Pengaturan Motor Induksi Menggunakan *Observer Self Constructing Fuzzy Neural Network* dengan Metode Algoritma Pelatihan *Levenberg Marquardt*

Suhariningsih[1,2)], **Soebagio**[2)], **Mauridhi Heri Purnomo**[2)]
[1)]Politeknik Elektronika Negeri Surabaya- ITS
[2)]Jurusan Teknik Elektro FTI, Institut Teknologi Sepuluh Nopember Surabaya
Kampus ITS Keputih Sukolilo Surabaya 60111
E-mail : nuning@eepis-its.edu

ABSTRAK

Dalam penelitian ini dikembangkan pengaturan kecepatan motor induksi 3 *phase* tanpa sensor yang dioperasikan dengan metoda *Field Oriented Vector* (FOC). Kecepatan motor diestimasi oleh suatu *observer* dengan suatu metoda *Self Constructing Fuzzy Neural Network* (SCFNN) dimana pelatihannya menggunakan metode algoritma pelatihan *Levenberg Marquardt* (LM), yang menggantikan metode Backpropagasi karena metode ini kurang cepat mencapai konvergen. Metode SCFNN mempunyai kemampuan untuk menggabungkan *Fuzzy* dan *Neural Networks*. Hasil simulasi menunjukkan sistem dapat mengestimasi fluksi dan kecepatan dengan kekonvergenan yang lebih cepat dari metode backpropagasi. Hasil estimasi dapat digunakan untuk mengidentifikasi kecepatan rotor motor induksi

Kata kunci: pengaturan kecepatan, motor induksi tanpa sensor, FOC, SCFNN *observer*, *Levenberg Marquardt*

ABSTRACT

This paper describes about development of 3 phase speed sensorless induction motor speed controller using Field Oriented Vector(FOC) method. Motor speed is estimated by an observer using Self Constructing fuzzy Neural Network (SCFNN) with Levenberg Marquardt(LM) learning algorithm method, that replaces backpropagation method because this method is slow to reach convergent. SCFNN method combines the fuzzy and neural network. The simulation results show that the system can estimate flux and speed of induction motor and it converges faster than backpropagation method. .The estimation result can be used to identify rotor speed of induction motor with good performance

Keywords: *speed control, induction motor, sensorless, FOC, SCFNN observer, Levenberg Marquardt*

PENDAHULUAN

Motor DC merupakan jenis motor yang paling ideal untuk kontrol elektrik karena kecepatannya dapat diatur dengan mudah dan tidak memerlukan konverter. Kelemahan motor DC adalah harganya relatif mahal, ukurannya relatif besar, adanya komutator dan sikat-sikat dalam motor, sehingga memerlukan perawatan yang rumit dan harus dilakukan rutin. Selama perawatan operasi sistem terhenti, tentu ini sangat tidak dikehendaki dalam industri, karena akan sangat mengganggu proses dan mengurangi hasil (produksi) industri, yang berdampak pada kerugian perusahaan .

Catatan: Diskusi untuk makalah ini diterima sebelum tanggal 1 Desember 2008. Diskusi yang layak muat akan diterbitkan pada Jurnal Teknik Elektro volume 9, nomor 1, Maret 2009.

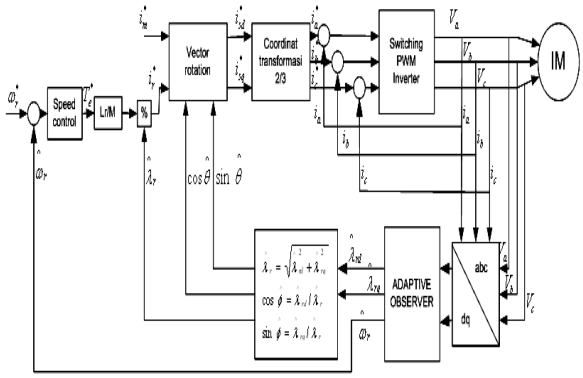
Kelebihan motor induksi selain kokoh, konstruksinya sederhana juga perawatannya mudah. Kelemahan motor induksi antara lain motor induksi merupakan motor yang tidak linier, metode untuk mengatur kecepatan rumit, disamping itu diperlukan suatu konverter yang dapat menimbulkan harmonisa. Namun setelah ditemukan suatu metode *Field Oriented Control(FOC)* yang mengubah sistem *couple* menjadi *decouple,* kesulitan dapat diatasi. Penggunaan metode ini menyebabkan motor induksi berfungsi seperti motor DC penguat terpisah. Dengan demikian motor induksi menggeser peranan motor DC dalam industri.

Dalam sistem pengaturan kecepatan motor induksi yang dioperasikan dengan metode FOC diperlukan suatu sensor kecepatan untuk

mengamati nilai kecepatannya. Hasil pengamatan dari sensor kecepatan motor ini dibandingkan dengan kecepatan *set point*, yang kemudian diumpankan ke kontroller untuk bisa mengendalikan kecepatan sehingga sesuai dengan harga *set point* yang di-*input*-kan. Biasanya letak sensor terlalu jauh dari sistem kontrol maka proses pengamatan sensor pada motor induksi ini membuat hasil pengukuran kecepatannya menjadi kurang akurat. Untuk mengatasi masalah tersebut diperlukan suatu *observer* yang berfungsi untuk mengamati besar torka dan arus, sehingga kecepatan motor dapat diprediksi. Karena itu dikembangkan suatu *observer* menggunakan metode *Selft Contructing Fuzzy Neural Network (SCFNN)*.

Sasaran dari penelitian ini adalah mengembangkan suatu *observer* metode *Self Constructing Fuzzy Neural Network (SCFNN*) dengan metode algorithm pelatihan *Levenberg Marquardt* yang menggantikan SCFNN dengan metode Backpropagasi yang dilakukan peneliti sebelumnya [1]. Metode ini digunakan untuk mengontrol kecepatan motor induksi 3 *phase*, sehingga motor mempunyai kinerja yang lebih bagus dengan pencapaian kekonvergenan yang lebih cepat.

Penelitian ini diharapkan memberi kontribusi terhadap upaya untuk terus mengembangkan metode pengaturan kecepatan pada motor induksi sehingga dapat meningkatkan kinerja yang lebih baik dari sistem yang menggunakan motor induksi. Dengan perbaikan kinerja sistem diharapkan meningkatkan efisiensi kerja pada dunia industri.



Gambar 1. Konfigurasi sistem *speed-sensorless vector control* untuk motor induksi dengan SCFNN[2]

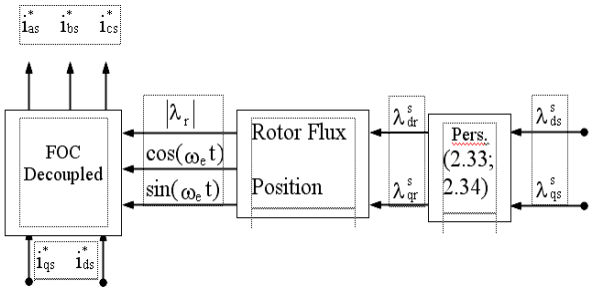
MODELING

Model Sistem

Mengembangkan dari beberapa penelitian yang telah dilakukan oleh Seong-Hwan Kim, dkk.[3], Faa-Jeng Lin, dkk[4] dan Iradiratu DPK.[2], maka blok diagram sistem yang dikembangkan pada penelitian ini terlihat pada gambar 1.

Field Oriented Control

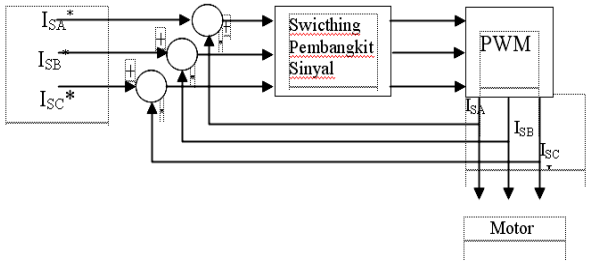
Field Oriented Control (FOC) adalah suatu metode pengaturan medan pada motor AC, dimana dari sistem *coupled* diubah menjadi sistem *decoupled*. Dengan sistem ini arus penguatan dan arus beban motor dapat dikontrol secara terpisah, dengan demikian torka dan fluksi juga dapat diatur secara terpisah. Diagram blok yang menggambarkan prinsip dasar sistem *decoupled field oriented control (FOC Decoupled)* motor induksi ditunjukkan pada gambar 2 [5].



Gambar 2. Diagram blok FOC *decoupled* motor induksi

Model *Inverter*

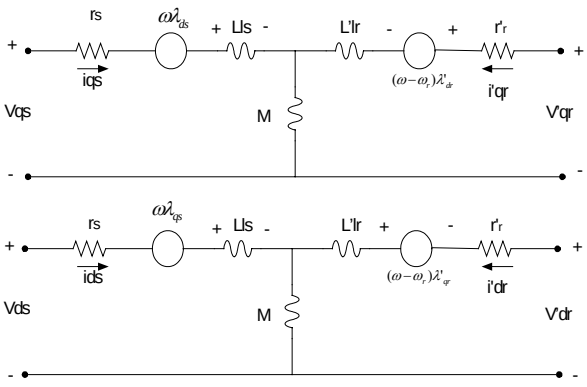
Vektor rotasi terhadap arus magnetisasi dan arus torsi menghasilkan arus fase referensi yang digunakan untuk sinyal kontrol PWM *inverter*. Tegangan yang dihasilkan *inverter* akan digunakan oleh stator motor induksi. Model PWM *inverter* ditunjukkan pada gambar 3.



Gambar 3. PWM *inverter*

Model Motor Induksi

Rangkaian ekivalen motor induksi dalam koordinat d-q dapat dilihat pada gambar 4.



Gambar 4. Rangkaian ekivalen motor induksi dalam koordinat d-q

Dengan memasukkan tegangan rotor (V_r = 0) pada rangkaian ekivalen motor induksi dalam koordinat d-q, maka didapat besaran tegangan stator yang merupakan fungsi dari arus stator dan arus rotor dalam bentuk matrik sebagai berikut :

$$\begin{bmatrix} v_{ds} \\ v_{qs} \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} R_s + pL_s & -\omega_s L_s & pM & -\omega_s M \\ \omega_s L_s & R_s + pL_s & \omega_s M & pM \\ pM & -(\omega_s - \omega_r)M & R_r + pL_r & -(\omega_s - \omega_r)L_r \\ (\omega_s - \omega_r)M & pM & (\omega_s - \omega_r)L_r & R_r + pL_r \end{bmatrix} \begin{bmatrix} i_{ds} \\ i_{qs} \\ i_{dr} \\ i_{qr} \end{bmatrix} \quad (1)$$

dengan: $p = \dfrac{d}{dt}$

Jika ditinjau pada koordinat *stationer* (ω_s = 0), maka persamaan 1 menjadi:

$$\begin{bmatrix} v_{ds} \\ v_{qs} \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} R_s + pL_s & 0 & pM & 0 \\ 0 & R_s + pL_s & 0 & pM \\ pM & \omega_r M & R_r + pL_r & \omega_r)L_r \\ -\omega_r M & pM & -\omega_r L_r & R_r + pL_r \end{bmatrix} \begin{bmatrix} i_{ds} \\ i_{qs} \\ i_{dr} \\ i_{qr} \end{bmatrix} \quad (2)$$

SELF CONSTRUCTING FUZZY NEURAL NETWORK DENGAN METODE PELATIHAN *LEVENBERG MARQUARDT*

Kontroler ini adalah sebuah kontroler *fuzzy*, sehingga masukannya adalah data *numeric* berupa nilai *error*. Struktur dasar dari sebuah *fuzzy neural network* adalah seperti pada gambar 5.[6]

Pada lapisan pertama hanya terjadi proses masukan berupa data *crisp* yaitu *error* (X_1) dan Delta *error* (X_2) untuk meneruskan sinyal ke lapisan berikutnya.

Pada lapisan kedua terjadi proses fuzzifikasi dan pembentukan *membership* fungsi. Fungsi yang dipergunakan adalah fungsi *Gaussian*.

$$u_{A_i^j} = exp\left(\frac{\left(x_i - m_{ji}\right)^2}{\sigma_{ji}^2}\right) \quad (3)$$

dengan m_{ji} dan σ_{ji} adalah rata-rata (*mean*) dan standar deviasi**.**

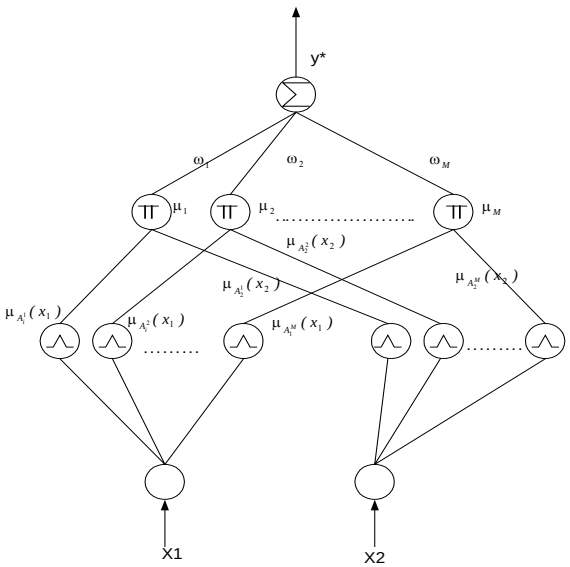
Lapisan ketiga merupakan penentuan kondisi awal dari aturan *fuzzy*. Langkah ini adalah untuk memperoleh hasil perkalian antara semua komponen input dari *error* dan delta *error* dengan persamaan untuk *rule node* ke-*j*:

$$u_j = u_{A_1^j}(x_1)u_{A_2^j}(x_2)\cdots u_{A_n^j}(x_n) = \prod_i u_{A_i^j}(x_i) \quad (4)$$

Dengan u_j adalah output *node rule* ke-*j*.

Lapisan keempat berfungsi untuk menjumlahkan seluruh sinyal masukan yang disimbolkan dengan Σ kemudian dirumuskan dalam persamaan y^* yang selanjutnya dilakukan proses defuzzikasi [6]

$$y^* = \sum_{j=1}^{M} w_j u_j \quad (5)$$



Gambar 5. Struktur dasar SCFNN

Algoritma On-Line pada SCFNN dengan Metode Pelatihan *Levenberg Marquardt*

Pada SCFNN terdapat dua jenis tipe algoritma pelatihan yaitu pelatihan struktur dan pelatihan parameter. Pelatihan struktur digunakan untuk

mencari *space input fuzzy logic partition* dan *fuzzy logic subject* yang bertujuan meminimalkan jumlah *rule* dan meminimalkan *fuzzy set* dalam semesta pembicaraan dari setiap variabel *input*. Pelatihan parameter menggunakan algoritma *supervised learning*, sedangkan untuk menentukan bobot dan parameter dari *membership function* diatur dengan algoritma pelatihan backpropagasi.

Fasa Pelatihan Struktur

Langkah pertama pada pelatihan struktur adalah menentukan perlu tidaknya melakukan pelatihan struktur. Jika $e_{\min} \le |e|$ atau $\Delta e_{\min} \le |\Delta e|$, dimana $e_{\min}$ dan $\Delta e_{\min}$ adalah konstanta positif, maka Pelatihan struktur diperlukan. Selanjutnya menentukan *node* baru (*membership function*) pada lapisan kedua dan menghubungkan *fuzzy logic rule* pada lapisan ketiga. Jika adanya satu *cluster* diberikan pada *input* akan menyebabkan adanya sebuah *rule fuzzy logic* pada lapisan ketiga, maka persamaan kekuatan penyulutan (*firing strength*) dari sebuah *rule* untuk setiap data masukan x_i dapat ditunjukkan sebagai sudut dimana data masukan memiliki hubungan terhadap *cluster* data. *Firing strength* diperoleh dari persamaan 4 yang digunakan sebagai pengukuran sudut :

$$D_j = u_j \quad j = 1,..., Q(t) \tag{6}$$

Dengan *Q(t)* adalah jumlah *rule* yang ada pada waktu *t.* Kriteria pembentukan *fuzzy rule* baru untuk data masukan baru dinyatakan sbb. Dengan menentukan pengukuran sudut maksimum $D_{\max}$

$$D_{\max} = \max_{1 \le j \le Q(t)} D_j \tag{7}$$

Jika $D_{\max} \le \overline{D}$, maka dibentuk *membership function* dengan $D \in (0,1)$. Kemudian *mean* dan standar deviasi dari *membership function* yang baru, dinyatakan lebih dulu dengan nilai tertentu secara heuristik atau cara lain. Jadi *mean* dan standar deviasi dari *membership function* baru sebagai berikut:

$$m_i^{(new)} = x_i \tag{8}$$

$$\sigma_i^{(new)} = \sigma_i \tag{9}$$

dengan x_i adalah data masukan yang baru dan σ_i adalah standard deviasi.

Untuk menghindari *membership function* yang baru sama dengan yang telah ada, kesamaan antara *membership function* yang lama dan baru harus diperiksa, yaitu dengan assumsi bahwa jika terdapat dua *fuzzy set* A dan B dengan *membership function* adalah $u_A(x) = exp\ [-(x - m_1)^2/\sigma_1^2]$ dan $u_B(x) = exp\ [-(x - m_2)^2/\sigma_2^2]$, dan asumsikan $m_1 \ge m_2$, maka $|A \cap B|$ dapat dihitung:

$$\begin{aligned} |A \cap B| = {} & \frac{1}{2}\frac{h^2(x)\left[m_2 - m_1 + \sqrt{\pi}(\sigma_1 + \sigma_2)\right]}{\sqrt{\pi}(\sigma_1 + \sigma_2)} + \\ & \frac{1}{2}\frac{h^2(x)\left[m_2 - m_1 + \sqrt{\pi}(\sigma_1 + \sigma_2)\right]}{\sqrt{\pi}(\sigma_1 + \sigma_2)} + \\ & \frac{1}{2}\frac{h^2(x)\left[m_2 - m_1 + \sqrt{\pi}(\sigma_1 + \sigma_2)\right]}{\sqrt{\pi}(\sigma_1 + \sigma_2)} \end{aligned} \tag{10}$$

dimana $h(x) = \max\{0,x\}$.

$$E(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{\sigma_1\sqrt{\pi} + \sigma_2\sqrt{\pi}|A \cup B|^*} \tag{11}$$

Pemeriksaan dilakukan pada semua variabel *input* x_i. Sedangkan nilai maksimumnya E_{max} didapat dengan:

$$E_{\max} = \max_{1 \le j \le Q(t)} E\left\{u\left(m_1^{(new)}, \sigma_1^{(new)}\right), \quad u\left(m_{j1}, \sigma_{j1}\right)\right\} \tag{12}$$

dimana $u(m_{j1}, \sigma_{j1})$ adalah *membership function Gaussian* dengan *mean* m_{j1} dan standar deviasi σ_{j1}; M(t) adalah jumlah *membership function* ke-i dari variabel *input*. Jika $E_{max} \le \overline{E}$ dengan $\overline{E} \in (0,1)$ adalah nilai yang sudah ditentukan, maka gunakan *membership function* baru dan jumlah M(t) adalah sebagai berikut:

$$M(t+1) = M(t) + 1 \tag{13}$$

Jadi pembentukan *membership function* berhubungan dengan pembentukan *rule fuzzy* baru dan bobot $\omega^{(new)}$.

Fasa Pelatihan Parameter

Algoritma pelatihan parameter dari SCFNN adalah menentukan *rule* yang adaptif untuk mengatur parameter-parameter jaringan, berdasarkan pasangan *input-output*. Jika parameter jaringan terdiri dari vektor parameter, maka proses pelatihan memperhitungkan vektor determinasi dari fungsi energi. Metode ini umumnya berdasar *rule* pelatihan backpropagasi karena vektor gradien dihitung dalam arah berlawanan terhadap *output* setiap *node*. Untuk

menjelaskan algoritma pelatihan parameter SCFNN menggunakan metode *supervised gradient decent*, asumsikan fungsi energi *E* didefinisikan sebagai:

$$E = \frac{1}{2}(\omega_m - \omega_r)^2 = \frac{1}{2}e_m^2 \tag{14}$$

Kemudian algoritma pelatihan parameter berdasarkan backpropagasi dijelaskan sebagai berikut [6]:

Lapisan keempat; bentuk *error* dipropagasi dan dihitung sebagai berikut:

$$\delta^4 = -\frac{\partial E}{\partial y^*} = \left[-\frac{\partial E}{\partial e_m}\frac{\partial e_m}{\partial y^*}\right] = \left[-\frac{\partial E}{\partial e_m}\frac{\partial e_m}{\partial \omega_r}\frac{\partial \omega_r}{\partial y^*}\right] \tag{15}$$

dan bobot di-*update* sebesar:

$$\Delta\omega_j = -\eta_\omega \frac{\partial E}{\partial \omega_j} = \left[-\eta_\omega \frac{\partial E}{\partial y^*}\right]\left(\frac{\partial y^*}{\partial \omega_j}\right) = -\eta_\omega \delta^4 u_j \tag{16}$$

dengan faktor η_ω adalah parameter *learning rate* dari bobot. Bobot pada lapisan keempat di-*update* dengan menggunakan persamaan:

$$\omega_j(N+1) = \omega_j(N) + \Delta\omega_j \tag{17}$$

dengan N adalah jumlah iterasi dari *node* ke-*j*

Lapisan ketiga; pada lapisan ini hanya *error* yang perlu dihitung dan dipropagasi:

$$\delta_j^3 = -\frac{\partial E}{\partial u_j} = \left[-\frac{\partial E}{\partial y^*}\right] * \left[\frac{\partial y^*}{\partial u_j}\right] = \delta^4 \omega_j \tag{18}$$

Lapisan kedua; error dihitung sebagai berikut:

$$\delta_{ji}^2 = -\frac{\partial E}{\partial u_{A_i^j}} = \left[-\frac{\partial E}{\partial y^*}\frac{\partial y^*}{\partial u_j}\right]\left[\frac{\partial u_j}{\partial u_{A_i^j}}\right] = \delta_j^3 \tag{19}$$

Hukum *update* dari m_{ji} adalah:

$$\Delta m_{ji} = -\eta_m \frac{\partial E}{\partial m_{ji}} = \left[-\eta_m \frac{\partial E}{\partial u_{A_i^j}}\frac{\partial u_{A_i^j}}{\partial m_{ji}}\right] = \eta_m \delta_{ji}^2 \frac{2\left(x_i^2 - m_{ji}\right)}{\left(\sigma_{ji}\right)^2} \tag{20}$$

Hukum *update* dari σ_{ji} adalah:

$$\Delta\sigma_{ji} = -\eta_\sigma \frac{\partial E}{\partial \sigma_{ji}} = \left[-\eta_\sigma \frac{\partial E}{\partial u_{A_i^j}}\frac{\partial u_{A_i^j}}{\partial \sigma_{ji}}\right] = \eta_\sigma \delta_{ji}^2 \frac{2\left(x_i^2 - m_{ji}\right)}{\left(\sigma_{ji}\right)^2} \tag{21}$$

dengan η_m dan η_σ adalah parameter *learning rate* dari *mean* dan standar devisasi fungsi *Gaussian*. *Mean* dan standar deviasi dari *membership function* pada lapisan ini di-*update* dengan:

$$m_{ji}(N+1) = m_{ji}(N) + \Delta m_{ji} \tag{22}$$

$$\sigma_{ji}(N+1) = \sigma_{ji}(N) + \Delta\sigma_{ji}) \tag{23}$$

Setelah didapatkan persamaan-persamaan ini maka disimulasikan rangkaian kontrol dengan *Self Constructing Fuzzy Neural Networks* dengan *plant* motor induksi.

Pada penelitian ini metode pelatihan Backpropagasi diganti dengan metode pelatihan *Levenberg Merquardt* (LM). Metode pelatihan LM ini merupakan kombinasi algoritma *Newton* dengan *Steepest Decent*. Bila metode *Gradient Descent* dinyatakan sebagai persamaan :

$$W_{kj}(t+1) = W_{kj}(t) + \alpha.\delta_k.Z_j \tag{24}$$

Persamaan di atas dapat disederhanakan menjadi

$$W_{k+1} = W_k + \alpha.g \tag{25}$$

Dimana *g* adalah vektor *gradient*. Bentuk persamaan *Newton* adalah:

$$W_{k+1} = W_k - A_k^{-1}.g \tag{26}$$

A_k adalah matrik Hessian (elemennya adalah turunan kedua *error* terhadap bobot) sebagai berikut:

$$A = \begin{bmatrix} \frac{\partial^2 E}{\partial W_1^2} & \frac{\partial^2 E}{\partial W_1 \partial W_1} & .... & .... & \frac{\partial^2 E}{\partial W_n \partial W_1} \\ \frac{\partial^2 E}{\partial W_1 \partial W_2} & \frac{\partial^2 E}{\partial W \partial W_2^2} & .... & .... & \frac{\partial^2 E}{\partial W_n \partial W_2} \\ .... & .... & .... & .... & .... \\ .... & .... & .... & .... & .... \\ \frac{\partial^2 E}{\partial W_1 \partial W_n} & \frac{\partial^2 E}{\partial W_2 \partial W_n} & .... & .... & \frac{\partial^2 E}{\partial W \partial W_{2n}^2} \end{bmatrix} \tag{27}$$

Matrik A dapat ditulus menjadi:

$$A = 2J^T J \quad (28)$$

dimana *J* adalah matrik Jacobian

Persamaan *update* bobot dengan metode pelatihan LM adalah :

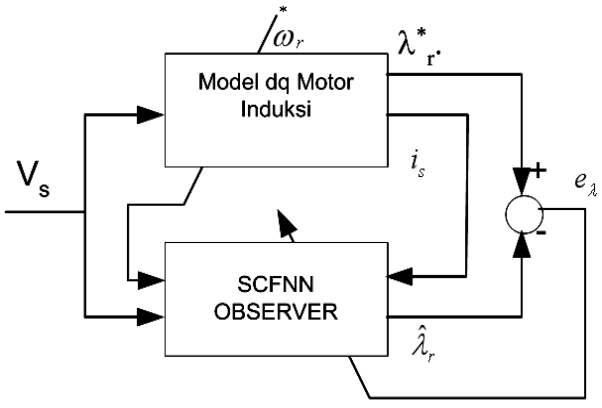
$$W_{k+1} = W_k - (J_k^T J_k + \mu I)^{-1} J_k^T e \quad (29)$$

Bila nilai $\mu = 0$, maka metode pelatihan LM akan sama dengan metode *Gauss Newton*, sedangkan bila μ maka metode pelatihan LM akan sama dengan Backpropagasi(*steepest descent*)[2].

Setelah didapatkan persamaan-persamaan ini maka disimulasikan rangkaian kontrol dengan *Self Constructing Fuzzy Neural Networks* metode pelatihan *Levenberg Marquardt* dengan *plant* motor induksi.

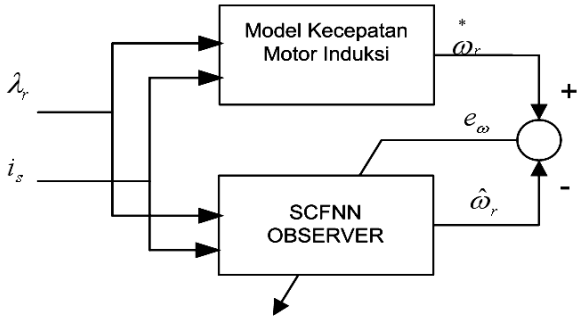
BLOK DIAGRAM *SELF CONTRUCTING FUZZY NEURAL NETWORK OBSERVER* (SCFNNO)

Blok diagram sistem kontrol kecepatan motor induksi tanpa sensor kecepatan dapat dilihat pada gambar 1. Bagian *adaptive observer* menggunakan *self contructing fuzzy neural network* terlihat pada gambar 6 dan gambar 7.



Gambar 6. Struktur estimasi fluksi

Gambar 6 adalah struktur estimasi untuk mendapatkan estimasi fluksi yang masing-masing terdiri dari fluksi *direct* λ_{dr} dan fluksi *quadrater* λ_{qr}. Hasil dua parameter ini digunakan untuk input estimasi kecepatan seperti terlihat pada gambar 7.



Gambar 7. Struktur estimasi kecepatan

Parameter motor induksi untuk data pelatihan SCFNN yang digunakan dalam mendapatkan target adalah:

a. Fluksi *direct* λ_{dr} terdiri atas I_{ds}, V_{ds}, V_{qs} dan ω_r
b. Fluksi *quadratur* λ_{qr} terdiri atas I_{qs}, V_{qs}, V_{qs} dan ω_r
c. Kecepatan ω_r terdiri atas λ_{dr}, λ_{qr} I_{ds}, dan I_{qs}.

Pembelajaran *Off-Line*

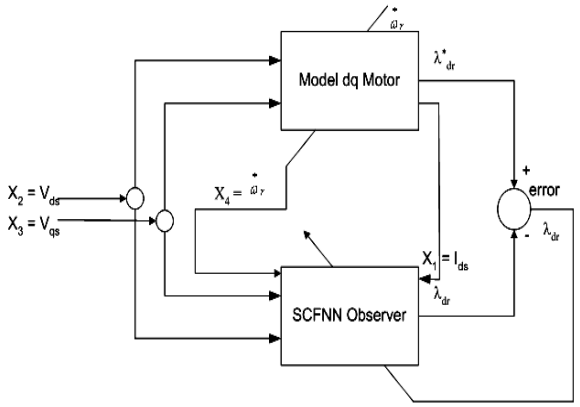
Metode pembelajaran untuk estimasi fluksi identifikasi kecepatan motor induksi tiga fasa menggunakan *self constructing fuzzy neural network*, dimana jaringannya terdiri dari empat lapisan, yaitu 4 *input*, *linguistic*, *precondition* dan 1 *output*. *Linguistic, precondition* dan *output* digunakan untuk mendapatkan nilai $\lambda_{dr}, \lambda_{qr}$ dan ω_r.

Proses pembelajaran menggunakan 4 *neuron input* yaitu V_s, I_s, ω_r^* dan λ_r. Pembelajaran dilakukan sebayak 5 *epoch*. Apabila hasil pembelajaran belum konvergen atau tidak sesuai target maka akan terjadi penambahan *membership function* baru. Penambahan akan berhenti bila hasil pembelajaran konvergen. Harga awal bobot ditentukan antara 0 dan 1 untuk mencari parameter optimal yang menghasilkan kinerja terbaik.

Dalam proses estimasi, ada tiga SCFNN untuk menyelesaikan estimasi fluksi *direct* λ_{dr}, fluksi *quadratur* λ_{qr}. Setelah selesai proses ini ada satu SCFNN menyelesaikan estimasi kecepatan ω_r.

Pembelajaran Off-line Fluksi Direcf λ_{dr}

Pembelajaran *off-line self constructing fuzzy neural network observer* untuk identifikasi fluksi *direct* λ_{dr}, ditunjukkan pada gambar 8.



Gambar 8. Sruktur estimasi fluksi *direct* λ_{dr} menggunakan SCFNNO

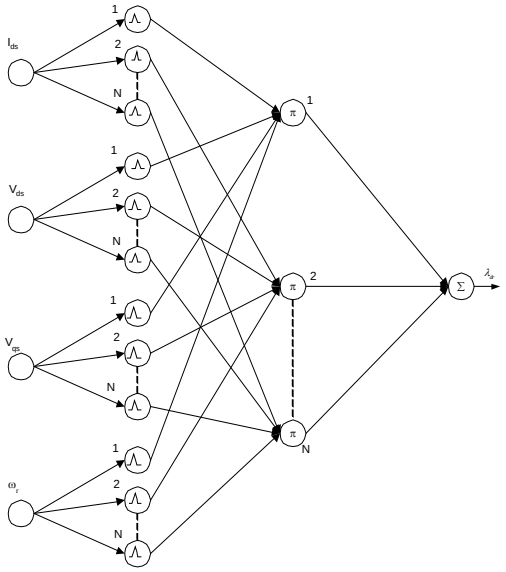
Gambar 8 menggambarkan tentang struktur dari *estimator* fluksi *direct* λ_{dr} menggunakan SCFNN dan input terdiri atas I_{ds}, V_{ds}, V_{qs} dan ω_r, masuk ke blok SCFNNO.

Blok model motor menentukan nilai kecepatan (kecepatan referensi ω_r^*) sehingga didapatkan fluksi *direct* referensi λ^*_{dr}. Bagian SCFNNO menghasilkan fluksi *direct* pelatihan $\hat{\lambda}_{dr}$. Selisih nilai referensi dan pembejaran didapatkan *error* atau fluksi *direct* estimasi.

Output dari SCFNNO didefinisikan sebagai fluksi *direct* pelatihan $\hat{\lambda}_{dr}$, yang kemudian digunakan sebagai *input* yang dapat diubah-ubah. Jika fluksi *direct* yang diestimasi merupakan deviasi dari fluksi *direct* sesungguhnya dan *error* hubungan antara flux dari model fluksi *direct* pelatihan $\hat{\lambda}_{dr}$ dan fluksi *direct* referensi λ^*_{dr}, maka error merupakan backpropagasi dari SCFNN dan pembebanan dari SCFNN adalah *adjusted on line* untuk mengurangi *error*. Akhirnya, *output* dari SCFNN merupakan model fluksi *direct* yang sesungguhnya.

Internal Struktur Dari SCFNNO

Gambar 9 menggambarkan tentang struktur dari SCFNNO dan dilatih dengan algoritma backpropagasi untuk mendapatkan hasil SCFNN yang mengikuti kecepatan yang sesungguhnya.



Gambar 9. Internal struktur SCFNNO

Jumlah blok SCFNNO ada 3 yaitu blok untuk SCFNNO λ_{dr}, λ_{qr} dan ω_r, blok SCFNNO untuk λ_{dr} dan λ_{qr} yang dihitung dulu, kemudian hasilnya dimasukkan ke blok SCFNNO ω_r.

Pada saat pelatihan, jumlah *rule* yang tercipta untuk masing-masing *input* bisa berbeda dari *rule* yang tercipta untuk satu blok SCFNNO, karena karakteristik I_{ds}, I_{qs}, V_{ds}, V_{qs} dan ω_r tidak sama.

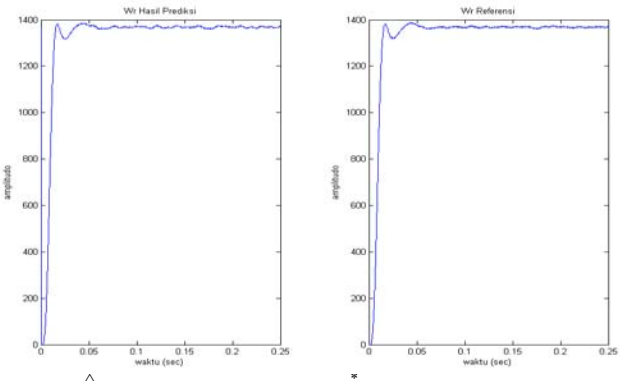
SIMULASI

Data motor induksi tiga fasa yang digunakan untuk simulasi adalah:

Rs (resistansi stator) = 176 ohm, Tegangan 115 Volt
Rr (resistansi rotor) = 190 ohm, Jumlah pasang kutub = 2
Ls (induktansi stator) = 3,79 H, Frekuensi 60 Hz
Lr (induktansi rotor) = 3,31 H
M (induktansi gandeng) = 3,21 H
J (momen inersia) = 0,0000105 Kg.m^2
Kd (konstanta gesek) = 1,9.10^{-5} Kg.m^2/s

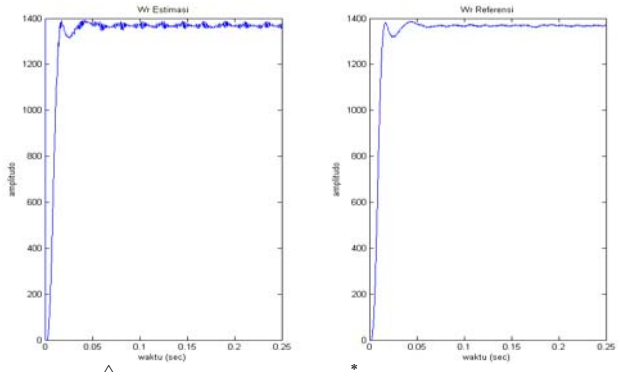
Gambar 10 dan gambar 11 merupakan hasil simulasi yang diperoleh untuk estimasi fluksi dan kecepatan rotor dengan menggunakan bahasa pemrograman C^{++}. Performansi yang ditunjukkan menampilkan pengambilan data pada kecepatan

referensi 1350 rpm**.** Gambar yang ditampilkan adalah hasil observer menggunakan SCFNNO pelatihan *Levenberg Marquardt* dan SCFNNO pelatihan Backpropagasi yang telah dilakukan peneliti sebelumnya [1].



$\hat{\omega}_r$ Prediksi ω_r^* Output Model Motor

Gambar 10. Grafik kecepatan fungsi waktu dengan SCFNNO *levenberg marquardt* nilai MSE 0,017175%



$\hat{\omega}_r$ Prediksi ω_r^* Output Model Motor

Gambar 11. Grafik kecepatan fungsi waktu dengan SCFNNO backpropagasi nilai MSE 0,030675%

Tabel l. Pengujian SCFNNO dengan pelatihan *levenberg marquardt* dan SCFNNO dengan pelatihan backpropagasi untuk beberapa data yang berbeda

N0	Kecepatan referensi (rpm)	Hasil prediksi pada	SCFNN Observer			
			Levenberq Marquardt		Backpropagasi	
			Jumlah epoch	MSE (%)	Jumlah epoch	MSE (%)
1	1350	λ_{dr}	2	0,010360	50	0.020035
		λ_{qr}	2	0,019923	50	0,081165
		ω_r	2	0,017175	50	0,030675
2	750	λ_{dr}	2	0,003539	50	0,056819
		λ_{qr}	2	0,024703	50	0,080676
		ω_r	2	0,021564	50	0,036961
3	500	λ_{dr}	2	0,027750	50	0,060614
		λ_{qr}	2	0,018828	50	0,125130
		ω_r	2	0,020052	50	0,049025

KESIMPULAN

Dari pengamatan dan analisis hasil simulasi SCFNNO pelatihan *Levenberg Marquardt* pada penelitian ini dan kemudian dibandingkan dengan SCFNNO pelatihan Backpropagasi yang telah dilakukan peneliti sebelumnya, maka dapat disimpulkan:

1. Pada simulasi pelatihan pada simulasi SCFNNO *Marquardt* diperlukan jumlah *epoch* yang lebih sedikit dengan error yang lebih kecil dibanding dengan SCFNNO Backpropagasi. Hal ini berarti bahwa metode SCFNNO pelatihan *Levenberg Marquardt* lebih cepat konvergen dibanding dengan SCFNNO dengan pelatihan Backpropagasi.
2. Simulasi estimasi kecepatan ($\hat{\omega}_r$) dengan SCFNNO pelatihan *Levenberg Marquardt* dengan pengambilan referensi kecepatan yang berbeda (yaitu 1350 rpm, 750 rpm, 500 rpm), nilai Mean Square Error (MSE) berkisar antara 0,017175% sampai 0,021564%.

   Dari data tersebut dapat disimpulkan bahwa nilai MSE SCFNNO pelatihan *Levenberg Marquardt* mempunyai kisaran nilai yang lebih rendah dari SCFNNO pelatihan Backpropagasi. Hal ini menunjukkan bahwa *Self Constructing Fuzzy Neural Network Observer* yang dirancang mampu mengestimasi fluksi dan kecepatan rotor dengan baik karena nilai yang diijinkan *Standart Error* Estimasi (SEE) kurang dari 5%

Dengan demikian metode SCFNNO pelatihan *Levenberg Marquardt* yang dirancang mempunyai performansi yang lebih baik dari pada SCFNNO pelatihan Backpropagasi sehingga dapat digunakan sebagai piranti alternatif untuk identifikasi kecepatan rotor motor induksi.

DAFTAR PUSTAKA

[1] Sutedjo, Soebagio dan Mauridhi Hery Purnomo, "Kendali Kecepatan Motor Induksi Tanpa Sensor Kecepatan Menggunakan Self Conructing Fuzzy Neural Network", *Seminar Nasional XII FTI-ITS*, 29 Maret 2005, hal. 414.

[2] Iradiratu, Mauridhi Hery Purnomo, Era Purwanto, "Perancangan Model Observer

Untuk Identifikasi Kecepatan Motor Induksi", *Procedings SMED*, Sep 2002.

[3] Seong-Hwan Kim,Tae-Sik Park,Ji-Yoon Too, and Gwi-Tae Park, "Speed-Sensorless Vector Control Of An Induction Motor Using Neural Network Speed Estimation", *IEEE Trans. On Industry Application*, vol. 48, No. 3, June 2001.

[4] F.J.Lin, W.J.Huang, and R.J.Wai, "A Supervisory Fuzzy Neural Network Control System For Tracking Periodic Inputs", *IEEE Trans. Fuzzy Syst*, vol.7, Feb.1999, pp.41-52.

[5] Fukuda and T. Shibata, "Theory And Applications Of Neural Network for Industrial Control System", *IEEE Trans. Ind. Electron*, vol.39, Dec. 1992, pp.472-489.

[6] Faa-Jeng Lin and Chih-Hong Lin, "A Permanent magnet Synchronous Motor ServoDrive Using Self Constructing fuzzy Neural Network Controller", *IEEE Trans. On Energy Conversion*, Vol. 19, No. 1, March 2004.

[7] Purwanto Era, *Studi Pengaturan Motor Induksi Dengan Metoda Vektor*, Master Tesis, Universitas Shizuoka Jepang, 1995.

[8] C. T. Lin, "A Neural Fuzzy Control system With Structure And Parameter Learning", *Fuzzy Sets Syst.*, vol. 70, no. 2-3, Mar. 1995, pp. 183-212.

[9] C. F. Juang and C. T.Lin, "An On-Line Self Constructing Neural Fuzzy Inference Network And Its Application", *IEEE Trans. Fuzzy Syst*, vol.6, Feb. 1998, pp. 12-32.

[10] Faa Jeng Lin, Rong Jong Wai, Chih Hong Lin dan Da Chung Liu, "Decouple Stator Flux Oriented Induction Motor Drive with Fuzzy Neural Networks Uncertainty Observer", *IEEE Trans. On Industrial Electronics*, vol.47, No. 2, April, 2000, pp.356–367.

[11] R.Krishnan, *VirginaTech. Blacksburg, VaElectric Motor Drives Modeling, Analysis and Control*, Prentice Hall International. Inc., New Jersey, 2001.

[12] Soebagio, Era Purwanto, "Algoritma genetika Untuk Optimasi Penentuan parameter Motor Induksi dengan Model d-q", *SMED*, 13 Juli 2000, hal. II-1.

[13] Soebagio, Mohammad Zuhri, "Pengaruh Perubahan Paremeter Terhadap Kinerja Motor Induksi Rotor Sangkar dan Rotor Belit", *Proc. SITIA*, Jurusan Teknik Elektro ITS, 2 Mei 2005, hal. 63-67.

[14] Mauridhi Hery P, Agus Kurniawan, *Supervised Neural Networks dan Aplikasinya*, Graha Ilmu, 2006.

[15] Soebagio, *Peran Pengemudian Elektris dalam menghadapi Kompetisi Global*, Pengukuhan Guru Besar, Surabaya, Januari, 2003.

[16] K. S.Narendra and K. Parthasarathy, "Identification And Control Of Dynamical Systems Using Neural Networks", *IEEE Trans Neural Networks*, vol. 1, Maret 1990, pp. 4–27.

[17] Boldea Ion, Nazar Syed A, *Vector Control AC Drives*, CRC Press, Inc.1992.

[18] Wiryajati K, et al, "Kontroller Konvensional Sebagai Model Referensi pada Pengaturan Motor Induksi Berbasis Field Oriented Control", *Proc. CECI&SITIA Control and Inteligent Teknology Application*, Surabaya, 2003, pp.D40-D43.