

# Pengaturan Kecepatan Motor DC Menggunakan Jaringan Syaraf Tiruan Feedforward Backpropagation

Sorikhi\*, Wahyudi\*\*, Iwan Setiawan\*\*

**Abstrak** - Jaringan syaraf backpropagation merupakan jaringan syaraf yang telah umum digunakan dalam beragam aplikasi kontrol. Dari segi arsitektur, backpropagation merupakan jaringan syaraf perceptron dengan banyak layer (multilayer), sehingga jaringan syaraf ini dikenal juga sebagai jaringan syaraf perceptron multilayer. Nama backpropagation diambil dari proses pembelajaran yang melibatkan penghantaran balik sinyal error dari output layer menuju layer dibawahnya hingga layer sebelum input layer.

Pada tugas akhir ini, jaringan syaraf backpropagation digunakan sebagai kontroler dalam pengaturan kecepatan motor DC. Skema yang digunakan dalam pengontrolan adalah fix stabilizing controller. Dalam skema ini, jaringan syaraf berperan sebagai controller dimana sinyal training yang digunakan berasal dari sinyal keluaran penguat proporsional.

Dengan menggunakan bobot dan bias awal 0,05, pengaruh parameter jaringan syaraf serta parameter pada skema kontroler fix stabilizing controller dapat diamati. Learning rate akan berpengaruh terhadap kecepatan plant mencapai kondisi tunak. Momentum juga mempunyai pengaruh yang sama dengan learning rate. Pengamatan juga dilakukan terhadap pengaruh jumlah hidden layer dalam jaringan serta pengaruh nilai penguat proporsional. Dari hasil percobaan diperoleh bahwa jaringan dengan satu hidden layer lebih cepat mencapai kondisi tunak dibandingkan jaringan dengan dua hidden layer. Nilai penguat proporsional dimana respon plant masih stabil adalah 2.

**Kata kunci** : jaringan syaraf tiruan, backpropagation, fix stabilizing controller, plant motor DC.

**Kata Kunci** : jaringan syaraf tiruan, backpropagation, fix stabilizing controller, plant motor DC.

## I. PENDAHULUAN

### 1.1 Latar Belakang

Beragam sistem kontrol melibatkan manusia sebagai pengendalinya, sehingga semakin banyak peralatan kontrol akan mengakibatkan besarnya biaya pengoperasian. Berangkat dari permasalahan tersebut muncul gagasan untuk memindahkan keahlian dari manusia ke dalam sistem kontrol.

Kontrol klasik (P, PI, PID) merupakan dasar dari sistem kontrol sederhana. Kesederhanaannya merupakan alasan utama kenapa kontrol ini begitu luas diterapkan di industri.

Kontrol klasik merupakan kontroler penguatan umpan balik tetap, oleh karena itu kontroler ini tidak mampu mengkompensasi variasi parameter dalam plant dan tidak dapat beradaptasi terhadap perubahan lingkungan.

Dalam teknik klasik, modelkan plant secara matematik dan penalaan parameter dari kontroler harus dilakukan terlebih dahulu sebelum mengimplementasikan kontrolernya. Kebanyakan sistem kontrol mempunyai perilaku non linier, lebih jauh, untuk memodelkan sistem ini seringkali mengalami kesulitan, kadangkala mustahil untuk penggunaan hukum-hukum fisika. Oleh karena itu, penggunaan kontrol klasik tidaklah cocok untuk aplikasi kontrol non linier.

Dalam tiga dekade terakhir telah dikenalkan teknik kontrol alternatif, seperti *neural network* (jaringan syaraf tiruan), adaptif, serta *fuzzy* kontrol sebagai alternatif teknik kontrol klasik. Perkembangan jaringan syaraf tiruan (JST) telah menginspirasi munculnya teknik kontrol yang lebih baik dan efisien. JST mempunyai kemampuan mempelajari kondisi dinamis sistem dengan mengestimasi fungsi *input-output* sistem. Dengan menggunakan JST, akan didapat suatu kontroler tanpa harus memodelkan plant secara matematik serta penalaan parameter kontroler.

### 1.2 Tujuan

Tujuan tugas akhir ini adalah untuk mengetahui unjuk kerja jaringan syaraf tiruan *backpropagation* yang diterapkan secara *on line* pada pengaturan kecepatan plant motor DC.

### 1.3 Batasan Masalah

Dalam pembuatan tugas akhir ini penulis membatasi permasalahan sebagai berikut:

1. Plant yang digunakan pada tugas akhir ini adalah plant motor DC tanpa beban.
2. Jaringan syaraf tiruan yang digunakan adalah *feedforward backpropagation* dengan jumlah neuron pada *input layer* sebanyak dua buah, empat buah pada *hidden layer*, serta satu buah neuron pada *output layer*.
3. Jumlah *hidden layer* pada jaringan syaraf yang diujikan adalah jaringan syaraf dengan satu *hidden layer*, dan jaringan syaraf dengan dua *hidden layer*.
4. Mikrokontroler AT89S51 pada plant hanya digunakan sebagai sarana komunikasi serial.
5. Tidak akan membahas mengenai proses *handshaking* pada komunikasi serial.
6. Bahasa pemrograman yang digunakan menggunakan Visual Basic 6.0.

\* Mahasiswa Jurusan Teknik Elektro UNDIP

\*\* Staff Pengajar Jurusan Teknik Elektro UNDIP

## II. DASAR TEORI

### 2.1 Jaringan Syaraf Tiruan<sup>[2]</sup>

Jaringan syaraf tiruan adalah sistem pengolahan informasi yang mempunyai karakteristik unjuk kerja tertentu sebagaimana jaringan syaraf biologis. Jaringan syaraf tiruan didasarkan pada beberapa asumsi :

1. Pengolahan informasi terjadi pada neuron.
2. Sinyal-sinyal dilewatkan antar neuron melalui rantai penghubung.
3. Masing-masing rantai penghubung akan mengalirkan sinyal yang ditransmisikan.
4. Masing-masing neuron menggunakan fungsi aktivasi pada jaringan masukannya untuk menentukan sinyal keluaran.

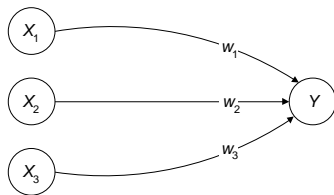
Jaringan syaraf tiruan dicirikan oleh (1) hubungan pola antar neuron, (2) metode untuk menentukan bobot dan (3) fungsi aktivasinya.

Sebuah jaringan syaraf terdiri dari banyak elemen pengolah sederhana yang disebut *neuron*, *unit*, *cell*, atau *node*. Masing-masing neuron dihubungkan ke neuron yang lain oleh rantai komunikasi terarah dengan masing-masing bobot yang berkaitan dengannya. Bobot merepresentasikan informasi yang digunakan oleh jaringan untuk memecahkan sebuah masalah.

Masing-masing neuron mempunyai *state internal*, disebut sebagai aktivasi atau level pengaktifan (*activation or activity level*) yang merupakan fungsi masukan. Biasanya, sebuah neuron mengirim aktivasinya sebagai sebuah sinyal ke beberapa neuron. Sebagai catatan bahwa neuron hanya dapat mengirim satu sinyal pada suatu waktu, meskipun sinyal itu dikirim ke beberapa neuron yang lain.

Gambar 1 menunjukkan sebuah neuron *Y* yang menerima input dari neuron  $X_1$ ,  $X_2$ , dan  $X_3$ . Pengaktifan (sinyal keluaran) dari neuron ini adalah  $x_1$ ,  $x_2$ , dan  $x_3$ , secara berturut-turut. Bobot pada hubungan dari  $X_1$ ,  $X_2$ , dan  $X_3$  ke neuron *Y* adalah  $w_1$ ,  $w_2$ , dan  $w_3$ . Jaringan masukan,  $y_{in}$ , ke neuron *Y* adalah jumlah dari sinyal terboboti (*weighted signal*) dari  $X_1$ ,  $X_2$ , dan  $X_3$ , yaitu :

$$y_{in} = w_1x_1 + w_2x_2 + w_3x_3.$$



Gambar 1 Neuron sederhana.

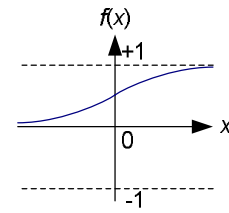
Aktivasi  $y$  dari neuron *Y* diberikan oleh suatu fungsi dari jaringan masukannya,  $y = f(y_{in})$ , sebagai contohnya adalah fungsi *sigmoid logistic* ( sebuah kurva berbentuk S) yang memenuhi persamaan :

$$f(x) = \frac{1}{1 + \exp(-x)}$$

atau fungsi aktivasi yang lain, semisal fungsi linier serta fungsi *threshold*<sup>[2][3]</sup>.

Dalam banyak kasus, digunakan fungsi aktivasi non linier. Fungsi non linier digunakan pada jaringan

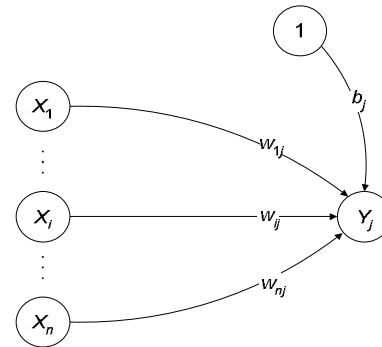
*multilayer* karena hasil pengumpanan sinyal melalui dua atau lebih *layer* dengan menggunakan fungsi linier akan diperoleh hasil yang tidak berbeda dengan apa yang diperoleh jika menggunakan *single layer* (lapis tunggal). Gambar 2 menunjukkan fungsi aktivasi *sigmoid logistic*.



Gambar 2 Fungsi aktivasi sigmoid logistic.

### 2.2 Jaringan Syaraf Tiruan Tipe Perceptron

Gambar 3 menunjukkan sebuah jaringan syaraf *perceptron* sederhana dengan  $n$  masukan dan satu keluaran, tanpa *hidden layer* (lapis tersembunyi).



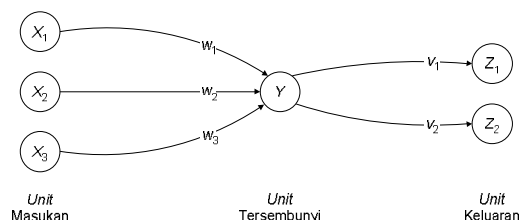
Gambar 3 Perceptron sederhana.

Neuron mempunyai sebuah *bias*  $b$ , yang dijumlahkan dengan masukan terboboti untuk membentuk masukan jaringan  $y_{in}$ :

$$y_{in} = w_{1j}x_1 + w_{2j}x_2 + \dots + w_{nj}x_n + b_j$$

Jumlah masukan ke dalam sebuah jaringan ditentukan oleh spesifikasi eksternal dari suatu permasalahan yang akan dipecahkan.

Umumnya satu neuron, bahkan dengan banyak masukan, kemungkinan tidaklah memadai untuk memecahkan suatu permasalahan. Oleh karena itu dibutuhkan jumlah neuron lebih dari satu yang dioperasikan secara paralel, dimana biasa disebut sebagai sebuah *layer* (lapis). Gambar 4 menunjukkan jaringan syaraf sangat sederhana yang terdiri dari tiga neuron pada *input layer*, satu neuron pada *hidden layer*, serta dua neuron pada *output layer*.



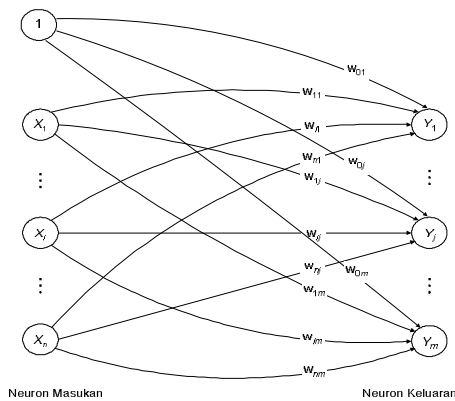
Gambar 4 Jaringan syaraf sangat sederhana.

Neuron dalam *layer* yang sama mempunyai karakter yang sama. Faktor kunci dalam menentukan karakter neuron tergantung dari fungsi aktivasi serta pola koneksi terboboti ketika suatu *layer* mengirim serta menerima sinyal. Dalam setiap *layer*, neuron-neuron biasanya mempunyai fungsi aktivasi yang sama dan pola koneksi terhadap neuron lain yang sama pula.

Penyusunan neuron ke dalam *layer* dan pola koneksi dalam dan antar *layer* disebut sebagai arsitektur jaringan. Kebanyakan jaringan syaraf mempunyai *input layer* dengan aktivasi setiap *unit* sama dengan sinyal masukan eksternal. Dengan demikian, pada *input layer* tidak terdapat adanya fungsi aktivasi.

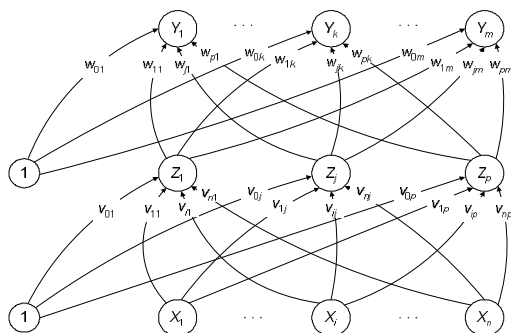
Jaringan syaraf seringkali diklasifikasikan sebagai *single layer* atau *multilayer*. Dalam menentukan jumlah *layer*, *unit* masukan tidak dihitung sebagai *layer*, karena pada *unit* ini tidak terjadi adanya proses komputasi. Dengan demikian, jumlah *layer* dalam suatu jaringan dapat ditentukan sebagai jumlah *layer* jalur interkoneksi terboboti antar neuron.

Gambar 5 merupakan contoh jaringan syaraf *single layer*. Jaringan syaraf ini mempunyai  $n$  neuron pada *input layer* dan  $m$  neuron pada *output layer*. Pada jaringan syaraf ini tidak terdapat adanya *hidden layer*.



Gambar 5 Jaringan syaraf *single layer*.

Biasanya dibutuhkan lebih dari satu *layer* untuk memecahkan suatu permasalahan. Jaringan dengan satu *hidden layer* sudah mencukupi untuk mengaproksimasi pemetaan dari pola masukan terhadap pola keluaran. Akan tetapi, dua *hidden layer* bisa membuat proses *training* menjadi lebih mudah dalam beberapa situasi. Jaringan yang terlihat pada Gambar 6 merupakan jaringan syaraf *multilayer* dengan satu *hidden layer*.



Gambar 6 Jaringan syaraf *multilayer*.

### 2.3 Algoritma Pelatihan Backpropagation<sup>[2]</sup>

Aturan pembelajaran *backpropagation* dikembangkan dari *delta rule* dengan menambahkan *hidden layer*. Melatih jaringan dengan menggunakan metode *backpropagation* melibatkan tiga tahapan: *feedforward* pola pelatihan masukan, *backpropagation* terhadap *error*, serta penyesuaian bobot.

Algoritma pelatihan *backpropagation* adalah sebagai berikut:

- Langkah 0. Inialisasi bobot.
- Langkah 1. Ketika kondisi berhenti salah, lakukan langkah 2-9.
- Langkah 2. Untuk masing-masing pasangan *training*, lakukan langkah 3-8.

*Feedforward*:

- Langkah 3. Masing-masing neuron masukan ( $X_i, i = 1, \dots, n$ ) menerima sinyal masukan  $x_i$  dan menghantarkan sinyal ini ke semua *unit* dalam *layer* di atasnya (*unit-unit* tersembunyi).
- Langkah 4. Masing-masing neuron tersembunyi ( $Z_j, J = 1, \dots, n$ ) menjumlahkan sinyal masukan terboboti,

$$z\_in_j = v_{0j} + \sum_{i=1}^n x_i v_{ij},$$

hitung fungsi aktivasi untuk menghitung sinyal keluaran,

$$z_j = f(z\_in_j),$$

lalu mengirim sinyal ini ke semua *unit* pada *layer* di atasnya (*unit* keluaran).

- Langkah 5. Masing-masing *unit* keluaran ( $Y_k, k = 1, \dots, m$ ) menjumlahkan sinyal masukan terboboti,

$$y\_in_k = w_{0j} + \sum_{j=1}^p z_j w_{jk}$$

serta menghitung fungsi aktivasinya untuk menghitung sinyal keluaran,

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

*Backpropagation of error*:

- Langkah 6. Masing-masing *unit* keluaran ( $Y_k, k = 1, \dots, m$ ) menerima pola target terhadap pola masukan, kemudian menghitung informasi *error*nya,

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

menghitung koreksi bobot,

$$\Delta w_{jk} = \alpha \delta_k z_j,$$

menghitung koreksi *bias*,

$$\Delta w_{0k} = \alpha \delta_k,$$

dan mengirim  $\delta_k$  ke *unit* di *layer* dibawahnya.

- Langkah 7. Masing-masing *unit* tersembunyi ( $Z_j, j = 1, \dots, p$ ) menjumlahkan masukan *deltanya* (dari *unit* pada *layer* di atasnya),

$$\delta\_in_j = \sum_{k=1}^m \delta_k w_{jk},$$

kalikan dengan turunan fungsi aktivasi untuk menghitung informasi *error*,

$\delta_j = \delta_{in_j} f'(z_{in_j}),$   
 menghitung koreksi bobot,  
 $\Delta v_{ij} = \alpha \delta_j x_i,$   
 kemudian menghitung koreksi bias,  
 $\Delta v_{0j} = \alpha \delta_j.$

Update bobot dan bias:

Langkah 8. Masing-masing unit keluaran ( $Y_k, k = 1, \dots, m$ ) update bias dan bobotnya ( $j = 0, \dots, p$ ):

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

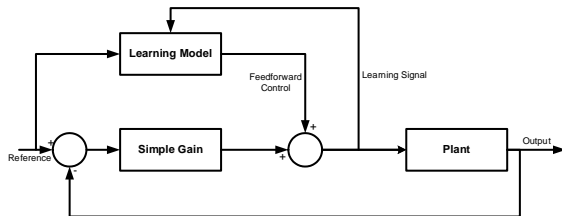
Masing-masing unit tersembunyi ( $Z_j, j = 1, \dots, p$ ) melakukan update bias dan bobotnya ( $i = 0, \dots, n$ ):

$$V_{ij}(\text{baru}) = v_{ij}(\text{lama}) + \Delta v_{ij}.$$

Langkah 9. Uji syarat henti.

### 2.4 Fixed Stabilizing Controller<sup>[3]</sup>

Salah satu skema *direct learning control* yang paling sederhana adalah sebagaimana terlihat pada Gambar 7, dimana sebuah kontroler *fixed, stabilizing*, linier, *feedback controller* digunakan untuk melatih jaringan syaraf tiruan. Kontroler linier didesain demikian hingga sistem loop tertutup menjadi stabil dalam setiap daerah operasi dan sinyal dari kontroler tersebut menjadi sinyal *training* bagi modul pembelajaran.

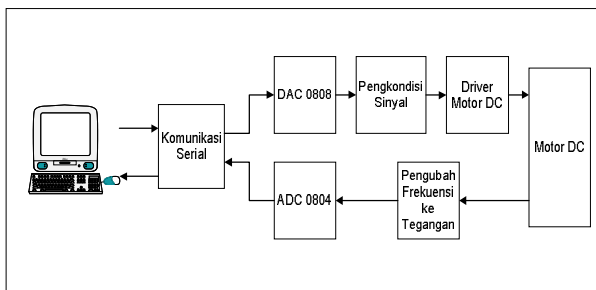


Gambar 7 Fixed Stabilizing Controller<sup>[3][8]</sup>.

Pada skema ini jaringan syaraf tiruan berperan sebagai model inverse dinamis dari *plant*. Model ini menggunakan sinyal referensi dan keluaran aktual *plant* sebagai masukan serta *error* kontrol umpan balik sebagai algoritma adaptasi. Ketika jaringan syaraf tiruan telah terlatih, maka sinyal *error* akan mengalami konvergensi menuju nol. Dengan demikian kontroler jaringan syaraf tiruan akan berlatih mengambil peran kontroler umpan balik.

### BAB III PERANCANGAN SISTEM

Secara umum sistem yang akan dibuat ditunjukkan pada Gambar 8.



Gambar 8 Diagram blok perancangan sistem.

Penjelasan dari masing – masing blok adalah sebagai berikut :

#### 1. DAC 0808

Rangkaian DAC berfungsi untuk mengubah data digital berupa masukan referensi kecepatan motor dari komputer menjadi data analog berupa tegangan.

#### 2. Pengkondisi sinyal

Pengkondisi sinyal berfungsi untuk menguatkan tegangan keluaran dari DAC. Keluaran DAC maksimal adalah 5 V, sedangkan tegangan maksimal yang dibutuhkan adalah 13 V sehingga perlu dikuatkan sebesar 2,6 kali.

#### 3. Driver motor DC

Untuk menggerakkan motor DC diperlukan arus yang cukup besar, sedangkan *output* pengkondisi sinyal tidak memiliki arus yang cukup untuk menggerakkan motor DC. Oleh karena itu, diperlukan sebuah penggerak motor DC yang tidak lain adalah penguat arus untuk menguatkan tegangan kontrol dari pengkondisi sinyal.

#### 4. Motor DC

Motor DC yang digunakan dilengkapi dengan optokopler beserta piringan berkode yang terpasang pada poros motor.

#### 5. Pengubah frekuensi ke tegangan

Rangkaian pada blok ini menggunakan LM2917N sebagai komponen utama yang berfungsi mengubah data frekuensi dari optokopler menjadi tegangan yang sesuai dengan besarnya frekuensi masukan.

#### 6. ADC 0804

Rangkaian ADC yang digunakan untuk mengubah besaran analog dari pengubah frekuensi ke tegangan menjadi data digital sehingga dapat diolah di komputer.

#### 7. Komunikasi serial

Mikrokontroler AT89S51 diposisikan sebagai sarana komunikasi serial, yang mengatur aliran data dari ADC ke komputer atau dari komputer ke DAC.

#### 8. Komputer.

Komputer digunakan sebagai unit kontroler untuk mengaplikasikan algoritma jaringan syaraf tiruan *backpropagation*. Disamping itu, komputer juga digunakan untuk menampilkan grafik respon dan menyimpan data ke dalam file untuk keperluan analisis.

### BAB IV PENGUJIAN DAN ANALISIS

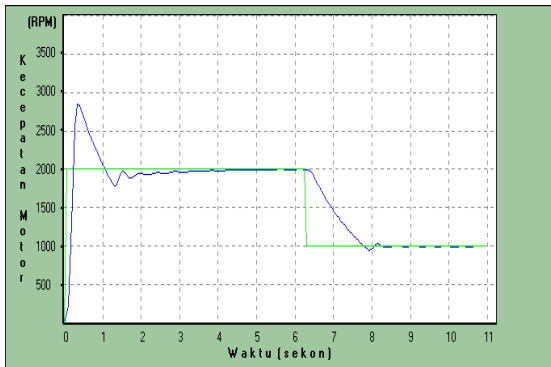
Pemilihan bobot dan *bias* awal akan menentukan kecepatan respon awal *plant*. Bobot dan *bias* awal yang dipilih harus dalam nilai yang relatif kecil. Nilai bobot dan *bias* awal yang dipilih untuk analisis adalah 0,05.

#### 4.1 Pengaruh Learning Rate terhadap Respon Transien Plant

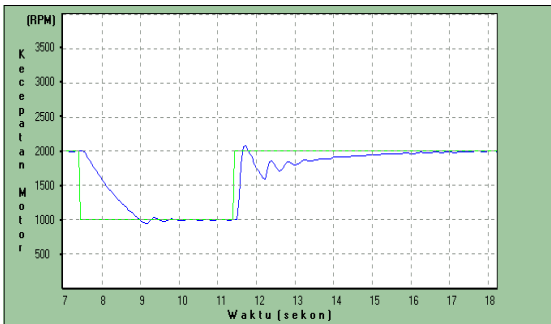
Untuk mengetahui pengaruh nilai *learning rate* pada unjuk kerja jaringan syaraf dilakukan dengan cara membandingkan dua nilai *learning rate* yang berbeda, yaitu 0,3 dan 0,7, sementara parameter lainnya diberikan dengan nilai tetap. Pada percobaan ini nilai *momentum*, jumlah *hidden layer*, dan penguat proporsional adalah 0,2, 2, dan 1,3.

Pada saat awal, respon *plant* mengalami *overshoot* sebesar 43,75 % dan waktu yang dibutuhkan untuk mencapai kondisi tunak pada kecepatan referensi 2000 RPM sekitar 5,6 detik. Dengan menggunakan

bobot-bobot hasil pembelajaran, dibutuhkan waktu sekitar 2,25 detik untuk mencapai kondisi tunak pada kecepatan 1000 RPM. Gambar 9 menunjukkan bagaimana respon transien *plant* pada saat awal dan pada saat kecepatan referensi diturunkan.

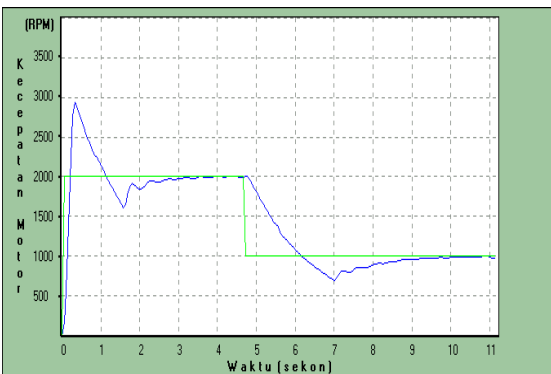


Gambar 9 Respon transien *plant* dari 2000 RPM ke 1000 RPM dengan nilai *learning rate* = 0,3, *momentum* = 0,2, jumlah *hidden layer* = 2, dan penguat proporsional = 1,3.

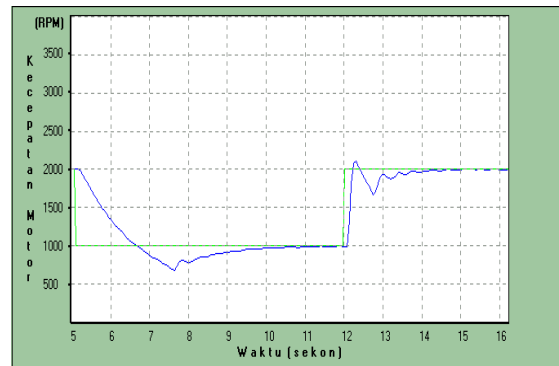


Gambar 10 Respon transien *plant* dari 1000 RPM ke 2000 RPM dengan nilai *learning rate* = 0,3, *momentum* = 0,2, jumlah *hidden layer* = 2, dan penguat proporsional = 1,3.

Gambar 10 menunjukkan pengaruh *learning rate* terhadap respon *plant* ketika kecepatan motor dinaikkan dari 1000 RPM menjadi 2000 RPM. Dengan menggunakan bobot-bobot hasil pembelajaran, respon *plant* mengalami *overshoot* 5 % dengan waktu penetapan sekitar 6,1 detik.



Gambar 11 Respon transien *plant* dari 2000 RPM ke 1000 RPM dengan nilai *learning rate* = 0,7, *momentum* = 0,2, jumlah *hidden layer* = 2, dan penguat proporsional = 1,3.



Gambar 12 Respon transien *plant* dari 1000 RPM ke 2000 RPM dengan nilai *learning rate* = 0,7, *momentum* = 0,2, jumlah *hidden layer* = 2, dan penguat proporsional = 1,3.

Gambar 11 menunjukkan perubahan respon transien *plant* dengan nilai *learning rate* = 0,7. Pada saat awal, respon *plant* mengalami *overshoot* sebesar 47,5 % dan waktu yang dibutuhkan untuk mencapai kondisi tunak sekitar 3,5 detik.

Gambar 12 menunjukkan pengaruh *learning rate* ketika kecepatan motor dinaikkan dari 1000 RPM menjadi 2000 RPM. Respon *plant* mengalami *overshoot* 5 % dengan waktu penetapan sekitar 2,7 detik.

Tabel 1 Pengaruh nilai *learning rate* terhadap respon *plant* pada saat nilai *momentum* = 0,2, jumlah *hidden layer* = 2, dan penguat proporsional = 1,3.

Parameter unjuk kerja	Respon awal <i>plant</i> mencapai referensi		Respon <i>plant</i> dari kecepatan 1000 RPM ke 2000 RPM	
	<i>Overshoot</i> maksimum	Waktu penetapan	<i>Overshoot</i> maksimum	Waktu penetapan
<i>Learning rate</i> = 0,3	43,75 %	5,6 detik	5 %	6,1 detik
<i>Learning rate</i> = 0,7	47,5 %	3,5 detik	5 %	2,7 detik

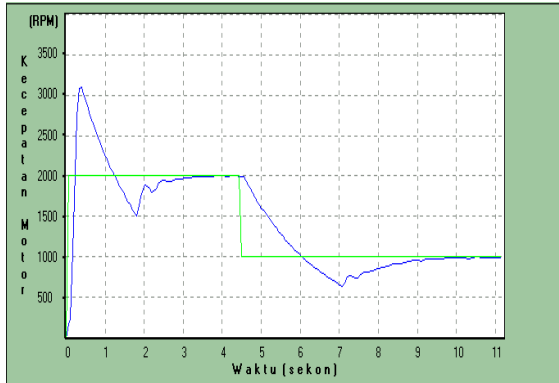
Berdasarkan Tabel 1 dan gambar respon transien *plant*, dapat disimpulkan bahwa penambahan nilai *learning rate* akan berpengaruh terhadap kecepatan respon transien *plant*, semakin besar nilai *learning rate* semakin cepat *plant* mencapai kondisi tunak. Penambahan nilai *learning rate* juga akan menambah *overshoot* *plant* awal serta terjadinya *undershoot* yang lebih besar.

#### 4.2 Pengaruh *Momentum* terhadap Respon Transien *Plant*

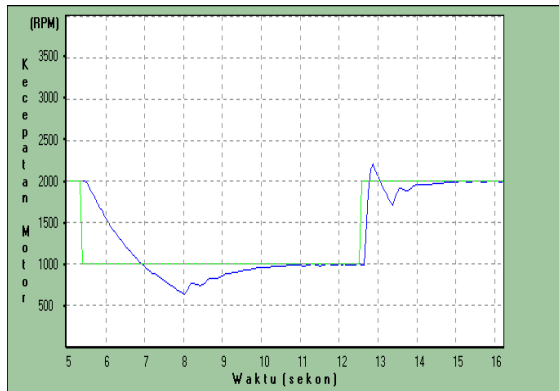
*Momentum* adalah parameter tambahan yang sebenarnya tidak harus ada dalam algoritma *backpropagation*, akan tetapi dengan adanya penambahan *momentum* telah terbukti semakin meningkatkan kecepatan pembelajaran jaringan syaraf mencapai sinyal terharap yang diinginkan.

Untuk mengetahui pengaruh langsung dari *momentum* terhadap respon transien *plant*, dilakukan pengujian terhadap dua jaringan syaraf dengan nilai *momentum* yang berbeda. Nilai *momentum* yang diujikan pada percobaan ini adalah 0,3 dan 0,6. Pada percobaan ini, untuk nilai *learning rate*, jumlah *hidden layer*, dan penguat proporsional secara berturut-turut adalah 0,7, 2, dan 1,4.

Gambar 13 menunjukkan respon *plant* pada saat awal. *Plant* mengalami *overshoot* sebesar 55 % dan waktu yang dibutuhkan untuk mencapai kondisi tunak pada kecepatan 2000 RPM sekitar 3,8 detik.

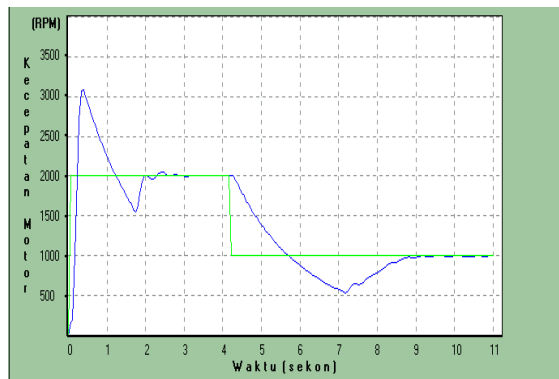


Gambar 13 Respon transien *plant* dari 2000 RPM ke 1000 RPM dengan nilai *learning rate* = 0,7, *momentum* = 0,3, jumlah *hidden layer* = 2, dan penguat proporsional = 1,4.

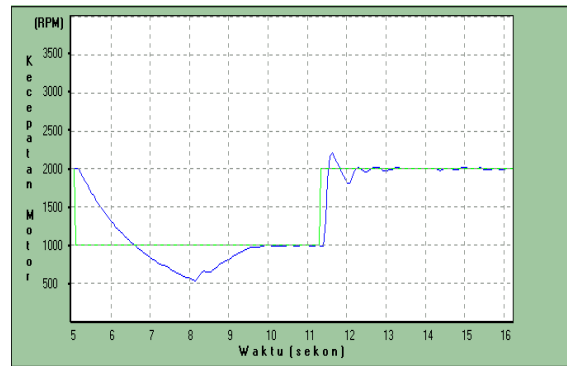


Gambar 14 Respon transien *plant* dari 1000 RPM ke 2000 RPM dengan nilai *learning rate* = 0,7, *momentum* = 0,3, jumlah *hidden layer* = 2, dan penguat proporsional = 1,4.

Pengaruh nilai *momentum* sebesar 0,3 dapat diketahui ketika kecepatan motor dinaikkan dari 1000 RPM menjadi 2000 RPM. Dengan menggunakan bobot-bobot hasil pembelajaran, respon *plant* mengalami *overshoot* 10 % dengan waktu penetapan yang dibutuhkan sekitar 2,5 detik. Gambar 14 menunjukkan pengaruh respon transien *plant* dengan nilai *momentum* sebesar 0,3.



Gambar 15 Respon transien *plant* dari 2000 RPM ke 1000 RPM dengan nilai *learning rate* = 0,7, *momentum* = 0,6, jumlah *hidden layer* = 2, dan penguat proporsional = 1,4.



Gambar 16 Respon transien *plant* dari 1000 RPM ke 2000 RPM dengan nilai *learning rate* = 0,7, *momentum* = 0,6, jumlah *hidden layer* = 2, dan penguat proporsional = 1,4.

Gambar 15 dan Gambar 16 menunjukkan perubahan respon transien *plant* untuk nilai *momentum* 0,6. Pada saat awal, respon *plant* mengalami *overshoot* sebesar 55 % dan waktu yang dibutuhkan untuk mencapai kondisi tunak sekitar 2,5 detik. Ketika kecepatan motor dinaikkan dari 1000 RPM menjadi 2000 RPM, respon *plant* mengalami *overshoot* 10 % dengan waktu penetapan yang dibutuhkan sekitar 1,5 detik.

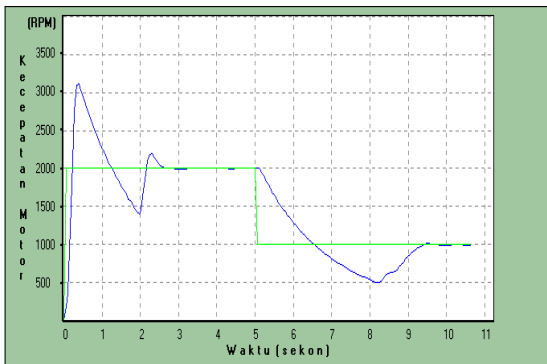
Tabel 2 Pengaruh nilai *momentum* terhadap respon *plant* pada saat nilai *learning rate* = 0,7, jumlah *hidden layer* = 2, dan penguat proporsional = 1,4.

Parameter unjuk kerja	Respon awal <i>plant</i> mencapai referensi		Respon <i>plant</i> dari kecepatan 1000 RPM ke 2000 RPM	
	<i>Overshoot</i> maksimum	Waktu penetapan	<i>Overshoot</i> maksimum	Waktu penetapan
<i>Momentu</i> $m = 0,3$	55 %	3,8 detik	10 %	2,5 detik
<i>Momentu</i> $m = 0,6$	55 %	2,5 detik	10 %	1,9 detik

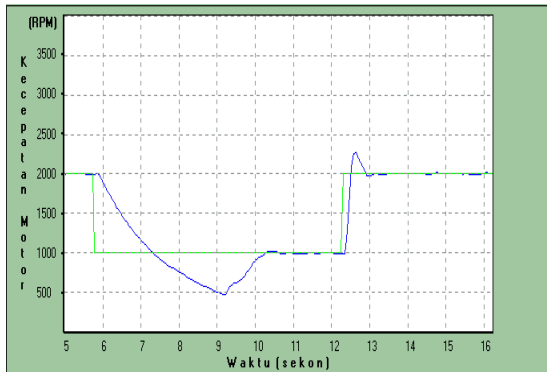
Dari Tabel 2 dan gambar respon transien *plant* dapat disimpulkan, penambahan nilai *momentum* akan berpengaruh terhadap respon *plant* mencapai kondisi tunak. Semakin besar nilai *momentum*, semakin cepat *plant* mencapai kondisi tunak. Penambahan *momentum* juga akan berdampak terhadap penambahan *undershoot* dari *plant*.

#### 4.3 Pengaruh Jumlah *Hidden Layer* terhadap Respon Transien *Plant*

Dalam beberapa referensi, penentuan jumlah *hidden layer* yang diperlukan dalam menyelesaikan suatu kasus tidak dijelaskan secara mendetail. Dalam hal ini tidak ada panduan yang jelas mengenai berapakah jumlah *hidden layer* yang paling optimal untuk memecahkan suatu permasalahan. Ada yang mengatakan bahwa satu *hidden layer* sudah mencukupi, namun dua *hidden layer* akan menjadikan fungsi jaringan syaraf sebagai aproksimator menjadi lebih baik<sup>[2]</sup>. Ada pula yang mengatakan bahwa pemilihan jumlah *hidden layer* lebih ditentukan secara empiris daripada perhitungan matematis<sup>[13]</sup>. Pada percobaan kali ini, parameter jaringan syaraf yang dipilih untuk nilai-*learning rate*, *momentum*, dan penguat proporsional secara berturut-turut adalah 0,6, 0,5, 1,2. Jumlah *hidden layer* yang dibandingkan adalah 1 dan 2.

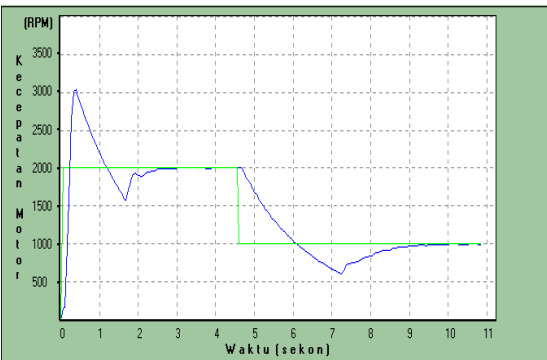


Gambar 17 Respon transien *plant* dari 2000 RPM ke 1000 RPM dengan nilai *learning rate* = 0,6, *momentum* = 0,5, jumlah *hidden layer* = 1, dan penguat proporsional = 1,2.

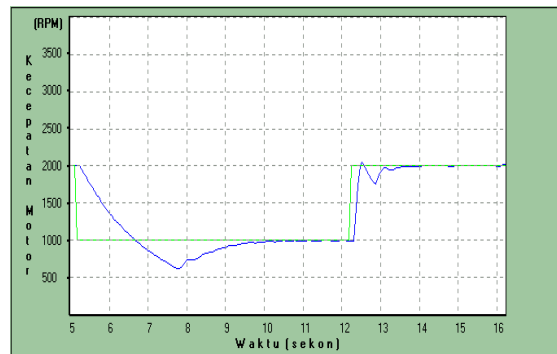


Gambar 18 Respon transien *plant* dari 1000 RPM ke 2000 RPM dengan nilai *learning rate* = 0,6, *momentum* = 0,5, jumlah *hidden layer* = 1, dan penguat proporsional = 1,2.

Gambar 17 dan Gambar 18 menunjukkan respon transien *plant* pada jaringan syaraf dengan satu *hidden layer*. Respon *plant* mengalami *overshoot* sebesar 55 % dan waktu yang dibutuhkan untuk mencapai kondisi tunak pada kecepatan 2000 RPM sekitar 2,5 detik. Pengaruh satu *hidden layer* terhadap unjuk kerja jaringan dapat diketahui ketika kecepatan motor dinaikkan dari 1000 RPM menjadi 2000 RPM sebagaimana ditunjukkan Gambar 18. Respon *plant* mengalami *overshoot* 12,5 % dengan waktu penetapan yang dibutuhkan sekitar 1 detik.



Gambar 19 Respon transien *plant* dari 2000 RPM ke 1000 RPM dengan nilai *learning rate* = 0,6, *momentum* = 0,5, jumlah *hidden layer* = 2, dan penguat proporsional = 1,2.



Gambar 20 Respon transien *plant* dari 1000 RPM ke 2000 RPM dengan nilai *learning rate* = 0,6, *momentum* = 0,5, jumlah *hidden layer* = 2, dan penguat proporsional = 1,2.

Gambar 19 dan Gambar 20 menunjukkan perubahan respon transien *plant* dengan dua *hidden layer*. Pada saat awal, respon *plant* mengalami *overshoot* sebesar 50 % dan waktu yang dibutuhkan untuk mencapai kondisi tunak pada kecepatan 2000 RPM sekitar 2 detik. Pada saat kecepatan motor dinaikkan dari 1000 RPM menjadi 2000 RPM, respon *plant* mengalami *overshoot* 1 % dengan waktu penetapan yang dibutuhkan sekitar 1,6 detik.

Tabel 3 Pengaruh jumlah *hidden layer* terhadap respon *plant* pada saat nilai *learning rate* = 0,7, jumlah *momentum* = 0,5, dan penguat proporsional = 1,4.

Parameter unjuk kerja	Respon awal <i>plant</i> mencapai referensi		Respon <i>plant</i> dari kecepatan 1000 RPM ke 2000 RPM	
	<i>Overshoot</i> maksimum	Waktu penetapan	<i>Overshoot</i> maksimum	Waktu penetapan
<i>Hidden layer</i> = 1	55 %	2,5 detik	12,5 %	1 detik
<i>Hidden layer</i> = 2	50 %	3 detik	1 %	1,6 detik

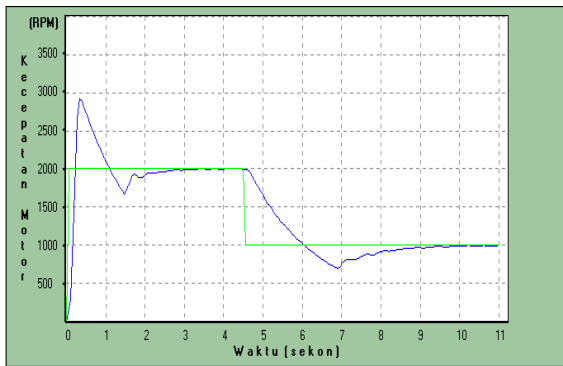
Dari Tabel 3 dapat disimpulkan, jaringan dengan satu *hidden layer* mempunyai waktu penetapan yang lebih cepat dibandingkan dengan jaringan dengan dua *hidden layer*. Pada jaringan dengan satu *hidden layer*, respon *plant* mengalami *overshoot* yang lebih besar jika dibandingkan jaringan dengan dua *hidden layer*.

#### 4.4 Pengaruh Penguat Proporsional terhadap Respon Transien *Plant*

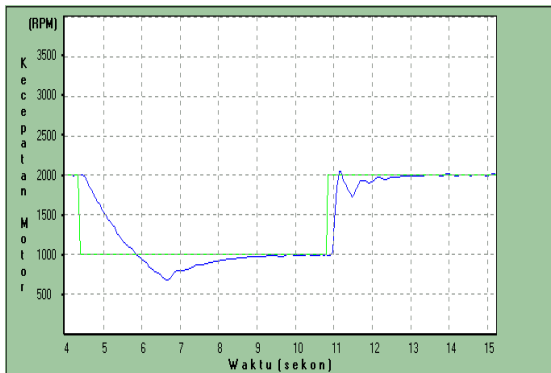
Dengan menggunakan skema kontroler *fix stabilizing controller*, maka peranan penguat proporsional sangat menentukan kondisi *plant* secara keseluruhan. Dikatakan sangat menentukan, karena sinyal terharap dari *plant* merupakan jumlah dari keluaran penguat proporsional dengan keluaran jaringan syaraf.

Parameter-parameter jaringan syaraf yang dipilih pada percobaan ini untuk *learning rate*, *momentum*, dan jumlah *hidden layer* secara berturut-turut adalah: 0,8, 0,2, dan 2. Penguat proporsional yang diujikan pada percobaan ini adalah 1,1 dan 1,5.

Pada saat awal, respon *plant* mengalami *overshoot* sebesar 49,9 % dan waktu yang dibutuhkan untuk mencapai kondisi tunak referensi 2000 RPM sekitar 3,2 detik. Setelah diperoleh bobot-bobot hasil pembelajaran, kecepatan motor diturunkan menjadi 1000 RPM. Keadaan ini dijelaskan pada Gambar 21.



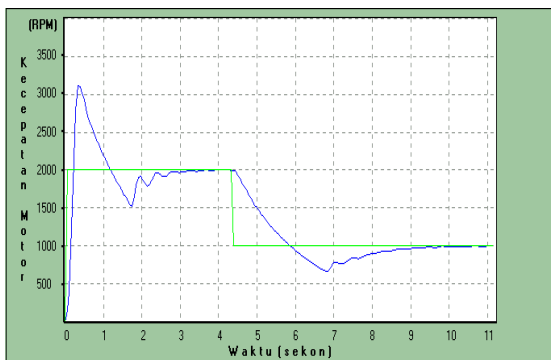
Gambar 21 Respon transien *plant* dari 2000 RPM ke 1000 RPM dengan nilai *learning rate* = 0,8, *momentum* = 0,2, jumlah *hidden layer* = 2, dan penguat proporsional = 1,1.



Gambar 22 Respon transien *plant* dari 1000 RPM ke 2000 RPM dengan nilai *learning rate* = 0,8, *momentum* = 0,2, jumlah *hidden layer* = 2, dan penguat proporsional = 1,1.

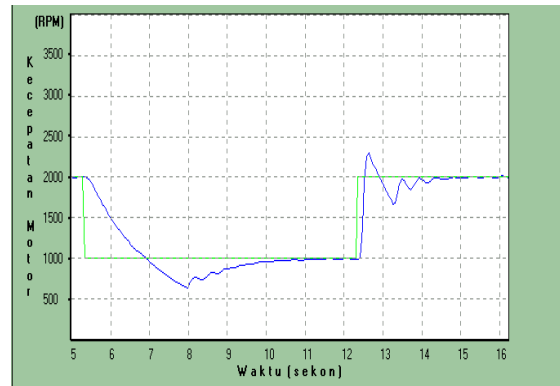
Gambar 22 menjelaskan penggunaan bobot-bobot hasil pembelajaran ketika kecepatan motor dinaikkan dari 1000 RPM menjadi 2000 RPM. Gambar 22 juga menunjukkan pengaruh penguat proporsional sebesar 1,1 akan memberikan nilai *overshoot* 1 % dengan waktu penyetapan sekitar 2,2 detik.

Gambar 23 dan Gambar 24 menunjukkan perubahan respon transien *plant* dengan penguat proporsional 1,5. Respon awal *plant* mengalami *overshoot* sebesar 55 % dan waktu yang dibutuhkan untuk mencapai kondisi tunak pada kecepatan referensi sekitar 3,6 detik.



Gambar 23 Respon transien *plant* dari 2000 RPM ke 1000 RPM dengan nilai *learning rate* = 0,8, *momentum* = 0,2, jumlah *hidden layer* = 2, dan penguat proporsional = 1,5.

Ketika kecepatan motor dinaikkan dari 1000 RPM menjadi 2000 RPM, respon *plant* mengalami *overshoot* 15 % dengan waktu penyetapan sekitar 2,5 detik. Peningkatan penguat proporsional akan memberikan *overshoot* yang lebih besar serta kecenderungan sistem mengalami osilasi.



Gambar 24 Respon transien *plant* dari 1000 RPM ke 2000 RPM dengan nilai *learning rate* = 0,8, *momentum* = 0,2, jumlah *hidden layer* = 2, dan penguat proporsional = 1,5.

Tabel 4 Pengaruh jumlah penguat proporsional terhadap respon *plant* pada saat nilai *learning rate* = 0,7, jumlah *momentum* = 0,5, dan jumlah *hidden layer* = 2.

Parameter unjuk kerja	Respon awal <i>plant</i> mencapai referensi		Respon <i>plant</i> dari kecepatan 1000 RPM ke 2000 RPM	
	<i>Overshoot</i> maksimum	Waktu penyetapan	<i>Overshoot</i> maksimum	Waktu penyetapan
Penguat proporsional = 1,1	49,9 %	3,2 detik	1 %	2,5 detik
Penguat proporsional = 1,5	55 %	3,6 detik	15 %	2,4 detik

Berdasarkan Tabel 4, dapat disimpulkan bahwa pertambahan penguat proporsional pada *plant* akan memberikan pengaruh semakin cepat *plant* mencapai kondisi tunak dengan disertai pertambahan *overshoot* maksimum *plant*. Untuk nilai penguat proporsional yang mendekati 2, *plant* cenderung mengalami osilasi (tidak stabil).

## BAB V PENUTUP

### 5.1 Kesimpulan

Berdasarkan hasil percobaan yang telah dilakukan dapat diambil beberapa kesimpulan sebagai berikut:

1. *Learning rate* pada jaringan syaraf *backpropagation* berpengaruh terhadap kecepatan *plant* mencapai kondisi tunak. Pemilihan *learning rate* yang relatif besar akan memberikan kecepatan respon yang lebih baik dibanding *learning rate* yang lebih kecil.
2. Pertambahan nilai *momentum* pada jaringan syaraf akan mempunyai pengaruh yang sama dengan pertambahan nilai *learning rate*. Semakin besar nilai *momentum*, jaringan syaraf akan semakin cepat melakukan aproksimasi.
3. Jaringan syaraf dengan satu *hidden layer* mempunyai respon transien yang lebih cepat dibandingkan



dengan jaringan syaraf dengan dua *hidden layer*. Respon jaringan syaraf dengan satu *hidden layer* juga memberikan nilai *overshoot* maksimum yang lebih besar jika dibandingkan dengan jaringan syaraf dengan dua *hidden layer*.

4. Semakin besar penguat proporsional, semakin cepat *plant* mencapai kondisi tunak. Pemilihan penguat proporsional tidak boleh terlalu besar, karena *plant* akan mengalami osilasi. Berdasarkan percobaan, nilai kritis penguat proporsional pada saat *plant* masih stabil adalah 2.
5. Berdasarkan *trial and error*, dengan menggunakan bobot dan *bias* awal sebesar 0,05, nilai-nilai parameter yang memberikan respon paling baik adalah *learning rate* = 0.75, *momentum* = 0,6, jumlah *hidden layer* = 2 dan penguat proporsional = 1.1.

## 5.2 Saran

1. Untuk lebih meningkatkan unjuk kerja jaringan syaraf, terkait dengan pengaturan nilai *learning rate* dan *momentum* dapat digabungkan dengan metode kontrol yang lain, seperti misalnya kontrol logika *fuzzy*. Penggabungan metode kontrol memungkinkan pengaturan kedua parameter ini dapat dilakukan secara otomatis, tanpa harus dilakukan pengaturan secara manual.
2. *Plant* motor DC ini dapat dikembangkan dengan memberi beban, sehingga akan dapat diketahui bagaimana unjuk kerja jaringan syaraf dalam merespon perubahan lingkungan.
3. Perluanya penggunaan PWM (*pulse width modulation*) sebagai pengganti penguat biasa pada *driver* motor, sehingga diperoleh respon yang lebih baik pada saat terjadi penurunan referensi kecepatan motor.
4. Untuk menambah ketelitian dalam akuisisi data, diperlukan adanya *interface* dengan resolusi yang lebih tinggi.

## DAFTAR PUSTAKA

1. Coughlin, Robert, Federick, Driscoll, *Penguat Operasional dan Rangkaian Terpadu Linier*, Jakarta, Erlangga.
2. Fausett, Laurene, *Fundamentals of Neural Networks: Architecture, Algorithms, and Applications*, Prentice Hall Inc., 1994.
3. Hagan Martin T., Demuth Howard B., *Neural Network for Control*, IEEE Journal.
4. Ibnu Malik, Moh., Anistardi, Bereksperimen dengan Mikrokontroler 8031, Cetakan ke-2, Elex Media Komputindo, 1999.
5. Malvino, Paul Albert, *Elektronika Komputer Digital: Pengantar Mikrokomputer (terj.)*, Edisi Ke-6, Erlangga, 1996.
6. Ogata, Katsuhiko, *Teknik Kontrol Otomatik, Jilid 1*, Erlangga, 1991.
7. Prasetya, Retna, Widodo, Catur Edi, *Interfacing Port Paralel dan Port Serial Komputer dengan Visual Basic 6.0*, Andi Yogyakarta.2004.
8. Setiawan, Iwan, *Jaringan Syaraf Tiruan untuk Aplikasi Pemodelan dan Pengendalian*, Jurusan

Teknik Elektro Fakultas Teknik Universitas Diponegoro.

9. Stich, Timothy James, Spoeerre, Julie K., Velasco, Thomas, *The Application of Artificial Neural Networks to Monitoring and Control of an Induction Hardening Process*, Journal of Industrial Technology, Volume16 No. 1, The Official Electronic Publication of the National Association of Industrial Technology, 2000.
10. Tocci, Ronald J., *Digital Systems: Principles and Application*, Edisi ke-5, Prentice Hall International Inc., 1991.
11. MSDN Library January 2001. Microsoft Developer Network. CD-ROM.
12. ....http://www.beyondlogic.org/serial/serial1.htm #41
13. ....http://cortex.snowseed.com/index.html
14. ....http://www.doc.ic.ac.uk/~nd/surprise\_96/journal/vol4/cs11/report.html
15. ....http://www.106.ibm.com/developerworks/linux/library/l-neural

## BIOGRAFI



Sorikhi dilahirkan di Pemalang tanggal 08 agustus 1979. Saat ini sedang menempuh pendidikan tinggi di Jurusan Teknik Elektro Universitas Diponegoro pada konsentrasi Kontrol. Bidang yang diminati adalah pemrograman komputer untuk aplikasi kontrol, pemrograman mikrokontroler. Email: [soryc2001@yahoo.com](mailto:soryc2001@yahoo.com)

Menyetujui dan Mengesahkan

Pembimbing I

Wahyudi, ST, MT.  
NIP. 132 086 662  
Tanggal .....

Pembimbing II

Iwan Setiawan, ST, MT.  
NIP. 132 283 183  
Tanggal .....