



Multi-Objective Optimum Design of a Neuro-Fuzzy Network Using a Combined PSO and DE Algorithm Based on Fuzzy Logic

ARTICLE INFO

Article Type

Original Research

Authors

Pourhashem H.¹ MSc,
Jamali A.*¹ PhD,
Narimanzade N.¹ PhD

How to cite this article

Pourhashem H, Jamali A, Narimanzade N. Multi-Objective Optimum Design of a Neuro-Fuzzy Network Using a Combined PSO and DE Algorithm Based on Fuzzy Logic. Modares Mechanical Engineering. 2019;19(2):475-482.

ABSTRACT

Because of the widespread application in complex modeling based on experimental data, neuro-fuzzy networks have attracted the attention of researchers. In the neuro-fuzzy inference system, the objective is to reduce the system's prediction error relative to the actual data. The regulation of parameters of neuro-fuzzy network is very important and affects its performance. So, a new optimization algorithm based on Particle Swarm Optimization (PSO) and Differential Evolution (DE) has been proposed. In this algorithm, the coefficients of the operator speed are calculated dynamically, using fuzzy logic. These coefficients are set according to the generation number and variance of the particles. Proposed operator leads the particles to explore and exploit the search domain more precisely. Next, the performance of the proposed algorithm is checked by optimizing three benchmarks and comparing it with the results, which are obtained by conventional PSO and DE. The results show that the proposed algorithm obtained better solution in comparison with DE and PSO and proved its performance and efficiency. Finally, a neuro-fuzzy system has been employed to forecast the time series of Mackey-Glass. The parameters of this neuro-fuzzy network are optimized by the new algorithm and the PSO and DE method multi-objectively and the Pareto charts obtained by each method of optimization are compared with each other, indicating the better performance of the new algorithm.

Keywords Optimization; Particle swarm optimization; Differential evolution; Fuzzy logic; Neuro-Fuzzy network

¹Dynamic-Control-vibration Department, Mechanical Faculty, University of Guilan, Rasht, Iran

*Correspondence

Address: Mechanical Faculty, University of Guilan, Persian gulf highway, Rasht, Iran
Phone: +98 (813) 33843854
Fax: +98 (813) 33691065
ali.jamali@guilan.ac.ir

Article History

Received: May 06, 2018
Accepted: October 30, 2018
ePublished: February 01, 2019

CITATION LINKS

[1] Particle swarm optimization [2] Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces [3] Parameter selection in particle swarm optimization [4] Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients [5] Fuzzy adaptive particle swarm optimization [6] Particle swarm optimiser with neighbourhood operator [7] Species-based particle swarm optimizer enhanced by memory for dynamic optimization [8] Particle swarm optimization using dynamic tournament topology [9] Enhanced particle swarm optimizer incorporating a weighted particle [10] An improved adaptive differential evolution algorithm for continuous optimization [11] A new adaptive differential evolution optimization algorithm based on fuzzy inference system [12] Self-Adapting control parameters in differential evolution: a comparative study on numerical benchmark problems [13] An improved differential evolution algorithm for the economic load dispatch problems with or without valve-point effects [14] Particle Swarm Optimization and Differential Evolution Algorithms: Technical Analysis, Applications and Hybridization Perspectives [15] ANFIS: Adaptive-network-based fuzzy inference system [16] A comparison of performance of several artificial intelligence methods for forecasting monthly discharge time series [17] Neuro-fuzzy methods for nonlinear system identification

طراحی بهینه چندهدفی شبکه فازی- عصبی توسط الگوریتم ترکیبی تکامل تفاضلی و تجمعی ذره براساس منطق فازی

حامد پورهایم MSc

گروه دینامیک و کنترل و ارتعاشات، دانشکده مهندسی مکانیک، دانشگاه گیلان، رشت، ایران

علی جمالی* PhD

گروه دینامیک و کنترل و ارتعاشات، دانشکده مهندسی مکانیک، دانشگاه گیلان، رشت، ایران

نادر نریمانزاده PhD

گروه دینامیک و کنترل و ارتعاشات، دانشکده مهندسی مکانیک، دانشگاه گیلان، رشت، ایران

چکیده

شبکه‌های فازی- عصبی به دلیل کاربرد گسترده‌ای که در مدل‌سازی فرآیندهای پیچیده براساس داده‌های آزمایشگاهی دارند مورد توجه محققین قرار گرفته‌اند. در سیستم استنتاج فازی- عصبی هدف کاهش خطای پیش‌بینی سیستم نسبت به داده‌های اصلی است. تنظیم پارامترهای شبکه فازی- عصبی از اهمیت بسیار بالایی برخوردار است و بر عملکرد آن تأثیرگذار می‌گردد. برای این منظور الگوریتم بهینه‌سازی جدیدی مبتنی بر ترکیب دو روش بهینه‌سازی تکامل تفاضلی و تجمعی ذره ارایه شده است. در این الگوریتم ضرایب سرعت عملگرها با استفاده از منطق فازی به صورت دینامیکی محاسبه می‌شوند. این ضرایب با توجه به شماره نسل و واریانس ذرات تنظیم می‌شوند. در الگوریتم پیشنهادی سعی شده است که در محل‌های کم جمعیت‌تر فضای جست‌وجو بیشتر کاوش شود و از گیرافتادن الگوریتم در نقاط بهینه محلی جلوگیری شود. در ادامه با بهینه‌سازی سه تابع معیار نتایج حاصل از الگوریتم تکامل تفاضلی و تجمعی ذرات و روش پیشنهادی با هم مقایسه می‌شوند و عملکرد قابل قبول الگوریتم ارایه شده، قابل لمس است. در انتها شبکه فازی- عصبی برای پیش‌بینی سری زمانی تابع مکی- گلاس تعریف شده است که با استفاده از الگوریتم ارایه شده و الگوریتم‌های تکامل تفاضلی و تجمعی ذره، توابع تعلق مربوط به ورودی و خروجی شبکه فازی- عصبی به صورت دودسته بهینه می‌شوند و نمودار پارتو به دست آمده از این روش‌ها با هم مقایسه می‌شوند که نشان‌دهنده عملکرد بهتر الگوریتم پیشنهادی است.

کلیدواژه‌ها: بهینه‌سازی، بهینه‌سازی تجمعی ذره، تکامل تفاضلی، منطق فازی، شبکه فازی- عصبی

تاریخ دریافت: ۹۷/۰۲/۱۶

تاریخ پذیرش: ۹۷/۰۸/۰۸

*نویسنده مسئول: ali.jamali@guilan.ac.ir

۱- مقدمه

الگوریتم‌های تکاملی، روش‌های مناسبی برای بهینه‌سازی توابع پیچیده به شمار می‌روند. روش‌های تکاملی نسبت به سایر روش‌ها، روش‌های مطمئن‌تری از نظر گرفتارنشدن در نقاط بهینه محلی هستند. از این رو در این زمینه فعالیت‌های بسیاری صورت گرفته است. مبنای این الگوریتم‌ها سوق دادن جمعیت اولیه به سمت بهترین مکان موجود در فضای بهینه‌سازی است. این الگوریتم‌ها معمولاً از پدیده‌های طبیعی الهام گرفته شده‌اند. بین الگوریتم‌های بهینه‌سازی تکاملی، دو الگوریتم تکامل تفاضلی و تجمعی ذره در سال‌های اخیر مورد توجه محققان قرار گرفته است. روش بهینه‌سازی تجمعی ذرات (PSO) توسط کندی و /برهاردت در سال ۱۹۹۵ ارایه شد و /استورن و /پرایس در سال ۱۹۹۷ روش تفاضل تکاملی را ارایه دادند [1, 2].

عملگرهای ژنتیکی نقش بسیار مهمی در کاوش فضای جست‌وجو و رسیدن الگوریتم به نقاط بهینه دارند، بنابراین تنظیم این پارامترها از اهمیت بسیار بالایی برخوردار است. الگوریتم بهینه‌سازی تجمعی ذرات برگرفته از زندگی ماهی‌ها و پرندگان

است. در واقع جمعیت اولیه‌ای برای آن در نظر گرفته می‌شود که هر ذره از آن همانند ماهی‌ها یا پرندگان دارای حافظه‌ای از بهترین مکان‌های کاوش شده هستند. در واقع برای هر ذره، دو حافظه بهترین مکان دست‌یافته شخصی و بهترین مکان دست‌یافته گروهی تعریف می‌شود. ذرات به سمت این مکان‌ها حرکت می‌کنند و در انتها به بهینه‌ترین نقطه در فضای جست‌وجو همگرا می‌شوند [1]. در این الگوریتم برای هر ذره در هر لحظه سرعتی در نظر گرفته می‌شود که این سرعت تابعی از مکان آن ذره و بهترین حافظه شخصی و گروهی همان ذره است. در واقع این سرعت مکان، لحظه بعد ذره را مشخص می‌سازد. معادلات ۱ و ۲ سرعت ذره و مکان ذره در لحظه بعد را مشخص می‌سازد.

$$\begin{aligned} v_{id}(t+1) &= w_i \times v_{id}(t) + c_1 \times r_1 \times (p_{id}(t) - x_{id}(t)) + c_2 \times r_2 \times (p_{gd}(t) - x_{id}(t)) \end{aligned} \quad (1)$$

$$x_{id}(t+1) = x_{id}(t) + v_{id}(t) \quad (2)$$

w_i ضریب وزنی اینرسی و c_1 و c_2 ضرایب سرعت هستند. p_{id} بهترین حافظه شخصی ذره، p_{gd} بهترین حافظه گروهی ذره و r_1 و r_2 بیانگر اعداد تصادفی هستند.

کارهای متفاوتی در بهبود عملکرد الگوریتم بهینه‌سازی تجمعی ذرات صورت گرفته است. به عنوان مثال می‌توان ضریب وزنی اینرسی را ثابت یا متغیر با زمان در نظر گرفت یا این که مقدار بیشینه‌ای برای این ضریب قائل شد [3]. ضرایب سرعت نیز می‌توانند به این صورت باشند. در یک پژوهش [4] معادله‌ای متناسب با زمان برای ضرایب سرعت ارایه شده است. این ضرایب در پژوهشی [5] با استفاده از قواعد فازی به دست آمده‌اند. از جمله فرضیات دیگری که در نظر گرفته شده است، می‌توان به جداسازی ذرات در همسایگی مشخص اشاره نمود که توسط عملگرهای متفاوت جداسازی انجام می‌شود [6-8]. در یک پژوهش [9] برای جلوگیری از گیرافتادن در نقاط بهینه محلی برای ذرات متفاوت وزن‌های مشخصی تعریف شده است.

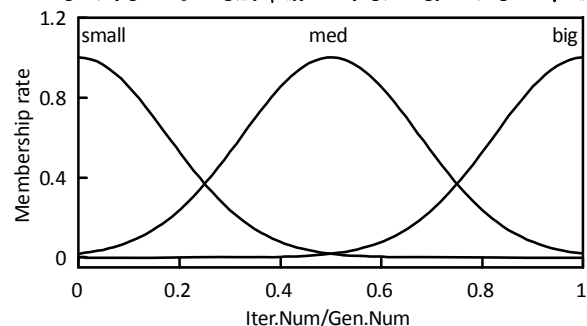
الگوریتم تکامل تفاضلی از دو بخش اصلی جهش و پیوند تشکیل شده است. در واقع جمعیت اولیه به عنوان جمعیت والد توسط عملگر جهش، جمعیت جدیدی به وجود می‌آورد و این جمعیت با جمعیت والد پیوند داده می‌شود و جمعیت نسل بعد را مشخص می‌سازد [2]. در این روش ابعاد بهینه‌سازی در هر ذره به عنوان کروموزوم‌های آن ذره در نظر گرفته می‌شوند. در عمل جهش ذرات والد که می‌توانند دو یا چند ذره باشند، ذره جدید را به وجود می‌آورند. در ادامه، کروموزوم‌های ذره نهایی به صورت تصادفی از دو ذره متناسب در جمعیت اولیه و ثانویه انتخاب می‌شود. معادلات ۳ و ۴ به ترتیب عملگرهای جهش و پیوند را مشخص می‌سازند.

$$v_i = x_{r1} + F \times (x_{r2} - x_{r3}) \quad (3)$$

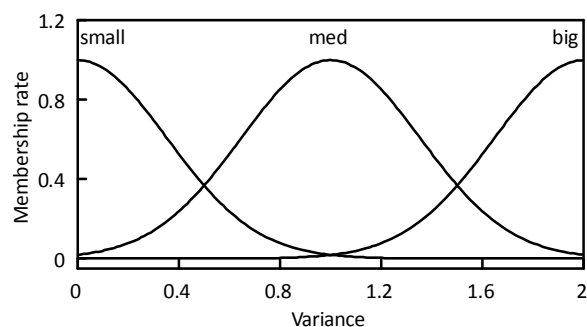
$$(u_i)_j = \begin{cases} (v_i)_j & r < Cr \\ or & j = n_i \\ (x_i)_j & otherwise \end{cases} \quad (4)$$

در معادله ۳، v_i ذره‌ای جهش‌یافته و x_{r1} ، x_{r2} و x_{r3} سه ذره از جمعیت اولیه هستند که به صورت تصادفی انتخاب شده‌اند. ضریب جهش F عددی ثابت در بازه (۰، ۱) است. عملگر جهش می‌تواند معادلات دیگری هم در خود جای دهد. برای مثال یک پژوهش [10] معادلات بیشتری ارایه داده است.

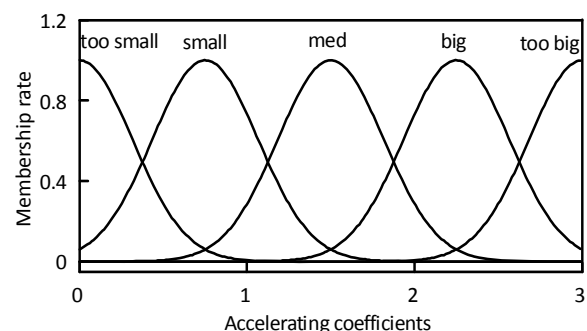
در معادله ۴ اندیس i بیانگر شماره هر ذره و اندیس j بیانگر هر کروموزوم ذره i ام است. همان طور که مشخص است، معادله ۴ را



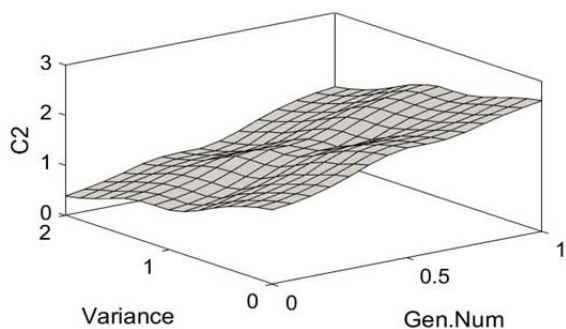
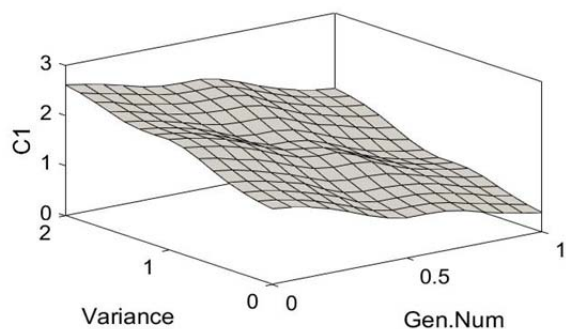
شکل (۱) تابع تعلق شماره نسل نسبی



شکل (۲) تابع تعلق ورودی دوم (واریانس)



شکل (۳) تابع تعلق ضرایب سرعت



شکل (۴) نمودارهای صفحات پایگاه قواعد جدول ۱

می‌توان به دو صورت تعریف نمود. C_r ثابت پیوند در بازه $(0, 1)$ و r عددی تصادفی در بازه $(0, 1)$ است. n_j نیز عددی تصادفی در بازه $(0, D)$ بوده که D همان ابعاد بهینه‌سازی است.

در روش بهینه‌سازی تکامل تفاضلی کارهای متعددی انجام گرفته است. یک پژوهش^[11] ضریب جهش را متغیر در نظر گرفته است و این ضریب را با استفاده از رویکرد فازی تعیین می‌کند. در پژوهش دیگری^[12] الگوریتم تکامل تفاضلی خود-تنظیم‌شونده با ضریب جهش و ضریب پیوند متغیر ارایه شده است. یک پژوهش^[13] روشی تصادفی برای انتخاب جمعیت اولیه معرفی نموده و در الگوریتم پیشنهادی خود از دو عملگر جهش و ضریب پیوند متغیر استفاده کرده است. همچنین پژوهش دیگری^[14] روشی ترکیبی مبتنی بر دو الگوریتم تکامل تفاضلی و تجمعی ذرات ارایه داده که با ترکیب عملگر جهش الگوریتم تفاضل تکاملی با سرعت در الگوریتم تجمعی ذرات، رابطه جدیدی برای سرعت در الگوریتم تجمعی ذرات تعریف نموده است.

۲- الگوریتم ترکیبی با ضرایب سرعت متغیر با زمان و وابسته به واریانس مبتنی بر تصمیم‌گیری فازی (FTVVAC)

در این قسمت الگوریتم جدیدی مبتنی بر روش فازی برای تنظیم پارامترهای الگوریتم تجمعی ذره و تکامل تفاضلی ارایه خواهد شد. در این الگوریتم عملگر جهش در الگوریتم تکامل تفاضلی و پارامتر سرعت ذرات در الگوریتم تجمعی ذرات با هم ترکیب شده و عملگر جدید جهش ارایه شده است، به نحوی که برای هر ذره جهش‌یافته متناسب با عضو λ_m جمعیت اولیه، دو حافظه گروهی و فردی تعریف می‌شود و ذره جهش‌یافته به سمت این دو حافظه سوق داده می‌شود. معادله ۵ عملگر جهش در الگوریتم پیشنهادی را نمایش می‌دهد.

$$v_i = x_r + c_1 \times r_1 \times (p_{id} - x_r) + c_2 \times r_2 \times (p_{gd} - x_r) \quad (5)$$

در معادله ۵، ضرایب به صورت تطبیقی تغییر خواهند کرد. برای تنظیم ضرایب موجود در این معادله از منطق فازی استفاده شده و سعی شده است تا تاثیر پراکندگی جمعیت بر ضرایب سرعت دیده شود. در نتیجه پایگاه قوانین فازی براساس واریانس جمعیت و شماره نسل، تنظیم خواهد شد. جدول ۱ این قوانین را مشخص می‌سازد. در واقع قوانین فازی طوری تنظیم شده‌اند که وقتی واریانس جمعیت بالا است یعنی ذره در نقاط خلوت‌تری قرار دارد، ذرات بیشتر به سمت تجربه شخصی‌شان حرکت کنند یا به عبارتی دیگر c_1 بزرگ‌تر شود. با گذشت زمان نیز ذرات به سمت تجربه گروهی حرکت می‌کنند، یعنی c_2 افزایش می‌یابد. توابع تعلق برای سیستم فازی با دو ورودی و دو خروجی در شکل‌های ۱-۳ آورده شده است. شکل ۴ نمودار صفحات تشکیل‌شده توسط پایگاه قواعد جدول ۱ را نشان می‌دهد.

جدول (۱) پایگاه قواعد تصمیم‌گیری فازی

ورودی‌ها		خروجی‌ها	
شماره نسل نسبی	واریانس	c_1 ضریب	c_2 ضریب
کم	بالا	خیلی بزرگ	خیلی کوچک
متوسط	بالا	بزرگ	کوچک
بالا	بالا	متوسط	متوسط
کم	متوسط	بزرگ	کوچک
متوسط	متوسط	متوسط	متوسط
بالا	متوسط	کوچک	بزرگ
کم	کم	متوسط	متوسط
متوسط	کم	کوچک	بزرگ
بالا	کم	خیلی کوچک	خیلی بزرگ

۳- نتایج تجربی برای توابع معیار

در ادامه برای بررسی میزان کارایی الگوریتم ارایه شده، سه تابع معیار معرفی می شود و با بهینه سازی این توابع با الگوریتم جدید ارایه شده و سایر الگوریتم ها، کارایی این روش ها را مورد مقایسه قرار می دهیم. معادله ۶ تابع معیار کروی، معادله ۷ تابع معیار رزنبراک و معادله ۸ تابع معیار آکلی را نمایش می دهند. نقطه بهینه در تابع کروی $x^* = (0, \dots, 0)$ ، در تابع رزنبراک $x^* = (1, \dots, 1)$ و در تابع آکلی نیز $x^* = (0, \dots, 0)$ بوده و مقدار این سه تابع در نقطه بهینه شان صفر است.

$$f_1 = \sum_{k=1}^d x_k^2 \quad (6)$$

$$f_2 = \sum_{k=1}^{d-1} 100 \times (x_{k+1} - x_k^2) + (1 - x_k)^2 \quad (7)$$

$$f_3 = -20e^{-0.2 \sqrt{\left(\frac{1}{d}\right) \sum_{k=1}^d x_k^2}} - e^{\frac{1}{d} \sum_{k=1}^d \cos(2\pi x_k)} + 20 + e \quad (8)$$

در جدول ۲، نتایج بهینه سازی توابع معیار برای تعداد نسل های ۲۰۰۰، تعداد متغیرهای طراحی ۳۰ (یعنی در معادلات توابع معیار $d = 30$ است) و برای تعداد جمعیت اولیه مختلف مشاهده می شود. اعداد موجود در این جدول نشان دهنده خطای مقدار توابع معیار در نقطه بهینه به دست آمده توسط روش های موجود با مقدار توابع معیار در نقاط بهینه واقعی را نشان می دهد. معادله ۹ این خطا را نمایش می دهد.

$$E = f(x_{\text{optimized}}) - f(x^*) \quad (9)$$

در معادله ۹، $f(x^*)$ مقدار تابع معیار در نقطه بهینه واقعی و $f(x_{\text{optimized}})$ مقدار تابع معیار در نقطه بهینه به دست آمده با روش های بهینه سازی است.

لازم به ذکر است که این نتایج حاصل میانگین ده بار اجرای

بهینه سازی است. هر چه مقدار E به دست آمده در جدول ۲ کوچک تر باشد، نشانگر عملکرد بهتر روش مربوطه است. همان طور که از نتایج ارایه شده در جدول مشخص است، روش مبتنی بر تصمیم گیری فازی در تابع کروی عملکرد بسیار مطلوبی دارد. در تابع رزنبراک نیز برای تعداد جمعیت بالای ۱۰۰ عدد، روش ترکیبی مبتنی بر منطق فازی عملکرد مطلوبی دارد و در تعداد جمعیت کمتر نیز این روش نسبت به سایر روش ها از عملکرد بهتری برخوردار است. برای تابع آکلی نیز روش مبتنی بر تصمیم گیری فازی نسبت به سایر روش ها به جواب بهینه تر دست یافته است.

جدول ۲) خطای بهینه سازی توابع معیار برای تعداد نسل های ۲۰۰۰ و ابعاد ۳۰

نام تابع معیار	جمعیت اولیه	DE	PSO	PSDE-FTVVAC
کروی (-۵۰۰، ۵۰۰)	۱۰۰	۰	۰/۰۵۶۹	۰
	۵۰	۰	۰/۳۸۱۸	۰
	۲۵	۰/۱۷۶۵	۲۳۶۷/۹	۰
رزنبراک (-۲/۰۴، ۲/۰۴)	۱۰۰	۱۵/۰۸۶	۳۲/۸۲۰۸	۰/۰۰۰۸
	۵۰	۲۳/۷۸۹۶	۵۵/۱۲۵۶	۴/۴۲۶
	۲۵	۲۷/۰۹۸۹	۹۵/۹۳۹۸	۱۸/۰۶۰۲
آکلی (-۳۲/۷، ۳۲/۷)	۱۰۰	۳/۳۹۴۹	۴/۹۶۷۴	۲/۲۰۸۶
	۵۰	۶/۲۵۵۱	۶/۳۲۴۶	۳/۱۲۰۱
	۲۵	۱۰/۹۷۴۵	۸/۴۷۹۱	۴/۲۰۴۶

جدول ۳ روش ارایه شده PSDE-FTVVAC را با روش های PSO-TVAC، MPSO-TVAC، HPSO-TVAC و PSO-DV برای جمعیت اولیه ۴۰ ذره با هم مقایسه می کند [۱۴]. در این مقایسه ابعاد d متغیر بوده و تعداد نسل های بهینه سازی نیز متفاوت در نظر گرفته شده است. معیار مقایسه نیز خطای مقادیر توابع معیار بوده که در معادله ۹ آورده شده است. هر چه این مقدار کمتر باشد، عملکرد بهتری را شاهد خواهیم بود. نتایج مطلوب به دست آمده توسط روش PSDE-FTVVAC در جدول ۳ بیانگر عملکرد خوب این روش است.

جدول ۳) مقایسه خطای روش های PSDE-FTVVAC با سایر روش های پیشنهادی [۱۴] برای توابع معیار مشخص

نام تابع معیار	تعداد ابعاد	تعداد نسل	PSO-TVAC	PSO-RANDW	PSO-TVAC	MPSO-TVAC	HPSO-TVAC	HPSO-FAC	PSDE-FTVVAC (this work)
کروی (-۱۰۰، ۱۰۰)	۱۰	۱۰۰۰	۰/۰۱	۰/۰۱	۰/۰۱	۰/۰۱	۰/۰۱	۰/۰۱	۰
	۲۰	۲۰۰۰	۰/۰۱	۰/۰۱	۰/۰۱	۰/۰۱	۰/۰۱	۰/۰۱	۱/۸۶E-۱۴۹
	۳۰	۳۰۰۰	۰/۰۱	۰/۰۱	۰/۰۱	۰/۰۱	۰/۰۱	۰/۲۳	۸/۴۵E-۸۷
رزنبراک (-۱۰۰، ۱۰۰)	۱۰	۳۰۰۰	۹/۹۵	۲/۱۰	۲۷/۱۱	۱۱/۱۲	۱۲/۹۷	۱۲/۶۹	۰
	۲۰	۴۰۰۰	۱/۴۹	۲۸/۱۸	۵۱/۵۶	۵۴/۴۰	۱۴/۰۹	۱۰/۱۳	۲/۳۱E-۲۴
	۳۰	۵۰۰۰	۲۸/۹۷	۳۵/۲۸	۶۳/۳۵	۱۳۵/۰۸	۱۳/۶۷	۷۰۶/۲۸	۱/۶۲E-۱۶

نیز کاهش می یابد تا ذرات به نقاط بهینه شخصی کمتر متمایل شوند. همچنین واریانس ذرات با افزایش نسل نسبت عکس دارد و کاهش پیدا می کند. در نتیجه در نسل های آخر ذرات کمترین واریانس و کمترین مقدار C_2 و بیشترین مقدار C_1 را دارند. همان طور که از نمودار ۱ مشخص است، در تابع رزنبراک در نسل های اولیه این ضرایب به مقدار برابر رسیده و به تغییرات با روند کندتری ادامه داده است. در واقع C_1 سریع کاهش می یابد و C_2 افزایش سریع تری دارد. دلیل این امر ذات تابع رزنبراک بوده که در آن تغییرات مقدار تابع در حوالی نقطه بهینه بسیار شدید است و ذرات در همان ابتدای بهینه سازی به مقادیر نزدیک به هم می رسند و واریانس آنها سریع کاهش پیدا می کند. این امر موجب سخت تر شدن بهینه سازی و کاهش سرعت همگرایی نیز می شود.

نمودار ۱، تغییرات ضرایب سرعت طی افزایش نسل برای بهینه سازی توابع معیار با ابعاد ۱۰ در جمعیت ۲۵ تایی با روش PSDE-FTVVAC را نمایش می دهد. با توجه به قواعد فازی تعبیه شده، همان طور که انتظار می رفت ضریب C_1 (ضریب سرعت حافظه شخصی) با گذشت زمان کاهش و ضریب C_2 (ضریب سرعت حافظه گروهی) با گذشت زمان افزایش می یابد. با توجه به این که افزایش نسل در فرآیند بهینه سازی نیازمند همگرایی ذرات بوده، در نتیجه این روند تغییرات در ضرایب سرعت مناسب است. اما علاوه بر شماره نسل، واریانس ذرات نیز در تغییرات این ضرایب تاثیرگذار هستند.

علت افزایش C_1 در طول افزایش نسل بدین منظور است تا ذرات بیشتر به سمت نقطه بهینه جهانی حرکت کنند و در این راستا C_2

ضرایب p, q و r با دریافت داده‌های یک مدل آموزش داده می‌شوند [17]. با توجه به شکل در لایه اول میزان عضویت ورودی‌ها به توابع تعلق سنجیده می‌شود. معادلات ۱۱ و ۱۲ این مقدار را به دست می‌آورد.

$$Q_{1,i} = \mu_{A_i}(x) \quad i = 1, 2 \quad (11)$$

$$Q_{1,i} = \mu_{B_{i-2}}(x) \quad i = 3, 4 \quad (12)$$

μ_{A_i} و $\mu_{B_{i-2}}$ همان توابع تعلق هستند که در این مقاله به صورت گاوسی در نظر گرفته شده‌اند.

$$\mu_{A_i}(x) = \exp\left(-\left(\frac{x-b_i}{a_i}\right)^2\right) \quad (13)$$

$$\mu_{B_{i-2}}(x) = \exp\left(-\left(\frac{x-b_i}{a_i}\right)^2\right) \quad (14)$$

ضرایب a و b در معادلات ۱۳ و ۱۴ می‌توانند منحصربه‌فرد باشند. مقادیر w_i و $\bar{w}_i$ در لایه دوم و سوم به صورت معادلات ۱۵ و ۱۶ هستند.

$$Q_{2,i} = w_i = \mu_{A_i} \times \mu_{B_i} \quad i = 1, 2 \quad (15)$$

$$Q_{3,i} = \bar{w}_i = \frac{w_i}{\sum_{i=1}^n w_i} \quad (16)$$

با داشتن مقادیر w_i و $\bar{w}_i$ مقادیر خروجی لایه‌های چهارم و پنجم به دست می‌آیند. معادلات ۱۷ و ۱۸ خروجی این لایه‌ها را نمایش می‌دهند.

$$Q_{4,i} = \bar{w}_i f_i = \bar{w}_i(p_i x + q_i y + r_i) \quad (17)$$

$$Q_{5,i} = \sum_{i=1}^n \bar{w}_i f_i = \frac{\sum_{i=1}^n w_i f_i}{\sum_{i=1}^n w_i} \quad (18)$$

در پیش‌بینی سری زمانی، هر نقطه با اطلاعات ۴ نقطه قبلی خود پیش‌بینی می‌شود. برای این منظور در بهینه‌سازی سیستم استنتاجی فازی- عصبی تطبیقی برای پیش‌بینی سری زمانی مکی- گلاس، ابتدا تعداد ۴۰۰ نقطه از داده‌های این سری به صورت تصادفی انتخاب می‌شود که هر کدام از آنها اطلاعات ۴ نقطه قبلی خود را دارند. این داده‌ها به عنوان داده‌های آموزشی در نظر گرفته می‌شوند. با آموزش سیستم استنتاج فازی- عصبی تطبیقی توسط نرم‌افزار MATLAB، ۲۰۰ داده دیگر پیش‌بینی می‌شود. همچنین ۴۰۰ داده اولیه نیز مجدداً با این روش پیش‌بینی می‌شود تا خطای حاصل از آن اندازه‌گیری شود.

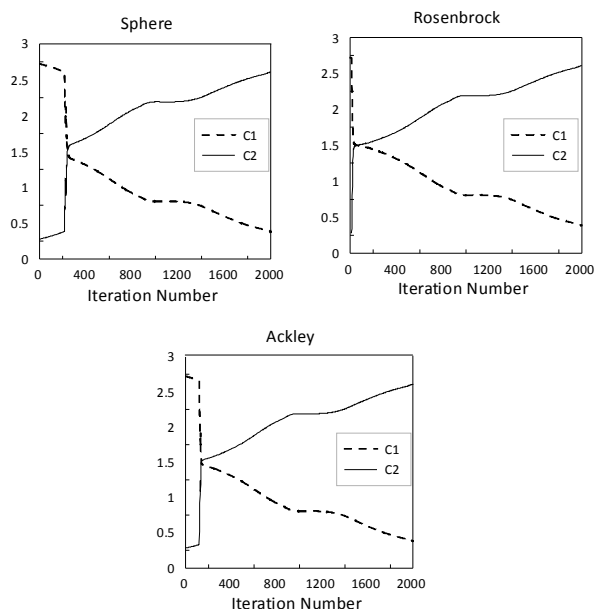
داده‌های آموزشی، پیش‌بینی و داده‌های ثانویه با مقادیر واقعی تابع مقایسه می‌شوند. خطای اندازه‌گیری این داده‌ها به عنوان اهداف بهینه‌سازی در نظر گرفته شده‌اند. یعنی بهینه‌سازی، دوهدفه است که توابع هزینه خطای پیش‌بینی داده‌های اولیه و خطای پیش‌بینی داده‌های ثانویه هستند. معادلات ۱۹ و ۲۰ این توابع هدف بهینه‌سازی را نمایش می‌دهند.

$$\text{Validation data error} = \sum_{i=1}^{200} (x_i - x'_i)^2 \quad (19)$$

$$\text{Training data error} = \sum_{j=1}^{400} (x_j - x'_j)^2 \quad (20)$$

در معادله ۱۹، i شماره داده‌های پیش‌بینی شده‌ای است که در داده‌های آموزشی نبودند و در معادله ۲۰، j شماره داده‌های آموزشی

همچنین موجب تأثیر بیشتر ذرات اولیه روی روند بهینه‌سازی می‌شود. همان طور که در جدول ۲ نیز مشخص است، بهینه‌سازی تابع روزنبراک دارای خطای بیشتری در تعداد نسل ۲۰۰۰ تایی نسبت به سایر توابع است. این در حالی است که مقدار توابع کروی و آکلی بازه تغییرات کمتری دارند و این امر موجب می‌شود واریانس آنها در نسل‌های اولیه سریع کاهش پیدا نکند، روند بهینه‌سازی بهتر انجام گیرد و در نتیجه بهینه‌سازی به نتایج مطلوب‌تری برسد.



نمودار ۱) نمودار تغییرات ضرایب سرعت نسبت به تعداد نسل

۳- بهینه‌سازی سیستم استنتاجی فازی- عصبی تطبیقی (ANFIS) برای مدل‌سازی فرآیندهای پیچیده

سیستم استنتاجی فازی- عصبی تطبیقی در سال ۱۹۹۳ توسط یانگ ارایه شد [15]. سیستم استنتاج فازی- عصبی تطبیقی در واقع یک سیستم فازی مبتنی بر سیستم استنتاج سوگنو فازی است که ترکیبی از هوش مصنوعی (ANN) و منطق فازی است. سیستم استنتاجی فازی- تطبیقی می‌تواند با دریافت داده‌های یک مدل، قوانین اگر و آن گاه را تعبیه کند و سایر داده‌های آن را پیش‌بینی کند.

در این مقاله هدف پیش‌بینی سری زمانی مکی- گلاس با استفاده از سیستم استنتاج فازی- عصبی است. معادله ۱۰۰ این سری زمانی را نمایش می‌دهد.

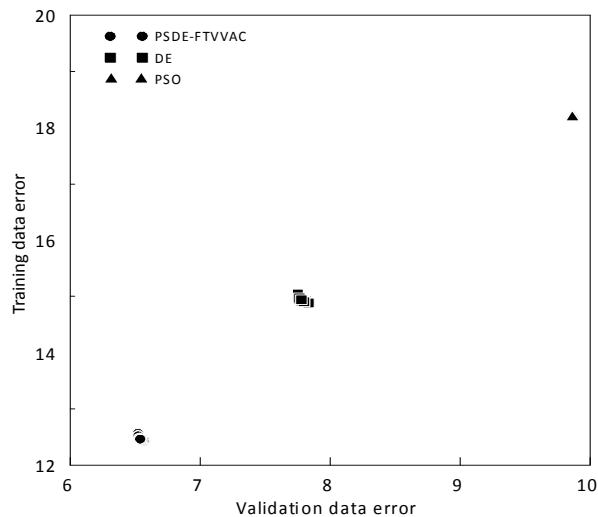
$$\frac{dx(t)}{dt} = \frac{\beta x(t-\tau)}{1+x^n(t-\tau)} - \gamma x(t) \quad (10)$$

ثابت‌های $\beta = 0.2$ ، $\gamma = 0.1$ و $n = 10$ و $\tau = 16$ در نظر گرفته شده‌اند.

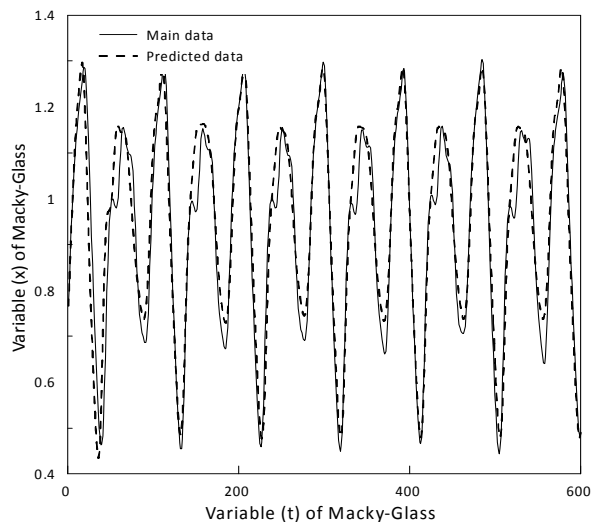
سیستم استنتاج فازی- عصبی تطبیقی از ۵ لایه تشکیل شده است. شکل ۵، سیستمی با دو ورودی x و y و یک خروجی f را نمایش می‌دهد [16]. قوانینی که شکل ۶ نمایش می‌دهد، عبارت از موارد زیر است:

۱- اگر x متعلق به A_1 باشد و y متعلق به B_1 ، آن گاه $f = p_1x + q_1y + r_1$

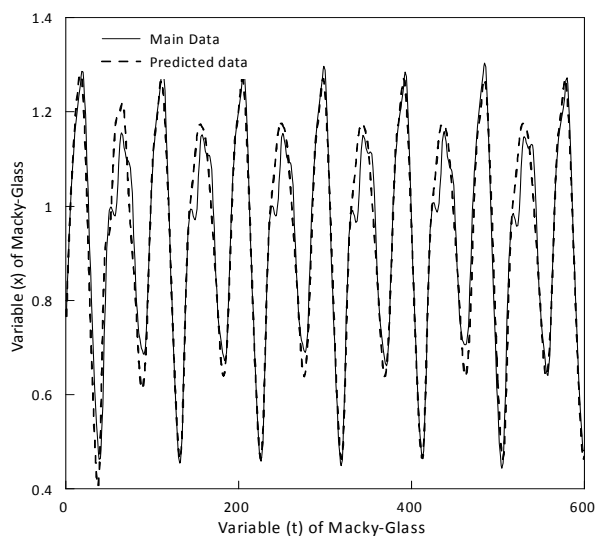
۲- اگر x متعلق به A_2 باشد و y متعلق به B_2 ، آن گاه $f = p_2x + q_2y + r_2$



نمودار ۲) مقایسه نمودارهای بهترین پارتوی به دست آمده با روش های متفاوت بهینه سازی



نمودار ۳) مقایسه نقاط پیش بینی شده (بهینه شده با PSO) با نقاط اصلی



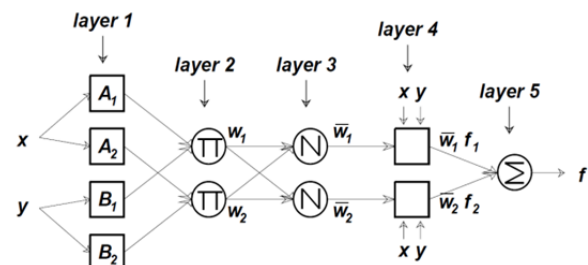
نمودار ۴) مقایسه نقاط پیش بینی شده (بهینه شده با DE) با نقاط اصلی

هستند که مجدداً پیش بینی شده اند. x_i و x_j مقدار واقعی هر داده و x'_i و x'_j مقدار پیش بینی شده همان داده است. متغیرهای طراحی بهینه سازی همان پارامترهای توابع تعلق هستند که برای هر کدام از ۴ داده ورودی، دو تابع تعلق تعریف شده است. همان طور که در معادلات ۱۳ و ۱۴ مشخص بوده، هر تابع تعلق شامل دو پارامتر اصلی a و b است (شکل ۶). در نتیجه پارامترهای ورودی بهینه سازی، ۱۶ عدد هستند. بهینه سازی چندهدفه تعریف شده در معادله ۲۱ نمایش داده شده است.

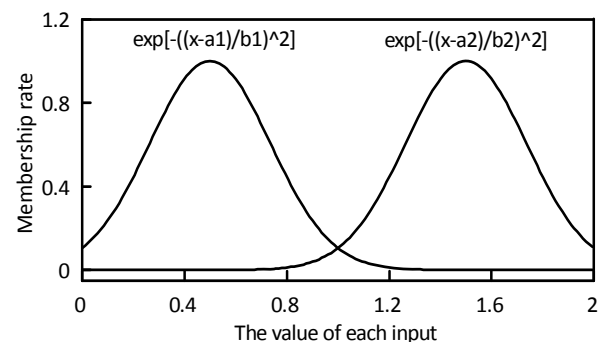
$$\text{Minimize } f(X) \\ = (\text{Validation data error}, \text{Training data error})$$

$$X = [a1_1, a1_2, a1_3, a1_4, a2_1, a2_2, a2_3, a2_4, b1_1, b1_2, b1_3, b1_4, b2_1, b2_2, b2_3, b2_4]$$

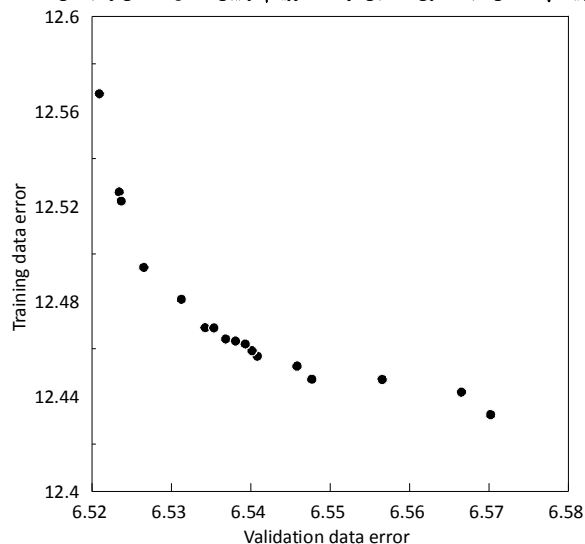
متغیرهای طراحی بهینه سازی در معادله ۲۱ همان پارامترهای توابع تعلق هستند که در شکل ۶ نمایش داده شده است. در انتها سیستم استنتاج فازی- عصبی تطبیقی تعریف شده با سه روش PSO، DE، PSDE-FTVVAC، بهینه و نتایج بهینه سازی به صورت پارتو نمایش داده می شوند و با هم مورد مقایسه قرار می گیرند. در نمودار ۲، پارتوی به دست آمده از هر روش در یک نمودار با هم مقایسه می شود. همان طور که از نمودار ۲ مشخص بوده، پارتوی به دست آمده توسط روش PSDE-FTVVAC نسبت به سایر پارتوها برتر بوده و به پاسخ بهتری رسیده است. در نمودارهای ۳-۵ مقادیر پیش بینی سیستم استنتاج فازی- عصبی تطبیقی بهینه شده با هر یک از روش های بهینه سازی معرفی شده آورده شده است. نمودارهای ۶-۸ نیز بهترین پارتوی به دست آمده توسط روش های بهینه سازی مذکور را نمایش می دهند.



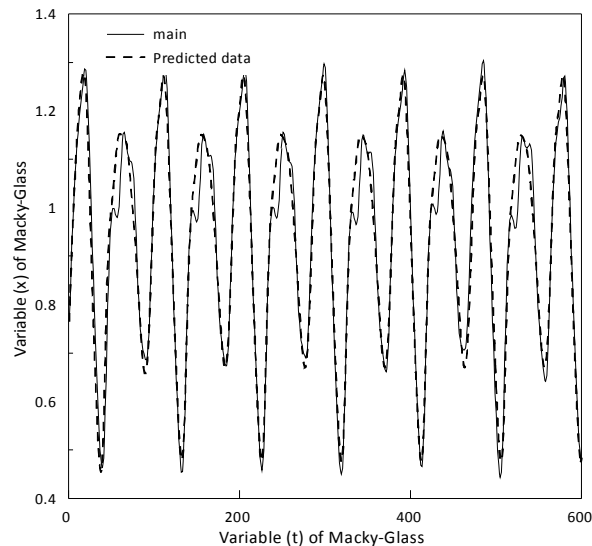
شکل ۵) ساختار سیستم استنتاج فازی- عصبی تطبیقی [۱۶]



شکل ۶) شکل کلی توابع تعلق هر ورودی سیستم فازی- عصبی که پارامترهای آن همان متغیرهای طراحی بهینه سازی هستند



نمودار ۸) نمودار بهترین پارتوی اهداف بهینه سازی (بهینه شده با PSDE-FTVVAC)



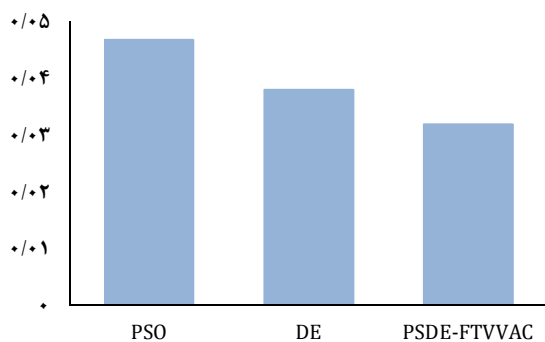
نمودار ۵) مقایسه نقاط پیش بینی شده (بهینه شده با PSDE-FTVVAC) با نقاط اصلی

۵- نتیجه گیری

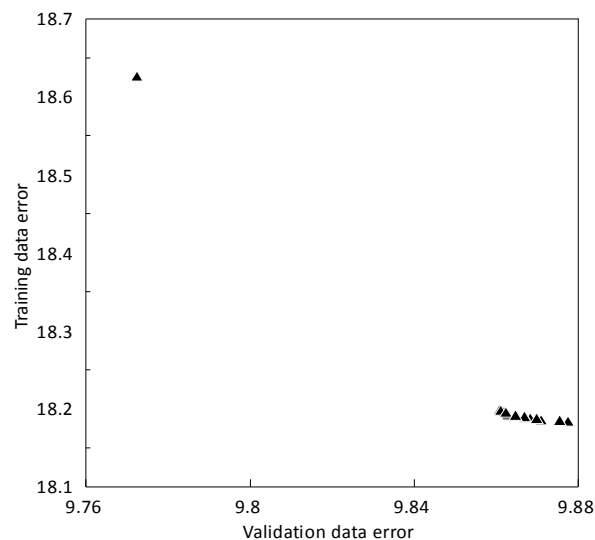
با توجه به نتایج پارتوهای نشان داده شده در نمودار ۲ می توان دریافت که روش $PSDE - FTVVAC$ عملکرد بهتری نسبت به دو روش دیگر داشته و منحنی پارتوی آن نسبت به منحنی های پارتوی روش های DE و PSO برتر است. این تفاوت با مقایسه نمودارهای ۳-۵ نیز توجیه می شود. به طوری که نمودار ۵، نمودار پیش بینی شده توسط روش $PSDE - FTVVAC$ را نمایش می دهد، در بسیاری از نقاط، نقاط پیش بینی شده روی نقاط اصلی منطبق شده اند.

برای بررسی بهتر نتایج عملکرد هر کدام از روش های DE ، PSO و $PSDE - FTVVAC$ به ازای یکی از نقاط پارتوی به دست آمده همان روش، میانگین خطای ۶۰۰ داده پیش بینی شده محاسبه شده است. این خطا در معادله ۲۲ با $\bar{e}$ نمایش داده شده است. در انتخاب نقطه پارتو، نقطه ای که تقریباً در مرکز منحنی پارتو قرار داشت، انتخاب شد. نمودار ۹، این مقدار خطای میانگین را در سه روش فوق مقایسه می کند. باید توجه داشت که مقادیر موجود در این نمودار مجموع خطای کل ۶۰۰ داده موجود تقسیم بر تعدادشان است. در انتها متغیرهای طراحی بهینه سازی در نقاط انتخاب شده پارتو در جدول ۴ نمایش داده شده است.

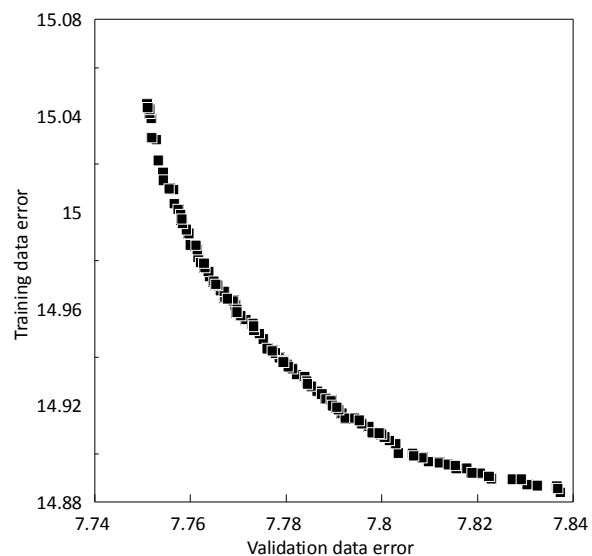
$$\bar{e} = \frac{\sum_{i=1}^{600} |e_i|}{600} \quad (22)$$



نمودار ۹) خطای میانگین بهینه سازی ANFIS در پیش بینی سری زمانی



نمودار ۶) نمودار بهترین پارتوی اهداف بهینه سازی (بهینه شده با PSO)



نمودار ۷) نمودار بهترین پارتوی اهداف بهینه سازی (بهینه شده با DE)

PSO	DE	PSDE – FTVVAC	Design Variable
۰/۲۰۱۸	۰/۱۹۵۷	۰/۲۴۷۳	b_{11}
۰/۱۵۴۸	۰/۲۳۰۳	۰/۲۱۴۵	a_{11}
۱/۲۳۶۶	۱/۰۴۳۶	۱/۰۸۴۳	b_{21}
۰/۵۲۶۴	۰/۵۷۹۱	۰/۶۱۹۶	a_{21}
۰/۷۵۴۷	۰/۷۸۵۸	۰/۷۱۵۴	b_{12}
۰/۶۴۱۴	۰/۶۰۵۵	۰/۴۸۰۰	a_{12}
۱/۸۴۳۷	۱/۳۹۴۰	۱/۲۵۰۵	b_{22}
۰/۴۱۶۳	۰/۶۰۱۸	۰/۵۷۴۵	a_{22}
۰/۶۳۳۰	۰/۹۳۲۱	۰/۸۹۵۹	b_{13}
۰/۴۹۰۳	۰/۳۲۹۴	۰/۶۷۰۵	a_{13}
۱/۶۶۶۰	۱/۳۱۳۲	۱/۶۷۳۴	b_{23}
۰/۶۶۴۳	۰/۴۹۳۶	۰/۴۱۴۷	a_{23}
۰/۷۳۱۰	۰/۶۲۶۱	۰/۶۶۰۶	b_{14}
۰/۵۰۷۲	۰/۶۴۱۸	۰/۲۵۶۵	a_{14}
۱/۸۲۲۰	۱/۲۹۸۶	۱/۱۵۰۹	b_{24}
۰/۱۹۴۵	۰/۵۳۴۸	۰/۶۳۲۹	a_{24}

با توجه به نمودار ۹ می‌توان دریافت که روش $PSDE - FTVVAC$ نسبت به روش DE خطا را حدود ۱۶/۱٪ و نسبت به روش PSO خطا را حدود ۳۱/۸٪ کاهش داده است. در مجموع با توجه به این که خطای میانگین بهینه‌سازی چندهدفه کاهش یافته است، می‌توان بهبود عملکرد را در روش $PSDE - FTVVAC$ شاهد بود.

تشکر و قدردانی: از کارکنان دانشکده مهندسی مکانیک که امکانات را برای انجام این پژوهش فراهم نمودند، تشکر می‌شود.

تأییدیه اخلاقی: محتویات علمی مقاله حاصل فعالیت علمی نویسندگان بوده و صحت و اعتبار نتایج بر عهده نویسندگان است.

تعارض منافع: مقاله حاضر هیچ گونه تعارض منافی با سازمان و اشخاص دیگر ندارد.

سهم نویسندگان: حامد پورهایم (نویسنده اول)، نگارنده مقدمه/پژوهشگر اصلی/نگارنده بحث (۳۴٪)؛ علی جمالی (نویسنده دوم)، روش‌شناس/پژوهشگر کمکی/نگارنده بحث (۳۳٪)؛ نادر نریمان‌زاده (نویسنده سوم)، روش‌شناس/پژوهشگر کمکی (۳۳٪)

منابع مالی: منابع مالی خاصی برای انجام این تحقیق استفاده نشده است.

منابع

- 1997;11(4):341-359.
- 3- Shi H, Liu S, Wu H, Li R, Liu S, Kwok N, et al. Oscillatory Particle Swarm Optimizer. *Applied Soft Computing*. 2018;73:316-327.
- 4- Ratnaweera A, Halgamuge SK, Watson HC. Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients. *IEEE Transactions on Evolutionary Computation*. 2004;8(3):240-255.
- 5- Nesamalar J, Venkatesh P, Raja S. Managing multi-line power congestion by using Hybrid Nelder-Mead-Fuzzy Adaptive Particle Swarm Optimization (HNM-FAPSO). *Applied Soft Computing*. 2016;43:222-234.
- 6- Wang H, Sun H, Li C, Rahnamayan S, Pan J. Diversity enhanced particle swarm optimization with neighborhood search. *Information Sciences*. 2013;223:119-135.
- 7- Luo W, Sun J, Bu C, Liang H. Species-based particle swarm optimizer enhanced by memory for dynamic optimization. *Applied Soft Computing*. 2016;47:130-140.
- 8- Wang L, Yang B, Orchard J. Particle swarm optimization using dynamic tournament topology. *Applied Soft Computing*. 2016;48:584-596.
- 9- Li NJ, Wang WJ, James Hsu CC, Chang W, Chou HG, Chang JW. Enhanced particle swarm optimizer incorporating a weighted particle. *Neurocomputing*. 2014;124:218-227.
- 10- Yi W, Zhou Y, Gao L, Li X, Mou J. An improved adaptive differential evolution algorithm for continuous optimization. *Expert Systems with Applications*. 2016;44:1-12.
- 11- Salehpour M, Jamali A, Bagheri A, Nariman-zadeh N. A new adaptive differential evolution optimization algorithm based on fuzzy inference system. *Engineering Science and Technology, an International Journal*. 2017;20(2):587-597.
- 12- Brest J, Greiner S, Boskovic B, Mernik M, Zumer V. Self-Adapting control parameters in differential evolution: A comparative study on numerical benchmark problems. *IEEE Transactions on Evolutionary Computation*. 2006;10(6):646-657.
- 13- Zou D, Li S, Wang GG, Li Z, Ouyang H. An improved differential evolution algorithm for the economic load dispatch problems with or without valve-point effects. *Applied Energy*. 2016;181:375-390.
- 14- Thangaraj R, Pant M, Abraham A, Bouvry p. Particle swarm optimization: Hybridization perspectives and experimental illustrations. *Applied Mathematics and Computation*. 2011;217:5208-5226.
- 15- Jang JSR. ANFIS: Adaptive-network-based fuzzy inference system. *IEEE Transactions on Systems, Man, and Cybernetics*. 1993;23(3):665-685.
- 16- Wang WC, Chau KW, Cheng CT, Qiu L. A comparison of performance of several artificial intelligence methods for forecasting monthly discharge time series. *Journal of Hydrology*. 2009;374(3):294-306.
- 17- Babuška R, Verbruggen H. Neuro-fuzzy methods for nonlinear system identification. *Annual Reviews in Control*. 2003;27(1):73-85.

1- Chen Y, Li L, Xiao J, Yang Y, Liang J, Li T. Particle swarm optimizer with crossover operation. *Journal of Engineering Applications of Artificial Intelligence*. 2018;70:159-169.

2- Storn R, Price K. Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*.