بهینه‌سازی ساختار الگوریتم درخت مدل خطی محلی با استفاده از الگوریتم بهینه‌سازی حدی

خلیل شریفی[(۱)] - محمدرضا احمدزاده[(۲)]

(۱) کارشناسی ارشد - دانشکده برق، دانشگاه آزاد اسلامی، واحد نجف‌آباد

(۲) استادیار - دانشکده برق و کامپیوتر، دانشگاه صنعتی اصفهان

تاریخ دریافت: پاییز ۱۳۸۸ تاریخ پذیرش: پاییز ۱۳۹۰

خلاصه: درخت مدل خطی محلی یا LOLIMOT که در آن از نوعی مدل فازی عصبی خطی محلی استفاده شده است، الگوریتمی بر اساس استراتژی تقسیم و حل می‌باشد که در آن حل مسئله پیچیده از طریق تقسیم مسئله به تعدادی زیر مسئله کوچک‌تر (و از این رو ساده‌تر) صورت می‌پذیرد. بنابراین مشخصات این مدل فازی- عصبی (زیرمسئله‌های کوچک‌تر شده) به مقدار زیادی به ساختار الگوریتم به کار برده شده جهت تقسیم‌بندی وابسته می‌باشد. الگوریتم LOLIMOT برای رسیدن به خروجی با خطای کمتر فضای مسئله را به تعدادی مدل خطی محلی یا LLM تقسیم می‌نماید و پس از پیدا کردن بدترین LLM (LLM با خطای بیشتر) با تقسیم آن به دو LLM الگوریتم را ادامه می‌دهد. در این الگوریتم در هر تکرار از آن بدترین LLM با نرخ تقسیم 1/2 در جهت‌های متعامد بر فضای ورودی تقسیم می‌شود. در این مقاله به کمک الگوریتم بهینه‌سازی حدی به بهینه‌سازی نرخ تقسیم می‌پردازیم، نتایج پیاده سازی حاکی از آن است که کارایی نسخه جدید الگوریتم LOLIMOT از نظر شاخص میانگین مربعات خطا بهتر از الگوریتم اولیه است.

کلمات کلیدی: مدل عصبی - فازی، شناسایی سیستم غیرخطی، درخت مدل خطی، الگوریتم بهینه سازی حدی (EO).

۱- مقدمه

سیستم‌های فازی دارای ساختار ویژه‌ای برای تحلیل و مدل سازی استدلال تقریبی هستند که تقریباً مانند آنچه انسانها در هنگام تصمیم‌گیری انجام می‌دهند می‌باشند. ساختار، تعداد و مجموعه‌ها و متغیرهای زبانی مناسب برای هر سیستم فازی (که کار خاصی را انجام می‌دهد) معمولاً توسط افراد خبره انتخاب می‌شود. عملی که در سیستم‌های فازی گوناگون صورت می‌گیرد، استخراج قوانین فازی توسط داده‌ها می‌باشد. ترکیب قدرت یادگیری شبکه‌های عصبی با دانش قرار گرفته در قوانین سیستم فازی به تولید سیستم‌های عصبی - فازی منجر شده است که به راحتی قوانین فازی را از روی داده‌ها استخراج می‌کند. الگوریتم یادگیری درخت مدل خطی محلی (LOLIMOT)[۱] یک روش یادگیری کاملاً خودکار با تعداد بسیار کم پارامترهای قابل تنظیم است که برای ایجاد و آموزش مدل‌های عصبی- فازی خطی محلی به کار می‌رود [۵-۲-۱]. یک مسئله مدل‌سازی پیچیده به تعدادی زیرمسئله کوچکتر و در نتیجه ساده‌تر تقسیم می‌شود از این رو دقت مدل‌های LLNF[۲] ضرورتاً به استراتژی تقسیم (نرخ تقسیم) وابسته است. اگرچه استفاده نکردن از روشهای بهینه‌سازی غیرخطی باعث افزایش سرعت الگوریتم LOLIMOT شده است اما ممکن است الگوریتم، مدل‌های محلی زیر بهینه و یا غیرضروری را تولید کرده و در نتیجه از قدرت الگوریتم در پیدا کردن ساختار بهینه کل کاسته شود. بنابراین یکی از نکات حیاتی برای بهبود عملکرد الگوریتم LOLIMOT بهینه‌سازی استراتژی تقسیم به وسیله روشهای جستجوی قدرتمند می‌باشد.

الگوریتم بهینه‌سازی حدی (EO)[۳] یک الگوریتم تکاملی است که برای حل مسائل بهینه‌سازی مورد استفاده قرار می‌گیرد. این الگوریتم برای تولید جوابهای بهتر، در هر مرحله یکی از اجزای آخرین جواب ایجاد شده را انتخاب و مقدار آن را با مقداری جدید جایگزین می‌کند. انتخاب یک جزء از بین اجزای جواب با توجه به شایستگی محلی آنها انجام می‌گیرد [۴-۳]. این مقاله شامل شش بخش می‌باشد. در بخش دوم این مقاله توصیف مدل فازی عصبی خطی محلی ارائه می‌شود. در بخش سوم الگوریتم یادگیری درخت مدل خطی محلی LOLIMOT تشریح خواهد شد و در بخش چهارم به معرفی روش پیشنهادی در توسعه درخت مدل خطی محلی با استفاده از الگوریتم بهینه‌سازی حدی می‌پردازیم. نتایج شبیه‌سازی و همچنین نتیجه‌گیری این مقاله به ترتیب در بخشهای پنجم و ششم ارائه شده است.

۲- مدل فازی عصبی خطی محلی

الف) سیستم فازی- عصبی

محاسبات فازی عصبی یکی از روشهای ترکیبی مدل سازی سیستم‌ها می‌باشد. یک سیستم فازی معمولاً با به خدمت گرفتن یک فرد خبره و فرموله کردن دانش خبره به مجموعه‌های زبانی و قوانین فازی طراحی می‌شود. یک انتخاب این است که قوانین فازی را از داده‌ها استخراج کنیم. ترکیب قابلیت‌های یادگیری شبکه‌های عصبی با قابلیت ارائه دانش توسط منطق فازی منجر به ایجاد سیستم‌های فازی عصبی شده است که قوانین فازی را از روی داده‌های آموزشی استخراج می‌کنند. معمولاً سیستم‌های فازی- عصبی به صورت معماری‌هایی شبیه شبکه‌های عصبی نمایش داده می‌شوند و با استفاده از روش‌های یادگیری آموزش می‌بینند.

ب) توصیف ریاضی مدل فازی عصبی خطی محلی

رویکرد اصلی در مدل‌های فازی عصبی خطی محلی تقسیم فضای ورودی به زیرمجموعه‌های کوچکتر با توابع اعتبار مربوط به هر مجموعه است که با ایجاد مدل‌های خطی کوچکتر (به جای کار با مدل غیرخطی اصلی) خروجی‌های مدل را بتوان محاسبه کرد. شکل (۱) ساختار یک مدل عصبی- فازی خطی محلی با M نرون و P ورودی را نشان می‌دهد. خروجی هر مدل خطی محلی به صورت زیر محاسبه خواهد شد.

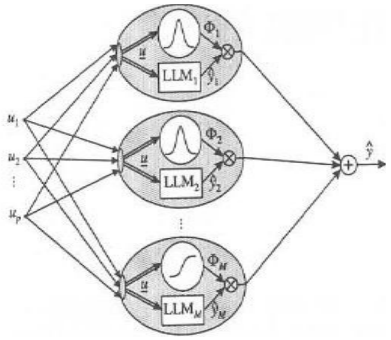
$$\hat{y}_i = w_{i0} + w_{i1}u_1 + w_{i2}u_2 + \cdots + w_{ip}u_p \qquad (۱)$$

و فرمول خروجی نهایی نیز برابر خواهد بود با:

$$\hat{y} = \sum_{i=1}^{M} \hat{y}_i \varphi_i(\underline{u}) \qquad (۲)$$

که در آن w_{ij} پارامتر iامین نرون و jامین ورودی بوده و M تعداد نرون‌ها می‌باشد و $\underline{u} = [u_1 u_2 \cdots u_p]^T$ ورودی مدل است. توابع اعتبار $\varphi_i(\underline{u})$ نیز به این صورت محاسبه می‌شوند:

$$\varphi_i(\underline{u}) = \frac{\mu_i(\underline{u})}{\sum_{j=1}^{M} \mu_j(\underline{u})} \qquad (۳)$$



شکل (۱): ساختار یک مدل عصبی- فازی خطی محلی با M نرون برای P ورودی

Fig. (1): The structure of a local linear neuro-fuzzy with M neurons for P input

$$\mu_i(\underline{u}) = \exp\left(\frac{-0.5\,(u_1 - c_{i1})^2}{\sigma_{i1}^2}\right)\times\ldots \times \exp\left(\frac{-0.5\,(u_p - c_{ip})^2}{\sigma_{ip}^2}\right) \qquad (۴)$$

بهینه‌سازی محلی پارامترهای هر مدل خطی به راحتی به روش حداقل مربعات حاصل می‌شود که این کار از طریق ایجاد یک ماتریس وزن و یک ماترس رگرسیون از ورودی‌ها و سپس به کارگیری روش کمترین مربعات انجام می‌شود. بنابراین ماتریس رگرسیون X از N نمونه داده و ماتریس وزن $\underline{Q}_i$ براساس فرمول‌های (۵) و (۶) به دست می‌آید:

$$\underline{X} = \begin{bmatrix} 1 & u_1(1) & u_2(1) & \cdots & u_p(1) \\ 1 & u_1(2) & u_2(2) & \cdots & u_p(2) \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & u_1(N) & u_2(N) & \cdots & u_p(N) \end{bmatrix} \qquad (۵)$$

$$Q_i = \begin{bmatrix} \varphi_i(\underline{u}(1)) & 0 & 0 & 0 \\ 0 & \varphi_i(\underline{u}(2)) & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \varphi_i(\underline{u}(N)) \end{bmatrix} \qquad (۶)$$

تخمین محلی پارامترهای هر مدل و خروجی هر مدل طبق فرمول (۷) محاسبه می‌شود.

$$\underline{\hat{W}}_i = [w_{i0} w_{i1} \ldots w_{ip}]^T$$
$$\hat{y}_i = \underline{X}\underline{\hat{W}}_i \;;\; \underline{\hat{W}}_i = (\underline{X}^T Q_i \underline{X})^{-1} \underline{X}^T Q_i \underline{Y} \qquad (۷)$$

و خطای هر مدل خطی محلی به صورت فرمول (۸) محاسبه می‌شود:

$$I_i = \sum_{j=1}^{N} \Phi_i(\underline{u}(j)) e^2(j) \qquad (۸)$$

که $e(j) = y(j) - \hat{y}(j)$ می‌باشد.

روشهای مختلفی برای بهینه‌سازی این مدل ارائه شده است که از این میان روش‌های مبتنی بر درخت به دلیل سادگی و شفافیت بسیار مناسب‌تر هستند [۵].

۳- الگوریتم یادگیری درخت مدل خطی محلی (LOLIMOT)

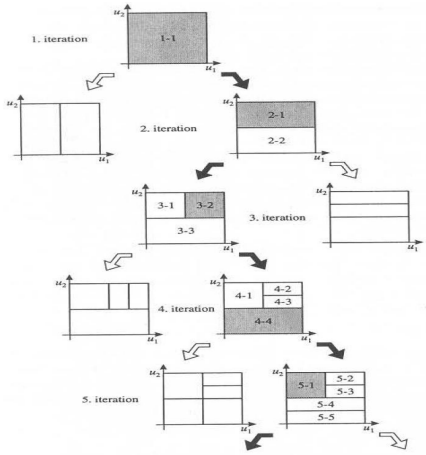
روش یادگیری درخت مدل خطی محلی (LOLIMOT) یک روش یادگیری افزایشی است که مبتنی بر تقسیم فضای ورودی است [۵]. الگوریتم LOLIMOT شامل ۵ مرحله است که به طور خلاصه در زیر آورده شده است:

۱- از یک مدل اولیه آغاز می‌شود: تعداد نرون‌ها M=1 و $\varphi_i(\underline{u}) = 1$ فرض می‌شود.

۲- بدترین مدل خطی محلی انتخاب می‌شود: بدترین LLM[۴] که بیشترین MSE[۵] را داراست انتخاب می‌شود.

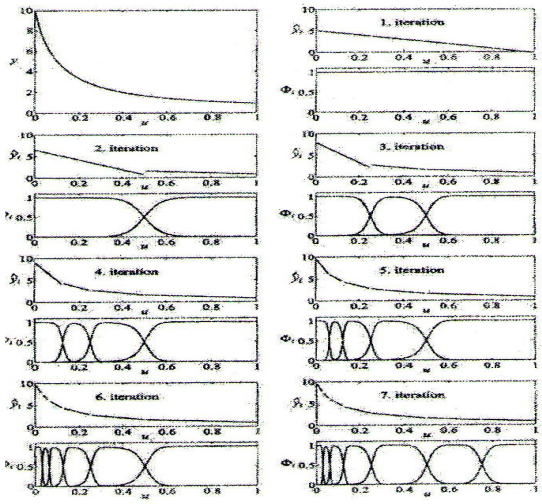
۳- بررسی کردن همه تقسیمات: بدترین مدل برای تقسیمات بیشتر در نظر گرفته می‌شود این تقسیمات به صورت عمود بر محور و در تمامی

ابعاد صورت می‌پذیرد برای تقسیم در هر بعد مراحل زیر انجام می‌شود:
الف) توابع عضویت چند بعدی برای دو مدل جدید به دست می‌آید.
ب) توابع اعتبار دو مدل جدید محاسبه می‌شود.
ج) پارامترهای دو مدل جدید به صورت محلی تخمین زده می‌شود.
د) شاخص خطای کلی مدل فعلی محاسبه می‌شود.
۴- بهترین حالتی که کمترین شاخص خطای کلی مدل را ایجاد کرد، انتخاب می‌شود. (M=M+1)
۵- در صورتی که شرط خاتمه برقرار باشد، الگوریتم متوقف می‌شود و در غیر این صورت از مرحله ۲ تکرار می‌شود.
برای درک بهتر الگوریتم فوق شکل (۲) را ملاحظه کنید.



شکل (۲): عملکرد الگوریتم جستجوی LOLIMOT در پنج مرحله تکرار نخست برای یک ورودی دو بعدی

Fig. (2): The action of search algorithm LOLIMOT in first five iterations for a two dimension input



شکل (۳): تخمین تابعy به وسیله درخت مدل خطی محلی در هفت مرحله تکرار نخست

Fig. (3): The estimation of y function by local linear tree in the first seven iterations

شکل (۳) ضمن ارائه یک مثال به تشریح الگوریتم LOLIMOT برای تخمین تابع یک بعدی y پرداخته است. تابع y به وسیله درخت مدل عصبی- فازی خطی محلی تخمین زده شده است. هفت مرحله تکرار نخست در این شکل نشان داده شده است.

۴- روش پیشنهادی درتوسعه درخت مدل خطی محلی با استفاده ازالگوریتم بهینه‌سازی حدی (EO)

الف) الگوریتم بهینه‌سازی حدی

الگوریتم EO یک الگوریتم تکاملی جدید است که بر اساس ویژگی خودسازمان ده بحرانی عمل می‌کند. در تحقیقات متعددی این الگوریتم برای حل مسائل بهینه‌سازی استفاده شده و جواب‌های مطلوبی ارائه داده است. برخلاف بسیاری از الگوریتم‌های تکاملی که در هر لحظه از چندین جواب تشکیل شده‌اند این الگوریتم تنها از یک جواب تشکیل شده است که در هر مرحله آن را بهبود می‌دهد. بهبود جواب در این الگوریتم با انتخاب یکی از اجزای جواب (جزء تعویضی) و جایگزین کردن مقدار آن با مقداری جدید انجام می‌شود. این انتخاب به گونه‌ای است که اجزا با شایستگی کمتر انتخاب می‌شوند. از دیگر مزایای الگوریتم EO علاوه بر داشتن تنها یک پارامتر برای تنظیم، می‌توان به ساختار بسیار ساده آن اشاره کرد [۷-۶-۴-۳]. الگوریتم EO به صورت شبه کد به صورت زیر می‌باشد:

۱- یک راه‌حل (پاسخ) را انتخاب می‌کنیم

$$x_{best} = x, c(x_{best}) = c(x) \quad , \; x = (x_1, x_2)$$

۲- برای این پاسخ:

آ) برازندگی λ_i را برای هر متغیر x_i محاسبه می‌کنیم.

ب) همه برازندگی‌ها را مرتب کرده و متغیر x_j را که دارای کمترین برازندگی است انتخاب می‌کنیم یعنی برای همه i: $\lambda_j \le \lambda_i$

پ) یک راه‌حل (پاسخ) x' را در همسایگی x انتخاب می‌کنیم به طوری که بایستی وضعیت متغیر x_j تغییر نماید.

ت) x را برابر 'x در نظر می‌گیریم ($x = x'$)

ث) اگر $c(x) < c(x_{best})$ است سپس $x_{best}=x$, $c(x_{best}) = c(x)$

۳- مرحله‌ی ۲ را به تعداد مطلوب تکرار می‌کنیم.

۴- d) $c(x_{best})$ و x_{best} را برمی‌گردانیم.

ب) ترکیب LOLIMOT و EO

ما از الگوریتم EO برای بهینه‌سازی نرخ تقسیم استفاده می‌کنیم. بدین منظور بدترین مدلی که در مرحله دوم تعیین می‌شود را یک پاسخ در نظر می‌گیریم به طوری که x_1 و x_2 دو مدلی هستند که از تقسیم این بدترین مدل ایجاد خواهند شد. در این الگوریتم برازندگی λ_i برابر عکس تابع هزینه iامین نرون تعریف می‌شود و $c(x)$ مجموع دو تابع هزینه متناظر با هریک از نرون‌های ایجاد شده خواهد بود یعنی $c(x) = I_1+I_2$ با مشخص شدن x_j پاسخ x به گونه‌ای تغییر خواهد یافت که تابع هزینه مربوط به بدترین مدل کمتر شده و در نتیجه پاسخ بهبود یابد. بر

این اساس الگوریتم EO جایگزین مرحله سوم الگوریتم LOLIMOT شده و بایستی برای تمامی ابعاد تکرار شود تا نهایتاً نرخ تقسیم بهینه به دست آید.

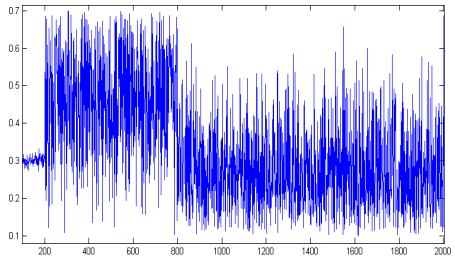
۵- نتایج شبیه سازی

در این قسمت عملکرد روش پیشنهاد شده برای شناسایی سیستم‌های Exchanger و Thermic-res-wall ارزیابی می‌گردد.

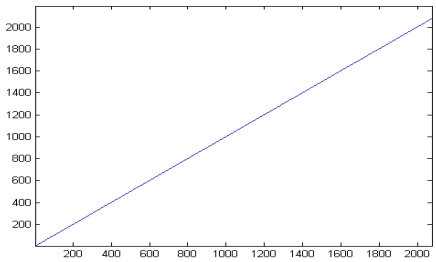
الف) شناسایی سیستم Exchanger

سیگنالهای ورودی- خروجی برای شناسایی سیستم در مرجع [۸] موجود می‌باشد. شکل (۴) 2000 نمونه از سیگنال‌های ورودی- خروجی برای شناسایی سیستم Exchanger را نشان می‌دهد. در ابتدا LOLIMOT اولیه برای شناسایی سیستم استفاده شده است.

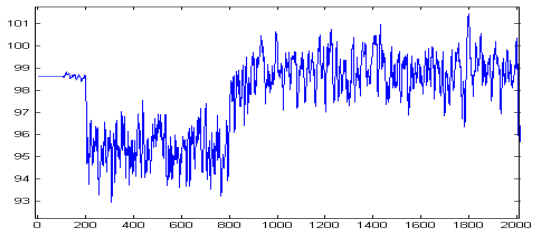
شکل (۵) خروجی حقیقی و خروجی تخمین زده شده سیستم و خطای بین آنها را در داده‌های آموزش نشان می‌دهد.



(الف)



(ب)



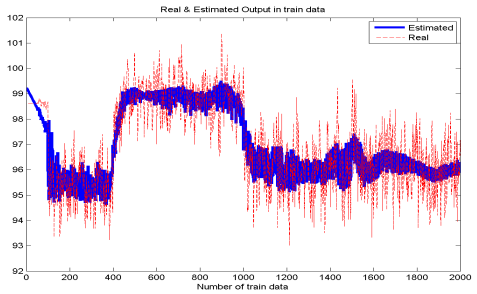
(ج)

شکل (۴): سیگنال‌های ورودی و خروجی برای شناسایی سیستم exchanger

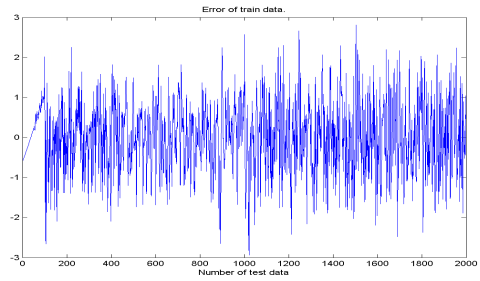
الف) سیگنال ورودی اول ب) سیگنال ورودی دوم ج) سیگنال خروجی

Fig. (4): The input and output signals to identify the exchanger system

(a) The first input signal (b) the second input signal (c) the output signal



(الف)



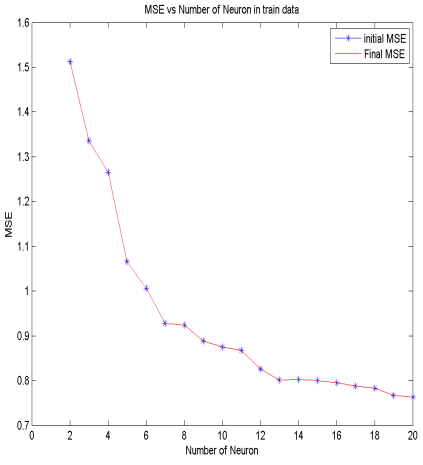
(ب)

شکل (۵): الگوریتم LOLIMOT: الف) خروجی واقعی و خروجی تخمین زده شده سیستم در داده‌های آموزش ب) خطای بر روی داده‌های آموزش

Fig. (5): The LOLIMOT algorithm: (a) the actual and estimated output system in the trained data. (b) the error on the trained data

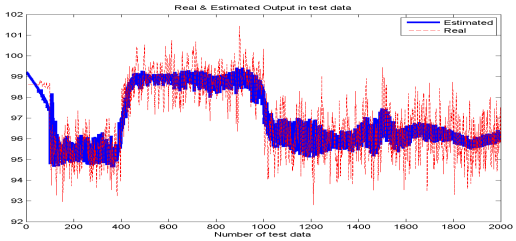
شکل (۶) MSE داده‌های آموزش را در هر تکرار از الگوریتم LOLIMOT نشان می‌دهد. در این حالت MSE داده‌های آموزش به ازای 20 نرون برابر 7633 می‌باشد. واضح است که با افزایش تعداد نرون MSE کاهش می‌یابد.

شکل (۷) نیز خروجی تخمین زده شده و خروجی حقیقی سیستم و خطای بین آنها را در داده‌های تست نشان می‌دهد.

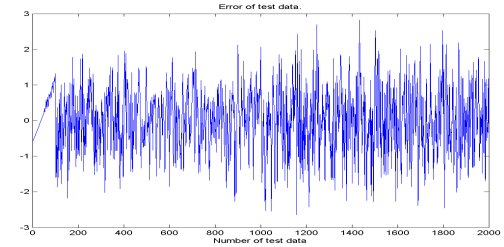


شکل (۶): نمودار MSE بر حسب تعداد نرون‌ها در داده‌های آموزش با استفاده از الگوریتم LOLIMOT

Fig. (6): The variations of MSE as a function of neuron numbers in the trained data using the LOLIMOT algorithm



(الف)



(ب)

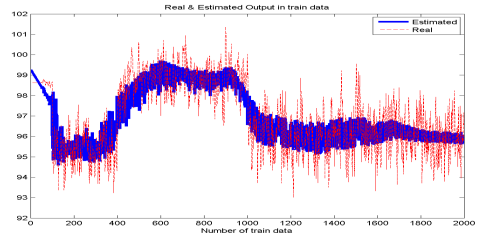
شکل (۷): الگوریتم LOLIMOT: الف) خروجی واقعی و خروجی تخمین زده شده سیستم در داده‌های تست ب) خطای بر روی داده‌های تست

Fig. (7): The LOLIMOT algorithm: (a) The actual and estimated output of the system in tested data (b) The error in tested data

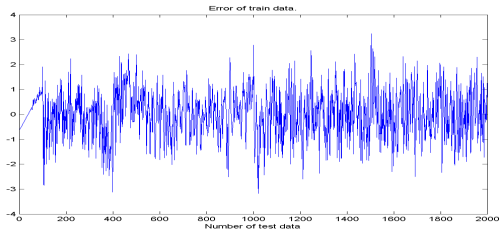
اکنون LOLIMOT توسعه یافته به وسیله الگوریتم EO برای شناسایی سیستم Exchanger در نظر گرفته می‌شود.

شکل (۸) خروجی تخمین زده شده و خروجی حقیقی سیستم و خطای بین آنها را در داده‌های آموزش نشان می‌دهد.

شکل (۹) نیز خروجی تخمین زده شده و خروجی حقیقی سیستم و خطای بین آنها را در داده‌های تست نشان می‌دهد.



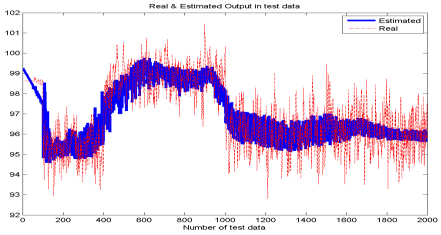
(الف)



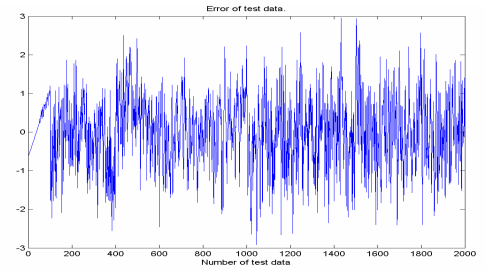
(ب)

شکل (۸): الگوریتم LOLIMOT توسعه یافته با EO : الف) خروجی واقعی و خروجی تخمین زده شده سیستم در داده‌های آموزش ب) خطای بر روی داده‌های آموزش

Fig. (8): The improved LOLIMOT algorithm with EO: (a) The actual and estimated output in the tested data. (b) The error in the tested data



(الف)



(ب)

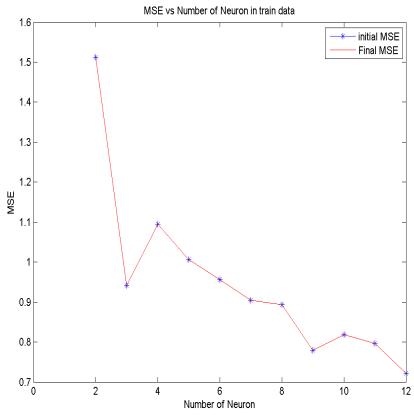
شکل (۹): الگوریتم LOLIMOT توسعه یافته با EO : الف) خروجی واقعی و خروجی تخمین زده شده سیستم در داده‌های تست ب) خطای بر روی داده‌های تست

Fig. (9): The improved LOLIMOT algorithm with EO: (a) The actual and the estimated output system in tested data. (b) The error in tested data

شکل (۱۰) MSE داده‌های آموزش را در هر تکرار از الگوریتم LOLIMOT نشان می‌دهد. مشاهده می‌شود که در این حالت سیستم با تعداد نرون خیلی کمتر (9 نرون) و همان خطای آموزش یعنی 7633 شناسایی می‌شود که این بیانگر قدرت الگوریتم ارتقا یافته می‌باشد.

شکل (۱۲) خروجی حقیقی و خروجی تخمین زده شده سیستم و خطای بین آنها را در داده‌های آموزش نشان می‌دهد.

در جدول (۱) خلاصه نتایج شبیه‌سازی‌های صورت گرفته داده شده است.



شکل (۱۰): نمودار MSE بر حسب تعداد نرون‌ها در داده‌های آموزش با استفاده از الگوریتم LOLIMOT توسعه یافته با EO

Fig. (10): The graph of MSE as a function of neuron number in trained data using improved LOLIMOT algorithm with EO

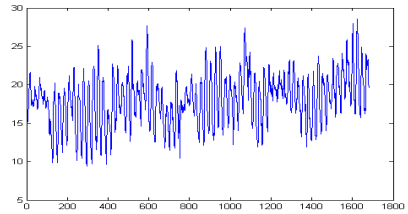
Table (1): The results obtained in identifying Exchanger system

جدول (۱): نتایج به دست آمده در شناسایی سیستم Exchanger

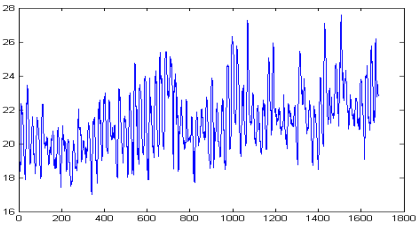
روش استفاده شده جهت شناسایی سیستم	LOLIMOT	LOLIMOT توسعه یافته با EO
تعداد نرون	۲۰	۹
MSE	۰/۷۶۳۳	۰/۷۶۳۳

ب) شناسایی سیستم Thermic-res-wall

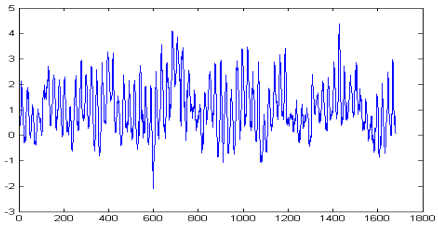
سیگنال‌های ورودی- خروجی برای شناسایی سیستم در مرجع [۸] موجود می‌باشد. شکل (۱۱) سیگنال‌های ورودی- خروجی را برای شناسایی سیستم Thermic-res-wall را نشان می‌دهد. در ابتدا LOLIMOT اولیه برای شناسایی سیستم استفاده شده است.



(الف)



(ب)



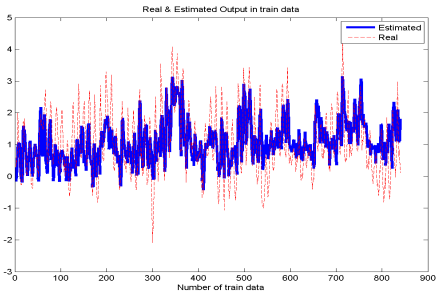
(ج)

شکل (۱۱): سیگنال‌های ورودی و خروجی برای شناسایی سیستم Thermic-res-wall

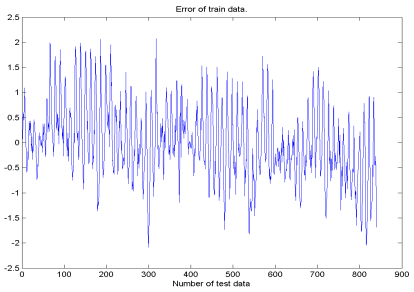
الف) سیگنال ورودی اول ب) سیگنال ورودی دوم ج) سیگنال خروجی

Fig. (11): The input and output signals to identify the Thermic-res-wall

(a) The first input signal (b) the second input signal (c) the output signal



(الف)



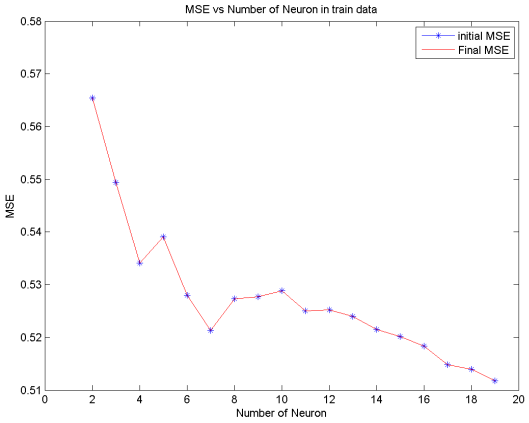
(ب)

شکل (۱۲): الگوریتم LOLIMOT: الف) خروجی واقعی و خروجی تخمین زده شده سیستم در داده‌های آموزش ب) خطای بر روی داده‌های آموزش

Fig. (12): The LOLIMOT algorithm: (a) The actual and estimated output to the system in trained data (b) The error in the trained data

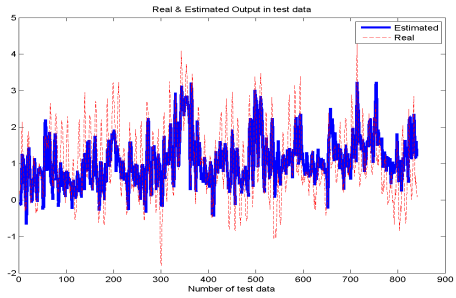
شکل (۱۳) MSE داده‌های آموزش را در هر تکرار از الگوریتم LOLIMOT نشان می‌دهد. در این حالت MSE داده‌های آموزش به ازای 19 نرون برابر 5118 می‌باشد. واضح است که با افزایش تعداد نرون MSE کاهش می‌یابد.

شکل (۱۴) نیز خروجی تخمین زده شده و خروجی حقیقی سیستم و خطای بین آنها را در داده‌های تست نشان می‌دهد.

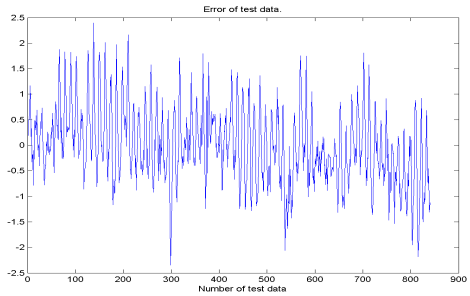


شکل (۱۳): نمودار MSE بر حسب تعداد نرون‌ها در داده‌های آموزش با استفاده از الگوریتم LOLIMOT

Fig. (13): The graph of MSE as a function of neuron numbers in the trained data using LOLIMOT algorithm



(الف)



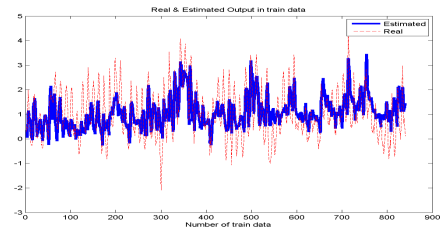
(ب)

شکل (۱۴): الگوریتم LOLIMOT: الف) خروجی واقعی و خروجی تخمین زده شده سیستم در داده‌های تست ب) خطای بر روی داده‌های تست

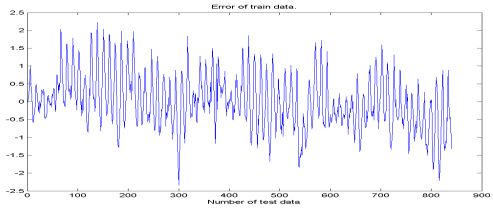
Fig. (14): The LOLIMOT algorithm: (a) The actual and estimated outputs of the system in tested data (b) The error in tested data

اکنون LOLIMOT توسعه‌یافته به وسیله الگوریتم EO برای شناسایی سیستم Thermic-res-wall در نظر گرفته می‌شود.

شکل (۱۵) خروجی تخمین زده شده و خروجی حقیقی سیستم و خطای بین آنها را در داده‌های آموزش نشان می‌دهد.



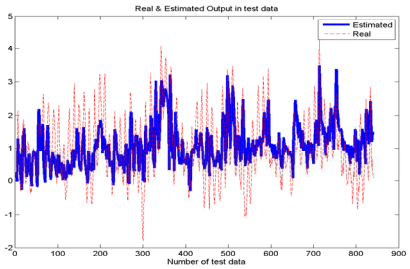
(الف)



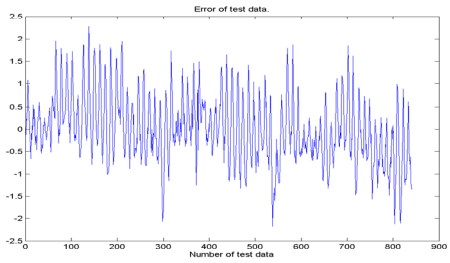
(ب)

شکل (۱۵): الگوریتم LOLIMOT توسعه یافته با EO: الف) خروجی واقعی و خروجی تخمین زده شده سیستم در داده‌های آموزش ب) خطای بر روی داده‌های آموزش

Fig. (15): The improved LOLIMOT with EO: (a) The actual and estimated outputs of the system in trained data (b) The error in the trained data



(الف)



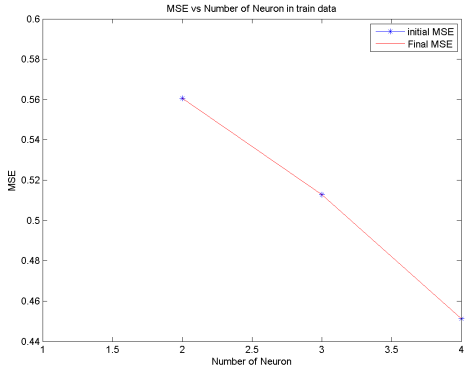
(ب)

شکل (۱۶): الگوریتم LOLIMOT توسعه یافته با EO: الف) خروجی واقعی و خروجی تخمین زده شده سیستم در داده‌های تست ب) خطای بر روی داده‌های تست

Fig. (16): The improved LOLIMOT algorithm with EO: (a) The actual and estimated outputs of the system in tested data (b) The error in the tested data

شکل (۱۶) نیز خروجی تخمین زده شده و خروجی حقیقی سیستم و خطای بین آنها را در داده‌های تست نشان می‌دهد.

شکل (۱۷) MSE داده‌های آموزش را در هر تکرار از الگوریتم LOLIMOT نشان می‌دهد. مشاهده می‌شود که در این حالت سیستم با تعداد نرون بسیار کمتر (4 نرون) و خطای آموزش کمتر یعنی 4512 شناسایی می‌شود که این بیانگر قدرت الگوریتم ارتقا یافته می‌باشد.



شکل (۱۷): نمودار MSE بر حسب تعداد نرون‌ها در داده‌های آموزش با استفاده از الگوریتم LOLIMOT توسعه یافته با EO

Fig. (17): The graph of MSE as a function of neuron numbers in the trained data using the improved LOLIMOT algorithm with EO

Table (2): The results obtained in identifying Thermic-res-wall
جدول (۲): نتایج به دست آمده در شناسایی سیستم Thermic-res-wall

روش استفاده شده جهت شناسایی سیستم	LOLIMOT	LOLIMOT توسعه یافته با EO
تعداد نرون	19	4
MSE	0.5118	0.4512

در جدول (۲) خلاصه نتایج شبیه‌سازی‌های صورت گرفته داده شده است.

۶- نتیجه‌گیری

روش یادگیری درخت مدل خطی محلی (LOLIMOT) یک روش مناسب و کارا برای شناسایی سیستم‌های غیرخطی می‌باشد. نرخ تقسیم در این الگوریتم همواره برابر 0.5 می‌باشد. از طرفی الگوریتم بهینه‌سازی حدی (EO) یک الگوریتم تکاملی جدید است که برای حل مسائل بهینه‌سازی استفاده شده و جواب‌های مطلوبی ارائه داده است. ما در این مقاله با استفاده از الگوریتم EO به بهینه‌سازی نرخ تقسیم در الگوریتم LOLIMOT پرداختیم و نشان دادیم که نسخه جدید الگوریتم به دست آمده کارآمدتر و قدرتمندتر از الگوریتم اولیه می‌باشد و با استفاده از این نسخه جدید در پایان شبکه‌ای خواهیم داشت که با تعداد مدل کمتر به خطای مطلوب‌تری دست خواهد یافت.

پی‌نوشت:

1- Local Linear Model Tree
2- Local Linear Neuro Fuzzy
3- Extremal Optimization
4- Local Linear Model
5- Mean Square Error

مراجع

[1] O. Nelles, "Nonlinear system identification with local linear neuro- fuzzy models", Ph.D. Thesis, TU Darmstadt, Automatisierungstechnik series., Shaker Varlag, Aachen, Germany, 1999.
[2] O. Nelles, A. Fink, R. Iserman, "Local linear model trees (LOLIMOT) for nonlinear system identification of cooling blast", EUFIT, p.p.1187-1191, Aachen, Germany, Sep. 1996.
[3] S. Boettcher, A.G. Oercus, "Extremal optimization: An evolutionary local- search algorithm", http:// arxiv. Org/ abs/cs. NE/020,030.
[4] P. Bak, C. Tang, K.W. Feld, "Self-organized criticaty", Phys. Rev. A, Vol.38, No.1, 1988.
[5] O. Nelles, "Nonlinear system identification", Springer, Meas. Scie. and Tec. Cre., Vol.13, No.4, 2001.
[6] S. Boettcher, A. Percus, "Natare's way of optimizing", Artificial Intell., Vol.119, no.1, pp.275-286. May 2000.
[7] Y.Z. Lu, M.R. Chen, Y.W. Chen, "Studies on extremal optimization and its applications in solving real world optimization problems", IEEE/FOCI, Honolulu, HI, pp.162-168, April 2007.
[8] http://homes. esat. kuleuven. be/-smc/daisy/