

## **BAB III**

### **Metode Penelitian**

#### **3.1. Variabel Penelitian dan Definisi Operasional**

Penelitian ini menggunakan satu definisi variabel operasional yaitu nilai harian indeks LQ 45.

#### **3.2. Populasi dan sampel**

Populasi objek penelitian adalah nilai harian seluruh indeks yang ada di BEJ. Nilai indeks yang menjadi sampel dalam penelitian ini adalah nilai indeks LQ 45 periode Januari 1997 – April 2007.

Hingga akhir tahun 2001 terdapat sebanyak 12 indeks harga saham yang resmi dikeluarkan oleh Bursa Efek Jakarta Yaitu IHSG (Indeks Harga Saham Gabungan), Indeks LQ 45, Indeks Syariah (*Jakarta Islamic Index / JII*), dan 9 indeks sektoral.

Pemilihan indeks LQ 45 dalam penelitian ini karena LQ 45 merupakan indeks yang dalam perhitungannya hanya melibatkan saham-saham yang aktif, memiliki kapitalisasi pasar yang besar dan memiliki fundamental yang baik. Indeks LQ 45 juga dipandang lebih mewakili kondisi pasar di Bursa Efek Jakarta dibandingkan IHSG (Agus Sartono dan Sri Zulaihati, 1998; Bima Putra, 2001 dalam Qiqin 2003). Periode penelitian yang digunakan adalah periode 2 Januari 1997 hingga 30 April 2007 (jumlah periode pengamatan 2520 hari perdagangan).

Indeks LQ 45 direview setiap 6 bulan sekali oleh otoritas bursa. Dalam menentukan saham-saham yang masuk dalam daftar Indeks LQ 45 tersebut, digunakan kriteria sebagai berikut (Bursa Efek Jakarta, 2000) :

1. Masuk dalam ranking 60 terbesar dari total transaksi saham di pasar reguler (rata-rata nilai transaksi selama 12 bulan terakhir).
2. Ranking berdasarkan kapitalisasi pasar (rata-rata kapitalisasi harian selama 12 bulan terakhir).
3. Telah tercatat di BEJ minimum 3 bulan.
4. Keadaan keuangan perusahaan dan prospek pertumbuhannya, frekuensi dan jumlah hari perdagangan transaksi pasar reguler.

Data tersebut dibagi menjadi dua bagian yaitu periode model dan periode estimasi. Periode model ditetapkan selama 2016 hari perdagangan (80% sampel) sedangkan periode estimasi ditetapkan selama 504 hari (20% sampel). Pada metode JST data pada periode model dibagi menjadi dua yaitu periode training (60% dari total sampel) dan periode evaluasi (20% dari total sampel).

Data pada periode model digunakan untuk menghasilkan model yang digunakan untuk melakukan peramalan. sedangkan data pada periode estimasi digunakan untuk melihat tingkat kesalahan(*error*) yang dihasilkan oleh metode-metode yang digunakan. Semakin kecil *error* yang dimiliki sebuah model maka semakin baik model tersebut digunakan dalam melakukan peramalan dalam analisis teknikal.

### **3.3. Jenis dan Sumber Data**

Adapun data yang dipakai dalam penelitian ini adalah data sekunder yang diambil dan diolah dari *JSX Statistics dan Indonesian Capital Market Directory* (ICMD) 1997-2007. Pemilihan Bursa Efek Jakarta sebagai sumber utama dalam penelitian ini karena Bursa Efek Jakarta merupakan pasar saham terbesar dan paling representatif di Indonesia.

Data ini diperoleh dengan cara membaca buku dan media cetak yang menyajikan data yang berhubungan dengan masalah yang diteliti, *Daily Trading JSX 2002-2005* dan *Pojok BEJ* Fakultas Ekonomi Universitas Diponegoro.

### **3.4. Metode Pengumpulan Data**

Metode pengumpulan data yang digunakan dalam penelitian ini adalah *non-participant observer*, dimana peneliti hanya mengamati data yang sudah tersedia tanpa ikut menjadi bagian dari suatu sistem data. Data yang dibutuhkan adalah nilai harian indeks LQ 45 periode 2 januari 1997 hingga 30 april 2007. Data diperoleh dari Bursa Efek Jakarta melalui *Pojok BEJ* Universitas diponegoro dan perpustakaan Fakultas Ekonomi undip.

### **3.5. Metode analisis**

#### **3.5.1 Multifraktal**

Ide dasar pengembangan eksponen Hurst adalah model otokorelasi. Pada otokorelasi biasa menggunakan data sebagai satu kesatuan deret waktu, sedangkan pada analisis R/S (*Rescaled range Analysis*, sebutan untuk

mendapatkan eksponen Hurst) data dipecah menjadi beberapa bagian, dan analisis R/S dilakukan terhadap masing-masing data yang terpecah. Misalkan kita memiliki data deret waktu  $Y_1, \dots, Y_N$  data ini kemudian dipecah menjadi beberapa bagian dengan panjang yang sama, dengan masing-masing terdiri atas  $y_1, \dots, y_t$ .

Nilai R diperoleh dari persamaan :

$$R_N = \text{Maks}X_{(t,N)} - \text{min}X_{(t,N)}$$

Nilai X diperoleh dari persamaan :

$$X_{t,N} = \sum_{u=1}^t (x_u - \mu_N)$$

Dimana  $\mu_N$  adalah rata-rata deret waktu selama periode N. Nilai S merupakan deviasi standard data deret waktu yang kita miliki. Dapat diperoleh dengan persamaan

$$S_N = \left[ \frac{1}{N} \sum_{i=1}^N (y_i - \hat{y}_N)^2 \right]^{\frac{1}{2}}$$

Rasio R/S dari R dan Deviasi Standard S dari deret waktu utama dapat dihitung dengan hukum empiris sebagai berikut (Yao dkk, 1999) :  $R/S = N^H$ .

Nilai Eksponen Hurst dapat dihitung sebagai berikut :

$$H = \log(R/S)/\log(N)$$

Dimana nilai H berada diantara 0 dan 1 ( $0 < H < 1$ ). Estimasi nilai H dapat diperoleh dengan melakukan perhitungan slope grafik  $\log R/S$  terhadap N menggunakan regresi.

Nilai eksponen Hurst (H) menggambarkan probabilitas bahwa dua event konsekutif dapat muncul. Jika nilai  $H = 0,5$  maka data deret waktu bertipe acak, terdiri atas event-event yang tidak berhubungan. Nilai H selain 0,5 menggambarkan bahwa objek observasi tidak independen, Ketika  $0 \leq H < 0,5$ , sistem yang diteliti merupakan deret ergodic dan antipersisten dengan frekuensi pembalikan yang tinggi dan volatilitas yang tinggi. Disamping kelaziman yang ada mengenai konsep pembalikan rata-rata pada literatur ekonomi dan keuangan, hanya ditemukan beberapa deret waktu antipersisten. Bagi kondisi ketiga ( $0,5 < H \leq 1,0$ ), H mendeskripsikan deret persisten dan adanya tren yang ditunjukkan oleh efek ingatan jangka panjang (*long memory effects*). Kekuatan bias bergantung pada seberapa besar nilai H diatas 0,5. Semakin rendah nilai H, lebih banyak *noise* pada sistem dan deret lebih mendekati keacakan.

**Tabel 3.1**

**Ringkasan Arti Nilai H**

<b>Nilai H</b>	<b>Arti</b>
$0 \leq H < 0,5$	<i>Ergodic</i> dan antipersisten, frekuensi pembalikan dan volatilitas tinggi
$H = 0,5$	Sepenuhnya acak
$0,5 < H \leq 1$	Persisten dan kecenderungan pembentukan tren dengan adanya efek ingatan jangka panjang ( <i>long memory effects</i> )

Sumber : Yao dkk, 1999 diringkas

### 3.5.2 Metode ARIMA

Teknik analisis data dengan metode ARIMA dilakukan karena merupakan teknik untuk mencari pola yang paling cocok dari sekelompok data (*curve fitting*), dengan demikian ARIMA memanfaatkan sepenuhnya data masa lalu dan sekarang untuk melakukan peramalan jangka pendek yang akurat (Sugiarto dan Harijono, 2000). ARIMA seringkali ditulis sebagai ARIMA (p,d,q) yang memiliki arti bahwa p adalah orde koefisien autokorelasi, d adalah orde / jumlah diferensiasi yang dilakukan (hanya digunakan apabila data bersifat non-stasioner) (Sugiharto dan Harijono, 2000) dan q adalah orde dalam koefisien rata-rata bergerak (*moving average*).

Peramalan dengan menggunakan model ARIMA dapat dilakukan dengan rumus :

$$Y_t = \gamma_0 + \partial_1 Y_{t-1} + \partial_2 Y_{t-2} + \dots + \partial_n Y_{t-p} - \lambda_1 e_{t-1} - \lambda_2 e_{t-2} - \lambda_n e_{t-q}$$

Keterangan :

B	: Koefisien Regresi
$Y_T$	: Variabel dependen pada waktu t
$Y_{t-1} \dots Y_{t-p}$	: Variabel lag
$e_t$	: Residual term
$W_1 \dots W_q$	: Bobot
$e_{t-1} \dots e_{t-p}$	: nilai sebelumnya atau residual

### 3.5.2.1 Stasioneritas data

Data yang tidak stasioner memiliki rata-rata dan varian yang tidak konstan sepanjang waktu. Dengan kata lain, secara ekstrim data stasioner adalah data yang tidak mengalami kenaikan dan penurunan. Selanjutnya regresi yang menggunakan data yang tidak stasioner biasanya mengarah kepada regresi lancung. Permasalahan ini muncul diakibatkan oleh variabel (dependen dan independen) runtun waktu terdapat tren yang kuat (dengan pergerakan yang menurun maupun meningkat). Adanya tren akan menghasilkan nilai  $R^2$  yang tinggi, tetapi keterkaitan antar variabel akan rendah (Firmansyah, 2000).

Model ARIMA mengasumsikan bahwa data masukan harus stasioner. Apabila data masukan tidak stasioner perlu dilakukan penyesuaian untuk menghasilkan data yang stasioner. Salah satu cara yang umum dipakai adalah metode pembedaan (*differencing*). Metode ini dilakukan dengan cara mengurangi nilai data pada suatu periode dengan nilai data periode sebelumnya

Untuk keperluan pengujian stasioneritas, dapat dilakukan dengan beberapa metode seperti *autocorrelation function (correlogram)*, uji akar-akar unit dan derajat integrasi.

a. Pengujian stasioneritas berdasarkan correlogram

Suatu pengujian sederhana terhadap stasioneritas data adalah dengan menggunakan fungsi koefisien autokorelasi (*autocorrelation function / ACF*). Koefisien ini menunjukkan keeratan hubungan antara nilai variabel yang sama tetapi pada waktu yang berbeda. Correlogram merupakan peta / grafik dari nilai ACF pada berbagai lag.

Secara matematis rumus koefisien autokorelasi adalah (Sugiharto dan Harijono, 2000:183) :

$$rk = \frac{\sum_{i=1}^{n-k} (Y_t - \bar{Y})(Y_{t-k} - \bar{Y})}{\sum_{i=1}^n (Y_t - \bar{Y})^2}$$

Untuk menentukan apakah nilai koefisien autokorelasi berbeda secara statistik dari nol dilakukan sebuah pengujian. Suatu runtun waktu dikatakan stasioner atau menunjukkan kesalahan random adalah jika koefisien autokorelasi untuk semua lag secara statistik tidak berbeda signifikan dari nol atau berbeda dari nol hanya untuk beberapa lag kedepan. Untuk itu perlu dihitung kesalahan standard dengan rumus :

$$se_{rk} = \frac{1}{\sqrt{n}}$$

Dimana n menunjukkan jumlah observasi. Dengan interval kepercayaan yang dipilih, misalnya 95 persen, maka batas signifikansi koefisien autokorelasi adalah :

$$- Z_{\alpha/2} \times se_{rk} \text{ s.d. } Z_{\alpha/2} \times se_{rk}$$

Suatu koefisien autokorelasi disimpulkan tidak berbeda secara signifikan dari nol apabila nilainya berada diantara rentang tersebut dan sebaliknya. Apabila koefisien autokorelasi berada diluar rentang, dapat disimpulkan koefisien tersebut signifikan, yang berarti ada hubungan signifikan antara nilai suatu variabel dengan nilai variabel itu sendiri dengan *time lag* 1 periode.

b. Uji akar-akar unit dan derajat integrasi

Sebuah tes stasioneritas (atau non-stasioneritas) yang menjadi sangat populer beberapa tahun belakangan adalah uji akar-akar unit (*unit root test*). Stasioneritas dapat diperiksa dengan mencari apakah data runtun waktu mengandung akar unit (*unit root*). Terdapat berbagai metode untuk melakukan uji akar unit diantaranya dickey-fuller, Augmented Dickey Fuller, Dickey-Fuller DLS (ERS), Philips-Perron, Kwiatkowski-Philips-Schmidt-Shin, Elliot-Rothenberg-Stock Point-Optimal, dan Ng-Perron. Dalam penelitian ini akan digunakan uji Augmented Dickey-Fuller untuk menentukan apakah suatu data runtun waktu mengandung akar unit atau bersifat non-stasioner.

Untuk memperoleh gambaran mengenai uji akar-akar unit, ditaksir model autoregresif berikut ini dengan OLS (Insukrindo, 1994; Gujarati, 1995 dalam Firmansyah, 2000) :

$$DX_t = a_0 + a_1 BX_t + \sum_{i=1}^k b_i B^i DX_t$$

$$DX_t = a_0 + a_1 T + a_2 BX_t + \sum_{i=1}^k d^i B_i DX_t$$

Dimana,  $DX_t = X_t - X_{t-1}$ ,  $BX = X_{t-1}$ , T = tren waktu,  $X_t$  = variabel yang diamati pada periode t. Selanjutnya dihitung statistik ADF. Nilai ADF digunakan untuk uji hipotesis bahwa  $a_1=0$  dan  $c_2=0$  ditunjukkan oleh nilai t statistik hitung pada koefisien  $BX_t$  pada persamaan diatas. Jumlah kelambanan k ditentukan oleh  $k=n^{1/5}$ , dimana n = jumlah observasi. Nilai kritis (tabel) untuk kedua uji terkait dapat dilihat pada Fuller, 1976; Guilky dan Schmidt, 1989 (Insukrindo, 1994:130 dalam Firmansyah, 2000). Runtun waktu yang diamati stasioner jika memiliki

nilai ADF lebih besar dari nilai kritis. Beberapa piranti lunak ekonometrika seperti EViews, SPlus, dan R menyediakan nilai kritis ini setiap kali kita melakukan running data.

Uji derajat integrasi adalah uji yang dilakukan untuk mengetahui pada derajat berapakah data yang diamati stasioner. Uji ini mirip atau merupakan perluasan uji akar-akar unit, dilakukan jika data yang diamati ternyata tidak stasioner sebagaimana direkomendasikan oleh uji akar-akar unit. Bentuk umum regresinya adalah :

$$D2X_t = e_0 + e_t BDX_t + \sum_{i=1}^k f_i B^i D2X_t$$

$$D2X_t = g_0 + g_1 T + g_2 BDX_t + \sum_{i=1}^k h_i B^i D2X_t$$

Dimana,  $D2X_t = DX_t - DX_{t-1}$ ,  $BDX_t = DX_{t-1}$ , selanjutnya pengujiannya sama dengan uji akar-akar unit. Jika pada derajat pertama ini data masih belum stasioner, maka uji integrasi perlu dilanjutkan pada derajat berikutnya sampai memperoleh suatu kondisi stasioner.

### 3.5.2.2. Tahapan Metode ARIMA

Metode ARIMA menggunakan pendekatan iteratif dalam mengidentifikasi suatu model yang paling tepat dari berbagai model yang ada. Model sementara yang telah dipilih diuji lagi dengan data historis untuk melihat apakah model sementara yang terbentuk tersebut sudah memadai atau belum. Model sudah dianggap memadai apabila residual (selisih hasil peramalan dengan data historis) terdistribusi secara acak, kecil dan independen satu sama lain. Langkah-langkah

penerapan metode ARIMA secara berturut-turut adalah : identifikasi model, estimasi parameter model, *diagnostic checking*, dan peramalan (*forecasting*)

a. Identifikasi model

Seperti yang dijelaskan sebelumnya bahwa model ARIMA hanya dapat diterapkan untuk deret waktu yang stasioner. Oleh karena itu, pertama kali yang harus dilakukan adalah menyelidiki apakah data yang kita gunakan sudah stasioner atau belum. Jika data tidak stasioner, yang perlu dilakukan adalah memeriksa pada perbedaan beberapa data akan stasioner, yaitu menentukan berapa nilai  $d$ . Proses ini dapat dilakukan dengan menggunakan koefisien ACF (*Auto Correlation Function*), atau uji akar-akar unit (*unit roots test*) dan derajat integrasi. Jika data sudah stasioner sehingga tidak dilakukan perbedaan terhadap data runtun waktu maka  $d$  diberi nilai 0.

Disamping menentukan  $d$ , pada tahap ini juga ditentukan berapa jumlah nilai lag residual ( $q$ ) dan nilai lag dependen ( $p$ ) yang digunakan dalam model. Alat utama yang digunakan untuk mengidentifikasi  $q$  dan  $p$  adalah ACF dan PACF (*Partial Auto Correlation Funtion / Koefisien Autokorelasi Parsial*), dan correlogram yang menunjukkan plot nilai ACF dan PACF terhadap lag.

Koefisien autokorelasi parsial mengukur tingkat keeratan hubungan antara  $X_t$  dan  $X_{t-k}$  sedangkan pengaruh dari time lag  $1, 2, 3, \dots, k-1$  dianggap konstan. Dengan kata lain, koefisien autokorelasi parsial mengukur derajat hubungan antara nilai-nilai sekarang dengan nilai-nilai sebelumnya (untuk time lag tertentu), sedangkan pengaruh nilai variabel time lag yang lain dianggap konstan. Secara matematis,

koefisien autokorelasi parsial berorde  $m$  didefinisikan sebagai koefisien *autoregressive* terakhir dari model AR( $m$ )

**Tabel 3.2**  
**Pola ACF dan PACF**

Tipe Model	Pola Tipikal ACF	Pola tipikal PACF
AR( $p$ )	Menurun secara eksponensial menuju nol	Signifikan pada semua lag $p$
MA( $q$ )	Signifikan pada semua lag $p$	Menurun secara eksponensial menuju nol
ARMA( $p,q$ )	Menurun secara eksponensial menuju nol	Menurun secara eksponensial menuju nol

Sumber : Gujarati 2003

#### b. Estimasi

Setelah menetapkan model sementara dari hasil identifikasi, yaitu menentukan nilai  $p$ ,  $d$ , dan  $q$ , langkah berikutnya adalah melakukan estimasi parameter autoregressive dan moving average yang tercakup dalam model (Firmansyah, 2000). Jika teridentifikasi proses AR murni maka parameter dapat diestimasi dengan menggunakan kuadrat terkecil (*Least Square*). Jika sebuah pola MA diidentifikasi maka *maximum likelihood* atau estimasi kuadrat terkecil, keduanya membutuhkan metode optimisasi non-linier (Griffiths, 1993), hal ini terjadi karena adanya unsur *moving average* yang menyebabkan ketidak linieran parameter (Firmansyah, 2000). Namun, saat ini sudah tersedia berbagai piranti lunak statistik yang mampu menangani perhitungan tersebut sehingga kita tidak perlu khawatir mengenai estimasi matematis.

#### c. Diagnostic Checking

Setelah melakukan estimasi dan mendapatkan penduga parameter, agar model sementara dapat digunakan untuk peramalan, perlu dilakukan uji kelayakan terhadap model tersebut. Tahap ini disebut *diagnostic checking*, dimana pada tahap ini diuji apakah spesifikasi model sudah benar atau belum. Pengujian kelayakan ini dapat dilakukan dengan beberapa cara.

- (1) Setelah estimasi dilakukan, maka nilai residual dapat ditentukan. Jika nilai-nilai koefisien autokorelasi residual untuk berbagai time lag tidak berbeda secara signifikan dari nol, model dianggap memadai untuk dipakai sebagai model peramalan.
- (2) Menggunakan statistik Box-Pierce Q, yang dihitung dengan formula :

$$Q = n \sum_{k=1}^m \hat{\rho}_k^2$$

Dimana :

n = jumlah sampel

m = jumlah lag, dan

$\hat{\rho}_k$  = nilai koefisien autokorelasi time lag k. Jika nilai Q hitung lebih kecil daripada  $\chi^2$  kritis dengan derajat kebebasan m, maka model dianggap memadai.

- (3) Menggunakan varian dari statistik Box-Pierce Q, yaitu statistik Ljung-Box(LB), yang dapat dihitung dengan :

$$LB = n(n+2) \sum_{k=1}^m \left( \frac{\hat{\rho}_k^2}{n-k} \right)$$

Sama seperti Q statistik, statistik LB mendekati  $\chi^2$  kritis dengan derajat kebebasan m. Jika statistik LB lebih kecil dari nilai  $\chi^2$  kritis, maka semua koefisien autokorelasi dianggap tidak berbeda dari nol, atau model telah dispesifikasikan dengan benar. Statistik LB dianggap lebih unggul secara statistik daripada Q statistik dalam menjelaskan sample kecil.

- (4) Menggunakan t statistik untuk menguji apakah koefisien model secara individu berbeda dari nol. Apabila suatu variabel tidak signifikan secara individu berarti variabel tersebut seharusnya dilepas dari spesifikasi model lain kemudian diduga dan diuji. Jika model sementara yang dipilih belum lolos uji diagnostik, maka proses pembentukan model diulang kembali. Menemukan model ARIMA yang terbaik merupakan proses iteratif.

d. Peramalan (*forecasting*)

Setelah model terbaik diperoleh, selanjutnya peramalan dapat dilakukan. Dalam berbagai kasus, peramalan dengan metode ini lebih dipercaya daripada peramalan yang dilakukan dengan model ekonometri tradisional. Namun, hal ini tentu saja perlu dipelajari lebih lanjut oleh para peneliti yang tertarik menggunakan metode serupa.

Berdasarkan ciri yang dimilikinya, model runtun waktu seperti ini lebih cocok untuk peramalan dengan jangkauan sangat pendek, sementara model struktural lebih cocok untuk peramalan dengan jangkauan panjang (Mulyono, 2000 dalam Firmansyah, 2000)

### 3.5.3 Metode Jaringan Syaraf Tiruan

Kelemahan JST dengan layar tunggal membuat perkembangan JST menjadi terhenti sekitar tahun 1970 an. Penemuan backpropagation yang terdiri atas beberapa lapisan membuka kembali cakrawala.(Siang, 2005). JST dengan layar tunggal memiliki keterbatasan dalam pengenalan pola. Kelemahan ini bisa ditanggulangi dengan menambahkan satu atau beberapa lapisan tersembunyi diantara layar masukan dan keluaran. Meskipun penggunaan lebih dari satu lapisan tersembunyi memiliki kelebihan manfaat untuk beberapa kasus, tetapi pelatihannya memerlukan waktu yang lama. Maka umumnya pelatihan dimulai dengan mencoba sebuah lapisan tersembunyi lebih dahulu.

Metode Jaringan Syaraf Tiruan yang akan digunakan adalah metode jaringan syaraf tiruan propagasi balik. Kusumadewi (2004) menjelaskan, propagasi balik menggunakan error *output* untuk mengubah nilai bobot-bobotnya dalam arah mundur (*backward*). Untuk mendapatkan error ini, tahap perambatan maju (*forward propagation*) harus dikerjakan terlebih dahulu.

Input yang akan digunakan dalam pelatihan ini adalah harga harian Indeks LQ45 dan harga Indeks LQ45 pada lag signifikan hasil estimasi dengan metode ARIMA. Diberikan dua lapisan tersembunyi dengan masing-masing memiliki 10 dan 5 unit tersembunyi didalamnya. Lapisan Output berisi harga saham pada periode t.

#### 3.5.3.1 Fungsi Aktivasi

Fungsi aktivasi yang digunakan dalam propagasi balik harus memenuhi beberapa syarat, yaitu : kontinu, terdiferensiasi dengan mudah dan merupakan fungsi yang tidak turun. Salah satu fungsi yang memenuhi ketiga syarat tersebut adalah fungsi sigmoid biner yang memiliki range (0,1) (Siang, 2005).

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

dengan turunan

$$f'(x) = f(x)(1 - f(x))$$

Fungsi lain yang sering dipakai adalah fungsi sigmoid bipolar yang bentuk fungsinya mirip dengan fungsi sigmoid biner, tapi dengan range (-1,1).

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

Dengan turunan

$$f'(x) = \frac{(1 + f(x))(1 - f(x))}{2}$$

Fungsi sigmoid memiliki nilai maksimum = 1. Maka untuk pola yang targetnya > 1, pola masukan dan keluaran harus terlebih dahulu ditransformasi sehingga semua polanya memiliki *range* yang sama seperti fungsi sigmoid yang dipakai. Alternatif lain adalah menggunakan fungsi aktivasi sigmoid hanya pada lapisan yang bukan lapisan keluaran. Pada lapisan keluaran, fungsi aktivasi yang dipakai adalah fungsi identitas :  $f(x) = x$

Data yang ada dapat ditransformasikan ke interval [0,1], tetapi akan lebih baik jika ditransofmasikan ke interval yang lebih kecil, misal pada interval [0,1

0,0] mengingat fungsi sigmoid merupakan fungsi asimtotik yang nilainya tidak pernah mencapai 0 maupun 1

Untuk tiap data dalam deret waktu, transformasi linier data ke interval [0,1 0,9] dapat dituliskan sebagai berikut :

$$x' = \frac{0,8(x - a)}{b - a} + 0,1$$

Dimana,

a = nilai minimum data deret waktu

b = nilai maksimum data deret waktu

x = nilai data

### 3.5.3.2 Pelatihan Standard Propagasi Balik

Pelatihan Jaringan Syaraf Tiruan Propagasi Balik meliputi 3 fase. Ketiga fase tersebut dapat dijelaskan sebagai berikut (Siang, 2005) :

#### **Fase pertama : propagasi maju**

Selama propagasi maju, sinyal masukan ( $=x_i$ ) dipropagasikan ke layar tersembunyi menggunakan fungsi aktivasi yang ditentukan. Keluaran dari setiap unit lapisan tersembunyi ( $=z_j$ ) tersebut selanjutnya dipropagasikan maju lagi ke layar tersembunyi di atasnya menggunakan fungsi aktivasi yang ditentukan. Demikian seterusnya hingga menghasilkan keluaran jaringan ( $=y_k$ ). Berikutnya, keluaran jaringan ( $=y_k$ ) dibandingkan dengan target yang harus dicapai ( $=t_k$ ). Selisih  $t_k - y_k$  adalah kesalahan yang terjadi. Jika kesalahan ini lebih kecil dari batas toleransi yang ditentukan, maka iterasi dihentikan. Akan tetapi apabila

kesalahan masih lebih besar dari batas toleransinya, maka bobot setiap garis dalam jaringan akan dimodifikasi untuk mengurangi kesalahan yang terjadi.

### **Fase Kedua : Propagasi Mundur**

Berdasarkan kesalahan  $t_k - y_k$ , dihitung faktor  $\delta_k(k=1,2,\dots,m)$  yang dipakai untuk mendistribusikan kesalahan di unit  $y_k$  ke semua unit tersembunyi yang terhubung langsung dengan  $y_k$ .  $\delta_k$  juga dipakai untuk mengubah bobot garis yang berhubungan langsung dengan unit keluaran. Dengan cara yang sama, dihitung faktor  $\delta_k$  di setiap unit di lapisan tersembunyi sebagai dasar perubahan bobot semua garis yang berasal dari unit tersembunyi pada lapisan dibawahnya. Demikian seterusnya hingga semua faktor  $\delta$  di unit tersembunyi yang berhubungan langsung dengan unit masukan dihitung.

### **Fase Ketiga : Perubahan Bobot**

Setelah semua faktor  $\delta$  dihitung, bobot semua garis dimodifikasi bersamaan. Perubahan bobot suatu garis didasarkan atas faktor  $\delta$  neuron di lapisan atasnya. Sebagai contoh, perubahan garis yang menuju ke layar keluaran didasarkan atas  $\delta_k$  yang ada di unit keluaran.

Ketiga fase tersebut diulang-ulang terus hingga kondisi penghentian dipenuhi. Umumnya kondisi penghentian yang sering dipakai adalah jumlah iterasi atau kesalahan. Iterasi akan dihentikan jika jumlah iterasi yang dilakukan sudah melebihi jumlah maksimum iterasi yang ditetapkan atau jika kesalahan yang terjadi sudah lebih kecil dari batas toleransi yang diizinkan.

### 3.5.3.3 Algoritma pelatihan

Algoritma standard yang digunakan dalam pelatihan Jaringan Syaraf Tiruan Feed Forward Back Propagation, yaitu gradient conjugate dan gradient conjugate with momentum seringkali terlalu lambat untuk keperluan praktis. Oleh karena itu, pada penelitian ini akan digunakan Algoritma *Gradient Conjugate With Adaptive Learning Rate and Momentum* (traingdx). Algoritma ini merupakan penggabungan dari *Algoritma Gradient Conjugate with Adaptive Learning*(traingda) dan *Gradient Conjugate With Momentum*(traingdm).

Pada standard backpropagation, perubahan bobot didasarkan atas gradien yang terjadi untuk pola yang dimasukkan saat itu. Modifikasi yang dapat dilakukan adalah melakukan perubahan bobot yang didasarkan atas arah gradien pola terakhir dan pola sebelumnya (disebut momentum) yang dimasukkan, jadi tidak hanya pola masukan terakhir saja yang diperhitungkan. (Siang, 2005)

Penambahan momentum dimaksudkan untuk menghindari perubahan bobot yang mencolok akibat adanya data yang sangat berbeda dengan yang lain (outlier). Apabila beberapa data terakhir yang diberikan ke jaringan memiliki pola yang serupa (berarti arah gradien sudah benar), maka perubahan bobot dilakukan secara cepat. Namun, apabila data terakhir yang dimasukkan memiliki pola yang berbeda dengan pola sebelumnya, maka perubahan dilakukan secara lambat. (Siang, 2005)

Dengan Penambahan momentum, bobot baru pada waktu ke  $(t+1)$  didasarkan atas bobot pada waktu  $t$  dan  $(t-1)$ . Disini harus ditambahkan dua variabel baru yang mencatat besarnya momentum untuk dua iterasi terakhir. Jika

$\mu$  adalah konstanta ( $0 \leq \mu \leq 1$ ) yang menyatakan parameter momentum maka bobot baru dihitung berdasarkan persamaan :

$$w_{kj}(t+1) = w_{kj}(t) + \alpha \delta_k z_j + \mu (w_{kj}(t) - w_{kj}(t-1))$$

dan

$$v_{ji}(t+1) = v_{ji}(t) + \alpha \delta_j x_i + \mu (v_{ji}(t) - v_{ji}(t-1))$$

Algoritma *gradient descent* dengan *adaptive learning rate*, dasarnya sama dengan algoritma *gradient descent* standard dengan beberapa perubahan. Pertama-tama dihitung terlebih dahulu nilai output jaringan dan error pelatihan. Pada setiap epoch, bobot-bobot baru dihitung dengan menggunakan *learning rate* yang ada. Kemudian dihitung kembali output jaringan dan error pelatihan. Jika perbandingan antara error pelatihan yang baru dengan error pelatihan lama melebihi maksimum kenaikan kinerja (`max_perf_inc`), maka bobot-bobot baru tersebut akan diabaikan, sekaligus nilai *learning rate* akan dikurangi dengan cara mengalikannya dengan `lr_dec`. Sebaliknya, apabila perbandingan antara *error* pelatihan baru dengan error pelatihan lama kurang dari maksimum kenaikan kinerja, maka nilai bobot-bobot akan dipertahankan, sekaligus nilai *learning rate* akan dinaikkan dengan cara mengalikannya dengan `lr_inc`.

Dengan cara ini, apabila *learning rate* terlalu tinggi dan mengarah ke ketidakstabilan, maka *learning rate* akan diturunkan. Sebaliknya jika *learning rate* terlalu kecil untuk menuju konvergensi, maka *learning rate* akan dinaikkan. Dengan demikian, maka algoritma pembelajaran akan tetap terjaga pada kondisi stabil.

*Algoritma gradient descent with momentum and adaptive learning*(traingdx) merupakan penggabungan antara algoritma *gradient descent with adaptive learning*(traingda) dan algoritma *gradient descent with momentum*(traingdm). Algoritma ini merupakan algoritma default yang digunakan oleh matlab karena memiliki performa kecepatan pelatihan yang tinggi.

Simulasi dilakukan baik pada periode training maupun periode testing. Salah satu cara yang dapat digunakan untuk melakukan evaluasi terhadap jaringan syaraf yang digunakan adalah dengan menggunakan analisis regresi terhadap respon jaringan dan target yang diharapkan. Analisis ini akan dilakukan dengan fungsi `postreg` yang terdapat pada MATLAB.

### 3.5.4 Pengukuran Kinerja

#### 3.5.4.1 Mean Squared Error

Dalam statistik, *Mean Squared Error* (MSE) sebuah estimator adalah nilai yang diharapkan dari kuadrat *error*. *Error* yang ada menunjukkan seberapa besar perbedaan hasil estimasi dengan nilai yang akan diestimasi. Perbedaan itu terjadi karena adanya keacakan pada data atau karena estimator tidak mengandung informasi yang dapat menghasilkan estimasi yang lebih akurat

$$MSE = \frac{1}{N} \sum_{t=h}^N (y_t - \hat{y}_t)^2$$

Dimana :

MSE = *Mean Squared Error*

N = Jumlah Sampel

$y_t$  = Nilai Aktual Indeks

$\hat{y}_t$  = Nilai Prediksi Indeks

#### 3.5.4.2 Komparasi Hasil Peramalan

Setelah nilai Mean Squared Error dari kedua metode didapatkan, maka akan dilakukan komparasi terhadap nilai MSE yang didapatkan pada periode testing (*out-sample*)

- Jika nilai  $MSE_{ARIMA} < MSE_{ANN}$  maka metode ARIMA memiliki performa lebih baik dibandingkan metode ANN karena memiliki tingkat kesalahan yang dihasilkan oleh ARIMA relatif lebih kecil.
- Sebaliknya, jika  $MSE_{ARIMA} > MSE_{ANN}$  maka metode ARIMA memiliki performa lebih buruk dibandingkan metode ANN karena tingkat kesalahan yang dihasilkan oleh metode ARIMA relatif lebih besar.