ISSN: 2339-2541
JURNAL GAUSSIAN, Volume 5, Nomor 4, Tahun 2016, Halaman 771-780
Online di: http://ejournal-s1.undip.ac.id/index.php/gaussian

PEMODELAN NEURO-GARCH PADA *RETURN* NILAI TUKAR RUPIAH TERHADAP DOLLAR AMERIKA

Umi Sulistyorini Adi[1], Budi Warsito[2], Suparti[3]
[1]Mahasiswa Departemen Statistika FSM Universitas Diponegoro
[2,3]Staff Pengajar Departemen Statistika FSM Universitas Diponegoro

ABSTRACT

Exchange rate can be defined as the value of a currency against other currencies. Exchange rates always fluctuate all the time. Very high fluctuations and unconstant becoming problem in forecasting where the data changed extremely. Most of economic data have heteroskedasticity characteristic analyzed using (Generalized Autoregressive Conditional Heteroskedasticity) GARCH models. Another model that commonly used as an alternative is Artificial Neural Network (ANN). However, both models have weaknesses. ARIMA models are linear, but the residual probably still contains non-linear relationship, while the ANN model used to non-linear relationship there is difficulty in determining the input. In this research combination of the two models is Neuro-GARCH model, with GARCH model used as input of ANN model. The purpose of this study was determined the best variance model Neuro-GARCH of return exchange rates rupiah against US dollar. The data used is daily return value of the rupiah (IDR) against the US dollar (USD) from August 27th, 2012 to March 31st, 2016. In this research, the mean model obtained is MA (1) and varian model is GARCH (1,1). The best model is Neuro-GARCH (2-10-1) which MSE smaller than the GARCH (1,1).

Keywords: exchange rate, return, GARCH, Neuro-GARCH.

1. PENDAHULUAN

Kurs dapat diartikan sebagai harga suatu mata uang terhadap mata uang lainnya[7]. Kurs mata uang selalu berfluktuasi setiap saat. Fluktuasi nilai tukar mata uang (kurs) rupiah terhadap dollar dapat dimodelkan menggunakan analisis runtun waktu karena merupakan himpunan observasi terurut. Data runtun waktu dapat dimodelkan menggunakan model *Autoregressive Moving Average* (ARMA). Model ARMA memiliki asumsi varian residual yang konstan, yang dikenal dengan istilah *homoscedasticity*.

Data kurs rupiah terhadap dollar mempunyai sifat *volatility clustering*[2]. Pada tahun 1982, Engle telah mengembangkan suatu model untuk mengestimasi perilaku volatilitas suatu data yang menimbulkan adanya *volatility clustering*, yang sering disebut *time varying variance* atau kasus heteroskedastisitas. Model yang digunakan untuk memodelkan kondisi ini adalah *Autoregressive Conditional Heteroskedasticity* (ARCH), dan pada tahun 1986 telah dikembangkan suatu model yaitu *Generalized Autoregressive Conditional Heteroskedasticity* (GARCH) oleh Bollerslev dan Taylor. Model GARCH memanfaatkan ketidakkonstanan varian residual dari data runtun waktu yang dapat menghasilkan nilai ramalan dan selang kepercayaan yang lebar dan bias.

Model ARMA merupakan metode peramalan yang bersifat linier. Salah satu syarat data *time series* dimodelkan ARMA adalah data tersebut harus stasioner. Metode lain yang sering dipakai dalam memodelkan data dengan fluktuasi sangat besar dan tidak tetap adalah *Artificial Neural Network* (ANN). ANN adalah pemodelan yang bersifat non linier. *Artificial Neural Network* (ANN) atau yang biasa disebut Jaringan Syaraf Tiruan (JST)

merupakan sistem pemroses informasi yang memiliki karakteristik mirip dengan jaringan syaraf pada makhluk hidup[10]. Dalam ARMA-ANN terdapat dua komponen yang harus diestimasi dari data, yaitu model ARMA digunakan untuk menyelesaikan kasus yang linier dan residual dari model linier yang masih mengandung informasi hubungan non-linier dimodelkan menggunakan ANN. Oleh karena itu, pada penelitian ini digunakan model alternatif lain, yaitu gabungan antara ANN dan GARCH yang disebut dengan Neuro-GARCH.

2. TINJAUAN PUSTAKA

Pada penelitian ini digunakan data *return*, karena salah satu keuntungan menggunakan data *return* nilai tukar adalah peningkatan dan penurunan harga nilai tukar akan semakin terlihat jelas jika diamati perbandingan nilai (t) dengan (t-1). *Return* diinterpretasikan sebagai harga relatif yang berubah mengikuti perbandingan *stock exchange*. Log *return* dirumuskan sebagai berikut[1]:

$$r_t = ln\left(\frac{P_t}{P_{t-1}}\right)$$

dengan: r_t = log *return* pada waktu ke-t, P_t = nilai kurs pada waktu ke-t, dan P_{t-1} = nilai kurs pada waktu t-1.

2.1 Analisis Runtun Waktu

Runtun waktu adalah serangkaian pengamatan terhadap suatu peristiwa, kejadian, gejala, atau variabel yang diambil dari waktu ke waktu, dicatat secara teliti menurut urutan-urutan waktu terjadinya dan kemudian disusun sebagai data. Adapun waktu yang digunakan dapat berupa harian, mingguan, bulanan, tahunan, dan sebagainya[4].

Model runtun waktu mengasumsikan bahwa data masukan harus stasioner. Maka perlu dilakukan suatu uji formal untuk mengetahui apakah terdapat komponen *trend* yang berupa *random walk* dalam data. Dalam penelitian ini digunakan uji *Augmented Dickey Fuller* untuk melihat apakah terdapat akar unit di dalam model atau tidak. Pengujian dilakukan dengan menguji hipotesis sebagai berikut:

$H_0 : \gamma = 0$ (terdapat akar unit sehingga data tidak stasioner)

$H_1 : \gamma < 0$ (tidak terdapat akar unit sehingga data stasioner)

Taraf signifikansi yang digunakan adalah α, dengan statistik uji:

$$ADF_{hitung} = \frac{\gamma}{se\ (\hat{\gamma})}$$

Kriteria ujinya adalah H_0 ditolak jika nilai ADF_{hitung} < nilai statistik ADF atau nilai probabilitas < α.

Dalam analisis runtun waktu dikenal adanya fungsi autokorelasi (ACF) dan fungsi autokorelasi parsial (PACF). ACF merupakan fungsi korelasi antara deret waktu Z_t dengan Z_{t+k} dan dapat dituliskan[11]:

$$\rho_k = \frac{Cov\ (Z_t, Z_{t+k})}{\sqrt{Var(Z_t)}\sqrt{Var(Z_{t+k})}} \quad \text{atau} \quad \rho_k = \frac{\gamma_k}{\gamma_0}$$

dimana $Var(Z_t) = Var(Z_{t+k}) = \gamma_0, \gamma_k$ dinamakan fungsi autokovarian. Sedangkan PACF merupakan hubungan antara Z_t dan Z_{t+k} dengan mengabaikan variabel Z_{t+k-1}, Z_{t+k-2}, …, Z_t dan dapat dituliskan sebagai berikut:

$$\phi_{kk} = \frac{\det(P_k^*)}{\det(P_k)} = \frac{\begin{vmatrix} 1 & \rho_1 & \rho_2 & \cdots & \rho_{k-2} & \rho_1 \\ \rho_1 & 1 & \rho_1 & \cdots & \rho_{k-3} & \rho_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \rho_{k-1} & \rho_{k-2} & \rho_{k-3} & \cdots & \rho_1 & \rho_k \end{vmatrix}}{\begin{vmatrix} 1 & \rho_1 & \rho_2 & \cdots & \rho_{k-2} & \rho_{k-1} \\ \rho_1 & 1 & \rho_1 & \cdots & \rho_{k-3} & \rho_{k-2} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \rho_{k-1} & \rho_{k-2} & \rho_{k-3} & \cdots & \rho_1 & 1 \end{vmatrix}}$$

Berikut ini adalah beberapa macam model runtun waktu Box Jenkins, yang pertama adalah model *Autoregressive* (AR). Bentuk dari suatu proses *autoregressive* order p atau AR(p) dapat dinyatakan sebagai:

$$Z_t = \phi_1 Z_{t-1} + \phi_2 Z_{t-2} + \cdots + \phi_p Z_{t-p} + a_t$$

yakni nilai sekarang suatu proses dinyatakan sebagai jumlah tertimbang nilai-nilai yang lalu ditambah satu sesatan sekarang. Jadi dapat dipandang Z_t diregresikan pada p nilai Z yang lalu. Kedua adalah model *Moving Average* (MA), proses *moving average* order q dapat ditulis MA (q), diidentifikasikan sebagai:

$$Z_t = a_t - \theta_1 a_{t-1} - \theta_2 a_{t-2} - \cdots - \theta_q a_{t-q}$$

Berikutnya adalah model *Autoregressive Moving Average* (ARMA). Bentuk umum dari model ARMA yang merupakan suatu perluasan dari model AR (p) dan MA (q) adalah:

$$Z_t = \phi_1 Z_{t-1} + \ldots + \phi_p Z_{t-p} + a_t - \theta_1 a_{t-1} - \ldots - \theta_q a_{t-q}$$

Dan yang keempat adalah model *Autoregressive Integrated Moving Average* (ARIMA). Model ARIMA merupakan suatu runtun waktu nonstasioner yang dapat diturunkan menjadi stasioner menggunakan *differencing*/diambil selisih ke-d yang mempunyai model *autoregressive* orde p dan model *moving average* orde q. Secara umum model ARIMA (p,d,q) dapat ditulis sebagai berikut:

$$\phi_p(B)(1-B)^d Z_t = \theta_q(B) a_t$$

Dalam pemodelan ARIMA terdapat beberapa prosedur[11]. Langkah pertama adalah membuat plot data runtun waktu, melalui pemeriksaan yang cermat dari plot kita biasanya mendapatkan informasi apakah data tersebut mengandung *trend*, musiman, *outlier*, varian tidak konstan, dan fenomena non stasioner lainnya. Selanjutnya dilakukan identifikasi karakteristik dari ACF dan PACF, order p dan q model ARMA (p,q) diidentifikasikan berdasarkan ciri-ciri ACF dan PACF yang ditunjukkan pada tabel berikut:

Tabel 1. Ciri-ciri Teoritis ACF dan PACF untuk Model Stasioner

Model	ACF	PACF
AR (p)	Turun cepat menuju nol secara eksponensial	Terpotong setelah lag ke-p
MA (q)	Terpotong setelah lag ke-q	Turun cepat menuju nol secara eksponensial
ARMA(p,q)	Terpotong setelah lag ke (q-p)	Terpotong setelah lag ke (p-q)

Selanjutnya adalah pendugaan parameter, yang dilakukan dari autoregresinya dan rata-rata bergerak yang termasuk dalam modelnya. Metode yang digunakan untuk melakukan pendugaan parameter pada model ARIMA adalah Maksimum Likelihood.

Kemudian dilakukan verifikasi model untuk mengetahui apakah model cocok dengan data pengamatan. Verifikasi model meliputi uji independensi residual dan uji normalitas residual. Hipotesis untuk uji independensi residual adalah:

H_0: $\rho_1 = \rho_2 = \ldots = \rho_m = 0$ (tidak ada korelasi residual antar lag)

H_1: minimal ada satu $\rho_k \neq 0$, dimana $k = 1, 2, \ldots, m$ (terdapat korelasi residual antar lag)

Taraf signifikansi yang digunakan adalah α, dengan statistik uji:

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

Kriteria ujinya adalah tolak H_0 jika $Q > X^2_{(\alpha;m-(p+q))}$atau *p-value* < α. Sedangkan untuk uji normalitas residual hipotesisnya adalah:

H_0 : Residual berdistribusi normal

H_1 : Residual tidak berdistribusi normal

Dengan taraf signifikansi α dan statistik uji:

$$JB = \frac{n}{6}\left(S^2 + \frac{(K-3)^2}{4}\right)$$

Kriteria ujinya adalah tolak H_0 jika JB > $X^2_{(2)}$ atau *p-value* < α.

Setelah didapatkan model ARIMA terbaik, maka perlu dilakukan pengujian apakah varian residual dari model tersebut bersifat tidak heteroskedastisitas (homogenitas). Jika residual data bersifat heteroskedastisitas maka dapat dikatakan terdapat efek ARCH/GARCH dalam model tersebut, sehingga perlu untuk dilakukan pemodelan ke dalam bentuk ARCH/GARCH yang dapat mengatasi ketidakkonstanan varian. Uji *Lagrange Multiplier* (LM) digunakan untuk menguji apakah terdapat efek heteroskedastisitas pada varian residual model runtun waktu[9].

Hipotesis:

H_0: $\alpha_1 = \cdots = \alpha_r = 0$ (tidak ada efek ARCH/GARCH dalam residual)

H_1: $\exists \alpha_i \neq 0, i = 1, \ldots, r$ (ada efek ARCH/GARCH dalam residual)

Taraf signifikansi yang digunakan adalah α, dengan statistik uji:

$$F = \frac{(SSR_0 - SSR_1)/r}{SSR_1/(n-2r-1)}$$

dimana $SSR_0 = \sum_{t=r+1}^{n}(a_t^2 - \bar{\omega})^2$, $SSR_1 = \sum_{t=r+1}^{n} \hat{e}_t^2$, $\bar{\omega}$ = rata-rata sampel dari a_t^2 ,n = banyaknya pengamatan, r = banyaknya lag yang diuji

Kriteria ujinya adalah tolak H_0 jika nilai F > $X^2_{(\alpha;r)}$ atau probabilitas < α.

Terdapat dua sifat penting yang sering dimiliki oleh data runtun waktu di bidang keuangan, khususnya untuk data *return*[6]. Pertama, distribusi probabilitas dari *return* bersifat *fat tails* dibandingkan dengan distribusi normal. Umumnya hal ini ditandai oleh harga *excess kurtosis* (yakni kurtosis ke-3) yang positif. Keadaan distribusi ini sering juga disebut bersifat *leptokurtik*. Kedua, adanya *volatility clustering*, hal ini sering juga disebut sebagai kasus *time-varying variance* atau kasus heteroskedastisitas. Model runtun waktu yang dapat digunakan untuk memodelkan data dengan sifat tersebut diantaranya adalah *Autoregressive Conditional Heteroskedasticity* (ARCH) yang dikemukakan oleh Engle. Bentuk umum dari model ARCH (r) adalah:

$$\sigma_t^2 = \alpha_0 + \alpha_1 a_{t-1}^2 + \alpha_2 a_{t-2}^2 + \cdots + \alpha_r a_{t-r}^2$$

Kemudian pada tahun 1986 Bollerslev memperkenalkan model *Generalized Autoregressive Conditional Heteroskedasticity* atau lebih dikenal dengan GARCH yang merupakan generalisasi dari model ARCH. Bentuk umum model GARCH (r,s):

$$\sigma_t^2 = \alpha_0 + \sum_{i=1}^{r} \alpha_i \, a_{t-i}^2 + \sum_{j=1}^{s} \beta_j \, \sigma_{t-j}^2$$

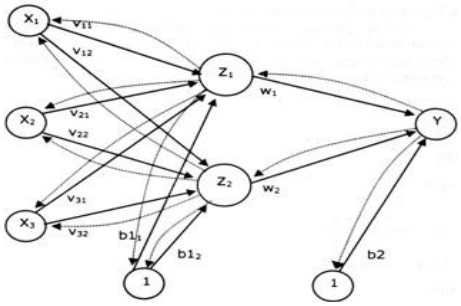
2.2 *Artificial Neural Network* (ANN)

Artificial Neural Network (ANN) merupakan salah satu sistem pemrosesan informasi yang didesain dengan menirukan cara kerja otak manusia dalam menyelesaikan suatu masalah dengan melakukan proses belajar melalui perubahan bobot. ANN melakukan pengenalan pola data masa lalu untuk dipelajari sehingga mampu memberikan keputusan terhadap data yang belum pernah dipelajari [3].

Neuron adalah bagian dasar dalam pemrosesan suatu jaringan. Neuron-neuron dalam satu lapisan akan dihubungkan dengan lapisan-lapisan sebelum dan sesudahnya, sehingga nantinya akan terbentuk lapisan *input* (*input layer*), lapisan tersembunyi (*hidden layer*), dan lapisan *output* (*output layer*). Sebuah neuron terdiri dari beberapa bagian, yaitu *input*, bobot, *processing unit*, dan *output*.

Berdasarkan jumlah lapisannya ANN dibagi menjadi dua, yaitu *single layer network* dan *multi layer network*[8]. Arsitektur yang digunakan dalam penelitian ini adalah *Multi*

Layer Network atau *Feed Forward Neural Network* (FFNN). Jaringan dengan banyak lapisan memiliki satu atau lebih lapisan yang terletak di antara lapisan *input* dan lapisan *output*, seperti terlihat pada gambar berikut:



Gambar 1. Jaringan Syaraf dengan Banyak Lapisan

Berikut rumus penentuan banyaknya neuron pada *hidden layer* dengan N_{in} adalah neuron *input*[12]:

Tabel 2. Penentuan Banyaknya Neuron *Hidden Layer*

N_{out} (Neuron *output*)	Rumus menghitung *hidden layer*
1	$2^{*}N_{in}+1$
2	$3^{*}N_{in}$
3	$\frac{2+N_{in}^{*}N_{out}+0{,}5N_{in}^{*}(N_{out}^{2}+N_{in})-3}{N_{in}+N_{out}}$
4	$\frac{(2^{*}N_{in})}{3}$
5	$\sqrt{N_{in}^{*}N_{out}}$
6	$2^{*}N_{in}$

Dalam ANN terdapat beberapa algoritma pelatihan yang bisa digunakan, salah satunya adalah *backpropagation*. Pelatihan *backpropagation* adalah metode pencarian titik minimum untuk mencari bobot dengan eror minimum menggunakan eror *output* untuk mengubah nilai bobot-bobotnya dalam arah mundur (*backward*). Terdapat tiga fase dalam JST *backpropagation*, yaitu fase *feed forward*, *backward*, dan perubahan bobot.

Fase *feed forward* adalah fase dimana tiap-tiap unit *input* $(x_i, i = 1,2,3,\dots,l)$ menerima sinyal x_i dan meneruskan sinyal tersebut ke semua unit pada lapisan yang ada di atasnya yaitu lapisan tersembunyi. Tiap-tiap unit pada suatu lapisan tersembunyi $(z_j, j = 1,2,3,\dots,m)$ menjumlahkan sinyal-sinyal *input* terbobot, selanjutnya $b1_j$ yang merupakan bias pada unit tersembunyi *j* dan aplikasi fungsi aktivasinya akan menghitung sinyal *output*nya dan mengirimkan sinyal tersebut ke semua unit di lapisan atasnya (unit-unit *output*). Tiap-tiap unit *output* $(y_k, k = 1,2,3,\dots,n)$ menjumlahkan sinyal-sinyal *input* terbobot. Dimana $b2_k$ merupakan bias pada unit keluaran *k*. Lalu menggunakan fungsi aktivasi untuk menghitung sinyal *output*nya dengan $y\_out_k = y = f(y\_in_k)$. *Output* (y_out) dibandingkan dengan target (t_k) yang harus dicapai. Selisihnya ($t_k - y\_out$) adalah eror (δ_k). Jika eror lebih kecil dari batas toleransi yang ditentukan, maka *epoch* dihentikan. Tapi jika eror masih lebih besar dari batas toleransi, maka bobot setiap garis dalam jaringan akan dimodifikasi untuk mengurangi kesalahan yang terjadi.

Pada fase *backward*, tiap-tiap unit *output* $(y_k, k = 1,2,3,\dots,n)$ menerima target pola yang berhubungan dengan pola *input* pembelajaran, kemudian menghitung informasi erornya. Kemudian hitung koreksi bobot yang nantinya akan digunakan untuk memperbaiki nilai w_{kj}. Hitung koreksi biasnya yang nantinya akan digunakan untuk

memperbaharui nilai $b2_k$, dan kirimkan δ_k ke unit-unit pada lapisan di bawahnya. Tiap-tiap unit tersembunyi $(z_j, j = 1,2,3,\dots,m)$ menjumlahkan delta *input*nya dari unit-unit yang berada pada lapisan di atasnya. Kalikan nilai ini dengan turunan dari fungsi aktivasinya untuk menghitung informasi eror. Selanjutnya hitung koreksi bobot yang digunakan untuk memperbaharui v_{ij} nanti dan hitung juga koreksi bias yang nantinya akan digunakan untuk memperbaiki nilai $b1_j$. Pada fase terakhir atau fase perubahan bobot, tiap-tiap unit *output* $(y_k, k = 1,2,3,\dots,n)$ memperbaiki bias dan bobotnya $(j = 1,2,3,\dots,m)$ kemudian tiap unit lapisan tersembunyi $\left(z_j, j = 1,2,3,\dots,m\right)$ memperbaiki bias dan bobotnya $(i = 1,2,3,\dots,l)$.

2.3 Neuro-GARCH

Model Neuro-GARCH merupakan model kombinasi dari JST dan GARCH. Pada model varian Neuro-GARCH ini, peubah *input* model JST adalah model GARCH. Misalkan model varian yang terbentuk adalah GARCH (1,1), maka *input* untuk model varian Neuro-GARCH yaitu σ_{t-1}^2 dan a_{t-1}^2, dengan target σ_t^2.

$$\hat{\sigma}_t^2 = f^o\left(\sum_{j=1}^{q} w_j^o f_j^h\left(\sum_{i=1}^{p} v_{j,i}^h \sigma_{t-1}^2 + v_{j,i}^h a_{t-1}^2 + b_j^h\right) + b^o\right)$$

Dalam pemodelan varian Neuro-GARCH ini, kriteria pemilihan model terbaik yang digunakan adalah *Mean Square Error* (MSE). MSE adalah rata-rata kesalahan kuadrat dari peramalan. Penggunaan MSE untuk berbagai aplikasi ANN dalam pemodelan dan peramalan runtun waktu sudah cukup memenuhi dan paling populer digunakan[5]. Nilai MSE dapat dihitung menggunakan rumus sebagai berikut:

$$MSE = \frac{\sum_{t=1}^{n}(Z_t - \hat{Z}_t)^2}{n}$$

3. METODE PENELITIAN

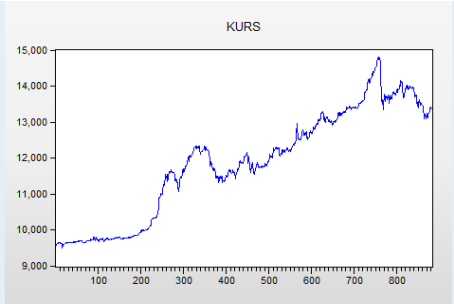
Data yang digunakan dalam penelitian ini adalah data sekunder, yaitu data kurs jual rupiah terhadap dollar Amerika dari tanggal 27 Agustus 2012 sampai dengan 31 Maret 2016, yang diperoleh dari Bank Indonesia *Official Website* (www.bi.go.id). Variabel yang digunakan dalam penelitian ini menggunakan data *return* dari kurs jual rupiah terhadap dollar Amerika sebanyak 881 data. Langkah-langkah yang dilakukan untuk menganalisis data penelitian adalah:

1. Menyiapkan data yang akan digunakan untuk penelitian.
2. Melakukan uji stasioneritas menggunakan uji *Augmented Dickey Fuller*.
3. Menentukan model Box Jenkins yang sesuai, yaitu model AR, MA, dan ARMA.
4. Mengestimasi parameter model Box Jenkins dan menguji signifikansi parameternya.
5. Melakukan verifikasi model Box Jenkins.
6. Memilih model terbaik di antara model yang mungkin berdasarkan MSE terkecil.
7. Melakukan pengujian efek ARCH/GARCH dengan uji *Lagrange Multiplier*.
8. Pembentukan model GARCH.
9. Menentukan *input* dan target berdasarkan model GARCH terbaik.
10. Membagi data sebanyak 90% untuk *training* dan 10% untuk *testing*.
11. Melakukan inisialisasi parameter.
12. Melakukan proses *training* dan *testing*.
13. Memilih model Neuro-GARCH terbaik berdasarkan MSE *testing*.
14. Membuat plot *training* dan *testing* berdasarkan model yang telah didapat.

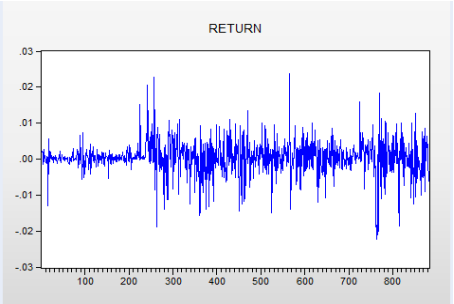
4. HASIL DAN PEMBAHASAN

4.1 Pemodelan ARIMA

Berikut ini adalah plot runtun waktu data kurs jual dan *return* kurs jual:



Gambar 2. Plot Data Kurs Jual



Gambar 3. Plot Data *Return* Kurs Jual

Gambar 2 menunjukkan bahwa data mengalami *trend* naik, sehingga menyebabkan data tidak stasioner. Salah satu cara untuk membuat data menjadi stasioner terhadap *mean* adalah transformasi menjadi data *return*. Plot *return* pada Gambar 3 menunjukkan bahwa data sudah stasioner, nilai *return* bertanda positif menunjukkan jika terjadi kenaikan harga dan bertanda negatif jika terjadi penurunan harga.

Untuk memperkuat asumsi stasioneritas secara visual maka dilakukan uji stasioneritas secara formal menggunakan uji *Augmented Dickey-Fuller*. Berikut hasil uji *Augmented Dickey-Fuller*:

Tabel 3. Uji *Augmented Dickey-Fuller*

$\mathbf{ADF_{hitung}}$	**Prob.***	***Test Critical Value 5%***	**α**
-27,07722	0,0000	-2,864615	0,05

Berdasarkan Tabel 3 dapat disimpulkan bahwa pada taraf signifikansi 5% data *return* kurs jual sudah stasioner karena nilai ADF_{hitung} = -27,07722 < *Test critical values 5%* = -2,864615 atau nilai probabilitas = 0,0000 < $\alpha = 0{,}05$.

Untuk mengidentifikasi model ARIMA yang mungkin dapat dilihat pada plot ACF dan PACF. Pada plot ACF terlihat bahwa lag ke-1 melebihi batas standar error, sehingga model yang mungkin dari MA adalah MA (1). Dan pada plot PACF terlihat bahwa lag ke-1 melebihi batas standar error, sehingga model yang mungkin dari AR adalah AR (1). Maka identifikasi model Box Jenkins yang mungkin adalah AR (1), MA (1), dan ARMA (1,1).

Setelah didapatkan model sementara yang sudah teridentifikasi, maka dilakukan pendugaan parameter model dan uji signifikansi koefisien, hasilnya adalah sebagai berikut:

Tabel 4. Uji Signifikansi Parameter Model ARIMA

Model	**Parameter**	**Koefisien**	**Probabilitas**
AR (1)	C	0,000376	0,0365
	ϕ (1)	0,088324	0,0089
MA (1)	C	0,000378	0,0339
	θ (1)	0,089075	0,0083
ARMA (1,1)	C	0,000376	0,0351
	ϕ (1)	$3{,}16 \times 10^{-5}$	0,9999
	θ (1)	0,088952	0,8131

Berdasarkan statistik uji signifikansi parameter di Tabel 4 dapat disimpulkan bahwa pada taraf signifikansi 5% model AR(1) dan MA(1) mempunyai parameter yang signifikan terhadap model, sedangkan model ARMA(1,1) semua parameternya tidak signifikan terhadap model.

Kemudian dilakukan verifikasi model pada model AR(1) dan MA(1), yang meliputi uji independensi residual dan uji normalitas residual. Berdasarkan uji independensi residual yang telah dilakukan diperoleh kesimpulan bahwa pada taraf signifikansi 5% tidak ada korelasi residual antar lag untuk semua model yang mungkin. Kemudian pada uji normalitas diperoleh kesimpulan bahwa residual tidak berdistribusi normal untuk semua model. Hal tersebut mengindikasikan adanya efek ARCH/GARCH dalam model.

Oleh karena model yang terbentuk masih lebih dari satu, yaitu AR(1) dan MA(1), maka dilakukan pemilihan model terbaik melalui nilai MSE terkecil. Model *mean* terbaik adalah model MA(1) dengan persamaan $Z_t = 0{,}000378 + 0{,}089075\ a_{t-1} + e_t$ dan nilai MSE sebesar 2,341852466$x10^{-5}$.

4.2 Pemodelan GARCH

Untuk mengetahui apakah model ARIMA perlu dimodelkan ke dalam bentuk ARCH/GARCH maka dilakukan pengujian efek ARCH/GARCH dalam residual model. Pengujian ini dilakukan menggunakan uji *Lagrange Multiplier* dengan hipotesis:

H_0: $\alpha_1 = \cdots = \alpha_r = 0$ (tidak ada efek ARCH/GARCH dalam residual)

H_1: $\exists \alpha_i \neq 0, i = 1, \ldots, r$ (ada efek ARCH/GARCH dalam residual)

Diperoleh kesimpulan bahwa pada taraf signifikansi 5% H_0 ditolak karena nilai F = 113,3497 > $\chi^2_{(0,05;1)} = 3{,}841$ atau probabilitas = 0,0000 < α = 0,05. Sehingga terdapat efek ARCH/GARCH dalam residual model MA(1).

Dari hasil uji *Lagrange Multiplier*, diketahui bahwa model MA(1) memiliki efek ARCH/GARCH. Kemudian dibentuk model ARCH(1), ARCH(2), GARCH(1,1), GARCH(1,2), GARCH(2,1), dan GARCH(2,2) karena model-model tersebut sudah cukup baik untuk memodelkan volatilitas dari data. Beberapa model ARCH/GARCH yang mungkin dilakukan verifikasi berdasarkan empat sifat model ARCH/GARCH kemudian dilakukan uji signifikansi parameter. Empat sifat model ARCH/GARCH (r,s) yaitu:

1. $\alpha_0 > 0$
2. $\alpha_i \geq 0$ untuk i = 1,2,..,r
3. $\beta_j \geq 0$ untuk j = 1,2,..,s
4. $\sum_{i=1}^{r} \sum_{j=1}^{s} (\alpha_i + \beta_j) < 1$

Dari pengujian yang telah dilakukan diperoleh hasil bahwa hanya model ARCH(1), ARCH(2), GARCH(1,1) dan GARCH(1,2) saja yang memenuhi empat sifat model ARCH/GARCH dan memiliki parameter yang signifikan terhadap model.

Oleh karena model yang terbentuk masih lebih dari satu, maka dilakukan pemilihan model terbaik di antara 4 model ARCH/GARCH yang mungkin melalui nilai MSE terkecil. Model terbaik adalah model GARCH(1,1) dengan persamaan $\sigma_t^2 = 5{,}69x10^{-7} + 0{,}122681a_{t-1}^2 + 0{,}864464\sigma_{t-1}^2$ dan nilai MSE sebesar 2,37489$x10^{-5}$.

4.3 Pemodelan Neuro-GARCH

Pemodelan Neuro-GARCH ini menggunakan jaringan yang mempunyai banyak lapisan. Jumlah lapisan *input* yang digunakan diambil dari banyaknya variabel yang terbentuk dari model GARCH(1,1). Berdasarkan model GARCH(1,1), *input* untuk model varian adalah σ_{t-1}^2 dan a_{t-1}^2, dengan target σ_t^2. *Hidden layer* yang digunakan sebanyak satu lapisan dengan jumlah unit yaitu 5, 10, 15, dan 20. Algoritma pembelajaran yang digunakan adalah *Backpropagation* dengan algoritma pelatihan Quasi Newton. Fungsi

aktivasi yang digunakan pada jaringan ini yaitu *logsig* (sigmoid biner) pada *hidden layer*, dan *purelin* (linier) pada lapisan *output*.

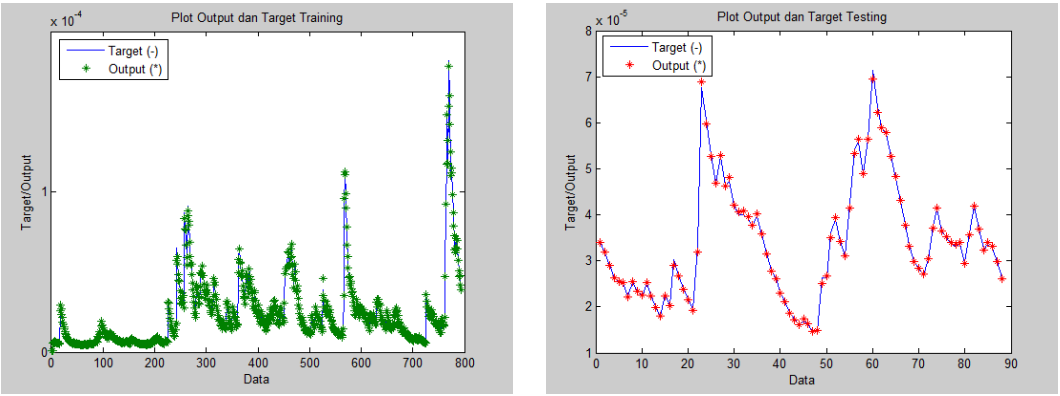
Setelah dilakukan proses *training* dan *testing*, maka diperoleh nilai MSE untuk masing-masing model dengan jumlah unit pada lapisan tersembunyi yang berbeda-beda.

Tabel 5. Hasil *Training* dan *Testing*

Jumlah Unit Lapisan	MSE *Training*	MSE *Testing*
2-5-1	$4{,}9101x10^{-13}$	$3{,}4430x10^{-13}$
2-10-1	$5{,}0656x10^{-13}$	$2{,}3006x10^{-13}$
2-15-1	$4{,}8054x10^{-13}$	$3{,}0725x10^{-13}$
2-20-1	$4{,}9647x10^{-13}$	$3{,}5272x10^{-13}$

Berdasarkan nilai MSE pada Tabel 5, model varian terbaik yang terpilih yaitu model Neuro-GARCH dengan jumlah unit lapisan 2-10-1 dengan MSE *testing* sebesar $2{,}3006x10^{-13}$.

Plot data varian hasil *training* dan *testing* model Neuro-GARCH terbaik ditunjukkan pada gambar berikut:



Gambar 4. Plot *Output* dan Target Data Varian

Berdasarkan Gambar 4 dapat dilihat bahwa tahapan *training* dan *testing* data varian mempunyai hasil yang baik atau bisa dikatakan cukup akurat, terbukti dari pola *output* yang berhimpit atau bahkan sama dengan pola target.

5. KESIMPULAN

Model *mean* yang terbentuk adalah MA(1) dengan persamaan $Z_t = 0{,}000378 + 0{,}089075\ a_{t-1} + e_t$. Sedangkan untuk model varian yang terbentuk adalah GARCH (1,1) dengan persamaan $\sigma_t^2 = 5{,}69x10^{-7} + 0{,}122681a_{t-1}^2 + 0{,}864464\sigma_{t-1}^2$. Model Neuro-GARCH terbaik adalah Neuro-GARCH (2-10-1), terdiri atas dua unit di lapisan input yaitu σ_{t-1}^2 dan a_{t-1}^2, sepuluh unit di lapisan tersembunyi, dan satu unit di lapisan output. Penentuan banyaknya neuron dalam *hidden layer* untuk mendapatkan model terbaik tidak mutlak harus menggunakan aturan perhitungan pada Tabel 2. Model Neuro-GARCH lebih bagus dibandingkan model GARCH karena nilai MSE pada model Neuro-GARCH (2-10-1) lebih kecil dibandingkan model GARCH (1,1).

6. DAFTAR PUSTAKA

[1] Chen, W. Y. 2005. *A Comparison of Forecasting Models for ASEAN Equity Markets*. Sunway Academic Journal, Vol. 2, Hal. 1-12.

[2] Elvitra, C. W. 2013. *Metode Peramalan Menggunakan Model Volatilitas Asymmetric Power Autoregressive Conditional Heteroskedasticity pada Return Nilai Tukar*

Rupiah terhadap Dollar. Tugas Akhir Sarjana Universitas Diponegoro: tidak diterbitkan.
[3] Kusumadewi, S. 2008. *Artificial Intelegence, Teknik dan Aplikasi*. Yogyakarta: Graha Ilmu.
[4] Makridakis, S. *et al.* 1999. *Forecasting: Methods and Applications*. New York: John Wiley & Sons Inc.
[5] Patterson, W. D. 1996. *Artificial Neural Network Theory and Applications*. Singapore: Prentice Hall.
[6] Rosadi, D. 2012. *Ekonometrika & Analisis Runtun Waktu Terapan dengan Eviews*. Yogyakarta: Andi Offset.
[7] Salvatore, D. 1997. *Ekonomi Internasional*. Jakarta: Penerbit Erlangga.
[8] Siang, J. J. 2005. *Jaringan Syaraf Tiruan & Programnya Menggunakan MATLAB*. Yogyakarta.
[9] Tsay, R. S. 2005. *Analysis of Financial Time Series* Second Edition. New York : A John Wiley & Sons, Inc.
[10] Warsito, B. 2009. *Kapita Selekta Statistika Neural Network*. Semarang: BP UNDIP Semarang.
[11] Wei, W. W. S. 2006. *Time Series Analysis: Univariate and Multivariate Methods*. Second Edition. Boston: Pearson Education Inc.
[12] Yasin, H., Suparti. 2014. *Pemodelan Volatilitas untuk Penghitungan Value at Risk (VaR) Menggunakan Feed Forward Neural Network dan Algoritma Genetika*. Media Statistika Vol. 7, No. 2:53-61.