

یادگیری شبکه عصبی با روش لونبرگ-مارکوارت بهبودیافته

رضا یظهري کرمانی^۱، محمد ملائی امامزاده^{۲*}، مجتبی برخوردار یزدی^۳

^۱دانشجوی کارشناسی ارشد، دانشکده فنی و مهندسی، دانشگاه شهید باهنر کرمان، کرمان، ایران

^۲استادیار، دانشکده فنی و مهندسی، دانشگاه شهید باهنر کرمان، کرمان، ایران

^۳دانشیار، دانشکده فنی و مهندسی، دانشگاه شهید باهنر کرمان، کرمان، ایران

چکیده

در این مقاله روشی جدید برای افزایش سرعت همگرایی و کارایی روش لونبرگ-مارکوارت ارائه می‌شود. روش لونبرگ-مارکوارت یک روش مبتنی بر نیوتن است که در بهینه‌سازی و تعیین ضرایب شبکه‌های عصبی دارای کارایی مناسبی در مقایسه با سایر روش‌ها از جمله روش پس‌انتشار خطا است. با این حال، عملکرد این روش به‌طور قابل توجهی به انتخاب ضریب دمپینگ مناسب بستگی دارد. از جمله روش‌های مختلف تعیین ضریب دمپینگ، روش جستجوی خطی مارکوارت و روش‌های مبتنی بر نرم خطا و مبتنی بر نرم ژاکوبین می‌باشند که در این مقاله با بررسی نقاط ضعف و قوت این روش‌ها، یک روش ترکیبی جهت افزایش سرعت همگرایی ارائه می‌شود. در روش پیشنهادی بازه جستجوی ضریب دمپینگ و در نتیجه مقدار پارامتر نرخ تنظیم کاهش یافته است. با انجام این اصلاحات دقت جستجوی ضریب دمپینگ افزایش یافته و سرعت همگرایی روش پیشنهادی افزایش می‌یابد. به منظور ارزیابی روش پیشنهادی، یادگیری یک شبکه عصبی برای شناسایی یک تابع غیرخطی پیچیده، یک مسأله رگرسیون و یک مسأله طبقه‌بندی مورد شبیه‌سازی قرار گرفته و نتایج به‌دست آمده نشان می‌دهد که روش پیشنهادی در مقایسه با سایر روش‌های مورد بررسی، دارای کارایی مناسبی بوده و توانسته است خطای یادگیری را به حد قابل قبولی کاهش داده و سرعت همگرایی بالاتری داشته باشد.

مقاله پژوهشی

تاریخ دریافت:

۱۴۰۳/۹/۱۹

تاریخ پذیرش:

۱۴۰۳/۱۲/۸

کلیدواژه‌ها:

ضریب دمپینگ، الگوریتم لونبرگ-مارکوارت، الگوریتم نیوتن، یادگیری شبکه‌های عصبی، بهینه‌سازی غیرخطی

نویسنده مسئول:

molaie@uk.ac.ir

doi : 10.22034/ABMIR.2025.22500.1082

۱- مقدمه

به روش‌های مرتبه دوم دارای محاسبات ساده‌تر و کمتری می‌باشند که در حالت کلی باعث می‌شود از نظر زمان، حافظه و پیچیدگی محاسبات، کم‌هزینه‌تر باشند، که برای طیف گسترده‌ای از مسائل بسیار کاربردی می‌باشند. اما در عین مزایایی که دارند، دارای معایبی مانند وابستگی به تعیین نرخ یادگیری و همگرایی کندتر نسبت به روش‌های مرتبه دوم هستند.

روش‌های بهینه‌سازی مرتبه دوم هم مشتق اول و هم مشتق دوم (ماتریس هسین) تابع هدف را لازم دارند مانند الگوریتم نیوتن [۵]. اگرچه این روش‌ها از نظر محاسباتی هزینه‌برتر هستند و نیاز به زمان، حافظه و محاسبات بیشتری دارند، اما مهم‌ترین عیب آن‌ها این است که ممکن است در نقاط دور از نقطه بهینه همگرا نشوند، اما در نقاط نزدیک به جواب بهینه کارآمدتر باشند [۶].

مشکلات بعدی روش نیوتن حجم محاسباتی بالا و مسأله تکنیکی است. برای رفع مشکل بار محاسباتی روش نیوتن، ابتدا روش گاوس-نیوتن و سپس روش لونبرگ-مارکوارت ارائه شده که روش لونبرگ-مارکوارت علاوه بر رفع مشکل حجم محاسبات، با افزودن پارامتری به نام ضریب دمپینگ مسأله تکنیکی و همچنین مشکل واگرایی در نقاط دور از نقطه بهینه (که در صورت منفی معین شدن ماتریس هسین در نقاط ابتدایی جستجو ایجاد می‌شود) را حل کرد و با ارائه استراتژی تعیین ضریب دمپینگ متغیر، روش لونبرگ-مارکوارت را به ترکیبی از روش گرادیان و روش نیوتن تبدیل کرد. در نقاط ابتدایی جستجو با انتخاب ضریب دمپینگ بزرگ رفتار روش به گرادیان نزدیک‌تر و مشکل واگرایی حل می‌شود و در نقاط انتهایی جستجو با انتخاب ضریب دمپینگ کوچک رفتار روش به نیوتن شبیه‌تر شده و سرعت همگرایی روش در نزدیکی جواب بهینه افزایش می‌یابد [۶]. به همین دلیل کارایی روش لونبرگ-مارکوارت به شدت به استراتژی مورد استفاده در انتخاب ضریب دمپینگ وابسته است. در گذشته روش‌های متفاوتی برای تعیین ضریب دمپینگ ارائه شده است [۷].

در سال ۱۹۴۴ در [۸] دو روش برای تعیین ضریب دمپینگ معرفی شد: (۱) انتخاب مقداری ثابت و (۲) با استفاده از نرم دوم ماتریس حساسیت در هر تکرار. در سال ۱۹۶۳ در [۹] مارکوارت روشی

شبکه‌های عصبی به عنوان روشی با کارایی بالا در کاربردهای متنوعی همچون طبقه‌بندی، خوشه‌بندی، تشخیص الگو، پیش‌بینی و مدل‌سازی در بسیاری از حوزه‌های علمی و عملی مورد استفاده قرار گرفته‌اند. این شبکه‌ها زیرمجموعه‌ای از یادگیری ماشینی هستند و از نظر عملکرد و دقت، در مقایسه با مدل‌های رگرسیون و روش‌های آماری سنتی، برتری قابل توجهی از خود نشان می‌دهند [۱]. یکی از جدی‌ترین چالش‌هایی که شبکه‌های عصبی با آن‌ها مواجه هستند مسئله یادگیری و تنظیم پارامترها است. یادگیری در شبکه‌های عصبی را می‌توان یک مسئله بهینه‌سازی دانست که در آن، هدف به دست آوردن ضرایب بهینه به گونه‌ای است یک تابع هزینه خاص مرتبط با عملکرد مسأله بهینه (کمینه یا بیشینه) شود. روش‌های بهینه‌سازی، عمدتاً به دودسته کلی تقسیم می‌شوند: روش‌های مبتنی بر گرادیان [۲] و روش‌های هوش جمعی [۳]. روش‌های بهینه‌سازی مبتنی بر گرادیان از قدرت جهت‌یابی گرادیان به منظور یافتن راه‌حل‌های بهینه استفاده می‌کنند مانند روش پس انتشار خطا. در مقابل، روش‌های هوش جمعی از رفتار جمعی مشاهده شده در طبیعت، به ویژه از حرکات هماهنگ موجودات اجتماعی، الهام می‌گیرند مانند الگوریتم ازدحام ذرات. محدودیت اصلی روش‌های مبتنی بر گرادیان نیاز به داشتن مشتق تابع هزینه است که در صورت دسترسی به مشتق، به دلیل سادگی و حجم کم محاسبات و در عین حال وجود تضمین همگرایی، این روش‌ها دارای کارایی مناسبی می‌باشند.

به دلیل مشتق‌پذیری خروجی شبکه‌های عصبی، می‌توان از روش‌های مبتنی بر گرادیان در زمینه یادگیری این شبکه‌ها استفاده کرد و به دلیل استفاده این روش‌ها از قدرت جهت‌یابی گرادیان، می‌توان انتظار داشت که روش‌های مبتنی بر گرادیان در زمینه یادگیری این شبکه‌ها عملکرد بهتری نسبت به روش‌های تکاملی داشته باشند.

به طور کلی روش‌های بهینه‌سازی مبتنی بر گرادیان به دودسته اصلی روش‌های مرتبه اول و مرتبه دوم تقسیم می‌شوند.

روش‌های بهینه‌سازی مرتبه اول تنها بر مشتق اول تابع هدف تکیه دارند مانند الگوریتم گرادیان کاهشی [۴]. این روش‌ها نسبت

۲- طرح مسئله و مدل‌سازی ریاضی

در این بخش به مدل‌سازی ریاضی مسئله مورد بررسی و معرفی روابط پرداخته می‌شود.

۲-۱ الگوریتم گرادیان کاهشی

الگوریتم گرادیان کاهشی متداول‌ترین و پرکاربردترین روش بهینه‌سازی مرتبه اول است که در آن، برای رسیدن به جواب بهینه (کمینه کردن تابع هزینه F) متغیرهای بهینه‌سازی (w) را در خلاف جهت گرادیان حرکت داده (به‌روزرسانی کرده) که این مطلب در رابطه زیر بیان شده است:

$$w_{k+1} = w_k - \alpha_k \times \nabla F(w_k) \quad (1)$$

که در آن w_k و w_{k+1} به ترتیب مقدار فعلی و مقدار به‌روزرسانی شده پارامترهای بهینه‌سازی، α_k طول گام و ∇F مشتق تابع هزینه نسبت به پارامترهای w_k است. البته اگر هدف از بهینه‌سازی یادگیری شبکه عصبی باشد به α_k به جای طول گام، نرخ یادگیری هم گفته می‌شود. برای به دست آوردن ∇F باید از تابع هزینه نسبت به تمامی پارامترها طبق رابطه (۲) مشتق گرفته شود:

$$\nabla F(w_k) = \left[\frac{\partial F}{\partial w_1} \quad \frac{\partial F}{\partial w_2} \quad \frac{\partial F}{\partial w_3} \quad \dots \quad \frac{\partial F}{\partial w_n} \right] \quad (2)$$

که در اینجا n تعداد متغیرها و w_i متغیر i -ام مسئله بهینه‌سازی است.

$$w = [w_1 \ w_2 \ w_3 \ \dots \ w_n] \quad (3)$$

روش‌های مرتبه اول به دلیل سادگی و کارایی، برای حل طیف وسیعی از مسائل مناسب هستند. با این حال، وابستگی به نرخ یادگیری و همگرایی کندتر، از معایب این روش‌ها به شمار می‌رود.

۲-۲ الگوریتم نیوتن

در روش نیوتن مشتق مرتبه دوم تابع هزینه (ماتریس هسین) برای تعیین جهت بهینه‌سازی و طول گام استفاده می‌شود. در روش گرادیان (مرتبه اول) مقدار به‌روزرسانی متناسب با ∇F است و با نزدیک شدن به جواب بهینه سرعت همگرایی کاهش می‌یابد که این مساله در روش نیوتن (مبتنی بر گرادیان مرتبه دوم) حل شده است و رابطه زیر برای به‌روزرسانی متغیرها استفاده می‌شود:

$$w_{k+1} = w_k - H(w_k)^{-1} \times \nabla F(w_k) \quad (4)$$

برای تخمین پارامترهای غیرخطی در مجموعه‌ای از معادلات معرفی کرده است. در این مقاله روش جستجوی خطی برای تعیین مقدار ضریب دمپینگ معرفی شد. در این روش در هر تکرار ضریب دمپینگ طی یک جستجوی خطی به گونه‌ای تنظیم می‌شود تا اطمینان حاصل شود که در مرحله بعد تخمین دقیق‌تری به وجود آمده باشد. در [۱۰] برای تعیین ضریب دمپینگ از نرم خطا استفاده شده است. بزرگ‌ترین مشکل استفاده از این روش، در نقاط دور از نقطه بهینه است. چرا که در این نقاط با افزایش مقدار ضریب دمپینگ خیلی بزرگ است و باعث کاهش شدید طول گام و در نتیجه کند شدن الگوریتم می‌شود.

در [۱۰] و در [۱۱] به معرفی و بررسی روش‌هایی که از ماتریس حساسیت برای به دست آوردن ضریب دمپینگ استفاده می‌کنند پرداخته شده است. یکی از این روش‌ها، استفاده از ضرایب قطر اصلی این ماتریس است. در این حالت امکان از بین رفتن پارامتر وجود دارد و این روش را ناکارآمد می‌کند. روش دیگر استفاده از بزرگ‌ترین مقدار قطر اصلی است در این حالت نیز شانس زیادی برای از بین رفتن پارامتر وجود دارد. به این دلیل که حدس اولیه پارامترها ممکن است در مناطقی باشد که ضریب دمپینگ کافی تولید نمی‌کنند. نتایج این مقالات نشان می‌دهد که روش معرفی شده توسط مارکوارت می‌تواند تا حد زیادی سرعت الگوریتم را زمانی که در ناحیه یک دره است، افزایش دهد.

در این مقاله ابتدا با بررسی فرضیات و تئوری مورد استفاده در ارائه روش لونبرگ-مارکوارت نشان داده شده است که روش ضریب دمپینگ مبتنی بر خطا باید دارای کارایی بیشتری باشد و در ادامه هم نشان داده شده است که ضریب دمپینگ نه تنها باید مضربی از نرم خطا باشد بلکه باید در محاسبه آن از ضریبی متغیر برای نرم خطا (برحسب تکرار شبیه‌سازی) مورد استفاده قرار گیرد. سپس برای تعیین این ضریب، از روش جستجوی خطی پیشنهاد شده است. از آنجایی که اطلاعات انجام شده به منظور هم خوانی هر چه بیشتر با تئوری و فرضیات مسأله است انتظار می‌رود که این اصلاحات سرعت همگرایی روش و کارایی آن را بهبود ببخشد.

$$E_k \equiv E(w_k) \quad (7)$$

در ادامه ∇F به صورت زیر به دست می‌آید:

$$\nabla F \equiv \nabla F(w_k) = \frac{\partial E^2}{\partial w} = 2 \times E \times \frac{\partial E}{\partial w} \quad (8)$$

که $\frac{\partial E}{\partial w}$ مشتق خطاها نسبت به تمام پارامترها است که همان ماتریس ژاکوبین است. پس می‌توان رابطه (۸) را به صورت زیر نوشت:

$$J \equiv J(w_k) \square \frac{\partial E}{\partial w_k} \Rightarrow \quad (9)$$

$$\nabla F(w_k) = \frac{\partial E^2}{\partial w} = 2 \times E \times J$$

در ادامه می‌توان ماتریس H را به صورت زیر به دست آورد:

$$H \equiv \nabla^2 F = \frac{\partial^2 E^2}{\partial w^2} = \frac{\partial(E \times J)}{\partial E} \square \frac{\partial E}{\partial w} + \frac{\partial(E \times J)}{\partial J} \square \frac{\partial J}{\partial w} \quad (10)$$

$$H \equiv \nabla^2 F = \frac{\partial^2 E^2}{\partial w^2} = JJ^T + E \square \frac{\partial^2 E}{\partial w^2} \quad (11)$$

با محاسبه مشتقات ذکر شده در رابطه (۱۰) می‌توان آن را به دو قسمت (اولی فاقد مشتق دوم و دومی شامل مشتق دوم) تقسیم کرد. با توجه به این که کارایی روش نیوتن برای نواحی نزدیک به مقدار بهینه است، در این نواحی مقدار خطا نزدیک به صفر می‌شود.

به دلیل این که در جمله $E \square \frac{\partial^2 E}{\partial w^2}$ (که شامل مشتق دوم است) مقدار خطا مستقیماً وجود دارد و در نزدیکی جواب بهینه تقریباً صفر و قابل صرف نظر است لذا با چشم پوشی از جمله دوم (مشتق دوم) در ماتریس هسین می‌توان بدون نیاز به انجام محاسبات

پیچیده مشتق مرتبه دوم $E \square \frac{\partial^2 E}{\partial w^2}$ و فقط با استفاده از مشتق مرتبه اول J ماتریس هسین را با دقت مناسبی به صورت زیر تقریب زد:

$$H_{GN} = \lim_{E \rightarrow 0} JJ^T + E \square \frac{\partial^2 E}{\partial w^2} \Rightarrow H_{GN} = JJ^T \quad (12)$$

$$w_{k+1} = w_k - H_{GN}^{-1} \nabla F = w_k - (JJ^T)^{-1} \times J^T E_k \quad (13)$$

که این روش به نام روش گaus-نیوتن نامیده شده است [۱۳].

که در آن H ماتریس هسین است و به صورت زیر محاسبه می‌شود:

$$H \equiv \nabla^2 F = \frac{\partial \nabla F}{\partial w} = \frac{\partial^2 F}{\partial w^2} \quad (5)$$

با استفاده از H در قانون به روزرسانی روش نیوتن، سرعت همگرایی روش در رسیدن به جواب بهینه (نسبت به روش گرادیان) بیشتر شده است، اما برای محاسبه ماتریس H ، محاسبات پیچیده تری لازم است. این الگوریتم با در نظر گرفتن اطلاعات مرتبه دوم، جهت حرکت و اندازه حرکت را به طور مناسب تری تعیین می‌کند که در سرعت همگرایی و نزدیک شدن به جواب بهینه مفید است. از معایب الگوریتم نیوتن می‌توان به روابط پیچیده لازم برای محاسبه مشتقات مرتبه دوم موجود در ماتریس هسین اشاره کرد و همچنین نیاز به محاسبه معکوس آن و همین طور حساسیت روش نسبت به انتخاب اولیه مقدار متغیرها را ذکر کرد. همین طور در این الگوریتم در صورت غیر محدب بودن مسئله بهینه سازی ممکن است به نقاط بهینه محلی یا نقاط زینی همگرا شود. از آنجایی که الگوریتم نیوتن از معکوس ماتریس هسین برای محاسبه جهت حرکت در هر تکرار استفاده می‌کند، اگر ماتریس هسین در نقطه خاصی دارای مقدار ویژه صفر یا مقدار ویژه بسیار کوچک باشد، این روش ممکن است با تکینگی مواجه شود. برای رفع این مشکل تحقیقات و پژوهش‌هایی انجام شده و روش‌هایی همانند روش گaus-نیوتن، لونبرگ-مارکوارت و ... معرفی شده‌اند. که بدون محاسبه مستقیم ماتریس هسین و با تقریب آن از حجم محاسبات کاسته و هم چنین مشکل تکینگی را حل کرده‌اند.

۲-۳ الگوریتم گaus-نیوتن

الگوریتم گaus-نیوتن به دنبال تخمین ماتریس هسین بدون محاسبه صریح آن است، که فرآیند بهینه سازی را کارآمدتر می‌کند و در عین حال مزایای اطلاعات مرتبه دوم را نیز حفظ می‌کند [۱۲]. برای محاسبه ماتریس هسین باید $\nabla^2 F$ محاسبه گردد. F تابع هزینه در مسئله بهینه سازی است و در این مقاله تابع هزینه به صورت زیر تعریف می‌شود:

$$F \equiv F(w_k) = \frac{1}{2} E(w_k)^T E(w_k) \quad (6)$$

که در آن E بردار خطا است و با توجه به مسئله تعریف می‌شود.

۴-۲ الگوریتم لونبرگ-مارکوارت

در الگوریتم گاوس-نیوتن هم مانند الگوریتم نیوتن ممکن است تکنیکی رخ دهد. برای رفع این مشکل و ایجاد تعادل بین نقاط قوت روش‌های نیوتن و گرادیان کاهش الگوریتم لونبرگ-مارکوارت یک ضریب دمپینگ به نام (λ) معرفی می‌کند که پایداری فرآیند بهینه‌سازی را بهبود می‌بخشد [۱۴]. ماتریس تقریب زده شده در روش لونبرگ-مارکوارت پس از اعمال ضریب λ را می‌توان با رابطه (۱۴) نشان داد.

$$H_{LM} = JJ^T + \lambda I \quad (14)$$

در روش لونبرگ-مارکوارت با تنظیم مقدار λ می‌توان بین الگوریتم گرادیان کاهش و نیوتن جابه‌جا شد به این صورت که اگر مقدار λ بزرگ انتخاب شود نحوه به‌روزرسانی مانند روش گرادیان کاهش می‌شود:

$$\text{if } \lambda \square \|J\|^2 \Rightarrow H_{LM} \square \lambda I, \quad (15)$$

$$w_{K+1} = w_K - \lambda^{-1} JE$$

اگر مقدار کوچک انتخاب شود نحوه به‌روزرسانی مانند روش گاوس-نیوتن می‌شود:

$$\text{if } \lambda \square \|J\|^2 \Rightarrow H_{LM} \square JJ^T = H_{GN} \quad (16)$$

۳- روش‌های تعیین ضریب دمپینگ

همان‌طور که قبلاً اشاره شد، با انتخاب ضریب دمپینگ، روش لونبرگ-مارکوارت می‌تواند بین دو روش نیوتن و روش گرادیان تغییر وضعیت دهد. در این راستا بهترین استراتژی در انتخاب مقدار ضریب دمپینگ به این صورت است که در گام‌های اولیه که ممکن است روش نیوتن به دلیل امکان عدم مثبت معین بودن ماتریس H و همچنین به دلیل بزرگ بودن طول گام، واگرا شود، مقدار ضریب دمپینگ را بزرگ انتخاب شود تا رفتار روش لونبرگ-مارکوارت شبیه روش گرادیان شده و با رابطه (۱۵) طول گام آن‌هم در این حالت که متناظر با معکوس ضریب دمپینگ است کاهش پیدا کند. در نتیجه روش لونبرگ-مارکوارت در گام‌های اول با انتخاب ضریب دمپینگ بزرگ مشابه گرادیان با طول گام کوچک بوده و همگرا خواهد بود. به تدریج با نزدیک شدن به جواب بهینه، دیگر

مشکل عدم مثبت معین بودن H در روش روش نیوتن وجود ندارد و از طرفی سرعت همگرایی روش گرادیان کم می‌شود و این حالت با کاهش دادن مقدار ضریب دمپینگ مشابه رابطه (۱۶) روش لونبرگ-مارکوارت به روش گاوس-نیوتن شبیه شده و با سرعت مناسبی به سمت جواب بهینه همگرا می‌شود.

۳-۱ تعیین ضریب دمپینگ با استفاده از روش جستجوی خطی مارکوارت^۱

در ارائه روش لونبرگ-مارکوارت برای اولین بار در [۸] ضریب دمپینگ ثابت فرض شده و در [۹] برای تنظیم مقدار بهینه ضریب دمپینگ در حین حل مسئله در روش لونبرگ-مارکوارت از روش جستجوی خطی مشابه الگوریتم ۱ استفاده شد. در این روش در هر تکرار ضریب دمپینگ طی یک جستجوی خطی به گونه‌ای تنظیم می‌شود تا اطمینان حاصل شود که در مرحله بعد تخمین دقیق‌تری به دست آید. در هر تکرار مقدار تابع هزینه به ازای مقدار λ و $\frac{\lambda}{\beta}$ محاسبه می‌شود که در روش مارکوارت $\beta=10$ در نظر گرفته شده است.

در صورت کاهش تابع هزینه مقدار جدید λ در تکرار بعد استفاده می‌شود اما اگر تابع هزینه افزایش یابد مقدار λ در β ضرب می‌شود. این عمل تا زمان ادامه پیدا می‌کند که تابع هدف کاهش می‌شود. در این صورت مقدار جدید λ در تکرار بعد استفاده می‌شود.

الگوریتم (۱): الگوریتم تعیین ضریب دمپینگ به روش

مارکوارت

Let $\beta=10$.

Let $\lambda^{(r-1)}$ denote the value of λ that was calculated from the previous iteration.

Let $\lambda^{(0)} = 10^{(2)}$

Compute $\varphi'(\lambda^{(r-1)})$ And $\varphi'(\frac{\lambda^{(r-1)}}{\beta})$

If $\varphi'(\frac{\lambda^{(r-1)}}{\beta}) \leq \varphi'$, Let $\lambda^{(r)} = \frac{\lambda^{(r-1)}}{\beta}$

¹ Damping Factor using Line Search: DFLS

$$F = E^T E = \sum (y_r[k] - y_n[k])^2 \quad (20)$$

که در آن y_n خروجی شبکه عصبی و y_r خروجی سیستم واقعی است که توسط شبکه عصبی، شناسایی و مدل‌سازی می‌شود. رابطه بین خروجی y و ورودی x در سیستم واقعی به صورت زیر به عنوان یک رابطه غیرخطی نمایش داده شده است.

$$y = \sin(x), \quad x \in [0, 2\pi] \quad (21)$$

در این شبکه نشان داده شده در شکل ۱، وزن‌ها و بایاس‌های لایه میانی و وزن‌ها و بایاس لایه خروجی به عنوان پارامترهای بهینه‌سازی تعریف شده است.

$$W_1 \begin{bmatrix} w_{1,1} \\ \vdots \\ w_{1,10} \\ b_{1,1} \\ \vdots \\ b_{1,10} \end{bmatrix} W_2 \begin{bmatrix} w_{2,1} \\ \vdots \\ w_{2,10} \\ b_{2,1} \end{bmatrix} W = \begin{bmatrix} W_1 \\ W_2 \end{bmatrix} \quad (22)$$

که در آن $w_{1,j}$ وزن لینک واسطه بین ورودی و نرون j -ام لایه میانی است. همچنین $b_{1,j}$ بایاس مورد استفاده در تابع فعال‌سازی نرون j -ام است که این تابع تانژانت هایپربولیک در نظر گرفته شده است و به صورت زیر تعریف شده است.

$$\tanh(x) = \frac{(e^x - e^{-x})}{(e^x + e^{-x})} \quad (23)$$

همچنین $w_{2,j}$ وزن لینک واسطه بین خروجی j -ام نرون لایه میانی تا نرون لایه خروجی است و $b_{2,1}$ بایاس مورد استفاده در رابطه نرون خروجی است.

با استفاده از سه روش معرفی شده مارکوارت، نرم خطا و نرم ژاکوبین، مسأله یادگیری شبکه عصبی برای شناسایی تابع غیرخطی (۲۱) مطابق شکل ۱ انجام شده است. در شکل ۲ نتایج بهینه‌سازی برحسب تکرار و در شکل ۳ نتایج برحسب زمان اجرای روش آمده است.

همان‌طور در شکل ۲ مشاهده می‌شود، تعیین ضریب دمپینگ با روش مبتنی بر نرم ژاکوبین (DFNJ) سرعت همگرایی و خطای نهایی مناسبی ندارد. روش مارکوارت (جستجوی خطی با $DFLS:\beta=10$) دارای نتایج بهتری است اما روش مبتنی بر نرم

If $\phi' \left(\frac{\lambda^{(r-1)}}{\beta} \right) > \phi'$, And $\phi' \left(\lambda^{(r-1)} \right) \leq \phi'$ Let $\lambda^{(r)} = \lambda^{(r-1)}$
Else if $\phi' \left(\frac{\lambda^{(r-1)}}{\beta} \right) > \phi'$, And $\phi' \left(\lambda^{(r-1)} \right) > \phi'$, increase λ by successive multiplication by β until for smallest k , $\phi' \left(\lambda^{(r-1)} \beta^k \right) \leq \phi'$. Let $\lambda^{(r)} = \lambda^{(r-1)} \beta^k$
Return $\lambda^{(r)}$

۲-۳ تعیین ضریب دمپینگ با استفاده از نرم خطا^۲

در مرجع [۱۱] برای تعیین ضریب دمپینگ از نرم خطا استفاده شده است. در روش تعیین ضریب دمپینگ مبتنی بر نرم خطای جستجو از نرم خطای جستجو برای انتخاب مقدار ضریب دمپینگ در هر مرحله استفاده می‌شود. تقریب ماتریس هسین با استفاده از این روش طبق رابطه (۱۷) صورت می‌گیرد:

$$H_E = JJ^T + \|E\| I \quad (17)$$

در مرجع [۱۵] و [۱۰] با بررسی روش نرم خطا روشی با استفاده از ضریب دمپینگ ۰٫۱ نرم خطا معرفی شده و باعث بهبود عملکرد این روش شده است.

$$H_E = JJ^T + 0.1 \|E\| I \quad (18)$$

۳-۳ تعیین ضریب دمپینگ با استفاده از نرم ژاکوبین^۲

در روش تعیین ضریب دمپینگ مبتنی بر نرم ماتریس ژاکوبین از نرم ماتریس ژاکوبین برای انتخاب مقدار ضریب دمپینگ در هر مرحله استفاده می‌شود. تقریب ماتریس هسین با استفاده از این روش طبق رابطه (۱۹) صورت می‌گیرد:

$$H_J = JJ^T + \|J\| I \quad (19)$$

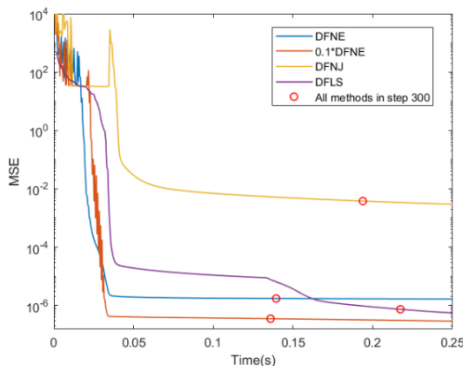
مزیت دو روش معرفی شده از نظر حجم محاسبات و زمان اجرا به پیچیدگی مسئله اضافه نمی‌کند و روش‌هایی ساده برای محاسبه مقدار ضریب دمپینگ هستند [۱۰].

۴- بررسی نتایج روش‌های قبلی

در این مقاله نتایج شبیه‌سازی الگوریتم لونبرگ-مارکوارت برای یک شبکه عصبی پرسپترون با یک لایه میانی و ۱۰ نرون در لایه میانی در ۳۰۰ تکرار برای تابع سینوس نمایش داده می‌شود. یادگیری شبکه عصبی به منظور کمینه کردن تابع هزینه زیر است:

² Damping Factor using Norm of Jacobian: DFNJ

¹ Damping Factor using Norm of Error: DFNE



شکل (۳): خطای یادگیری برحسب زمان اجرا برای تابع غیرخطی
(۲۱)

۵- روش پیشنهادی برای تعیین ضریب دمپینگ^۱

با مقایسه رابطه (۱۱) با رابطه (۱۴) می‌توان چنین نتیجه گرفت که در روش لونبرگ-مارکوارت از تقریب زیر برای ساده‌سازی استفاده شده است:

$$\begin{cases} H \equiv JJ^T + E \frac{\partial^2 E}{\partial w^2} \Rightarrow \lambda I \approx E \frac{\partial^2 E}{\partial w^2} \\ H_{LM} = JJ^T + \lambda I \end{cases} \quad (24)$$

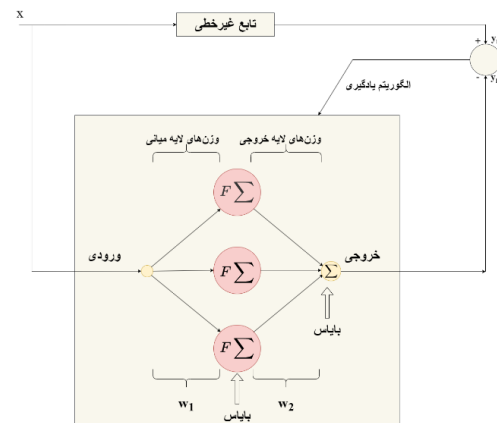
از رابطه (۲۴) می‌توان چنین نتیجه گرفت که در روش مارکوارت، ماتریس قطری λI باید جایگزین ماتریس $E \frac{\partial^2 E}{\partial w^2}$ شود. از آنجایی که ماتریس قطری λI نمی‌تواند با ماتریس $E \frac{\partial^2 E}{\partial w^2}$ مساوی باشد، در روش پیشنهادی فرض می‌شود که حداقل نرم آن‌ها برابر هم باشد:

$$\|\lambda I^*\| = \left\| E \frac{\partial^2 E}{\partial w^2} \right\| \Rightarrow \lambda^* = \left\| E \frac{\partial^2 E}{\partial w^2} \right\| \quad (25)$$

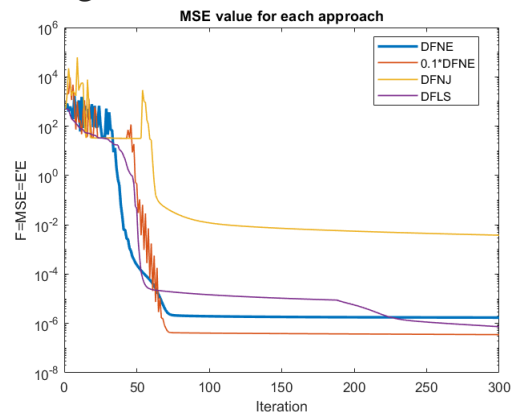
اگر از مقدار λ^* رابطه فوق استفاده شود خطای بین تقریب H_{LM} و ماتریس هسین اصلی کمترین مقدار خود می‌شود و انتظار می‌رود که تقریب H_{LM} به ازای λ^* ، بهترین نتایج را به همراه داشته باشد. البته در روش مارکوارت سعی شده تا به جای رابطه فوق، مقدار ضریب دمپینگ را با انجام عملیات جستجوی محدود به دست آورد ولی لزوماً این ضریب دمپینگ، مقدار بهینه نخواهد بود.

خطا (DFNE) دارای همگرایی سریع‌تر و خطای نهایی کمتری نسبت به سایر روش‌ها است که البته در نتایج آن گاهی نوسان هم دیده می‌شود.

در شکل ۳ مقدار تابع هدف برای هرکدام از روش‌ها برحسب زمان اجرا در هر تکرار نشان داده شده است. می‌توان مشاهده کرد که روش نرم خطا علاوه بر خطای نهایی کمتر به زمان کمتری نسبت به روش مارکوارت نیاز دارد.



شکل (۱): یادگیری شبکه عصبی به منظور یادگیری تابع غیرخطی



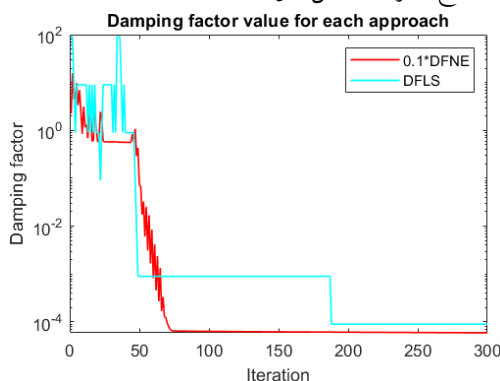
شکل (۲): خطای یادگیری برحسب تکرار برای تابع غیرخطی (۲۱)

در این مقاله سعی شده است تا با ترکیب روش تعیین ضریب دمپینگ مبتنی بر نرم خطا و روش جستجوی خطی مورد استفاده در روش مارکوارت مشکل نوسانی بودن روش مبتنی بر نرم خطا را حل کرده و سرعت همگرایی را هم نسبت به سایر روش‌ها بهبود بخشید.

¹ Damping Factor using the Proposed Method: DFPM

در روش جستجوی خطی مورد استفاده در روش مارکوارت از نرخ تنظیم $\beta=10$ استفاده شده است که λ به صورت ۱۰ برابر در هر جستجو افزایش یا کاهش می‌یابد. در روش پیشنهادی چون بجای λ در روش مارکوارت از $\eta_k \|E\|$ استفاده شده است، و این دو پارامتر (λ در روش مارکوارت و η در روش پیشنهادی) دارای بازه متفاوتی می‌باشند، لذا بهتر است از نرخ تنظیم جدیدی (متناظر با این روش) استفاده شود.

در شکل ۴ مقدار ضریب دمپینگ در هر تکرار برای دو روش نرم خطا و روش مارکوارت نمایش داده شده است. مشاهده می‌شود که مقدار $\lambda_E = 0.1 \|E\|$ تقریباً به صورت پیوسته (با کمی نوسان) تغییر می‌کند ولی در روش مارکوارت مقدار λ (از روش جستجوی خطی محدود) فقط چهار مقدار گسسته دارد و تغییر آن ناگهانی است لذا نمی‌تواند بهترین λ ممکن را محاسبه کند. علت این تغییرات ناگهانی انتخاب مقدار بزرگ $\beta=10$ است که باید بتواند کل بازه $10^{-5} < \lambda < 10^2$ را جستجو کند. ولی در روش پیشنهادی این مقاله به جای λ از مقدار $\eta \|E\|$ استفاده شده است که تغییرات $\|E\|$ تا حدی از تغییرات خیلی زیاد در بازه η جلوگیری می‌کند و سبب می‌شود که ناحیه مورد جستجو برای η خیلی کمتر از ناحیه جستجوی λ شود. همین مسأله سبب شده است که در روش پیشنهادی جستجو η دقیق‌تر انجام شود و انتظار می‌رود نتایج بهتری حاصل شود.



شکل (۴): مقدار ضریب دمپینگ در هر تکرار در تعیین ضریب دمپینگ به روش نرم خطا و مارکوارت
با توجه به شکل ۴ در روش مارکوارت بازه تغییرات ضریب دمپینگ به صورت زیر است:

از طرف دیگر در روش تعیین ضریب دمپینگ مبتنی بر نرم خطا، در ماتریس H_E می‌توان چنین نتیجه گرفت که در این روش از تقریب زیر استفاده شده است:

$$\begin{cases} H \equiv JJ^T + E \left[\frac{\partial^2 E}{\partial w^2} \right] \Rightarrow \left\{ \left\| \frac{\partial^2 E}{\partial w^2} \right\| \leq 0.1 \right. \\ \left. H_E = JJ^T + 0.1 \|E\| \right\} \quad (26) \\ \lambda_E = 0.1 \|E\| \end{cases}$$

که در این روش سعی شده است تا ضریب دمپینگ متناظر با خطا تغییر کند و هیچ تنظیم دیگری ندارد و اجرای آن بسیار ساده است و همچنین در شبیه‌سازی‌های شکل ۲، مشاهده می‌شود این روش نسبت به روش مارکوارت (که برای تعیین ضریب دمپینگ از جستجوی خطی استفاده می‌کند) دارای سرعت همگرایی بیشتری است و این نتایج بهتر به دلیل همخوانی بهتر مقدار ضریب دمپینگ به دست آمده از این روش (۲۶) با ضریب دمپینگ بهینه به دست آمده در رابطه (۲۵) است. البته در نتایج شکل ۲ و ۳ مشاهده می‌شود که نوسان‌هایی وجود دارد. دلیل هم آن است که در این تقریب (۲۶)، $\frac{\partial^2 E}{\partial w^2}$ برابر با مقدار ثابتی فرض شده است و در طول اجرای روش از ضریب ثابت ۰.۱ بجای آن استفاده می‌شود.

در این مقاله با در نظر گرفتن موارد فوق و این که ضریب دمپینگ بهینه باید تا حد امکان مقدار رابطه (۲۵) را نمایندگی کند، در روش پیشنهادی ابتدا پارامتر η_K به صورت زیر تعریف شده است:

$$\eta_K \left[\left\| \frac{\partial^2 E}{\partial w^2} \right\| \right] \Rightarrow E \left[\frac{\partial^2 E}{\partial w^2} \right] \equiv \eta_K \|E\| I \quad (27)$$

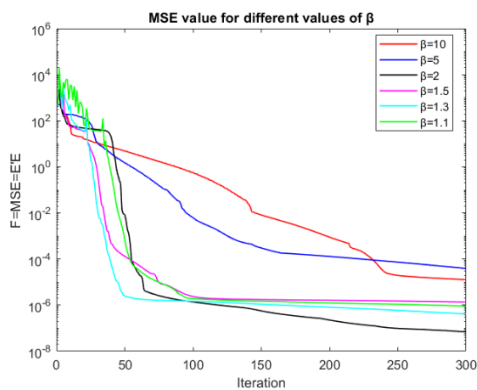
از آنجایی که محاسبه $\frac{\partial^2 E}{\partial w^2}$ نیاز به مشتق مرتبه دوم دارد و محاسبات آن بسیار پیچیده است در روش پیشنهادی، به جای محاسبه مقدار η_K از رابطه (۲۷)، پیشنهاد شده است که مقدار η_K را در هر گام توسط یک مسأله جستجوی خطی به دست آید و روش پیشنهادی به صورت زیر ارائه می‌شود:

$$H_{new} = JJ^T + \eta_K \|E\| I \quad (28)$$

که ماتریس H_{new} تخمین زده شده به اندازه کافی به ماتریس هسین واقعی نزدیک است و از طرفی نیاز به انجام محاسبات پیچیده $\frac{\partial^2 E}{\partial w^2}$ برطرف شده است.

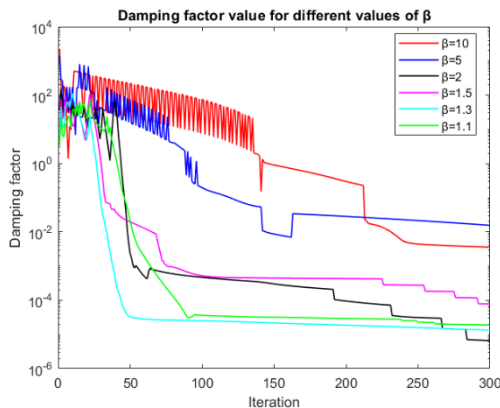
در شکل ۷ نتایج حاصل از تعیین ضریب دمپینگ به روش جستجوی خطی مارکوارت، روش مبتنی بر نرم خطا، روش مبتنی بر نرم ژاکوبین و روش پیشنهادی ($\beta=2$)، رسم شده است. همان‌طور که مشاهده می‌شود در روش پیشنهادی دیگر نوسانات روش نرم خطا وجود ندارد و همچنین سرعت همگرایی افزایش یافته و خطای نهایی کمتر شده است که نشان‌دهنده عملکرد خوب روش پیشنهادی نسبت به سایر روش‌ها است.

در شکل ۸ مقدار ضریب دمپینگ به ازای روش‌های متفاوت نمایش داده شده است و نشان می‌دهد که ضریب دمپینگ به دست آمده در روش پیشنهادی به خوبی توانسته است کل بازه موردنظر را در حین مسأله بهینه‌سازی پوشش دهد و همین مسأله سبب نتایج مناسب در شکل ۷ شده است.



شکل (۵): نتایج به دست آمده به ازای مقادیر متفاوت نرخ تنظیم

ضریب دمپینگ در روش پیشنهادی برای تابع غیرخطی (۲۱)



شکل (۶): مقدار ضریب دمپینگ در هر تکرار به ازای مقادیر متفاوت

نرخ تنظیم ضریب دمپینگ برای تابع غیرخطی (۲۱)

$$\left. \begin{aligned} \lambda_{200} &= 0.9 \times 10^{-5} \\ \lambda_0 &= 0.9 \times 10^2 \end{aligned} \right\} \Rightarrow \left| \frac{\lambda_0}{\lambda_{200}} \right| = 10^7 \quad (29)$$

در روش پیشنهادی داریم:

$$\left. \begin{aligned} \lambda_{200} &= \eta_{200} \|E\|_{200} = 0.9 \times 10^{-5} \\ \lambda_0 &= \eta_0 \|E\|_0 = 0.9 \times 10^2 \end{aligned} \right\} \Rightarrow \frac{\eta_{200} \lambda_0}{\eta_0 \lambda_{200}} = \frac{\lambda_{200}}{\lambda_0} \times \frac{\|E\|_0}{\|E\|_{200}} \quad (30)$$

حال اگر بازه تغییرات نرم خطا را مشابه شکل ۳ و ۴ در نظر بگیریم:

$$\left. \begin{aligned} \|E\|_{200} &\square 0.6 \times 10^{-5} \\ \|E\|_0 &\square 2 \times 10^0 \end{aligned} \right\} \Rightarrow \frac{\|E\|_0}{\|E\|_{200}} = 3.3 \times 10^5 \quad (31)$$

بنابراین از رابطه (۳۰) می‌توان نتیجه گرفت:

$$\left| \frac{\eta_0}{\eta_{200}} \right| = \frac{\lambda_0}{\lambda_{200}} \square \frac{\|E\|_{200}}{\|E\|_0} = 30 \quad (32)$$

لذا در روش پیشنهادی بازه تغییرات η ($\frac{\eta_{\max}}{\eta_{\min}} \square 30$) بسیار کم‌تر

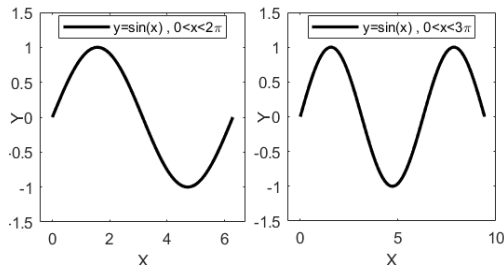
از بازه تغییرات λ در روش مارکوارت ($\frac{\lambda_{\max}}{\lambda_{\min}} \square 10^7$) است و

در نتیجه دیگر لازم نیست در روش پیشنهادی از نرخ تنظیم بزرگی (مانند $\beta=10$ در روش جستجوی مارکوارت) استفاده شود. در نتیجه با انتخاب β کوچک‌تر، ضمن این که کل بازه η مورد جستجو قرار می‌گیرد، اندازه پرش‌های η بسیار کمتر شده و در نتیجه جستجو دقیق‌تر انجام می‌شود.

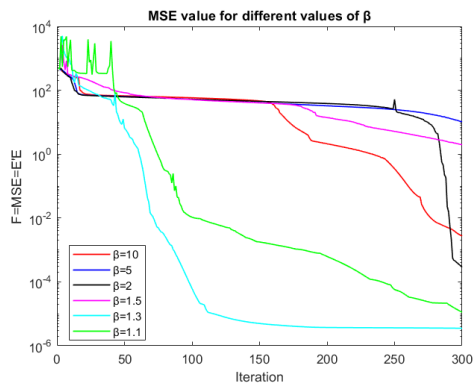
در شکل ۵، نتایج روش پیشنهادی به ازای مقادیر متفاوت نرخ تنظیم ضریب دمپینگ (β) آورده شده است. همچنین در شکل ۶ مقدار ضریب دمپینگ به ازای مقادیر مختلف نرخ تنظیم ضریب دمپینگ (β) رسم شده است.

همان‌طور که در شکل ۶ مشاهده می‌شود بهترین نتایج حاصل به ازای مقدار $\beta=2$ ، روش پیشنهادی کمترین خطای حالت دائم را نشان می‌دهد. با افزایش مقدار β بیشتر از ۲ همگرایی خطای یادگیری کمتر می‌شود که به دلیل عدم دقت در جستجوی خطی به ازای β های بزرگ است که این مسأله در شکل ۶ هم به صورت نوسانات زیاد در η به دست آمده به ازای مقادیر بزرگ $\beta=5$ و $\beta=10$ دیده می‌شود و این نشان می‌دهد که مسأله جستجوی خطی به ازای مقادیر بزرگ نرخ تنظیم $\beta > 2$ به درستی انجام نمی‌شود.

با پیچیده‌تر شدن مسئله مقدار $\frac{\partial^2 E}{\partial w^2}$ هم غیرخطی‌تر شده و در نتیجه پیدا کردن η_K به نحوی که مطابق رابطه (۲۷) بتواند معادل $\left\| \frac{\partial^2 E}{\partial w^2} \right\|$ باشد سخت می‌شود. لذا برای این مسئله انتظار می‌رود که نرخ جستجوی β کوچک‌تر باشد تا جستجو دقیق‌تر انجام شود. در شکل ۱۰ مشاهده می‌شود که نتایج بهینه‌سازی برای $\beta < 1.3$ بهتر از نتایج $\beta > 1.5$ است. علت این مسئله می‌تواند در شکل ۱۱ دیده شود. مشاهده می‌شود که ضریب دمپینگ به ازای نرخ جستجوی بالا ($\beta > 1.5$) بیشتر حالت نوسانی و جهش‌های بالا دارد و نمی‌تواند به درستی به دست آید. برای $\beta < 1.3$ مقدار به دست آمده برای ضریب دمپینگ لامبدا حالت نوسانی ندارد و به همین دلیل در شکل ۱۰ مشاهده می‌شود که به ازای $\beta = 1.1$ و $\beta = 1.3$ نتایج بهینه‌سازی نسبت به $\beta > 1.5$ بهتر است.

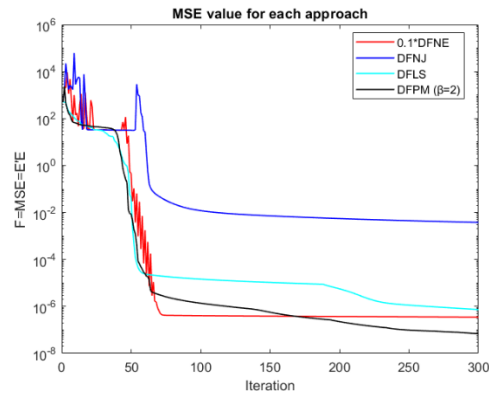


شکل (۹): توابع مورد بررسی معرفی شده در روابط (۲۱) و (۳۳)



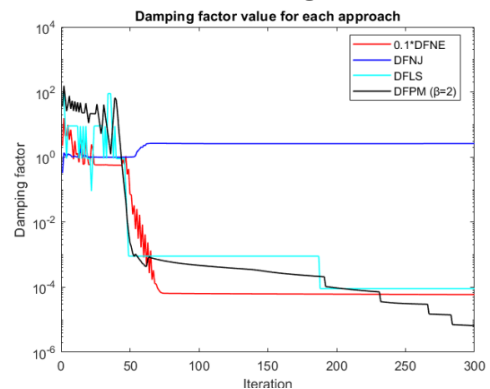
شکل (۱۰): نتایج به دست آمده به ازای مقادیر مختلف نرخ تنظیم

ضریب دمپینگ برای تابع غیرخطی (۳۳)



شکل (۷): مقایسه نتیجه روش پیشنهادی با روش‌های پیشین برای

تابع غیرخطی (۲۱)



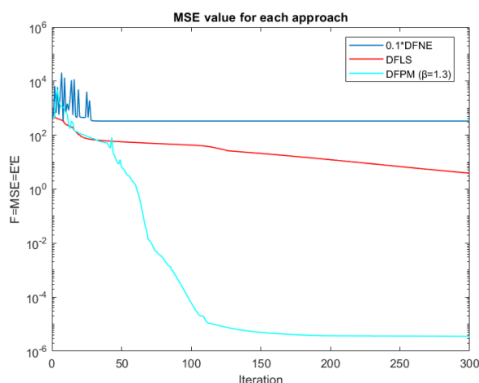
شکل (۸): مقایسه مقدار ضریب دمپینگ در روش پیشنهادی و

روش‌های پیشین برای تابع غیرخطی (۲۱)

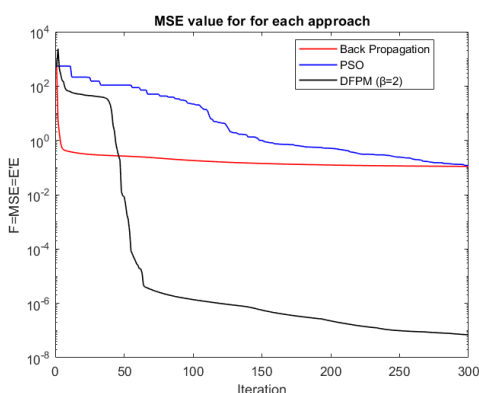
در شکل ۹ تابع $y = \sin(x)$ برای $0 \leq x \leq 2\pi$ که یک تابع غیرخطی با پیچیدگی متوسط است، رسم شده است و در کنار آن تابع $y = \sin(x)$ برای $0 \leq x \leq 3\pi$ نیز رسم شده است. مشاهده می‌شود با افزایش دامنه x ، پیچیدگی این تابع غیرخطی نیز افزایش یافته است. نتایج قبلی برای شناسایی تابع $y = \sin(x)$ برای $0 \leq x \leq 2\pi$ انجام شده است. در ادامه برای بررسی اثر پیچیدگی مسئله بر مقدار بهینه β (در روش پیشنهادی) و همچنین بررسی عملکرد روش پیشنهادی در مقایسه با سایر روش‌ها، شبیه‌سازی‌ها برای شناسایی تابع (۳۳) انجام شده است.

$$y = \sin(x), \quad x \in [0, 3\pi] \quad (۳۳)$$

در شکل ۱۰، نتایج روش پیشنهادی به ازای مقادیر مختلف نرخ تنظیم برای تابع (۳۳) نمایش داده شده است.

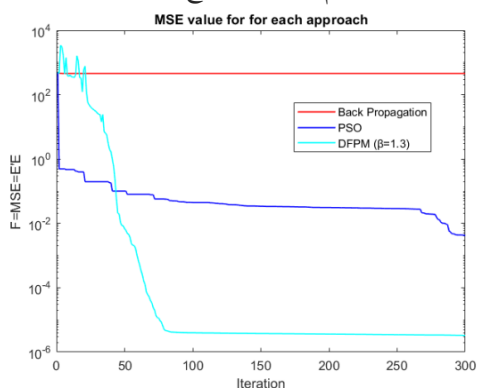


شکل (۱۲): مقایسه روش‌های معرفی شده برای تابع غیرخطی (۳۳)



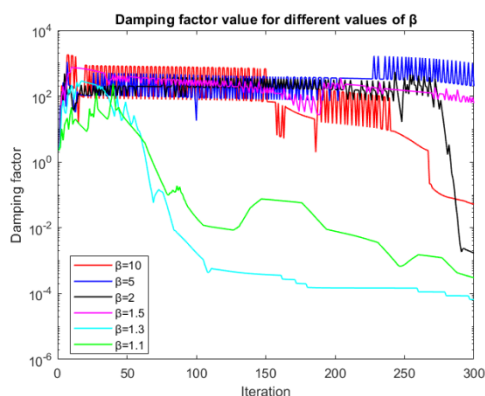
شکل (۱۳): مقایسه روش پیشنهادی با روش پس انتشار خطا و

روش ازدحام ذرات برای تابع غیرخطی (۲۱)



شکل (۱۴): مقایسه روش پیشنهادی با روش پس انتشار خطا و

روش ازدحام ذرات برای تابع غیرخطی (۳۳)



شکل (۱۱): مقدار ضریب دمپینگ در هر تکرار به ازای مقادیر

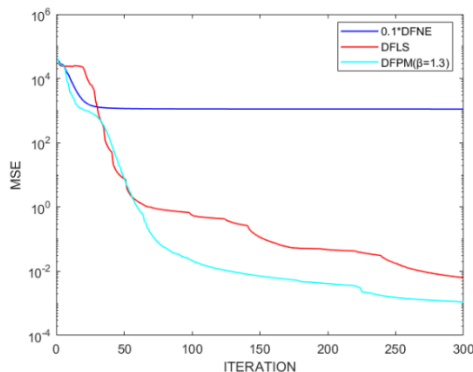
مختلف نرخ تنظیم ضریب دمپینگ برای تابع غیرخطی (۳۳)

شکل ۱۲ نتیجه شبیه‌سازی روش‌های مختلف برای رابطه (۳۳) را نشان می‌دهد. همان‌طور که مشاهده شد روش پیشنهادی برای تابع (۲۱) عملکرد بهتری نسبت به سایر روش‌ها داشت، با پیچیده‌تر شدن تابع مورد بررسی روش نرم خطا عملکرد مناسبی ندارد و همان‌طور که در شکل ۱۲ مشاهده می‌شود کارایی خود را در این حالت از دست می‌دهد. از طرفی روش مارکوارت به‌کندی به سمت جواب بهینه حرکت می‌کند، اما روش پیشنهادی با سرعت مناسبی به سمت جواب بهینه همگرا شده است و عملکرد مناسبی را نشان می‌دهد.

در انتها نتایج ناشی از روش پیشنهادی و روش‌های رایج پس‌انتشارخطا و ازدحام ذرات برای یادگیری شبکه عصبی به‌منظور مدل‌سازی تابع (۲۱) و (۳۳) به ترتیب در شکل‌های ۱۳ و ۱۴ نمایش داده شده است. که عملکرد بسیار خوب روش پیشنهادی از نظر سرعت همگرایی و خطای نهایی را در مقایسه با روش‌های رایج نشان می‌دهد.

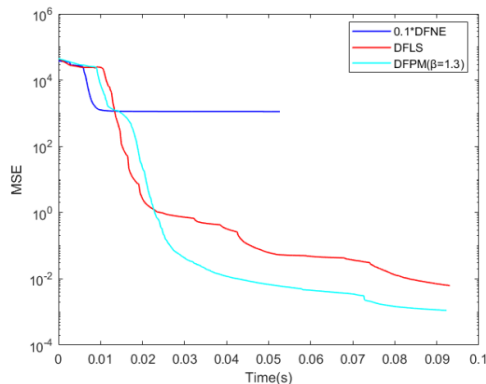
در انتها لازم است تاکید شود که β بهینه با توجه به پیچیدگی مسئله مقداری بین ۱٫۱ تا ۲ پیشنهاد می‌شود که طراح باید با توجه به پیچیدگی مسئله موردنظر β را از این بازه انتخاب کند.

تابع در شکل ۱۶ و ۱۷ قابل مشاهده است. که می‌توان عملکرد مناسب روش پیشنهادی (DFPM) هم برحسب تکرار و هم برحسب زمان اجرا در مقایسه با دو روش نرم خطا (DFNE) و مارکوارت (DFLS) را مشاهده کرد. در شکل ۱۸ می‌توان مشاهده کرد که روش پیشنهادی بازه بزرگ‌تر و پیوسته‌ای از مقادیر ضریب دمپینگ را نسبت به روش نرم خطا پوشش می‌دهد و همچنین دارای نوسانات و پرش‌های کمتری در ضریب دمپینگ نسبت به روش مارکوارت است و بازه تغییرات را به‌طور پیوسته کاملاً پوشش می‌دهد که باعث جستجوی بهتر فضا و به دست آمدن نتایج بهتر با استفاده از روش پیشنهادی شده است.



شکل (۱۶): مقایسه روش پیشنهادی با روش نرم خطا و مارکوارت

برای تابع غیرخطی (۳۴) برحسب تعداد تکرار



شکل (۱۷): مقایسه روش پیشنهادی با روش نرم خطا و مارکوارت

برای تابع غیرخطی (۳۴) برحسب زمان اجرا

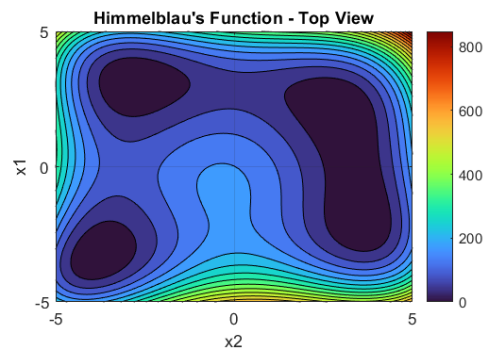
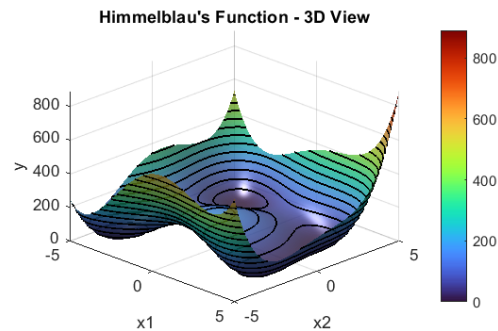
۶- کاربرد روش پیشنهادی برای توابع پیچیده‌تر

در این بخش به منظور ارزیابی و بررسی بیشتر کارایی روش پیشنهادی در مواجهه با مسائل پیچیده‌تر، شبیه‌سازی‌ها روی سه مسأله دیگر انجام شده و در ادامه نتایج مورد بررسی قرار می‌گیرد.

۶-۱ مدل‌سازی تابع غیرخطی هیمل بلاو^۱

برای شبیه‌سازی اول، تابع هیمل بلاو انتخاب و شبیه‌سازی شده است. همان‌طور که در شکل ۱۵ مشاهده می‌شود این تابع دارای ۴ نقطه زینی است [۱۶]:

$$y = (x_1^2 + x_2 - 11)^2 + (x_1 + x_2^2 + -7)^2 \quad (34)$$

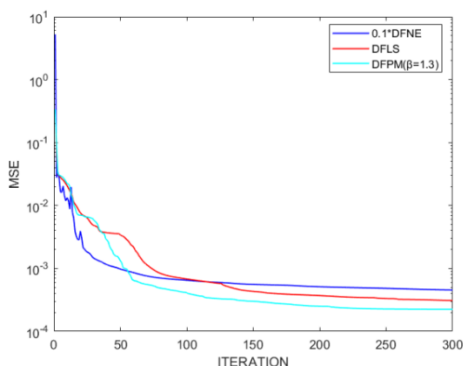


شکل (۱۵): نمایش سه بعدی تابع رابطه (۳۴)

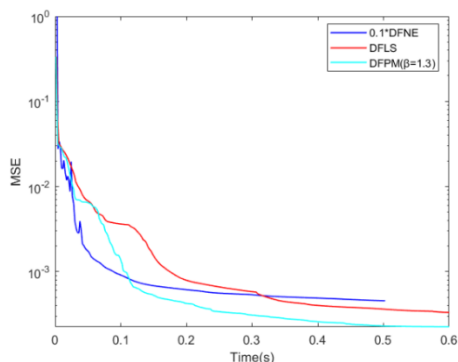
برای شناسایی این تابع مشابه بخش‌های قبل، از یک شبکه عصبی با یک لایه میانی شامل ۱۰ نورون استفاده شده است و لایه ورودی شامل ۲ نورون است که در مجموع شبکه عصبی مورد استفاده شامل ۴۱ وزن است. روش پیشنهادی و سایر روش‌های مورد بررسی برای یادگیری این شبکه عصبی استفاده شده است که نتایج شبیه‌سازی روش پیشنهادی در مقایسه با سایر روش‌ها برای این

¹ HimmelBlau's Function

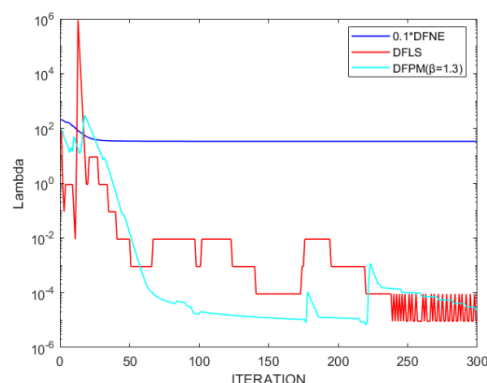
روش پیشنهادی دارای نوسانات و پرش‌های کمتری در ضریب دمپینگ است و بازه تغییرات را به‌طور پیوسته پوشش می‌دهد در حالی که در روش مارکوارت (DFLS) ضریب دمپینگ دارای پرش‌های بزرگی بوده است. این مسئله، منجر به عملکرد بهتر روش پیشنهادی در تخمین ضریب دمپینگ شده و با ایجاد تعادل بین پویایی و پیوستگی، عملکرد بهتری را ارائه می‌دهد.



شکل (۱۹): مقایسه روش پیشنهادی با روش نرم خطا و مارکوارت برای مسئله رگرسیون برحسب تعداد تکرار



شکل (۲۰): مقایسه روش پیشنهادی با روش نرم خطا و مارکوارت برای مسئله رگرسیون برحسب زمان اجرا



شکل (۱۸): مقایسه مقدار ضریب دمپینگ روش پیشنهادی با روش نرم خطا و مارکوارت برای تابع غیرخطی (۳۴)

۲-۶ مسئله رگرسیون بر روی مجموعه داده‌های واقعی

برای شبیه‌سازی دوم از مجموعه داده بار برودتی ساختمان موجود در مخازن یادگیری ماشینی UCI^۱ استفاده شده است. ورودی مسئله شامل هشت ویژگی (۱-وسعت نسبی ساختمان نسبت به ساختمان‌های دیگر، ۲-زیربنا، ۳-مساحت دیوارها، ۴-مساحت سقف، ۵-ارتفاع ساختمان، ۶-جهت ساختمان، ۷-وسعت سطح نورگیر (شیشه‌ای) و ۸-نسبت سطوح شیشه‌ای به کل سطوح) و یک داده خروجی (بار برودتی ساختمان) است که کل داده‌های موجود ۷۶۸ داده است [۱۷].

به‌منظور یادگیری از یک شبکه عصبی با یک‌لایه میانی شامل ۲۰ نورون استفاده شده است و لایه ورودی شامل ۸ نورون است که در مجموع شبکه عصبی مورد استفاده شامل ۲۰۱ وزن است. از ۵۰۰ داده اول برای یادگیری این شبکه عصبی استفاده شده است که در شکل ۱۹ و ۲۰ خطای یادگیری به ازای روش‌های مختلف قابل مشاهده است. همان‌طور که در شکل ۱۹ و ۲۰ مشاهده می‌شود روش پیشنهادی (DFPM) دارای نتایج بهتری در آموزش شبکه عصبی برای این مسئله رگرسیون است.

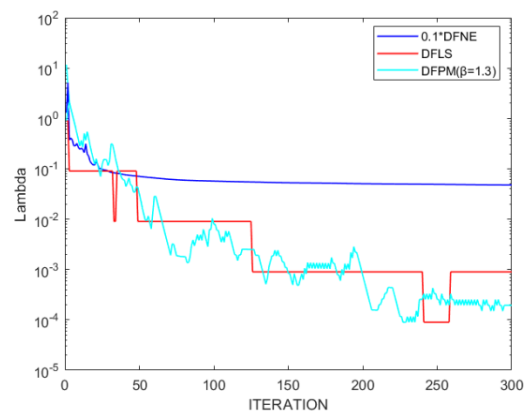
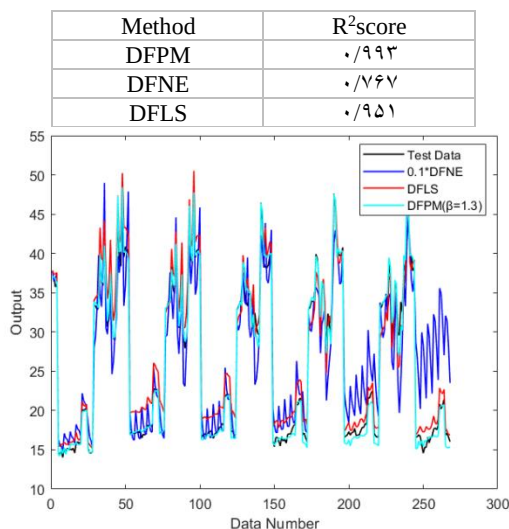
در شکل ۲۱ مشاهده می‌شود که روش مارکوارت و روش پیشنهادی، در مقایسه با روش نرم خطا، تغییرات ضریب دمپینگ را با دامنه وسیع‌تری پوشش می‌دهند، که نشان‌دهنده پویایی بیشتر این دو روش است و انتظار می‌رود نتایج بهتری حاصل شود. در مقایسه بین روش مارکوارت و روش پیشنهادی، مشاهده شد که

¹ UCI Machine Learning Repository

و ۱ (وجود احتمال سرطان) است که این مجموعه شامل ۵۷۰ داده است [۱۸].

به منظور یادگیری از یک شبکه عصبی با یک لایه میانی شامل ۱۰ نورون استفاده شده است و لایه ورودی شامل ۵ نورون است که در مجموع این شبکه عصبی شامل ۷۱ وزن است. همچنین از ۴۰۰ داده اول برای یادگیری این شبکه عصبی در مرحله آموزش

جدول (۱): مقایسه شاخص R^2 برای روش‌های مورد بررسی



شکل (۲۱): مقایسه مقدار ضریب دمینگ روش پیشنهادی با روش

نرم خطا و مارکوارت برای مسئله رگرسیون

پیشنهادی با ایجاد تعادل بین پویایی و پیوستگی، عملکرد بهتری را ارائه می‌دهد.

در مرحله ارزیابی تعداد ۲۶۸ داده باقیمانده برای تست مورد استفاده قرار گرفته که نتایج به دست آمده در شکل ۲۲ قابل مشاهده است. برای نمایش دقیق‌تر و امکان بررسی بیشتر داده‌ها، در شکل ۲۳ فقط نتایج به دست آمده برای داده‌های ۱۳۵ تا ۱۴۸ رسم شده است که به وضوح در این شکل دقت بیشتر روش پیشنهادی مشاهده می‌شود که به خوبی داده‌های تست را دنبال کرده و تخمین دقیق‌تری را نسبت به سایر روش‌ها ارائه کرده است.

یکی از معیارهای مورد استفاده در ارزیابی مسائل رگرسیون، معیار R^2 score است که در رابطه ۳۵ تعریف شده است. هرچه این معیار به ۱ نزدیک‌تر باشد نشان‌دهنده کارایی بیشتر روش است. برای هر روش از رابطه ۳۵ محاسبه شده و در جدول ۱ قابل مشاهده است.

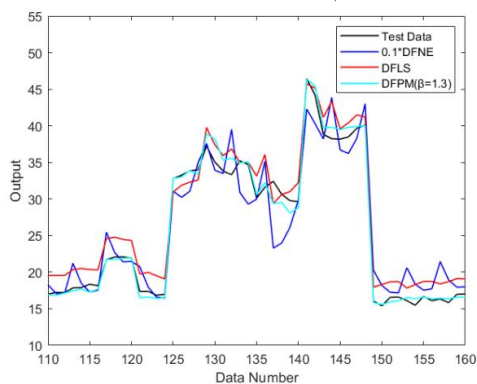
$$R^2 = 1 - \frac{\sum_{i=1}^N \text{Real data} - \text{Prediction}}{\sum_{i=1}^N \text{Real data} - \text{Mean of real data}} \quad (35)$$

که نتایج جدول ۱ عملکرد مناسب‌تر روش پیشنهادی در مقایسه با سایر روش‌ها برای مسائل رگرسیون را نشان می‌دهد.

۳-۶ مسئله طبقه‌بندی بر روی مجموعه داده‌های واقعی

برای شبیه‌سازی سوم، یک مسئله طبقه‌بندی انتخاب شده است که شامل مجموعه داده‌های سرطان سینه جمع‌آوری شده توسط دکتر ویلیام از بیماران دانشگاه پزشکی ویسکانسین است. در این مجموعه، داده‌های ورودی از تصاویر اشعه ایکس به دست آمده‌اند و داده خروجی می‌تواند دو مقدار ۰ (عدم وجود احتمال سرطان)

شکل (۲۲): مقایسه تخمین بر روی داده‌های تست با روش پیشنهادی، روش نرم خطا و مارکوارت برای مسئله رگرسیون



شکل (۲۳): نمایش داده‌های ۱۱۰ تا ۱۶۰ از شکل ۲۲

استفاده شده است که در شکل ۲۴ و ۲۵ میانگین خطای مرحله یادگیری در ۵۰ بار اجرای شبیه‌سازی به ازای روش‌های مختلف قابل مشاهده است. همان‌طور که در دو شکل ۲۴ و ۲۵ مشاهده می‌شود روش پیشنهادی دارای عملکرد مناسبی است و زودتر خطای یادگیری را کاهش داده است.

مارکوارت و روش پیشنهادی مشاهده می‌شود که در روش پیشنهادی ضریب دمپینگ به‌دست‌آمده دارای دامنه نوسانات و پرش‌های کمتری است و کل بازه تغییرات را به‌طور پیوسته پوشش داده است و به همین دلیل این روش دارای نتایج بهتری است. در مرحله ارزیابی، تعداد ۱۷۰ داده ورودی و خروجی مورد استفاده قرار گرفته که نتایج به‌دست‌آمده در شکل ۲۷ برای هر سه روش قابل مشاهده است. در ستون اول سطر اول، تعداد تشخیص درست عدم سرطان و ستون دوم سطر اول تعداد تشخیص اشتباه عدم سرطان است. ستون اول سطر دوم تشخیص اشتباه وجود سرطان است و در ستون دوم سطر دوم، تعداد تشخیص درست سرطان است که برای ارزیابی از معیار پوشش و دقت که به ترتیب در روابط (۳۶) و (۳۷) آمده، استفاده شده است.

$$Recall = \frac{True\ Positive}{True\ Positive + False\ negative} \quad (36)$$

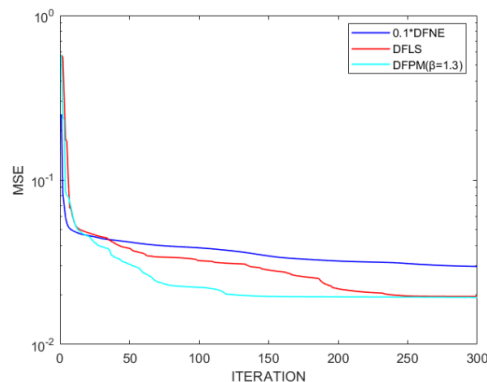
$$Accuracy = \frac{Number\ of\ correct\ prediction}{Total\ number\ of\ predictions} \quad (37)$$

در جدول ۲ نتایج به‌دست‌آمده از شبیه‌سازی ذکر شده است. در سطر دوم این جدول خطای یادگیری روش‌ها، در سطر سوم مقدار معیار پوشش و در سطر چهارم میزان دقت هر روش نشان داده شده است که روش پیشنهادی بهترین نتایج را داشته است. مشاهده می‌شود که روش پیشنهادی دارای نتایج بهتری است و دارای دقت و عملکرد مناسبی در مسائل طبقه‌بندی است.

Confusion Matrix			
True Class	Class 0	Class 1	
	DFPM:37 DFSL:37 0.1DFNE:37	DFPM:2 DFSL:2 0.1DFNE:2	
Class 1	DFPM:12 DFSL:13 0.1DFNE:17	DFPM:119 DFSL:118 0.1DFNE:114	
		Predicted Class	
		Class 0	Class 1

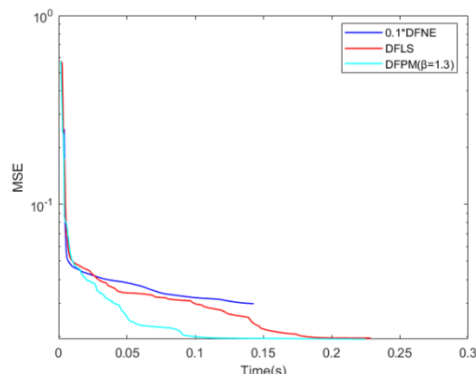
شکل (۲۷): نمایش ماتریس درهم‌ریختگی برای مسئله طبقه‌بندی

در شکل ۲۶ مقدار ضریب دمپینگ برای هر سه روش مورد بررسی، نمایش داده شده است. همان‌طور که مشاهده می‌شود، در



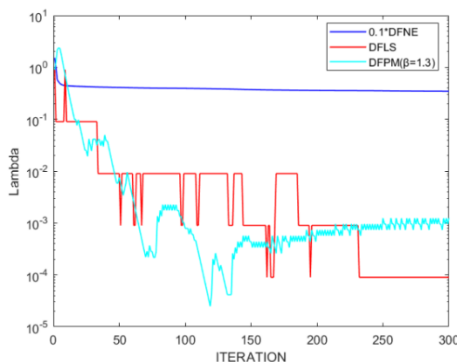
شکل (۲۴): مقایسه روش پیشنهادی با روش نرم خطا و مارکوارت

برای مسئله طبقه‌بندی برحسب تعداد تکرار



شکل (۲۵): مقایسه روش پیشنهادی با روش نرم خطا و مارکوارت

برای مسئله طبقه‌بندی برحسب زمان اجرا



شکل (۲۶): مقایسه مقدار ضریب دمپینگ روش پیشنهادی با روش

نرم خطا و مارکوارت برای مسئله طبقه‌بندی

روش مارکوارت و روش پیشنهادی بازه تغییرات ضریب دمپینگ به نسبت روش نرم خطا بیشتر است. که انتظار می‌رود نتایج این دو روش بهتر از روش نرم خطا شود. همچنین در بین دو روش

یافته با شرایط جدید مشکل نوسانی بودن را حل کرده و سرعت همگرایی را هم بهبود بخشید. در ادامه با بیان دلیل، بهترین ضریب برای نرخ تنظیم ضریب دمپینگ (β) در روش پیشنهادی تعیین شد. روش پیشنهادی و اصلاحات اعمال شده در آن با تئوری و فرضیات موجود در روش لونبرگ-مارکوارت هماهنگی بیشتری نسبت به سایر روش‌ها دارد و در نتیجه نتایج بهتری به دست می‌آید. روش پیشنهادی نتایج قابل توجهی را در مقایسه با روش‌های قبلی ارائه می‌دهد و علاوه بر این که به خطای نهایی کمتری می‌رسد نوسانات روش نرم خطا را به طور کلی برطرف و همگرایی در هر تکرار را هم تضمین می‌کند. در ادامه روش‌های مذکور برای آموزش شبکه عصبی به منظور شناسایی یک تابع غیرخطی پیچیده، مسئله رگرسیون و مسئله طبقه‌بندی با داده‌های واقعی شبیه‌سازی شدند. در نتایج به دست آمده مشاهده شد که روش پیشنهادی در مقایسه با دو روش دیگر عملکرد بهتری داشته و روش پیشنهادی در این مقاله می‌تواند به عنوان یک ابزار مفید برای حل مسائل بهینه‌سازی در شبکه‌های عصبی مورد استفاده قرار گیرد.

جدول (۲): مقایسه معیارهای عملکرد روش‌های مورد بررسی در مسئله طبقه‌بندی

Method	DFPM	DFNE	DFLS
MSE	۰/۰۱۹	۰/۰۳	۰/۰۱۹۴
Recall	۹۰/۸٪	۸۷/۰۲٪	۹۰/۵٪
Accuracy	۹۱/۷٪	۸۸/۸۲٪	۹۱/۱٪

۷- نتیجه‌گیری

در این مقاله، روش‌های مختلف تعیین ضریب دمپینگ در الگوریتم لونبرگ-مارکوارت، شامل روش‌های مارکوارت، روش‌های مبتنی بر نرم خطای جستجو و مبتنی بر نرم ماتریس ژاکوبین، با یکدیگر مقایسه شده‌اند. نتایج شبیه‌سازی‌ها نشان می‌دهد که تعیین ضریب دمپینگ به روش مارکوارت دارای کارایی خوبی است. در سایر روش‌ها، روش مبتنی بر نرم خطا دارای نتایج مناسب‌تری است که حتی نسبت به روش مارکوارت در مواردی نتایج بهتر می‌شود. البته در روش ضریب دمپینگ مبتنی بر نرم خطا، گاهی نوسان وجود دارد. در این مقاله سعی شده است تا با ترکیب روش تعیین ضریب دمپینگ مبتنی بر نرم خطا و روش جستجوی خطی مورد تطبیق

References

- [1] O. I. Abiodun, A. Jantan, A. E. Omolara, K. V. Dada, N. A. Mohamed, and H. Arshad, "State-of-the-art in artificial neural network applications: A survey," *Heliyon*, vol. 4, no. 11, p. e00938, Nov. 2018, doi: 10.1016/j.heliyon.2018.e00938.
- [2] M. Sh. Daoud, M. Shehab, H. M. Al-Mimi, L. Abualigah, R. A. Zitar, and M. K. Y. Shambour, "Gradient-Based Optimizer (GBO): A Review, Theory, Variants, and Applications," *Archives of Computational Methods in Engineering*, vol. 30, no. 4, pp. 2431–2449, May 2023, doi: 10.1007/s11831-022-09872-y.
- [3] X. Wei and H. Huang, "A survey on several new popular swarm intelligence optimization algorithms," 2023, doi: 10.21203/rs.3.rs-2450545/v1.
- [4] S. Ruder, "An overview of gradient descent optimization algorithms," Sep. 2016.
- [5] B. T. Polyak, "Newton's method and its use in optimization," *Eur J Oper Res*, vol. 181, no. 3, pp. 1086–1096, Sep. 2007, doi: 10.1016/j.ejor.2005.06.076.
- [6] M. S. Osigbeme, C. Osuji, M. O. Onyesolu, and U. P. Onochie, "Comparison of the Artificial Neural Network's Approximations for the Levenberg-Marquardt Algorithm and the Gradient Descent Optimization on Datasets," *Artificial Intelligence Evolution*, pp. 24–38, Mar. 2024, doi: 10.37256/aie.5120243781.
- [7] O. Umar, I. M. Sulaiman, M. Mamat, M. Y. Waziri, and N. Zamri, "On damping parameters of Levenberg-Marquardt algorithm for nonlinear least square problems," *J Phys Conf Ser*, vol. 1734, no. 1, p. 012018, Jan. 2021, doi: 10.1088/1742-6596/1734/1/012018.
- [8] K. Levenberg, "A method for the solution of certain non-linear problems in least squares," *Q Appl Math*, vol. 2, no. 2, pp. 164–168, 1944, doi: 10.1090/qam/10666.
- [9] D. W. Marquardt, "An Algorithm for Least-Squares Estimation of Nonlinear Parameters," *Journal of the Society for Industrial and Applied Mathematics*, vol. 11, no. 2, pp. 431–441, Jun. 1963, doi: 10.1137/0111030.
- [10] M. K. Transtrum and J. P. Sethna, "Improvements to the Levenberg-Marquardt algorithm for nonlinear least-squares minimization," Jan. 2012.

- [11] J. J. Moré, "The Levenberg-Marquardt algorithm: Implementation and theory," 1978, pp. 105–116. doi: 10.1007/BFb0067700.
- [12] Y. Wang, "Gauss–Newton method," WIREs Computational Statistics, vol. 4, no. 4, pp. 415–420, Jul. 2012, doi: 10.1002/wics.1202.
- [13] C. Chen, S. Reiz, C. Yu, H.-J. Bungartz, and G. Biros, "Fast Approximation of the Gauss-Newton Hessian Matrix for the Multilayer Perceptron," Oct. 2019.
- [14] M. R. Osborne, "Nonlinear least squares — the Levenberg algorithm revisited," The Journal of the Australian Mathematical Society. Series B. Applied Mathematics, vol. 19, no. 3, pp. 343–357, Jun. 1976, doi: 10.1017/S033427000000120X.
- [15] A. Suratgar, M. B. Tavakoli, and A. Hoseinabadi, "Modified Levenberg-Marquardt method for neural networks training," in Proceedings - Wec 05: Fourth World Enformatika Conference, 2005. doi: 10.5281/zenodo.1333881.
- [16] Y. H. Irawan and P. T. Lin, "Parametric optimization technique for continuous and combinational problems based on simulated annealing algorithm," Journal of Energy, Mechanical, Material, and Manufacturing Engineering, vol. 8, no. 2, pp. 75–82, 2023, doi: 10.22219/jemmmme.v8i2.29556.
- [17] Tsanas and A. Xifara, "Accurate quantitative estimation of energy performance of residential buildings using statistical machine learning tools," Energy and Buildings, vol. 49, pp. 560–567, 2012, doi: 10.1016/j.enbuild.2012.03.003.
- [18] S. S. Mohammadi, M. Salehirad, M. M. Emamzadeh et al., "Improved equilibrium optimizer for accurate training of feedforward neural networks," Optical Memory and Neural Networks, vol. 33, pp. 133–143, 2024, doi: 10.3103/S1060992X24700048.

پیوست ۱

درروش لونیبرگ-مارکوارت به‌روزرسانی متغیرهای بهینه‌سازی (وزن‌های شبکه عصبی) در جهت عکس‌گردان $(JJ^T + \lambda I)^{-1} \times J^T E_k$ انجام می‌شود. از آنجایی که ماتریس $JJ^T + \lambda I$ یک ماتریس مثبت معین است لذا معکوس آن هم معنی $[JJ^T + \lambda I]^{-1}$ مثبت معین است. از طرفی ماتریس مثبت معین دارای خاصیت زیر است:

$$\forall x \neq 0 \Rightarrow x^T H x > 0. \quad (38)$$

فرض کنید تابع هزینه زیر برای روش لونیبرگ-مارکوارت طبق رابطه (۶) تعریف‌شده باشد:

$$F = \frac{1}{2} E^T E, \quad E = Y_{ref}(X) - Y(X, W) \quad (39)$$

در رابطه فوق Y_{ref} داده‌های خروجی سیستم اصلی هستند که مستقل از w (وزن‌های شبکه عصبی) هستند و فقط Y که خروجی شبکه عصبی است به w ربط دارد.

همچنین اگر فرض شود که تغییرات Δw_k (فرضاً اسکالر است) در هر گام کوچک باشد. در این صورت اگر بر اثر اعمال تغییرات Δw_k ، تابع هزینه F هم دارای تغییرات ΔF شود، داریم:

$$\frac{\Delta F}{\Delta w} \approx \frac{\partial F}{\partial w} \Rightarrow \Delta F \approx \frac{\partial F}{\partial w} \cdot \Delta w \quad (40)$$

همچنین در حالت چند متغیره (W یک بردار است) به‌طور مشابه رابطه زیر برقرار است:

$$\begin{cases} \Delta F = \frac{\partial F}{\partial w} \cdot \Delta w \\ \frac{\partial F}{\partial w} = E^T \cdot \frac{\partial E}{\partial w} \end{cases} \Rightarrow \Delta F = \left[E^T \cdot \frac{\partial E}{\partial w} \right]^T \cdot \Delta w \quad (41)$$

که با توجه به رابطه (۳۹)، $\frac{\partial E}{\partial w}$ به‌صورت زیر تعریف‌شده است.

$$\frac{\partial E}{\partial w} = -\frac{\partial Y}{\partial w} = -J, \quad J \approx \frac{\partial Y}{\partial w} \quad (42)$$

و ΔF به‌صورت زیر است:

$$\Delta F = -E^T J^T \Delta w \quad (43)$$

که در این مقاله Δw به‌صورت زیر تعریف‌شده است:

$$\Delta w = -H^{-1} \nabla F = -H^{-1} E J^T \quad (44)$$

با جایگذاری رابطه (۴۴) در رابطه (۴۳) داریم:

$$\Delta F = -E^T J^T H^{-1} J < 0 \quad (45)$$

حال با توجه به این که H^{-1} ماتریس مثبت معین است و J یک بردار است، $J^T H^{-1} J$ مقداری مثبت است در نتیجه ΔF منفی می‌شود یعنی این که تابع هزینه در اثر تغییرات Δw آمده درروش لونیبرگ-مارکوارت کاهشی است. این کاهش ادامه دارد تا زمانی که ΔF صفر شود. $\Delta F = -E^T J^T H^{-1} J$ یعنی تا زمانی که یا E

صفر شود یا J صفر شود. چون $J = \frac{\partial Y}{\partial w}$ لذا صفر شدن J همان

نقطه و جواب بهینه است. لذا روش حتماً به جواب بهینه $E=0$ یا

$\frac{\partial Y}{\partial w} = 0$ می‌رسد. قبلاً از شرط اینکه Δw به اندازه کافی کوچک

باشد در روابط، استفاده کرده بودیم و به این شرط نشان داده شد

که درروش مارکوارت به ازای Δw ارائه‌شده حتماً ΔF منفی (F)

کاهشی است. درروش جستجوی خطی در هر تکرار λ (η)

درروش پیشنهادی) چندین بار تغییر می‌کند تا مطمئن شویم که

رابطه $F_k > F_{k+1}$ اولاً کاهشی و دوماً دارای بیشترین کاهش است،

لذا روش جستجوی خطی خودش هم (مستقل از اثبات فوق)

تضمین کننده کاهشی بودن تابع هزینه است.

Neural Network Learning Using an Improved Levenberg-Marquardt Algorithm

Reza Yazhari Kermani¹, Mohammad Mollaie Emamzadeh^{2*}, Mojtaba Barkhordari Yazdi³

¹MSc. Student, Department of Electrical Engineering, Shahid Bahonar University of Kerman, Kerman, Iran

²Assistant Professor, Department of Electrical Engineering, Shahid Bahonar University of Kerman, Kerman, Iran

³Associate Professor, Department of Electrical Engineering, Shahid Bahonar University of Kerman, Kerman, Iran

Article Information

Original Research Paper

Received:

2024 December 9

Accepted:

2025 February 26

Keywords:

Damping Factor, Levenberg-Marquardt Algorithm, Newton Algorithm, Neural Network Learning, Non-linear Optimization

Corresponding Author*:

molaie@uk.ac.ir

Abstract

This paper presents a new method to improve the convergence speed and efficiency of the Levenberg-Marquardt algorithm. The Levenberg-Marquardt algorithm is a Newton-based method that is efficient in optimization and determining the weights of neural networks compared to other methods, including the backpropagation method. However, the performance of this method depends significantly on the selection of an appropriate damping factor. Among the various methods for determining the damping factor, the Marquardt line search method and methods based on error norm and based on Jacobian norm are mentioned, which in this paper, by examining the strengths and weaknesses of these methods, a combined method is presented to increase the convergence speed. In the proposed method, the search range of the damping factor and, as a result, the value of the adjustment rate parameter is reduced. By making these corrections, the accuracy of the damping factor search is increased and the convergence speed of the proposed method is increased. In order to evaluate the proposed method, learning a neural network to identify several problems including a nonlinear complex function, a regression and a classification task has been simulated and the results show that the proposed method has good efficiency compared to other methods studied and has been able to reduce the learning error to an acceptable level and has a higher convergence speed.



: 10.22034/ABMIR.2025.22500.1082

E-ISSN: [2821-2037](#) /© 2024. Published by Yazd University This is an open access article under the CC BY 4.0 License (<https://creativecommons.org/licenses/by/4.0/>).

