itu juga dapat merusak pipa-pipa yang ada di dalam *boiler*. Sebaliknya bila *level* fluida cair kurang dari *level* yang ditentukan, maka fluida gas yang dihasilkan tidak akan maksimal untuk memutar turbin. *Level* ini sangat ditentukan oleh flow fluida cair yang masuk dan flow steam yang keluar. Pengendalian *level* yang selama ini digunakan pada *real plant* adalah dengan controller PID. Pengendali konvensional ini dirasa kurang mampu memberikan performansi yang baik karena sifatnya yang linier padahal proses-proses pada industri biasanya nonlinear, multivariabel dan komplek, sehingga pada Tugas Akhir ini akan dilakukan penelitian tentang pengendalian *level* dengan jaringan syaraf tiruan.. Setelah dilakukan pembelajaran, JST dapat menggantikan fungsi kontroler untuk menghasilkan *level* yang mendekati set point. Kemampuan JST yang dapat meniru dan mengotomatisasi kepakaran manusia ini menjadikannya paling tepat untuk menyelesaikan sistem-sistem yang nonlinear.

1.2 Rumusan Masalah

Adapun perumusan masalah yang diangkat pada pengerjaan Tugas Akhir ini yaitu:

1. Bagaimana melakukan identifikasi *plant* dan membuat kontroler dengan jaringan syaraf tiruan.
2. Bagaimana melakukan pengujian dan analisa jaringan syaraf tiruan sebagai sistem kontrol *level*

1.3 Batasan Masalah

Agar dapat lebih memusatkan penelitian tugas akhir ini, maka diberikan batasan masalah dalam pelaksanaannya nanti, yaitu :

1. Penelitian dilakukan pada di Boiler B-1102 (Daekyung Machinery (Korea)) PT. Petrokimia Gresik.
2. Pemodelan dengan JST hanya menggunakan data input output, bukan dengan model matematis.

3. Algoritma JST yang digunakan adalah *Levenberg Marquardt*.
4. *Software* yang digunakan untuk perancangan sistem kontrol adalah Matlab 7.7.0.(R2008b)

1.4 Tujuan

Tujuan dari Tugas Akhir ini adalah merancang sistem pengendalian *level* pada boiler di PT. Petrokimia Gresik dengan jaringan syaraf tiruan dan menganalisa hasil perancangan.

1.5 Manfaat

Manfaat dari Tugas Akhir ini adalah sistem kontrol yang dirancang bisa dipakai untuk memperbaiki sistem pengendalian *level* sebelumnya, sehingga dapat menjaga agar boiler bisa menghasilkan produk *steam* yang optimal.

1.6 Metodologi Penelitian

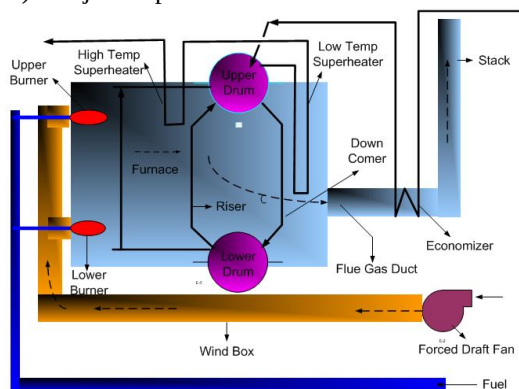
Metodologi yang digunakan dalam menyelesaikan penelitian ini adalah sebagai berikut:

1. Studi Literatur
2. Studi Lapangan dan Pengambilan Data
3. Perancangan dan Pembuatan Program
4. Analisis Hasil Pemrograman/Simulasi
5. Penyusunan Laporan Tugas Akhir.

BAB II. TEORI PENUNJANG

2.1 Boiler

Secara sederhana boiler adalah seperangkat alat konversi energi yang merubah energi panas menjadi energi tekanan dengan memanfaatkan perubahan wujud zat cair (fluida) menjadi uap

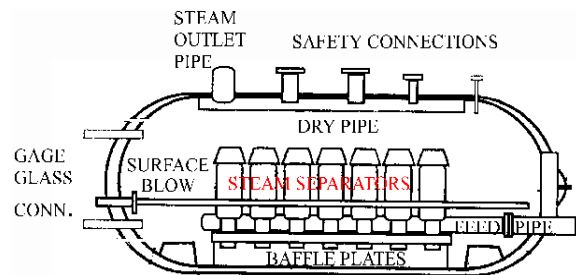


Gambar 2.1 Boiler b-1102 [petrokimia gresik]

Boiler B-1102 adalah salah satu boiler yang memproduksi steam sebagai penggerak turbin sebagai pembangkit listrik pada PT. Petrokimia Gresik. Untuk lebih jelasnya bisa dilihat pada gambar 2.1

2.2 Steam Drum

Fungsi dari steam drum adalah mengumpulkan dan memisahkan uap air, selain itu juga menyediakan dan mendistribusikan udara pada boiler ke evaporator. Gambar 2.2 adalah gambar penampang dari steam drum .



Gambar 2.2 [2]

Pada dasarnya komposisi dalam steam drum adalah sebagian air dan sebagian lagi uap. Masukan dari sistem feed water yaitu dibawah ketinggian air untuk menjaga gangguan dari permukaan air. Air meninggalkan drum menuju tabung-tabung *down comer* yang dalam hal ini adalah mud drum. Mud drum berfungsi mengumpulkan lumpur atau kotoran yang berasal dari sistem boiler yang menggunakan prinsip gravitasi. Karena posisi mud drum yang terletak di bawah sistem boiler. Air kemudian masuk dalam tabung-tabung riser di mana panas akan diberikan untuk mengubah air menjadi uap.

Level dalam *Steam Drum* harus dijaga untuk menghindari masuknya steam yang terlalu kering atau terlalu basah ke dalam turbin yang akan menimbulkan korosi dan dapat membahayakan proses selanjutnya.

2.3 Jaringan Syaraf Tiruan

Jaringan syaraf tiruan (JST) adalah suatu sistem pemrosesan informasi yang cara kerjanya memiliki kesamaan tertentu dengan jaringan syaraf biologis. Jaringan syaraf tiruan (JST) adalah salah satu algoritma pembelajaran mesin yang meniru cara kerja jaringan syaraf makhluk hidup. Jaringan syaraf tiruan (*artificial neural network*) merupakan jaringan dari banyak unit pemroses kecil (disebut neuron) yang masing-masing melakukan proses sederhana, yang ketika digabungkan akan menghasilkan perilaku yang kompleks. Jaringan syaraf tiruan dapat digunakan sebagai alat untuk memodelkan hubungan yang kompleks antara masukan (*input*) dan keluaran (*output*) pada sebuah sistem untuk menemukan pola-pola pada data.

Algoritma pembelajaran yang digunakan dalam penelitian ini adalah algoritma *Levenberg Marquardt*. Algoritma *Levenberg Marquardt* memiliki kelebihan karakteristiknya lebih cepat konvergen (*rapid convergence*), namun demikian algoritma *Levenberg Marquardt* membutuhkan penurunan yang lebih rumit dibanding algoritma *backpropagation*.

Data *training* adalah satu set *input* $u(k)$ yang berpasangan dengan *output* yang diinginkan $y(k)$ atau dapat ditulis :

$$Z^N = \{[u(k), y(k)] | k = 1, \dots, N\} \quad \dots(1)$$

Tujuan pembelajaran ini menentukan bobot yang mungkin dari pasangan data yang diberikan :

$$Z^N \rightarrow w \quad \dots(2)$$

Sehingga jaringan akan mengeluarkan perkiraan *output* $\hat{y}(k)$ yang sama atau mendekati *output* $y(k)$. Perkiraan *error* akan didekati dengan *mean square error criterion*:

$$V_N(w, Z^N) = L^1(w) - \frac{1}{2N} \sum [y(k) - \hat{y}(k|w)]^T [y(k) - \hat{y}(k|w)] \quad \dots(3)$$

Bobot yang diperoleh :

$$w = \arg \min_w V_N(w, Z^N) \quad \dots(4)$$

$$w^{(i+1)} = w^{(i)} + \mu^{(i)} f^{(i)} \quad \dots(5)$$

Dengan fariabel $w^{(i)}$ adalah bobot saat ini, $f^{(i)}$ adalah arah pencarian dan $\mu^{(i)}$ adalah besar langkah (*step size*).

Levenberg Marquardt adalah metoda standar untuk meminimalisasi *mean square error criterion*. Pada algoritma ini mempunyai parameter λ untuk menjaga konvergensi. Harga λ dikendalikan dengan rasio antara penurunan harga aktual dan harga prediksi.

2.4 Direct Inverse Control

Merupakan konsep yang paling dasar dari kontrol yang berdasarkan jaringan syaraf yang menggunakan *inverse* sebagai kontrol proses. Konsep yang paling dasar itu disebut *direct inverse control*. Prinsip dari proses ini dapat dideskripsikan sebagai berikut:

$$y(t+1) = g[y(t), \dots, y(t-n+1), u(t), \dots, u(t-m)] \quad \dots(6)$$

Jaringan yang digunakan untuk melatih proses *inverse* adalah:

$$\hat{u}(k) = \hat{g}^{-1}[y(t+1), y(t), \dots, y(t-n+1), u(t), \dots, u(t-m)] \quad \dots(7)$$

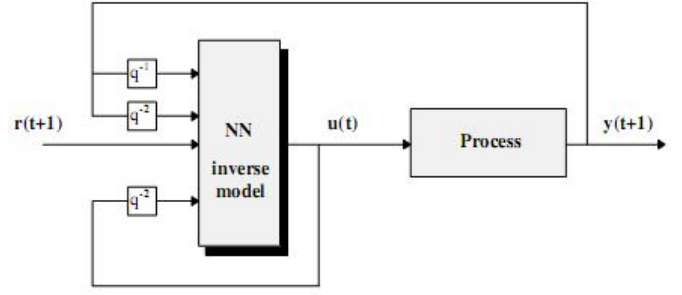
Kemudian model *inverse* diaplikasikan sebagai kontrol untuk suatu proses dengan memasukkan *output* yang diinginkan. Sebelum sistem kontrol aktual bekerja maka model *inverse* harus dilatih. Metode pembelajaran untuk kontrol berbasis jaringan syaraf tiruan dapat dibedakan menjadi 2 metode, yaitu:

1. Generalized Training

Pada metode ini jaringan syaraf tiruan *training* secara *offline* untuk meminimalisasi *mean square error* (MSE) di sinyal kontrol yang akan diterapkan pada proses melalui percobaan sinyal kontrol yang dihasilkan dari jaringan.

2. Specialized Training

Metode ini berfungsi untuk meminimalisasi nilai *mean square error* (MSE) antara sinyal referensi dan *output* dari proses. Metode ini telah berjalan dengan baik secara *online* dengan menggunakan *recursive training* algoritma.



Gambar 2.3 Direct Inverse Control.

BAB III. PERANCANGAN DAN METODOLOGI

3.1 Pemodelan Plant dengan JST

Pada penelitian ini data *input* yang digunakan untuk pemodelan plant adalah *flow Boiler Feed Water* dan *flow Steam* yang dihasilkan, sedangkan data *output*-nya adalah *level*.

Adapun proses pemodelannya ada dua tahap, yaitu *training* dan validasi. Arsitektur JST yang dipakai adalah *Multi Layer Perceptron*, yang terdiri dari *input layer*, *hidden layer*, dan *output layer*. Fungsi aktivasi pada *hidden layer* menggunakan *tangent hyperbolic* sedangkan pada *output layer* menggunakan fungsi aktivasi *linier*. Algoritma JST yang digunakan adalah *Levenberg Marquardt*. Struktur model JST yang dipakai adalah NNARX (*Neural Network AutoRegresive with eXternal Input*).

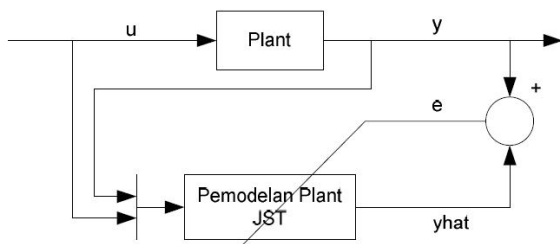
Gambar 3.1 menunjukkan bahwa tidak ada *feedback* pada proses pemodelan. Variabel “u” merupakan data input *plant*, sedangkan “y” merupakan data output *plant*. Data-data tersebut terlebih dahulu dinormalisasi untuk kemudian dimodelkan dengan JST. Kedua data ini dilatih dengan JST dan kemudian divalidasi, sehingga menghasilkan data output JST (variabel “yhat”). *Output* JST dibandingkan dengan output *plant* sehingga menghasilkan error “e”. Nilai error ini yang dipakai untuk meng-update nilai bobot JST secara terus menerus atau biasa disebut dengan proses iterasi. Proses iterasi dilakukan terus sampai didapatkan nilai output model JST paling bagus. Kriteria yang digunakan untuk menilai output model JST adalah *Root Mean Square Error* (RMSE) dan *Variance Accounted For* (VAF). RMSE adalah akar rata-rata total kuadrat *error* yang terjadi antara output model dan output proses. Output model *plant* akan semakin baik apabila mempunyai nilai RMSE yang kecil (mendekati nol), sebaliknya output dinilai jelek jika mempunyai nilai RMSE yang besar. Sedangkan kriteria berdasarkan VAF adalah semakin besar nilai VAF (mendekati nilai 100) maka semakin besar tingkat keberhasilan *training*. Penentuan nilai RMSE digunakan persamaan berikut:

$$RMSE = \sqrt{\frac{\sum_{i=1}^N (y_i - \hat{y}_i)^2}{N}} \quad \dots(8)$$

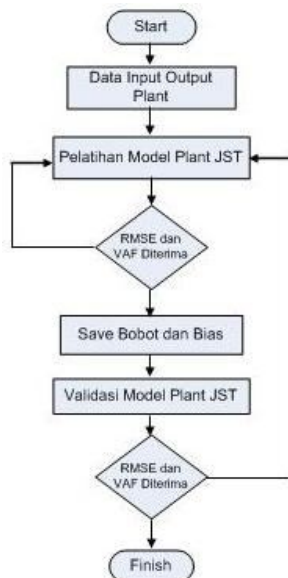
Dengan y_i adalah output proses, $\hat{y}_i$ adalah output model, dan N adalah jumlah data. Sedangkan untuk persamaan VAF adalah sebagai berikut :

$$VAF = \left\{ 1 - \frac{\text{var}[y(t) - \hat{y}(t)]}{\text{var}[y(t)]} \right\} \times 100\% \quad \dots(9)$$

Software untuk proses pemodelan dirancang supaya proses iterasi berhenti pada suatu nilai error minimal dengan memenuhi kriteria RMSE dan VAF. Bobot pada iterasi terakhir disimpan dan akan dipakai pada proses validasi. Data yang dipakai pada proses validasi berbeda dengan data yang dipakai pada proses *training*. Tujuannya untuk melihat keakurasian model yang dipakai pada proses *training*. Apabila *training* sudah menghasilkan nilai RMSE dan VAF yang bagus, tetapi setelah bobot tersebut digunakan pada proses validasi dan menghasilkan nilai RMSE dan VAF yang belum memenuhi target yang ditentukan, maka kita harus mengulang proses *training* dengan merubah-ubah parameternya sampai menghasilkan bobot yang paling baik untuk digunakan dalam proses *training* maupun validasi.



Gambar 3.1 Blok Diagram Pemodelan *Plant* JST



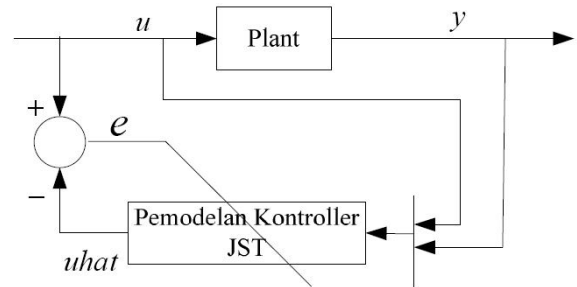
Gambar 3.2 Flowchart Pemodelan *Plant*

3.2 Pemodelan Kontroler dengan JST

Pada pemodelan kontroler, data input output yang dipakai adalah kebalikan dari pemodelan *plant*. Data input yang dipakai adalah *level steam drum* (%) sebagai data

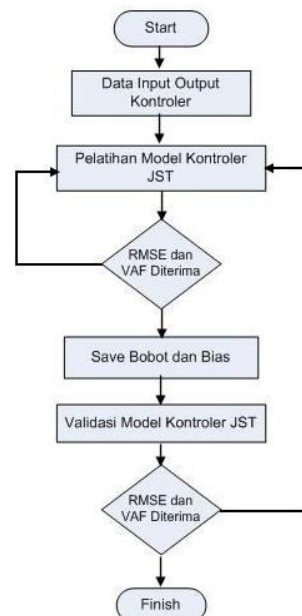
input, sedangkan *flow feed water* (ton/hr) dan *flow steam* (ton/hr) sebagai data *output*.

Pada gambar 3.3 terlihat bahwa pemodelan kontroler JST menghasilkan output yang ditunjukkan dengan variabel "*uhat*". Nilai output model kontroler "*uhat*" dibandingkan dengan input *plant* yang ditunjukkan dengan variabel "*u*", sehingga diperoleh suatu nilai error "*e*". Selanjutnya langkah pemodelan kontroler sama dengan pemodelan *plant*.



Gambar 3.3 Blok Diagram Pemodelan Kontroler JST

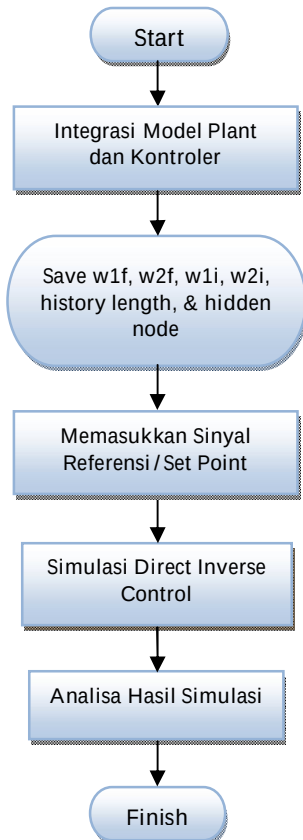
Pertama adalah *training* untuk mendapatkan nilai output model yang bagus sesuai kriteria RMSE dan VAF kemudian menyimpan bobotnya. Kedua adalah proses validasi dengan memakai bobot yang telah disimpan dari proses *training* dan memakai struktur jaringan yang sama dengan proses *training*. Setelah dilakukan iterasi berulang-ulang pada akhirnya diperoleh bobot baru. Bobot dari pemodelan kontroler ini dipakai sebagai *inverse*, sedangkan bobot dari pemodelan *plant* dipakai sebagai *forward*. Kedua bobot tersebut selanjutnya dipakai pada simulasi *direct inverse control*. Alur pemodelan kontroler jaringan syaraf tiruan ditunjukkan oleh flowchart dibawah ini:



Gambar 3.4 Flowchart Pemodelan Kontroler

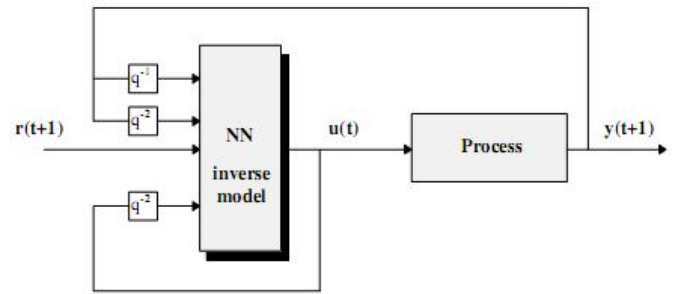
3.3 Direct Inverse Control

Direct inverse control merupakan solusi paling sederhana yang dapat diterapkan pada sistem kontrol berbasis jaringan syaraf tiruan. Jika didiskripsikan, *Direct Inverse Control* merupakan kontrol yang dihubungkan secara seri antara model *inverse* (model kontroler JST) dengan model *forward* (model *plant* JST). Gambar 7 menunjukkan langkah-langkah yang digunakan dalam simulasi *direct inverse control*.



Gambar 3.5 Flowchart Simulasi *Direct Inverse Control*

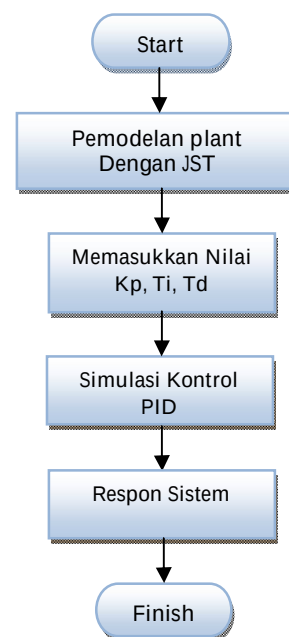
Direct inverse control berbasis JST adalah kontrol yang dihubungkan secara seri antara model kontroler JST dengan model *plant* JST, seperti yang ditunjukkan pada gambar 7. Dimana model kontroler JST digunakan sebagai controller. Dari nilai bobot, jumlah *history length* dan *hidden node* hasil *training* pemodelan *plant* JST dan *training* pemodelan kontroler JST digunakan untuk simulasi *direct inverse control* berbasis JST. Gambar 2.8 menunjukkan alur simulasi *direct inverse control* berbasis JST. Untuk mengetahui tingkat keberhasilan hasil simulasi *direct inverse control* berbasis JST ini dilakukan uji tracking setpoint dengan parameter nilai maksimum overshoot (Mp) dan nilai settling time (ts) dari respon kontrol yang di hasilkan. Maksimum overshoot (Mp) adalah nilai puncak kurva tanggapan diukur dari satuan. Sedangkan waktu turun/settling time (ts) adalah waktu yang diperlukan untuk menanggapi kurva agar dapat mencapai dan tetap berada dalam gugus akhir ukuran yang disederhanakan dengan presentase mutlak harga akhirnya (biasanya 2% atau 5%).



Gambar 3.6 Blok Diagram Direct Inverse Control

3.4 Kontrol PID

Pengendalian *level* pada *Steam drum* yang dipakai di *plant* menggunakan kontrol PID (*Proportional Integral Derivatif*), maka dari itu perlu dilakukan simulasi kontrol PID tersebut. Simulasi kontrol PID dilakukan dengan Jaringan Syaraf Tiruan dengan memasukkan parameter kontrol K_p , T_i , dan T_d ke dalam program *Direct Inverse Control*. Untuk mengetahui tingkat keberhasilan hasil simulasi kontrol PID *real plant* ini dilakukan uji *tracking set point* dengan parameter nilai *maximum overshoot* (Mp) dan nilai *settling time* (ts) dari respon kontrol yang dihasilkan (sama seperti uji *set point* pada analisis respon DIC). *Maximum overshoot* (Mp) adalah nilai puncak kurva tanggapan diukur dari satuan. Sedangkan waktu turun/*settling time* (ts) adalah waktu yang diperlukan untuk menanggapi kurva agar dapat mencapai dan tetap berada dalam gugus akhir ukuran yang disederhanakan dengan presentase mutlak harga akhirnya. Gambar 2.9 menunjukkan diagram alir dari simulasi sistem pengendalian dengan PID.

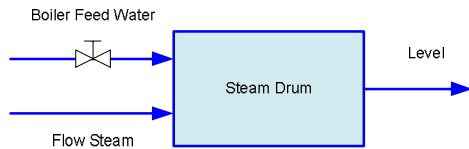


Gambar 3.7 Flowchart Simulasi Kontrol PID

BAB IV. SIMULASI DAN ANALISIS DATA

4.1 Data Input-Output Plant steam drum

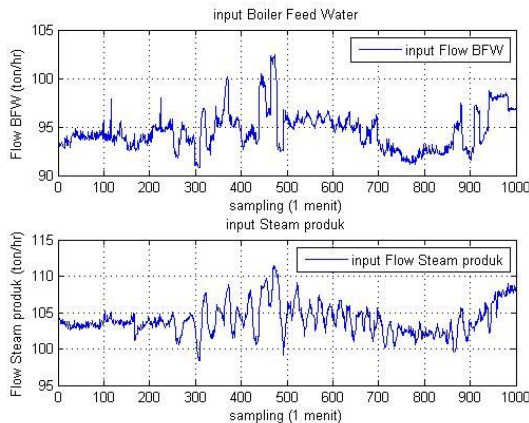
Pada proses pengendalian *level steam* drum di Boiler B-1102 PT. Petrokimia Gresik terdapat variabel *input-output* yang berperan di dalamnya. Data tersebut digunakan untuk simulasi sistem kontrol. Data *input* berupa *flow feed water* (ton/hr) dan *flow steam* (ton/hr), sedangkan data *output* adalah *level* pada *steam drum* (%).



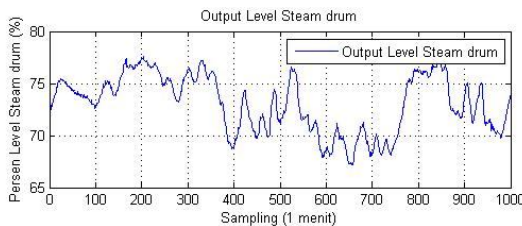
Gambar 4.1 Input-output plant level pada steam drum

Data input dan data output diambil dari PHD (Process Historical Data) yang diambil dari DCS (Distributed Control System). Data tersebut merupakan data *real plant* yang diambil setiap 1 menit selama 2 hari, jumlah data yang ada masing-masing adalah 1000. Selama selang waktu tersebut *plant* berjalan normal dalam artian tidak ada kegiatan *maintenance* terhadap *plant* yang menyebabkan *plant shutdown*.

Gambar 4.2 menunjukkan grafik dari *input plant* yaitu *flow feed water* (ton/hr) dan *flow steam* (ton/hr), sedangkan gambar 4.3 menunjukkan *output* adalah *level* pada *Steam Drum* (%). Masing-masing data berjumlah 1000 data, dengan 750 data dipakai untuk proses *training*, sedangkan yang 250 data dipakai untuk proses validasi pada setiap pemodelan *plant* maupun pemodelan kontroler JST.



Gambar 4.2 Grafik Profil Data Input Plant



Gambar 4.3 Grafik Profil Data Output Plant

4.2 Pemodelan Plant dengan Jaringan Syaraf Tiruan

Pemodelan *plant* dilakukan dengan dua tahap, yaitu *training* dan validasi. Sebelum pemodelan harus dilakukan normalisasi data, normalisasi data yaitu kegiatan mengolah data dengan cara *men-scaling* data yang akan dipakai sehingga data yang dipakai nantinya memiliki *range* dari 0 (nol) sampai 1 (satu). Tujuan dari normalisasi data ini adalah supaya program Jaringan Syaraf Tiruan (JST) bisa berjalan dengan cepat dalam melakukan eksekusi data baik tahap *training*, tahap validasi, ataupun tahap simulasi kontrol. Data yang dipakai untuk proses pemodelan dan perancangan kontrol berjumlah 750 data. Data diperoleh secara langsung dari *plant* berupa data *real plant*. Setelah dilakukan beberapa percobaan maka diambil 750 data untuk *training* dan 250 data untuk validasi. Dalam pemodelan JST, baik tahap *training* ataupun validasi, data *real plant* tersebut merupakan target dari model JST.

Arsitektur JST yang dipakai dalam pemodelan adalah MLP (*Multy Layer Perceptron*), artinya ada *input layer*, *hidden layer*, dan *output layer*. *Input layer* ada dua *node*, yaitu *node input* pertama berupa nilai flow BFW dari (*Boiler Feed Water*) dalam Ton/hr dan *node input* kedua berupa nilai *flow steam* dalam Ton/hr. *Hidden layer* ada dua *hidden node* yang masing-masing memakai fungsi aktivasi *tangent hyperbolic*. *Output layer* ada satu *node* berupa nilai *level* pada *steam drum* (%). *Output layer* memakai fungsi aktivasi *linier*. Algoritma JST yang dipakai adalah *Lavenberg Marquardt*. Sedangkan struktur model JST adalah NNARX (*Neural Network AutoRegressive with eXternal input*). Variabel-variabel yang dipakai dalam pemodelan JST antara lain:

- u , untuk variabel *input*
- y , untuk variabel *output*
- $w1f$, untuk variabel bobot dari *input layer* ke *hidden layer*
- $w2f$, untuk variabel bobot dari *hidden layer* ke *output layer*.

Persamaan *output* model JST adalah sebagai berikut:

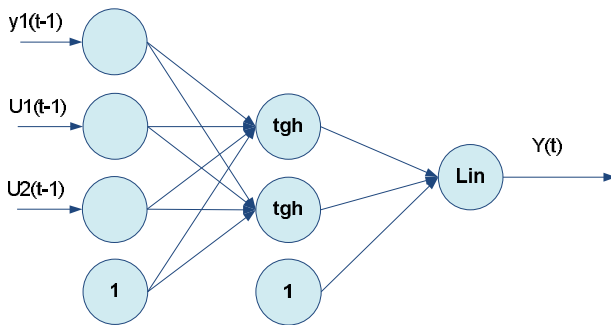
$$\hat{y} = f \left(\sum_{j=1}^n w_{i,j} u_j + w_{i,0} \right) \quad \dots (4.1)$$

Dengan:

- $\hat{y}$: *output* model JST
- f : fungsi aktivasi
- $w_{i,j}$: nilai bobot
- u_j : nilai *input*
- $w_{i,0}$: bias

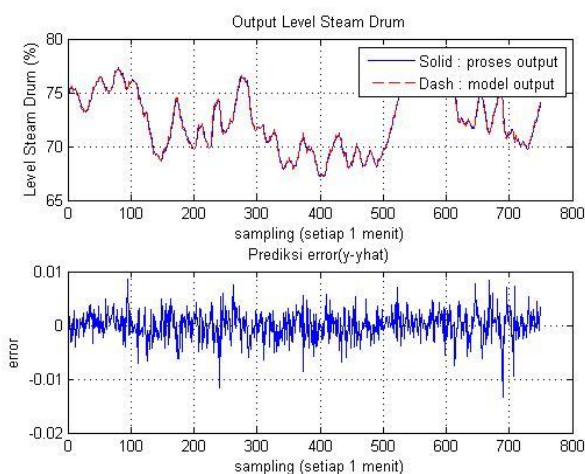
Tahap pertama dari pemodelan *plant* JST adalah *training*. Untuk mendapatkan model JST terbaik maka dilakukan eksperimen dengan cara mengubah-ubah parameter JST seperti jumlah *hidden node* dan *history length*. Eksperimen yang dilakukan adalah mengubah *hidden node* dari 1 sampai 20 *hidden node* dengan *history length* dari 1 sampai 5, dengan kolaborasi masing-masing

pada setiap kali eksperimen. Kriteria yang dipilih dari model JST adalah nilai RMSE (*Root Mean Square Error*) paling mendekati nol dan nilai VAF (*Variance Accounted For*) paling mendekati 100. Model JST yang disimpan (*forward*) untuk dipakai dalam simulasi kontrol adalah dengan *history length* 1 dan *hidden node* 2. Model tersebut memiliki nilai RMSE sebesar 0.0019 dan nilai VAF sebesar 99.7204. Gambar 4.4 menunjukkan model JST tahap pemodelan *plant* dengan dua *input node*, dua *hidden node*, dan satu *output node* dengan *input bias* pada masing-masing *hidden node* dan *output node*.

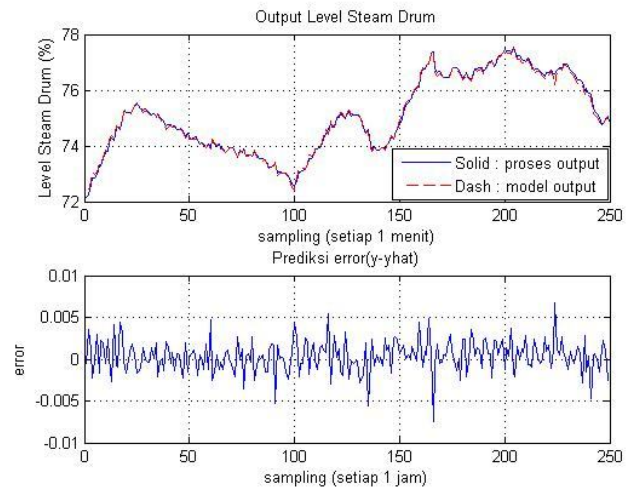


Gambar 4.4 Model JST Tahap Pemodelan *Plant*

Dari tahapan training akan dihasilkan bobot (w_{1f} dan w_{2f}) yang kemudian disimpan dalam *file forward*. Proses validasi menggunakan struktur JST yang digunakan pada tahap training dengan menggunakan bobot (w_{1f} dan w_{2f}) dalam *file forward*, pasangan data yang digunakan adalah 250 data lain yang tidak digunakan pada proses training. Tujuan dari validasi adalah untuk mengetahui berapa keakuratan *output model plant* JST yang telah dibangun dengan *output plant* sebenarnya. Adapun nilai RMSE (*Root Mean Square Error*) sebesar 0.0010 dan VAF (*Variance Accounted For*) sebesar 99.7056.



Gambar 4.5 Grafik hasil *training* pemodelan *plant* JST dan nilai *error*



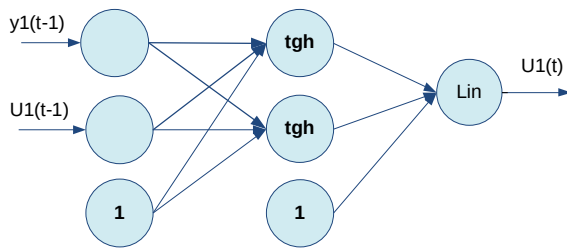
Gambar 4.6 Grafik Hasil Validasi JST dan Nilai *Error*

4.3 Pemodelan Kontroler dengan Jaringan Syaraf Tiruan

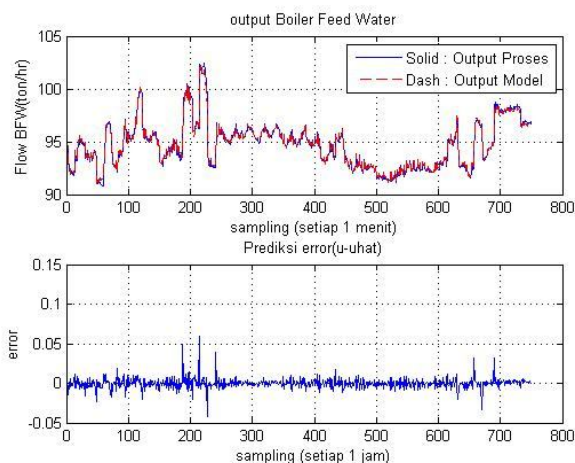
Hampir sama seperti pada proses pemodelan *plant*, pemodelan *kontroller* dengan JST memiliki langkah-langkah yang sama, hanya saja yang membedakan adalah struktur JST yang digunakan dan pasangan data *input-output*nya. Struktur JST pada pemodelan *kontroller* merupakan kebalikan dari struktur JST yang digunakan dalam pemodelan *plant* dan pasangan data yang digunakan adalah *level steam drum* (%) sebagai data *input*, sedangkan *flow feed water* (ton/hr) dan *flow steam* (ton/hr) sebagai data *output*.

Tahap pertama *training*, yaitu untuk mencari struktur JST untuk pemodelan *kontroller* yang paling baik, maka dilakukan eksperimen dengan merubah nilai *history length* dan jumlah *hidden node* untuk mendapatkan nilai RMSE yang paling kecil (paling mendekati nol) dan nilai VAF yang paling besar (paling mendekati 100). Eksperimen yang telah dilakukan adalah mengubah *hidden node* dari 1 sampai 20 *hidden node* dengan *history length* dari 1 sampai 5. Pada akhirnya dipilih model JST *kontroller* dengan *history length* 1 dan *hidden node* 2 karena dianggap model tersebut merupakan model paling sesuai dengan kriteria diatas, seperti terlihat pada Gambar 4.7. Gambar tersebut adalah arsitektur JST yang paling baik untuk digunakan dalam pemodelan *kontroller*. Terlihat bahwa arsitektur JST itu memiliki 2 *node input* dan 1 *node output* dengan 2 *hidden node*. Fungsi aktivasi yang dipakai dari *input layer* ke *hidden layer* adalah *tangent hyperbolic*, sedangkan dari *hidden layer* ke *output layer* adalah fungsi aktivasi *linier*.

Model JST *kontroller* tersebut menghasilkan RMSE sebesar 0,0014 dan VAF sebesar 99.6019 untuk *output BFW*. Bobot (w_{1i} dan w_{2i}) kemudian disimpan dalam *file inverse* untuk kemudian dilakukan validasi.

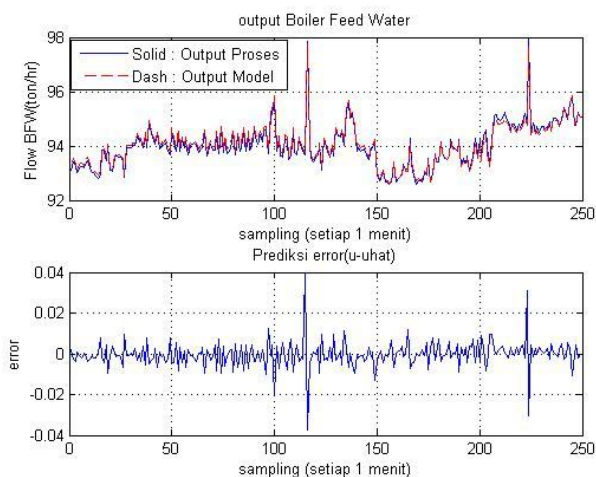


Gambar 4.7 Model JST untuk Pemodelan Kontroler



Gambar 4.8 Grafik BFW hasil training pemodelan kontroller JST dan nilai error

Pada proses *validasi* pasangan data yang digunakan adalah 250 data yang tidak digunakan pada proses *training*. Pada proses *validasi* dihasilkan nilai nilai RMSE sebesar 0.0016 dan VAF sebesar 96.0779 untuk BFW .

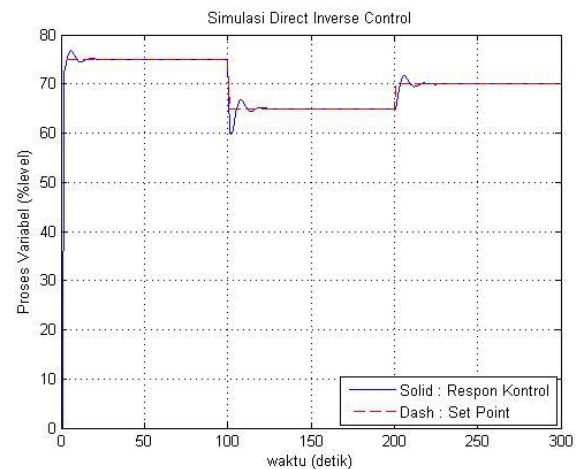


Gambar 4.9 Grafik BFW hasil validasi pemodelan kontroller JST dan nilai error

4.4 Simulasi *Direct Inverse Control*

Dari pemodelan *plant* dan pemodelan pengendali dengan JST yang telah dilakukan sebelumnya, didapatkan bobot dari pemodelan *plant* yaitu $W1f$ dan $W2f$ yang disimpan dalam file *forward* sedangkan bobot dari pemodelan pengendali yaitu $W1i$ dan $W2i$ yang disimpan dalam file *inverse*. Kedua jenis bobot disimpan beserta jumlah *hidden node* dan *history length* model JST. Selanjutnya simulasi DIC dilakukan dengan cara *load w1f, w2f, w1i, w2i, history length, dan hidden node* dari masing-masing model yang telah disimpan.

Untuk menguji DIC dilakukan uji respon DIC dengan cara *tracking set point*, yaitu memberi nilai *set point* yang berbeda pada sistem pengendali. Simulasi dilakukan dengan mengubah nilai *set point* sebanyak tiga kali, yaitu pada ketinggian level 75%, 65% dan 70% . Untuk menganalisa grafik respon sistem pengendali maka dihitung nilai *Maximum overshoot* (M_p), *Rise time* (t_r), *Delay time* (t_d), *Peak time* (t_p), dan nilai *Settling time* (t_s). Gambar 4.22 berikut ini adalah grafik respon sistem *Direct Inverse Control*. Garis merah merupakan nilai *set point* sedangkan respon berwarna biru adalah grafik respon sistem *Direct Inverse Control*. Nilai *set point* diberikan naik dan diturunkan lagi (tidak *linier*) bertujuan untuk melihat sistem pengendali bisa merespon pada nilai *set point* yang berubah-ubah.



Gambar 4.10 Respon Sistem *Direct Inverse Control*

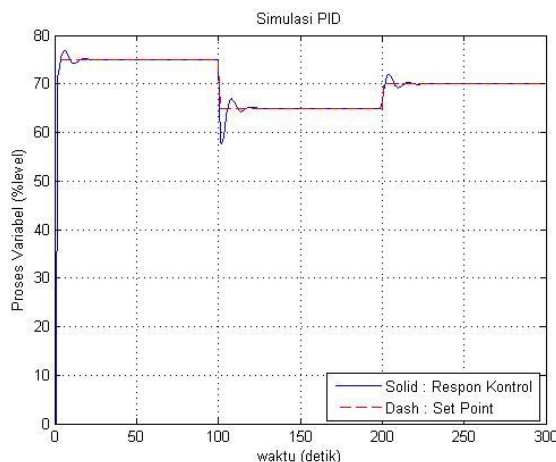
Maximum overshoot adalah nilai deviasi maksimum yang terjadi pada respon yang diukur dari keadaan *steady*. *Delay time* adalah waktu yang diperlukan oleh respon untuk mencapai setengah dari nilai akhir waktu pertama. *Rise time* adalah waktu yang diperlukan respon untuk naik dari keadaan 10% sampai 90%. *Peak time* adalah waktu yang diperlukan respon mencapai puncak pertama. *Time constant* adalah waktu yang diperlukan respon untuk mencapai 63% keadaan *steady*. Sedangkan *settling time* adalah waktu yang diperlukan respon untuk mencapai keadaan *steady* dengan nilai sebesar 2% atau 5%. Berikut ini adalah tabel karakteristik respon sistem kontrol pada saat dilakukan *tracking set point*:

Tabel 4.1 Karakteristik Respon Sistem *Direct Inverse Control*

Karakteristik Respon Sistem	Set Point		
	75% (4,533m)	65% (3,868m)	70% (4,230m)
Maximum Overshoot (%)	2.24	7.6	2.23
Settling Time (detik)	30	29	28
Peak Time (detik)	5	7	6
Rise Time (detik)	3.9	4	4
Delay Time (detik)	2.4	2.0	2.1

4.5 Simulasi Kontrol PID

Simulasi pengendali PID dilakukan menggunakan jaringan syaraf tiruan dengan memasukkan parameter pengendali PID ke dalam program Matlab. Parameter pengendali PID *real plant* ada $K_p = 0.8$; $T_i = 0.5$ detik; dan $T_d = 0$ detik. Untuk mengetahui respon sistem dilakukan pengujian yang sama dengan pengujian pada model *Direct Inverse Control*, yaitu dengan melakukan *tracking set point* (uji step). Pertama diberi *set point* 75% kemudian 65% dan yang terakhir 70%. Pada gambar 4.21 menunjukkan grafik respon sistem pengendali PID dengan sinyal uji step. Garis merah merupakan nilai *set point*. Nilai *set point* diberikan naik dan diturunkan lagi (step) bertujuan untuk melihat respon dari sistem pengendali pada nilai *set point* yang berubah-ubah.



Gambar 4.9 Respon Sistem Kontrol PID

Tabel 4.2 Karakteristik Respon Sistem Kontrol PID

Karakteristik Respon Sistem	Set Point (level)		
	75% (4,533m)	65% (3,868m)	70% (4,230m)
Maximum Overshoot (%)	2.26	11.71	2.25
Settling Time (detik)	31	30	29
Peak Time (detik)	6	8	6
Rise Time (detik)	4.1	4.2	4
Delay Time (detik)	2,5	2,2	2,1

4.6 Analisa Perbandingan Performansi Pengendali PID dan DIC

Dari Pengujian *tracking set point* (uji step) kedua model pengendali dalam hal ini DIC (*Direct Inverse Control*) dan pengendali PID, terlihat bahwa kedua pengendali mampu untuk mengikuti set point dengan baik. Akan tetapi seperti yang terlihat pada gambar 4.23, respon DIC yang diwakili oleh warna merah, menunjukkan respon sistem yang lebih baik dari pada respon pengendali PID yang diwakili oleh warna hijau. Perbedaan yang paling mencolok terlihat pada *tracking set-point* level 65% (3.868 meter), yaitu *tracking* dimana *set-point* tiba-tiba diturunkan dari harga awal yaitu level 75% (4.533 meter). *Maximum Overshoot* respon pengendali PID lebih besar dari pada respon DIC, yaitu sebesar 11.71% sedangkan DIC sendiri sebesar 7.6%. Dalam mencapai kondisi *steady state* DIC juga lebih cepat dari pada pengendali PID walaupun tidak terlalu signifikan perbedaannya.

V. KESIMPULAN DAN SARAN

5.1 Kesimpulan

Dari serangkaian metodologi, pengujian, analisa serta pembahasan yang telah dilakukan didapat beberapa kesimpulan diantaranya:

1. Model JST terbaik pada pemodelan *plant* didapat saat *history length* 1 dan jumlah *hidden node* 2 dengan menghasilkan $RMSE = 0.0019$ dan $VAF = 99.7204$ dan model JST terbaik pada pemodelan pengendali didapat saat *history length* 1 dan *hidden node* 2 dengan menghasilkan $RMSE = 0.0014$ dan $VAF = 99.6019$.
2. Respon sistem pengendali dengan simulasi *Direct Inverse Control* dapat mengikuti *set-point* yakni; DIC menghasilkan $M_p = 2.24\%$, $t_s = 30$ detik, dan pengendali PID menghasilkan $M_p = 2.26\%$, $t_s = 31$ detik, pada *set-point* pengendalian level 75% (4.533 meter).
3. Hasil pengujian performansi pengendali PID dan DIC menunjukkan bahwa kedua pengendali telah mampu memberikan hasil respon yang baik. Namun DIC sedikit lebih unggul dibandingkan pengendali PID

5.2 Saran

Saran yang dapat disampaikan adalah *software* pengendali jaringan syaraf tiruan bisa diaplikasikan secara *real plant*, tentu saja demi mendapatkan hasil keluaran yang lebih baik dan stabil sehingga mampu meningkatkan efektivitas dari proses di dalam *steam drum* boiler B-1102 sehingga menghasilkan kualitas *steam* yang lebih baik.

DAFTAR PUSTAKA

- Atieq, Nurul. Ya'umar. 2009. **Perancangan Sistem Pengendalian Temperatur pada *Additional Firing* dengan Metode Jaringan Syaraf Tiruan di *Waste Heat Boiler* PT. Petrokimia Gresik.** Surabaya: Tugas Akhir Jurusan Teknik Fisika. ITS.
- Norgaard, Magnus. 2000. **Neural Network Based Control System Design TOOLKIT Version 2.0.** Denmark: Department of Automation. Department of Mathematical Modelling. Technical University of Denmark.
- Norgaard, Magnus. 2000. **Neural Network Based System Identification TOOLBOX.** Denmark: Department of Automation. Department of Mathematical Modelling. Technical University of Denmark.
- Siang, Jong Jek. 2005. **Jaringan Syaraf Tiruan & Pemrogramannya Menggunakan Matlab.** Yogyakarta: ANDI.
- Syahir Roshihan. Ya'umar. 2008. **Identifikasi Proses Menggunakan Jaringan Syaraf Tiruan pada Steam Drum di PT. Petrokimia Gresik.** Surabaya: Tugas Akhir Jurusan Teknik Fisika. ITS.
- www.mathworks.com

BIODATA PENULIS



Nama : Sofidul Aris
TTL : Sidoarjo, 11 Desember 1986
Alamat : Kauman KedungCangkring no.9
01/01 Jabon Sidoarjo
E-mail : aris@ep.its.ac.id
vastrolorde.00@gmail.com
Riwayat Pendidikan :
2004 – sekarang : Teknik Fisika ITS, Surabaya
2001 – 2004 : SMA Negeri 3 Sidoarjo
1998 – 2001 : SMP Negeri 1 Porong
1992 – 1998 : MI Nahdhotul Ulama