J. DIFERENSIAL/E-ISSN:2775-9644
Vol.7(2),2025, 150–163
doi:https://doi.org/10.35508/jd.v7i2.22535

ARTIKEL PENELITIAN

Implementasi Runtun Waktu Samar Orde Tinggi Yolcu-Egrioglu-Aladag dalam Melakukan Peramalan

Luthfia Amanah[1,*], **Lusi Harini**[1]

[1]*Program Studi Matematika, Universitas Negeri Yogyakarta, Indonesia*
[*]*Penulis korespondensi: luthfiaamanah.2021@student.uny.ac.id*

Diterima: 7 Juni 2025; Direvisi: 4 September 2025; Disetujui: 11 September 2025; Dipublikasi:1 November 2025.

Abstrak: Peramalan memegang peran penting dalam pengambilan keputusan strategis di berbagai bidang, mulai dari ekonomi hingga manajemen publik. Namun, metode peramalan tradisional seperti ARIMA dan regresi linier memiliki keterbatasan dalam menangani karakteristik data runtun waktu yang sering kali berpola tidak linier, bersifat tidak stasioner, dan mengalami fluktuasi secara tiba-tiba. Penelitian ini mengusulkan pendekatan runtun waktu samar atau *Fuzzy Time Series* (FTS) yang lebih fleksibel dalam memodelkan ketidakpastian dan pola kompleks tanpa bergantung pada asumsi linearitas. Tujuan penelitian ini adalah meramalkan jumlah penduduk di Kabupaten Sleman menggunakan runtun waktu samar orde tinggi Yolcu-Egrioglu-Aladag (YEA). Data yang digunakan berupa data tahunan jumlah penduduk Kabupaten Sleman dari tahun 1998 hingga 2024, sebanyak 27 data observasi yang dibagi menjadi 80% data *training* dan 20% data *testing*. FTS YEA menggunakan operasi irisan untuk menyederhanakan input pada model orde tinggi, melakukan fuzzifikasi dengan *Fuzzy C-Means* dan membentuk relasi fuzzy melalui *Feedforward Neural Network*dengan algoritma Levenberg-Marquardt, serta defuzzifikasi. Hasil penelitian menunjukkan bahwa model YEA menghasilkan nilai *Mean Absolute Percentage Error* (MAPE) yang kecil pada data testing, sehingga baik untuk meramalkan jumlah penduduk di Kabupaten Sleman. Nilai MAPE dari FTS YEA yaitu 2,22% *(training)* dan 1,00% *(testing)*.

Kata Kunci: Peramalan, Runtun Waktu Samar, Orde Tinggi Yolcu-Egrioglu-Aladag, *Fuzzy C-Means*, *Feedforward Neural Network*, Jumlah Penduduk

Abstract: Forecasting plays an important role in strategic decision-making in various fields, ranging from economics to public management. However, traditional forecasting methods such as ARIMA and linear regression have limitations in handling the characteristics of time series data that are often non-linearly patterned, non-stationary, and experience sudden fluctuations. This research proposes a fuzzy time series (FTS) approach that is more flexible in modeling uncertainty and complex patterns without relying on the assumption of linearity. The purpose of this research is to forecast the population in Sleman Regency using the Yolcu-Egrioglu-Aladag (YEA) higher order fuzzy time series. The data used is annual data on the population of Sleman Regency from 1998 to 2024, a total of 27 observational data divided into 80% training data and 20% testing data. The YEA FTS uses a slice operation to simplify inputs in higher-order models, fuzzification with Fuzzy C-Means and forming fuzzy relations through a Feedforward Neural Network with the Levenberg-Marquardt algorithm, and defuzzification. The results show that the YEA model produces a small

© Amanah dan Harini

Mean Absolute Percentage Error (MAPE) value on the testing data, so it is good for forecasting the population in Sleman Regency. The MAPE value of the YEA FTS is 2.22% (training) and 1.00% (testing).

Keywords: Forecasting, Fuzzy Time Series, High Order Yolcu-Egrioglu-Aladag, Fuzzy C-Means, Feedforward Neural Network, Population.

1. Pendahuluan

Peramalan memainkan peran penting dalam pengambilan keputusan di berbagai bidang. Bidang-bidang seperti ekonomi, kesehatan, pemasaran, dan manajemen rantai pasokan sangat bergantung pada hasil peramalan untuk menentukan langkah-langkah strategis, seperti memperkirakan permintaan produk, menganalisis tren pasar, atau merencanakan distribusi sumber daya [1]. Peramalan bekerja dengan menganalisis data historis untuk mengidentifikasi pola atau tren yang terjadi di masa lalu, kemudian menggunakan pola tersebut untuk memproyeksikan kondisi di masa depan. Metode peramalan dapat melibatkan teknik statistik, algoritma matematika, atau model berbasis kecerdasan buatan untuk menghasilkan peramalan yang akurat [2].

Metode peramalan tradisional seperti ARIMA dan regresi linier memiliki keterbatasan dalam menangani data runtun waktu yang kompleks. Metode-metode ini membutuhkan asumsi tertentu, seperti linearitas dan stasioneritas data, yang sering kali tidak terpenuhi dalam data dunia nyata. Data runtun waktu sering dipengaruhi oleh perubahan pola akibat tren jangka panjang, fluktuasi musiman, siklus tertentu, serta gangguan acak. Tantangan terbesar dalam peramalan adalah mengelola gangguan yang tidak terduga dalam data dan memastikan bahwa model tetap relevan dengan perubahan yang terus terjadi [3].

Peramalan menggunakan metode runtun waktu samar *Fuzzy Time Series* (FTS) menawarkan pendekatan alternatif yang lebih fleksibel dalam menangani data runtun waktu kompleks dan tidak memenuhi asumsi linearitas serta stasioneritas. Runtun waktu samar bekerja dengan mengubah data numerik menjadi himpunan samar, yang memungkinkan model untuk menangani ketidakpastian dan pola non-linear dalam data. Metode ini mampu menangkap hubungan yang tidak jelas atau kabur antara variabel dalam waktu, memungkinkan peramalan yang lebih baik meskipun terdapat fluktuasi atau ketidakpastian dalam data. Selain itu, runtun waktu samar juga dapat mengakomodasi perubahan tren dan gangguan acak yang sering kali terjadi pada data dunia nyata, sehingga menghasilkan peramalan yang lebih akurat dan dapat diandalkan [4].

Metode peramalan berbasis runtun waktu samar telah banyak dikembangkan dalam beberapa penelitian terdahulu, seperti penelitian oleh Yolcu et al. (2016) yang mengusulkan metode runtun waktu samar orde tinggi berbasis operasi irisan untuk menganalisis hubungan samar yang lebih kompleks. Metode ini selanjutnya disebut sebagai model YEA (Yolcu-Egrioglu-Aladag), menunjukkan kemampuan menghasilkan peramalan dengan tingkat akurasi tinggi pada berbagai dataset, seperti data pendaftaran Universitas Alabama dan indeks pasar saham Istanbul, dengan rata-rata MAPE sebesar 0,48%.

Penelitian ini menggunakan data jumlah penduduk Kabupaten Sleman dari tahun 1998 hingga 2024 sebagai studi kasus. Data ini dipilih karena mencerminkan karakteristik runtun waktu yang dinamis, dengan pola pertumbuhan yang dipengaruhi oleh berbagai faktor eksternal seperti migrasi, kebijakan sosial, dan perubahan demografis. Implementasi metode FTS YEA kemudian akan dievaluasi menggunakan *Mean Absolute Percentage Error* (MAPE). Hasil penelitian ini diharapkan dapat menunjukkan kelayakan metode FTS YEA dalam meramalkan jumlah penduduk di masa depan secara akurat.

2. Metode Penelitian

2.1. Deskripsi Data

Pada penelitian ini, metode runtun waktu samar orde tinggi Yolcu-Egrioglu-Aladag (FTS YEA) diterapkan pada data runtun waktu jumlah penduduk di Kabupaten Sleman dari tahun 1998 sampai dengan 2024 sebanyak 27 data. Data tersebut merupakan data sekunder yang diperoleh dari Badan Pusat Statistik Kabupaten Sleman melalui website https://slemankab.bps.go.id/id. Data tersebut disajikan dalam tabel berikut:

Tabel 2.1: Data Jumlah Penduduk di Kabupaten Sleman Tahun 1998-2024

Tahun	**Jumlah Penduduk**
1998	828.960
1999	838.628
2000	850.176
2001	862.314
⋮	⋮
2021	1.136.474
2022	1.47.562
2023	1.157.290
2024	1.125.571

Data jumlah penduduk selanjutnya dibagi kedalam data training sebanyak 80% dan data testing sebanyak 20%.

Tabel 2.2: Pembagian Data Training dan Testing

Data Training (80%)	**Data Testing** (20%)
1998 - 2019	2020 - 2024

2.2. Runtun Waktu Samar Orde Tinggi Yolcu-Egrioglu-Aladag (FTS YEA)

Peramalan menggunakan runtun waktu samar orde tinggi dilakukan dengan langkah-langkah sebagai berikut [5]:

1. Melakukan fuzzifikasi pada data runtun waktu yang bersifat crisp menggunakan *Fuzzy C-Means* (FCM).
    a. Input data dan parameter awal
    Misalkan data runtun waktu $X = \{x_1, x_2, \ldots, x_n\}$ dengan n adalah jumlah observasi dalam runtun waktu dan c adalah jumlah himpunan samar yang ingin dibentuk atau jumlah klaster, di mana $2 \leq c \leq n$. Menentukan indeks kekaburan (fuzziness indekx) β yang mengontrol tingkat kekaburan antar klaster, di mana $\beta > 1$. Menentukan error terkecil ε yang menunjukan toleransi perbedaan antara hasil iterasi dalam proses optimasi. Nilai ε digunakan sebagai kriteria konvergensi dan menentukan maksimum iterasi. Proses iterasi dihentikan jika perubahan fungsi objektif antar iterasi lebih kecil dari ε.
    b. Inisialisasi pusat klaster dan matriks keanggotaan
    Inisialisasi derajat keanggotaan awal u_{ji}, yaitu nilai awal keanggotaan setiap data x_j

terhadap klaster i sebagai berikut:

$$\begin{aligned} 0 &\leq u_{ji} \leq 1, \forall i, j \\ 0 &\leq \sum_{j=1}^{n} u_{ji} \leq n, \ \forall i \\ & \sum_{i=1}^{c} u_{ji} = 1, \ \forall j \end{aligned} \tag{2.1}$$

Berdasarkan nilai awal ini, pusat klaster v_i dapat dihitung menggunakan Persamaan 2.2

$$v_i = \frac{\sum_{j=1}^{n} u_{ji}^{\beta} x_j}{\sum_{j=1}^{n} u_{ji}^{\beta}} \tag{2.2}$$

di mana v_i adalah pusat klaster ke-i.

c. Menghitung fungsi objektif
Setelah menemukan pusat klaster v_i, fungsi objektif J_β dihitung untuk mengevaluasi kualitas klasterisasi. Fungsi objektif dirumuskan dengan Persamaan 2.3 berikut:

$$J_\beta(X, V, U) = \sum_{j=1}^{n} \sum_{i=1}^{c} u_{ji}^{\beta} d^2(x_j, v_i) \tag{2.3}$$

di mana $d(x_j, v_i)$ adalah jarak Euclidean antara obseervasi x_j dan pusat klaster v_i. Semakin kecil jaraknya, semakin besar nilai u_{ij}, yang menunjukkan tingkat keanggotaan observasi tersebut dalam klaster i.

d. Memperbarui derajat keanggotaan
Untuk setiap observasi x_j, diperbarui derajat keanggotaan u_{ij} dengan Persamaan 2.4:

$$u_{ji} = \frac{1}{\sum_{k=1}^{c} \left(\frac{d(x_j, v_i)}{d(x_j, v_k)} \right)^{\frac{2}{\beta - 1}}} \tag{2.4}$$

di mana J_β merupakan fungsi objektif yang menunjukkan total kesalahan kuadrat berbobot dari klasterisasi. Nilai J_β diminimalkan selama iterasi.

e. Iterasi hingga didapatkan konvergensi
Langkah c dan d diulang hingga perbedaan nilai J_β antar iterasi lebih kecil dari ambang batas ε yang ditetapkan ($J_\beta < \varepsilon$).

f. Hasil fuzzifikasi
Setelah konvergensi, setiap observasi x_j memiliki derajat keanggotaan u_{ji} terhadap setiap klaster. Nilai u_{ji} membentuk matriks keanggotaan samar, yang menjadi input untuk langkah-langkah berikutnya.
Berikut adalah contoh data runtun waktu dengan delapan observasi, yaitu, 15, 25, 40, 30, 20, 55, 65, 80. Misalkan jumlah himpunan samar c adalah 3.

Tabel 2.3: Contoh Ilustrasi Fuzzifikasi

	Pusat Himpunan 1 (v_1)	**Pusat Himpunan 2** (v_2)	**Pusat Himpunan 3** (v_3)
	22,7579	52,2581	76,3750
Observasi keanggotaan dalam himpunan samar			
t X_t	**Himpunan 1** (L_1)	**Himpunan 2** (L_2)	**Himpunan 3** (L_3)
1 15	0,9439	0,0411	0,0150
2 25	0,9907	0,0072	0,0020
3 40	0,2986	0,6328	0,0685
4 30	0,8800	0,0979	0,0221
5 20	0,9906	0,0071	0,0023
6 55	0,0090	0,9698	0,0211
7 65	0,0386	0,4017	0,5597
8 80	0,0045	0,018	0,9768

2. Mendefinisikan hubungan samar dengan *Feedforward Neural Network* (FFANN) dan menggunakan algoritma Levenberg-Marquadt.
Langkah-langkah dalam melakukan pelatihan FFANN menggunakan algoritma pembelajaran Levenberg-Marquadt sebagai berikut [6]:

   a. Memasukkan data input dan target dengan operasi irisan
   Data input ke FFANN adalah vektor nilai keanggotaan u_{ji} yang diperoleh dengan menerapkan operasi irisan pada nilai keanggotaan setiap observasi dari runtun waktu tertunda. Vektor nilai keanggotaan input didefinisikan sebagai:

$$\mu(X_t^{LTS}) = \mu(X_{t-1}) \cap \mu(X_{t-2}) \cap \ldots \cap \mu(X_{t-}) \tag{2.5}$$

   di mana adalah orde runtun waktu samar yang menentukan jumlah keterlambatan (*lag*). Target FFANN adalah vektor nilai keanggotaan samar dari observasi di waktu saat ini t.
   Misalkan, dapat dipertimbangkan nilai keanggotaan sampel yang diberikan Tabel 3. Untuk model persamaan runtun waktu samar orde keiga, di mana ingin diperoleh peramalan untuk $X_{t=4}$. Dalam kasus ini:

$$\begin{aligned} \mu(X_{t-1=3}) &= (0,2986 \quad 0,6328 \quad 0,0685) \\ \mu(X-t-2=2) &= (0,9907 \quad 0,0072 \quad 0,0020) \\ \mu(X_{t-3=1}) &= (0,9439 \quad 0,0411 \quad 0,0150) \end{aligned}$$

   Dan dengan demikian,

$$(\mu(X_{t-1=3}) \cap \mu(X_{t-2=2}) \cap \mu(X_{t-3=1})) = \mu(X_t^{LTS}) = (0,2986 \quad 0,0072 \quad 0,0020)$$

Tabel 2.4: Contoh Input dan Target FFANN untuk Identifikasi Relasi Samar

Training Sample	t	Input 1 $\mu_{L_1}(X_t^{LTS})$	Input 2 $\mu_{L_2}(X_t^{LTS})$	Input 3 $\mu_{L_3}(X_t^{LTS})$	Target 1 $\mu_{L_1}(X_t^{LTS})$	Target 2 $\mu_{L_2}(X_t^{LTS})$	Target 3 $\mu_{L_3}(X_t^{LTS})$
1	4	0,2986	0,0072	0,0020	0,8800	0,0979	0,0221
2	5	0,2986	0,0072	0,0020	0,9906	0,0071	0,0023
3	6	0,2986	0,0071	0,0023	0,0090	0,9698	0,0211
4	7	0,0090	0,0071	0,0023	0, 0386	0,4017	0,5597
5	8	0,0090	0,0071	0,0023	0,0045	0,0187	0,9768

   b. Melakukan pembagian data menjadi dua bagian, yaitu:

1) Data *Training* yang digunakan untuk melatih model.
2) Data *Testing* yang digunakan untuk mengevaluasi model setelah setiap iterasi pelatihan.

c. Menginisiasi bobot dan bias awal jaringan dengan bilangan acak [-1, 1].
d. Menentukan parameter yang dibutuhkan, yaitu:
   1) Inisialisasi *epoch* = 0
   2) Parameter Marquadt atau *learning rate* $(\lambda > 0)$.
   3) Parameter faktor Tau (τ) untuk dibagikan atau dikalikan dengan parameter Marquadt.
e. Menentukan parameter jumlah neuron di hidden layer dalam proses jaringan saraf tiruan.
f. Vektor nilai keanggotaan $\mu(X_t^{LTS})$ dipetakan menjadi input x_i untuk digunakan dalam perhitungan *feedforward* jaringan saraf tiruan. Perhitungan Feedforward untuk setiap unit input $(x_i,\ i = 1, 2, \ldots, n)$ menerima sinyal masukan selanjutnya diteruskan ke seluruh unit pada *hidden layer*. Masing-masing unit *hidden layer* $(z_j,\ j = 1, 2, \ldots, p)$ menjumlahkan sinyal-sinyal input berbobot (v_{ij}) dan bias $(b1_j)$.

$$z_{in_j} = b1_j + \sum_{i=1}^{n} x_i v_{ij} \tag{2.6}$$

Hasil z_{in_j} diproses menggunakan fungsi aktivasi sigmoid untuk menghaslkan sinyal output.

$$z_j = f(z_{in_j}) = \frac{1}{1 + e^{-z_{in_j}}} \tag{2.7}$$

Selanjutnya sinyal tersebut dikirimkan ke seluruh unit pada lapisan atasnya.
Di mana

z_{i_j}	= sinyal awal yang masuk pada *hidden layer* ke-j
$b1_j$	= bobot awal dari bias *input* pada *hidden layer* ke-j
x_i	= input ke-$i = 1, 2, \ldots, n$
v_{ij}	= bobot awal dari *input layer* ke-i pada *hidden layer* ke- j
$f_(z_{in_j})$	= fungsi aktivasi pada *hidden layer* ke-j

g. Setiap unit lapisan output $(Y_k,\ k = 1, 2, \ldots, m)$ menjumlahkan sinyal-sinyal input berbobot (w_{jk}) dan bias $(b2_k)$.

$$y_{in_k} = b2_k + \sum_{j=1}^{n} z_j w_{jk} \tag{2.8}$$

Hasil y_{in_k} diproses menggunakan fungsi aktivasi sigmoid untuk menghasilkan sinyal output.

$$y_k = f(y_{in_k}) = \frac{1}{e^{-y_{in_k}}} \tag{2.9}$$

Selanjutnya sinyal tersebut dikirimkan ke seluruh unit pada lapisan atasnya.
Di mana

y_{in_k}	= sinyal input yang masuk pada *output layer* ke-k
$b2_k$	= bobot awal dari bias *hidden* pada *output layer* ke-k
z_j	= output dari setiap neuron *hidden layer* ke-j
w_{jk}	= bobot awal dari*hidden layer* ke-j pada *output layer* ke-k
$f(y_{in_k})$	= fungsi aktivasi pada *output layer* ke-k
y_k	= output dari setiap neuron *output layer* ke-k

h. Menghitung error dan MSE
Untuk menghitung error digunakan rumus:

$$e_r = t_r - y_r \tag{2.10}$$
$$e = [t_1 - y_1 \quad t_2 - y_2 \quad t_r - y_r]^T \tag{2.11}$$

Di mana
r = input ke-r
t_r = target keluaran yang diharapkan pada observasi ke-r
e_r = kesalahan pada unit keluaran untuk input ke-r
y_r = keluaran aktual untuk input ke-r
Menentukan fungsi kesalahan yang akan diminimalkan menggunakan *Mean Square Error* (MSE) dengan Persamaan 2.12:

$$MSE = \frac{1}{n}\sum_{r=1}^{n}(t_r - y_r)^2 \tag{2.12}$$

Di mana
n = jumlah data
t_r = target keluaran yang diharapkan pada observasi ke-r
y_r = keluaran aktual untuk input ke-r

i. Perhitungan *backforward* untuk tiap unit output dihitung error neuron (y_k, $k = 1, 2, \ldots, m$).

$$\delta 2 = (t_r - y_r) f'(y_{in_k}) \tag{2.13}$$
$$\varphi 2_{jk} = \delta 2_k z_j \tag{2.14}$$
$$\beta 2_k = \delta 2_k \tag{2.15}$$

Kemudian menghitung koreksi bobot untuk memperbaiki nilai w_{jk}

$$\Delta W_{jk} = \varphi 2_{jk} \tag{2.16}$$

Untuk memperbaiki nilai $\beta 2_k$, maka hitung koreksi bias

$$\Delta b2_k = \beta 2_k \tag{2.17}$$

Di mana
$\delta 2_k$ = *error neuron* untuk tiap *output layer* ke-k
$\varphi 2_{jk}$ = bobot *hidden layer* ke-j pada *output layer* ke-k
$\beta 2_k$ = bias pada output ke-k
Δw_{jk} = koreksi bobot untuk memperbaiki nilai w_{jk}
$\Delta b2_k$ = koreksi bias untuk memperbaiki nilai $\beta 2_k$

j. Error neuron dihitung setiap unit lapisan tersembunyi ($z_j, j = 1, 2, \ldots, p$)

$$\delta_{in_j} = \sum_{k=1}^{m} \delta 2_k W_{jk} \tag{2.18}$$
$$\delta 1_j = \delta_{in_j} f'(z_{in_j}) \tag{2.19}$$
$$\varphi 1_{ij} = \delta 1_j x_i \tag{2.20}$$
$$\beta 1_j = \delta 1_j \tag{2.21}$$

Selanjutnya menghitung koreksi bobot untuk memperbaiki nilai v_{ij}, yaitu:

$$\Delta v_{ij} = \varphi 1_{ij} \tag{2.22}$$

Selain itu, dihitung koreksi bias

$$\Delta b1_j = \beta 1_j \tag{2.23}$$

Di mana
$\delta 1_j$ = *error neuron* untuk tiap *output layer* ke-j
$\varphi 1_{ij}$ = bobot *hidden layer* ke-i pada *output layer* ke-j
$\beta 1_j$ = bias pada output ke-j
Δv_{ij} = koreksi bobot untuk memperbaiki nilai v_{ij}
$\Delta b1_j$ = koreksi bias untuk memperbaiki nilai $\beta 1_j$

k. Membentuk matriks Jacobian J yaitu matriks yang berisikan nilai koreksi bobot dan bias dari seluruh jaringan. Matriks Jacobian digunakan untuk mendapatkan bobot baru.

$$J = [\varphi 1_{11} \ \dots \ \varphi 1_{np} \quad \beta 1_1 \ \dots \ \beta 2_p \quad \varphi 2_{11} \ \dots \ \varphi 2_{pm} \quad \beta 2_1 \ \dots \ \beta 2_m] \tag{2.24}$$

l. Menghitung bobot baru

$$W_{baru} = W_{lama} - [J^T J + \lambda I]^{-1} J^T e \tag{2.25}$$

Di mana
J = matriks Jacobian
J^T = transpos matriks Jacobian
I = matriks identitas
e = nilai error

m. Mengevaluasi fungsi error
Terdapat dua kondisi yang mungkin terjadi pada fungsi error untuk bobot yang diperbarui $E(W_{baru})$, yaitu:
1) Jika $E(W_{baru}) \leq E(W_{lama})$ maka mengurangi parameter Marquadt $\lambda = \frac{\lambda}{\tau}$, di mana τ adalah faktor skala dan $\tau > 1$ dan $epoch = epoch + 1$, kemudian kembali ke langkah f.
2) Jika $E(W_{baru}) > E(W_{lama})$ maka meningkatkan parameter Marquadt $\lambda = \lambda \cdot \tau$, kemudian kembali ke langkah 1.
Di mana
λ = Parameter Marquadt atau *learning rate*
τ = Parameter faktor Tau

n. Proses pelatihan berhenti jika $epoch \geq epoch$ maksimal atau ketika kriteria *early stopping* terpenuhi, yaitu saat validasi error tidak mengalami penurunan signifikan dalam beberapa iterasi berturut-turut.

o. Hasil identifikasi hubungan samar
Setelah pelatihan selesai, FFAN dapat memperamalan vektor nilai keanggotaan samar $(\hat{\mu}_{L_i}(X_t))$ untuk waktu t, berdasarkan nilai keanggotaan dari waktu sebelumnya $(t-1, t-2, \dots, t-n)$.

p. Output FFANN
Output FFANN berbentuk vektor keanggotaan samar:

$$\hat{\mu}_{L_i}(X_t) = [\hat{\mu}_{L_1}(X_t), \hat{\mu}_{L_2}(X_t), \dots, \hat{\mu}_{L_c}(X_t)] \tag{2.26}$$

3. Defuzzifikasi hasil peramalan
Output dihasilkan menggunakan derajat keanggotaan dari c himpunan samar untuk observasi runtun waktu tertunda $t(\mu_{L_i}(X_t))$ sebagai input ke FFANN. Output yang dihasilkan FFANN berupa peramalan samar, yakni derajat keanggotaan observasi pada t. Pada proses defuzzifikasi, peramalan samar distandarisasi menjadi bobot dengan Persamaan 2.27, lalu

peramalan defuzzifikasi dihitung menggunakan Persamaan 2.28

$$w_{it} = \frac{\hat{\mu}_{Li}(X_t)}{\hat{\mu}_{L_1}(X_t) + \hat{\mu}_{L_2}(X_t) + \ldots + \hat{\mu}_{L_c}(X_t)} \tag{2.27}$$

$$\hat{X}_t = \sum_{i=1}^{c} w_{it} v_i \tag{2.28}$$

Di mana $\hat{\mu}_{L_i}(X_t),\ t = 1, 2, \ldots, c$, adalah peramalan samar untuk observasi pada t, terdiri dari nilai keanggotaan yang menjadi output dari FFANN. Bobot untuk proses defuzzifikasi peramalan samar direpresentasikan sebagai $w_i,\ i = 1, 2, \ldots, c$.
Jika dipertimbangkan contoh kumpulan data yang diberikan pada Tabel 2.3 dan diasumsikan bahwa output jaringan saraf untuk ketiga pengamatan $(t = 4)$ adalah 0,35; 0,61; 0,12, maka bobot dapat diperoleh sebagai berikut:

$$\begin{aligned}
w_{14} &= \frac{\hat{\mu}_{L_1}(X_4)}{\hat{\mu}_{L_1}(X_4) + \hat{\mu}_{L_2}(X_4) + \hat{\mu}_{L_3}(X_4)} = \frac{0,35}{0,35 + 0,61 + 0,12} = 0,3241 \\
w_{24} &= \frac{\hat{\mu}_{L_2}(X_4)}{\hat{\mu}_{L_1}(X_4) + \hat{\mu}_{L_2}(X_4) + \hat{\mu}_{L_3}(X_4)} = \frac{0,61}{0,35 + 0,61 + 0,12} = 0,5648 \\
w_{34} &= \frac{\hat{\mu}_{L_3}(X_4)}{\hat{\mu}_{L_1}(X_4) + \hat{\mu}_{L_2}(X_4) + \hat{\mu}_{L_3}(X_4)} = \frac{0,12}{0,35 + 0,61 + 0,12} = 0,1111 \\
\hat{X}_t &= \sum_{i=1}^{3} w_{i4} v_i = 0,3241 \times 22,7579 + 0,5648 \times 52,2581 + 0,1111 \times 76,3750 \\
&= 45,3754
\end{aligned}$$

2.3. Hyperparameter Tuning

Hyperparameter tuning merupakan proses pemilihan dan penyesuaian nilai hiperparameter yang menentukan struktur serta mekanisme pembelajaran suatu model. Hiperparameter tidak dapat dipelajari langsung dari data selama proses pelatihan, melainkan harus ditentukan sebelum model dilatih. Beberapa contoh hiperparameter dalam jaringan saraf tiruan meliputi jumlah neuron dalam setiap lapisan, jumlah lapisan tersembunyi, fungsi aktivasi, algoritma optimasi, laju pembelajaran, jumlah epoch, serta ukuran batch. Pemilihan nilai hiperparameter yang optimal berperan penting dalam meningkatkan kinerja model [7].

2.4. Mean Absolute Percentage Error (MAPE)

Mean Absolute Percentage Error (MAPE) merupakan metode yang digunakan untuk mengukut tingkat kesalahan dalam peramalan dengan membandingkan kesalahan absolut pada setiap periode terhadap nilai observasi aktual.

$$MAPE = \frac{\sum_{t=1}^{n} \left| \frac{A_t - \hat{A}_t}{A_t} \right|}{n} \times 100\% \tag{2.29}$$

Dengan
A_t = nilai aktual pada waktu ke-t
$\hat{A}_t$ = nilai peramalan pada waktu ke-t
n = jumlah data

3. Pembahasan

3.1. Peramalan Menggunakan Runtun Waktu Samar Orde Tinggi Yolcu-Egrioglu-Aladag (FTS YEA)

1. Melakukan fuzzifikasi pada data runtun waktu yang bersifat crisp menggunakan *Fuzzy C-Means* (FCM)
   Diketahui data runtun waktu x_j, yaitu jumlah penduduk di Kabupaten Sleman pada tahun 1998-2024, dengan $j = 1, 2, \ldots, 27$. Ditentukan:
   a) Jumlah data $(n) = 27$
   b) Jumlah klaster $(c) = 3$
   c) Indeks kekaburan $(\beta) = 2$
   d) Kemungkinan error terkecil $(\varepsilon) = 10^{-5}$
   e) Maksimum iterasi $(MaxIter) = 100$

   Didapatkan nilai final derajat keanggotaan dan pusat klaster sebagai berikut.

Tabel 3.1: Nilai Final Derajat Keanggotaan

L_1	L_2	L_3
0,957602	0,026962	0,015436
0,972038	0,017902	0,010060
0,986505	0,008714	0,004780
0,996827	0,002069	0,001104
⋮	⋮	⋮
0,011885	0,373120	0,614995
0,007207	0,160933	0,831860
0,002935	0,051577	0,945488
0,012801	0,625474	0,361725

Tabel 3.2: Nilai Final Pusat Klaster

v_1	v_2	v_3
872.659	1.089.389	1.173.149

2. Mendefinisikan hubungan samar dengan *Feedforward Neural Network* (FFANN) dan Algoritma Levenberg-Marquadt
   a) Data input dan target
   Akan dilakukan model peramalan runtun waktu samar orde enam sehingga diterapkan operasi irisan dengan *lag* 6 menggunakan Persamaan 2.5. Oleh karena itu, peramalan dimulai dari $X_{t=7}$:

$$\begin{aligned}
\mu(X_{t-1=6}) &= (0,994799 \quad 0,0034459 \quad 0,001742)\\
\mu(X_{t-2=5}) &= (0,999850 \quad 0,000990 \quad 0,000051)\\
\mu(X_{t-3=4}) &= (0,996827 \quad 0,002069 \quad 0,001104)\\
\mu(X_{t-4=3}) &= (0,986505 \quad 0,008714 \quad 0,004780)\\
\mu(x_{t-5=2}) &= (0,972038 \quad 0,017902 \quad 0,010060)\\
\mu(X_{t-6=1}) &= (0,957602 \quad 0,026962 \quad 0,015436)
\end{aligned}$$

   sehingga diperoleh input:

$$\begin{aligned}
&\big(\mu(X_{t-1=6}) \cap \mu(X_{t-2=5}) \cap \mu(X_{t-3=4}) \cap \mu(X_{t-4=3}) \cap \mu(X_{t-5=2}) \cap \mu(X_{t-6=1})\big)\\
&\qquad = \mu(X_7^{LTS}) = (0,957062 \quad 0,000990 \quad 0,000051)
\end{aligned}$$

Didapatkan hasil input dan target FFANN sebagai berikut:

Tabel 3.3: Input dan Target FFANN

Training Sample	t	Input 1 $\mu_{L_1}(X_t^{LTS})$	Input 2 $\mu_{L_2}(X_t^{LTS})$	Input 3 $\mu_{L_3}(X_t^{LTS})$	Target 1 $\mu_{L_1}(X_t^{LTS})$	Target 2 $\mu_{L_2}(X_t^{LTS})$	Target 3 $\mu_{L_3}(X_t^{LTS})$
1	7	0,957602	0,000099	0,000051	0,980103	0,013373	0,006525
2	8	0,972038	0,000099	0,000051	0,954030	0,031242	0,014729
3	9	0,954030	0,000099	0,000051	0,917217	0,056901	0,025882
4	10	0,917217	0,000099	0,000051	0,064933	0,826953	0,108114
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
18	24	0,000368	0,005238	0,366921	0,011885	0,373120	0,614995
19	25	0,000563	0,006430	0,366921	0,007207	0,160933	0,831860
20	26	0,003865	0,036696	0,366921	0,002935	0,051577	0,945488
21	27	0,002935	0,051577	0,366921	0,012801	0,625474	0,361725

b) Pembentukan parameter FFANN
Pada tahap ini dilakukan proses *hyperparameter tuning* untuk menentukan kombinasi parameter terbaik pada *neuron hidden layer, learning rate, batch size,* dan *epoch*. Adapun nilai *hyperparameter* yang diuji untuk *neuron hidden layer* yaitu 7,8, *learning rate* yaitu 0,001; 0,01, *batch size* yaitu 16,32 dan maksimum *epoch* yaitu 50, 100. Pemilihan parameter terbaik didasarkan pada nilai MSE terkecil yang dihasilkan dengan arsitektur jaringan paling sederhana selama proses pelatihan model. Diperoleh hasil tuning dan kombinasi parameter berikut:

Tabel 3.4: Kombinasi *Hyperparameter Tuning*

Neuron Hidden Layer	*Learning Rate*	*Batch Size*	*Epoch*	Waktu *Training* (detik)	MSE
7	0,001	16	50	4,72	0,16
7	0,001	16	100	9,25	0,13
7	0,001	32	50	4,64	0,12
7	0,001	32	100	7,42	0,11
7	0,01	16	50	4,29	0,03
7	0,01	16	100	7,80	0,03
7	0,01	32	50	4,23	0,08
7	0,01	32	100	8,01	0,04
8	0,001	16	50	5,17	0,08
8	0,001	16	100	8,19	0,11
8	0,001	32	50	4,28	0,22
8	0,001	32	100	7,94	0,15
8	0,01	16	50	4,51	0,06
8	0,01	16	100	8,37	0,02
8	0,01	32	50	4,37	0,07
8	0,01	32	100	8,40	0,03

Didapatkan model terbaik sebagai berikut:

Tabel 3.5: Parameter Model FFANN

Karakteristik	Spesifikasi
Arsitektur	1 *hidden layer*
Neuron input	3
Neuron hidden	8
Neuron output	3
Fungsi aktivasi	Sigmoid
Inisialisasi bobot	Random
Parameter Marquadt/*Learning rate* (λ)	0,01
Parameter Faktor Tau (τ)	3
Batch size	16
Maksimum *epoch*	100

c) Pelatihan menggunakan algoritma Levenberg-Marquadt
Proses pelatihan dijalankan dengan batas maksimum 100 *epoch*. Selama proses pelatihan, nilai *lost function* menunjukkan tren penurunan yang konsisten, menandakan bahwa model berhasil meminimalkan kesalahan pada data pelatihan. Pada *epoch* ke-100, diperoleh nilai *Mean Square Error* (MSE) sebesar $3,21 \times 10^{-8}$, dengan parameter Marquadt berada pada nilai $4,11 \times 10^{-9}$. Nilai MSE yang sangat kecil ini mengindikasikan bahwa hasil prediksi model sangat mendekati nilai target, sehingga performa model dapat dikategorikan sangat baik dalam hal akurasi prediksi terhadap data pelatihan.

Tabel 3.6: Output Pelatihan FFANN

t	Output 1	Output 2	Output 3
7	0,960198	0,031489	0,21720
8	0,999066	0,000779	0,010401
9	0,913319	0,067671	0,025736
10	0,077209	0,833026	0,097853
⋮	⋮	⋮	⋮
19	0,003480	0,009714	0,987763
20	0,007372	0,037401	0,953705
21	0,012951	0,072952	0,918658
22	0,009438	0,137889	0,859868

Tabel 3.7: Output Pengujian FFANN

t	Output 1	Output 2	Output 3
23	0,024862	0,625519	0,367711
24	0,013309	0,168762	0,823147
25	0,012955	0,153387	0,839260
26	0,009883	0,056002	0,940568
27	0,019371	0,614025	0,381346

3. Defuzzifikasi hasil peramalan
Menggunakan Persamaan 2.27, untuk ketiga pengamatan pada $t = 7$ diperoleh bobot sebagai berikut:

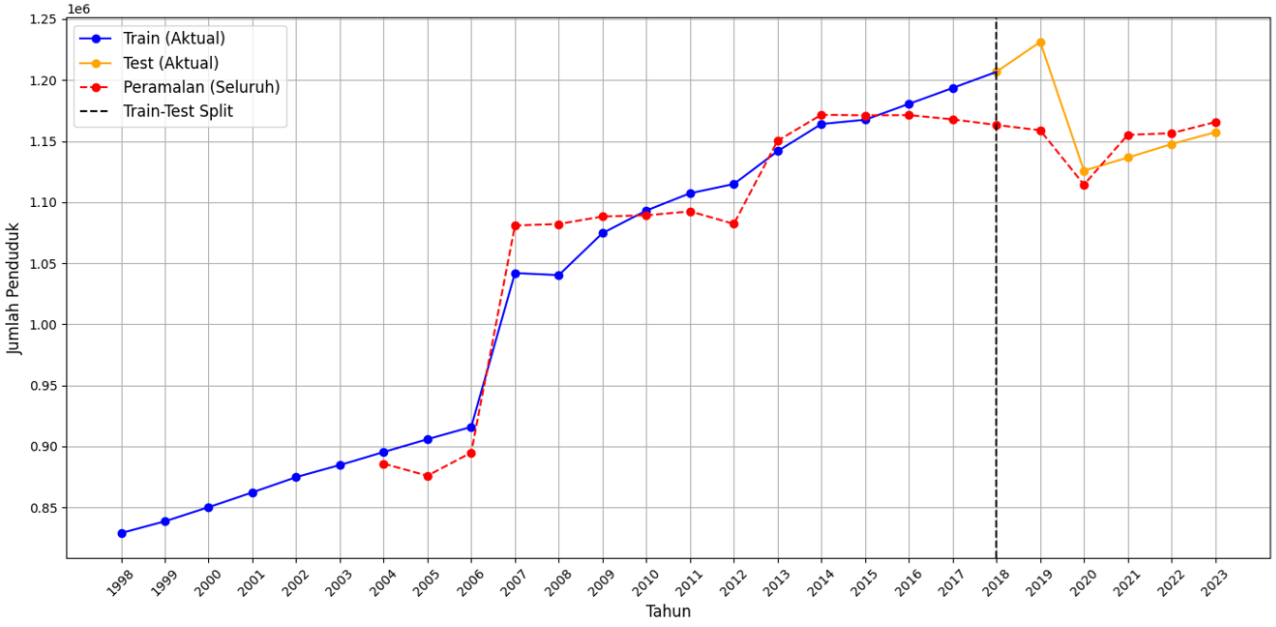
$$w_{17} = \frac{\hat{\mu}_{L_1}(X_7)}{\hat{\mu}_{L_1}(X_7) + \hat{\mu}_{L_2}(X_7) + \hat{\mu}_{L_3}(X_7)}$$

$$
\begin{aligned}
&= \frac{0,960198}{0,960198 + 0,031489 + 0,021720} \\
&= 0,947495 \\
w_{27} &= \frac{\hat{\mu}_{L_2}(X_7)}{\hat{\mu}_{L_1}(X_7) + \hat{\mu}_{L_2}(X_7) + \hat{\mu}_{L_3}(X_7)} \\
&= \frac{0,031498}{0,960198 + 0,031489 + 0,021720} \\
&= 0,031072 \\
w_{37} &= \frac{\hat{\mu}_{L_3}(X_7)}{\hat{\mu}_{L_1}(X_7) + \hat{\mu}_{L_2}(X_7) + \hat{\mu}_{L_3}(X_7)} \\
&= \frac{0,021720}{0,960198 + 0,031489 + 0,021720} \\
&= 0,021433
\end{aligned}
$$

Kemudian dihitung defuzzifikasi hasil peramalan menggunakan Persamaan 2.28 sebagai berikut:

$$
\begin{aligned}
\hat{X}_7 &= \sum_{i=1}^{3} w_{i7}\, v_i \\
&= 0,947495 \times 872.659 + 0,031072 \times 1.089.389 + 0,021433 \times 1.173.149 \\
&= 885.833,6 \approx 885.834
\end{aligned}
$$

Sehingga hasil peramalan jumlah penduduk Kabupaten Sleman untuk $t = 7$ atau tahun 2024 adalah 885.834 jiwa.



Gambar 3.1: Peramalan Jumlah Penduduk di Kabupaten Sleman dengan FTS YEA

3.2. Perhitungan Akurasi Menggunakan MAPE

Untuk mengetahui nilai kesalahan setiap metode digunakan MAPE. Berikut tabel nilai MAPE untuk data *training* dan *testing*:

Tabel 3.8: Nilai MAPE Data Training dan Testing FTS YEA

MAPE *Training* (80%)	MAPE *Testing* (20%)
2,22%	1,00%

4. Kesimpulan

Berdasarkan pembahasan mengenai metode Runtun Waktu Samar Orde Tinggi Yolcu-Egrioglu-Aladag (FTS YEA) untuk peramalan jumlah penduduk di Kabupaten Sleman, diperoleh nilai *Mean Absolute Percentage Error* (MAPE) sebesar 2,22% pada data training dan 1,00% pada data testing. Nilai MAPE yang rendah ini menunjukkan bahwa FTS YEA memiliki tingkat kesalahan yang kecil, sehingga dapat dikatakan akurat dan andal dalam meramalkan jumlah penduduk di Kabupaten Sleman.

Referensi

[1] P. Gupta, H. Ladia, K. Kakkar, K. Rai, Y. Agrawal, R. Mamgain, and N. Gaur, "Implementation of demand forecasting – a comparative approach," *Journal of Physics: Conference Series*, vol. 1714, no. 1, 2021. View online.

[2] F. Petropoulos, D. Apiletti, V. Assimakopoulos, M. Z. Babai, D. K. Barrow, S. B. Taieb, C. Bergmeir, R. J. Bessa, J. Bijak, J. E. Boylan, *et al.*, "Forecasting: theory and practice," *International Journal of forecasting*, vol. 38(3), pp. 705–871, 2022. View online.

[3] M. Pfenninger, D. N. Bigler, S. Rikli, and J. Osterrieder, "Wasserstein gan: Deep generation applied on financial time series," *SSRN Electronic Journal*, vol. 825215(825215), pp. 1–32, 2021. View online.

[4] P. O. Lucas, O. Orang, P. C. L. Silva, E. M. Mendes, and F. G. Guimaraes, "A tutorial on fuzzy time series forecasting models: Recent advances and challenges," *Learning Nonlinear Models*, vol. 19(2), pp. 29–50, 2022. View online.

[5] O. C. Yolcu, U. Yolcu, E. Egrioglu, and C. H. Aladag, "High order fuzzy time series forecasting method based on an intersection operation," *Applied Mathematical Modelling*, vol. 40(19-20), pp. 8750–8765, 2016. View online.

[6] A. P. Windarto, D. Nasution, A. Wanto, F. Tambunan, M. S. Hasibuan, M. N. H. Siregar, M. R. Lubis, S. Solikhun, Y. Fadhillah, and D. Nofriansyah, *Jaringan Saraf Tiruan: Algoritma Prediksi dan Implementasi*. Yayasan Kita Menulis, 2020. View online.

[7] J. Moolayil, *Learn Keras for Deep Neural Networks: A Fast-Track Approach to Modern Deep Learning with Phyton. In Learn Keras for Deep*. Springer, 2018. View online.

Format Sitasi IEEE:
L. Amanah dan L. Harini. *"Implementasi Runtun Waktu Samar Orde Tinggi Yolcu-Egrioglu-Aladag dalam Melakukan Peramalan"*, *Jurnal Diferensial*, vol. 7(2), pp. 150-163, 2025.

This work is licensed under a Creative Commons "Attribution-ShareAlike 4.0 International" license.