Proceedings of the
47th Annual Iranian Mathematics Conference,
Kharazmi University, Karaj, Iran, 28-31 August 2016

مجموعه مقالات چهل و هفتمین کنفرانس ریاضی ایران
دانشگاه خوارزمی، کرج، ۱۰-۷ شهریور ۱۳۹۵

یافتن پارامترهای بهین تقریب‌گر موجک توسط شبکه‌های عصبی مصنوعی

اژدر سلیمانپور باکفایت* و مهتا برجیسی

دانشگاه فرهنگیان، تهران
دانشجو معلم، دانشگاه فرهنگیان، پردیس علامه طباطبایی، ارومیه

چکیده

در این مقاله، تقریب توابع توسط موجک‌ها به فرمی جدید ارائه شده است. برخی از پارامترهای بهین موجک در عمل تقریب، اعداد صحیح می‌باشند که در این مقاله این پارامترها، اعداد حقیقی فرض شده‌اند. این ابتکار موجب شده است که عمل تقریب با کمترین تعداد توابع پایه‌ای انجام شده و در نتیجه دقت زیاد شده و زمان اجرا کاهش یابد. برای یافتن پارامترهای بهین در حالت مذکور از شبکه عصبی چندلایه استفاده شده است. عمل یادگیری شبکه عصبی توسط روش بهینه‌سازی نامقید تندترین کاهش (کاهش در جهت گرادیان) انجام شده است. با ذکر مثال‌هایی تاثیر شبکه‌ها در یافتن پارامترهای بهین تقریب‌گرهای موجک توضیح داده شده است.

واژه های کلیدی: تقریب توابع، تقریب‌گرهای موجک، شبکه‌های عصبی مصنوعی.
رده‌بندی موضوعی ریاضی (2010): 42C40، 97N40.

۱ مقدمه

موجک‌ها از جمله ابزار محاسباتی برای تقریب توابع هستند که کاربردهای فنی و مهندسی و نظری زیادی دارند[۷]. از کاربردهای موجک‌ها می‌توان به پردازش تصویر، انگشت‌نگاری و پایدارسازی سیستم‌های کنترلی اشاره کرد [۷]. ابزار دیگر برای تقریب توابع که می‌توان توسط آن مسائل علمی را مورد تجزیه و تحلیل قرار داد، علم شبکه‌های عصبی مصنوعی است.
بحث شبکه های عصبی مصنوعی، نشات گرفته از ساختار شبکه عصبی زیستی انسان است که بر اساس آن الگو، عمل پردازش محاسبات انجام می‌شود. اخیراً توسط شبکه‌های عصبی، خیلی از الگوریتم‌ها و محاسبات ریاضی انجام می‌شود. مواردی چند از توانایی‌های شبکه‌های عصبی عبارتند از: تقریب یک تابع مانند $f : R^n \longrightarrow R^m$ با هر درجه دقت، شناسایی الگوها، پیش بینی، کنترل و پایدارسازی سیستم‌ها و غیره. از کاربردهای شبکه‌ها در ریاضی می‌توان به موارد زیر اشاره کرد: در [۴] توسط شبکه‌های عصبی، معادلات دیفرانسیل معمولی و جزئی و معادلات انتگرال غیرخطی حل شده‌اند. در [۵] ریشه‌های معادلات جبری غیرخطی $f(x) = ۰$ با استفاده از شبکه‌های عصبی پیدا شده‌اند. در تمام کارهای انجام شده فوق معماری شبکه‌های عصبی مختلف است در برخی شبکه چندلایه و در برخی RBF و یا بازگشتی بکار رفته است[۱] . در این مقاله، ما از شبکه های چند لایه [۱] برای تقریب استفاده می‌کنیم و منظور از شبکه عصبی همان شبکه عصبی مصنوعی است.

۲ تقریب توسط موجک‌ها

در نظریه موجک، توابع پایه‌ای زیر به‌عنوان توابع متعامد در نظر گرفته می‌شوند.

$$\psi_{j,k}(x) = ۲^{\frac{j}{۲}}\psi(۲^j x - k), \quad x \in \mathbb{R}, j, k \in \mathbb{Z}. \tag{۱.۲}$$

هدف، بررسی بسط توابع موجود در $L^۲(\mathbb{R})$ بر حسب توابع پایه‌ای $\psi_{j,k}(x)$ ها می‌باشد. با فرض $\psi(x) = e^{-x^۲}$ تابع دلخواه $f \in L^۲(\mathbb{R})$ به صورت زیر تقریب زده می‌شود.

$$f(x) = \sum_{j\in\mathbb{Z}} \sum_{k\in\mathbb{Z}} C_{j,k}\psi_{j,k}(x). \tag{۲.۲}$$

می‌دانیم با اعمال شرایط تعامد روی $\psi_{j,k}$ ها، ضرایب $C_{j,k}$ بصورت خاصی برحسب تابع f معین می‌شوند. ضریب $۲^{j/۲}$ در توابع $\psi_{j,k}(x)$ بخاطر این است که نرم هر $\psi_{j,k}$ برابر یک باشد. فرض می‌کنیم $f \in L^۲(\Gamma)$ که در آن $\Gamma = [a, b]$ یا $\Gamma = \mathbb{R}$. برای تقریب f توسط بسط (۲.۲) دو سوال اساسی این است که تعداد ضرایب $C_{j,k}$ چند تا باید باشد؟ در صورت مشخص شدن تعداد آنها آیا ممکن است این ضرایب را مستقل از

*مسئول مکاتبات و سخنران

[۱] Multi-Layer-Perceptron

پارامترهای j و k پیدا کرد؟
یافتن پارامترهای تقریب‌گر موجک (۲.۲) با استفاده از روش‌های معمول مشکلات خاص خود را دارد چرا که در مواردی ممکن است تابع f در دسترس نبوده و گذشته از آن به علت زیاد بودن توابع پایه‌ای در تقریب، زمان اجرا زیاد شود. یک پاسخ برای سوال‌های فوق این است که پارامترهای j و k را بعنوان پارامترهای مستقل از هم در نظر بگیریم و در حالت کلی فرض کنیم $j,k \in \mathbb{R}$. با این فرض توانایی تقریب توابع پایه‌ای همچنان حفظ شده و یک مزیت مهم که در مثال‌ها نیز توضیح داده خواهد شد این است که از تعداد کم توابع پایه‌ای استفاده می‌شود. برای توضیح بیشتر به تابع زیر توجه می‌کنیم.

$$f(x) = e^{-(x-\sqrt{2})^2}, \quad x \in [a,b]. \tag{۳.۲}$$

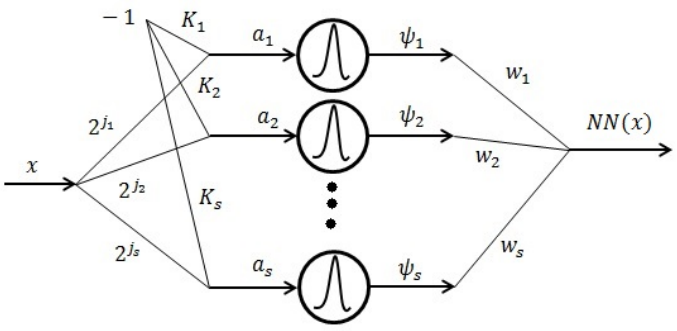
می‌دانیم در بین توابع پایه‌ای $\psi_{j,k}$، به علت صحیح بودن k,j هیچ یک از $\psi_{j,k}$ها نمی‌توانند برابر $f(x)$ باشند و باید ترکیبی از آنها $f(x)$ را ایجاد کنند. پس نمی‌توان توسط فقط یک تابع پایه‌ای مانند $\psi_{j,k}$ این تابع را تقریب زد، اما وقتی فرض کنیم پارامترهای j و k اعداد حقیقی باشند آنگاه فقط با یک تابع پایه‌ای می‌توان تابع f را تقریب زد. در حالت کلی برای همه توابع موجود در فضای $L^2(\Gamma)$ همین موضوع قابل طرح است. اگر پارامترهای j، k و $C_{j,k}$ مستقلاً محاسبه شوند آنگاه سوال اساسی این است که با چه الگوریتمی این پارامترها را باید پیدا کرد؟ در این مقاله برای یافتن پارامترهای مذکور بطور مستقل از هم، از شبکه عصبی چندلایه استفاده می‌کنیم.

۱.۲ ساختار شبکه عصبی موجک

در این بخش خواهیم دید یافتن پارامترهای تقریب‌گر موجک با کمترین تعداد $\psi_{j,k}$ها امکان‌پذیر است. یک شبکه عصبی چندلایه فقط با یک لایه پنهان را در نظر می‌گیریم. این شبکه دارای پارامترهایی بنام وزن‌ها و بایاس‌ها می‌باشد که مقدار بهین آنها جهت تقریب یک تابع باید پیدا شوند. چنین شبکه‌ای را با علامت $s-1-1$ نشان میدهند که در آن یک ورودی، یک خروجی موجود بوده و s تعداد نرونها در لایه پنهان است. روش یافتن پارامترهای بهین شبکه، یادگیری شبکه[۲] نام دارد. برای این کار دو روش به نام‌های الگوریتم پس‌انتشار خطا[۳](Bp) و روش تندترین کاهش[۴] (SD) وجود دارد. در واقع این دو روش معادل هستند. در این مقاله برای یادگیری شبکه از روش SD استفاده می‌شود. جامع [۵] بودن تقریب توسط شبکه‌های یک لایه توسط قضیه هچ-نیلسون [۶] به صورت زیر انجام شده است [۲].

قضیه ۱.۲. (هچ-نیلسون) هر تابع پیوسته $f : I^n \to \mathbb{R}^m$ را که در آن $I = [0,1]$ است، می‌توان دقیقاً با یک شبکه عصبی چندلایه با n ورودی، $2n+1$ واحد مخفی و m واحد خروجی نشان داد.

توجه داریم یکی دیگر از مزایای شبکه‌ها در امر تقریب نسبت به درونیابی در محاسبات عددی این است که شبکه می‌تواند دارای چندین خروجی و چندین ورودی باشد. به عبارت دیگر درونیابی توابعی مانند $f : \mathbb{R}^n \to \mathbb{R}^m$ به راحتی انجام می‌شود. در حالیکه در محاسبات عددی این امر کاری مشکل محسوب می‌شود. اکنون هدف ما این است که تابعی مانند f را به صورت (۲.۲) توسط شبکه عصبی تقریب بزنیم. برای این کار از شبکه با مشخصات شکل ۱ استفاده می‌کنیم. پارامترهای شبکه شامل وزن‌ها و بایاس‌ها در شبکه شکل ۱ طوری تعریف شده‌اند که تمام جملات و ویژگی‌های



شکل ۱: ساختار شبکه چندلایه.

بسط (۲.۲) حفظ شود. چون در این حالت پارامترهای j، k و $C_{j,k}$ مستقل از هم پیدا می‌شوند لذا اندیس آنها را مجزا از همدیگر فرض کرده و به

[۲] Network Learning
[۳] Backpropagation
[۴] Steepest Descent
[۵] Universal
[۶] Hecht-Neilsen

جای دو اندیس، یک اندیس در نظر گرفته می‌شود. در نتیجه بسط (۲.۲) با در نظر گرفتن شکل ۱ به صورت زیر خلاصه می‌شود.

$$\psi_i(x) = \psi_i(a_i) = \sqrt{۲^{j_i}}\psi(a_i), \quad \psi(x) = e^{-x^۲}. \tag{۴.۲}$$

$$f(x) = \sum_{i=۱}^{s} w_i\sqrt{۲^{j_i}}e^{-(۲^{j_i}x-k_i)^۲} = \sum_{i=۱}^{s} w_i\sqrt{۲^{j_i}}e^{-a_i^۲} = \sum_{i=۱}^{s} w_i\psi_i(x). \tag{۵.۲}$$

ملاحظه می‌کنیم که معادلات ریاضی شبکه واقع در شکل۱ به صورت (۴.۲) و (۵.۲) هستند. توجه داریم که تعداد نرون‌ها در لایه پنهان به تعداد s نرون می‌باشد. تعداد دقیق s را نمی‌توان قبل از شروع یادگیری نوشت اما می‌توان از روشی مانند روش رشد و هرس [۷] استفاده کرد. به عبارت دیگر، با تعداد کم مانند یک یا دو نرون شروع کرده و با زیاد شدن خطا آن را افزایش داده و با کم شدن خطا تعداد آنها کاهش می‌یابد. نهایتاً با یک تعداد ثابت نرون و دقت مطلوب آموزش شبکه تمام می‌شود.
برای بروز کردن پارامترهای j_i، k_i و w_i از روش SD استفاده می‌کنیم. مراحل کار به این ترتیب است. ابتدا با توجه به تابع داده شده f یک مجموعه داده‌های آموزشی به صورت زیر ایجاد می‌کنیم.

$$T = \{(x_۱, f_۱), (x_۲, f_۲), \ldots, (x_p, f_p)\}. \tag{۶.۲}$$

در این مجموعه، x_iها از بازه‌ای انتخاب می‌شوند که می‌خواهیم تابع را در آن بازه تقریب بزنیم. هدف از آموزش شبکه این است که پارامترهای آن را طوری بیابیم که خروجی شبکه به ازای هر x_i یعنی $NN(x_i)$ برابر $f(x_i)$ باشد. این هدف زمانی محقق می‌شود که تابع هدف زیر می‌نیمم شود.

$$(Min)\ \ E = \sum_{x_i \in T} \left(f(x_i) - NN(x_i)\right)^۲. \tag{۷.۲}$$

اکنون برای استفاده از روش SD جهت می‌نیمم کردن تابع هدف (۷.۲) باید پارامترهای شبکه به صورت زیر بروز شوند.

$$w_{i+۱} = w_i - \eta_۱ \frac{\partial NN}{\partial w_i} . \frac{\partial E}{\partial NN}, \tag{۸.۲}$$

$$j_{i+۱} = j_i - \eta_۲ \frac{\partial NN}{\partial j_i} . \frac{\partial E}{\partial NN}, \tag{۹.۲}$$

$$k_{i+۱} = k_i - \eta_۳ \frac{\partial NN}{\partial k_i} . \frac{\partial E}{\partial NN}, \tag{۱۰.۲}$$

که در آن η_i ها نرخ یادگیری هستند و بهتر است کوچک اختیار شوند. توجه داریم که اگر نرخ یادگیری خیلی کوچک باشد زمان اجرا زیاد شده و در صورتیکه این نرخ بزرگ باشد روند همگرایی، همراه با نوسانات زیاد خواهد شد. در حالت کلی با مقادیر کوچک نرخ، روند همگرایی کامل می‌شود. مشتقات جزئی مربوط به روابط (۸.۲) الی (۱۰.۲) عبارتنداز:

$$\begin{aligned}
\frac{\partial E}{\partial NN} &= -۲ \sum_{x_i \in T} \left(f(x_i) - NN(x_i)\right), \\
\frac{\partial NN}{\partial w_i} &= \psi_i, \qquad (۱۱.۲) \\
\frac{\partial NN}{\partial k_i} &= w_i\sqrt{۲^{j_i}}\left(-۲a_i\right)(-۱)e^{-a_i^۲} = ۲w_i a_i \psi_i, \qquad (۱۲.۲) \\
\frac{\partial NN}{\partial j_i} &= w_i\psi_i \times (ln(۲)) \left(\frac{۱}{۲} - ۲a_i x_i ۲^{j_i}\right). \qquad (۱۳.۲)
\end{aligned}$$

به این ترتیب روش SD برای آموزش شبکه کامل می‌شود. مراحل یادگیری به این صورت است. یک x_i به شبکه اعمال شده و سپس مقدار خطا بدست می‌آید برای اینکه مقدار E کاملاً بدست آید این کار را برای تمام x_i ها تکرار می‌کنیم. پس از اتمام اعضای مجموعه T گوییم یک سیکل [۸] کامل شده است. بعد از یک سیکل پارامترهای شبکه بروز می‌شوند و این تکرارها تا رسیدن به خطای مطلوب در سیکل‌های متوالی ادامه پیدا می‌کنند. به این نوع یادگیری بسته‌ای گفته می‌شود. از مفاهیم مهم و حائز اهمیت در هر الگوریتم عددی موضوع همگرایی و پایداری آن است. ممکن است روشی پایدار

[۷] Growing and Pruning
[۸] Epoch

باشد اما همگرا نباشد. در نتیجه موضوع همگرایی به لحاظ ریاضی اهمیت زیادی دارد. همگرایی روش این مقاله شامل همگرایی دو الگوریتم است. یکی از آنها شبکه عصبی و دیگری روش بهینه‌سازی SD می‌باشد. تضمین همگرایی شبکه عصبی چندلایه با یک لایه پنهان توسط قضیه کولموگوروف و قضیه هچ-نیلسون و چند قضیه دیگر قابل حصول است [۲]. روش SD یک روش بهینه‌سازی نامقید است که در جهت عکس گرادیان تابع هدف، عمل جستجو را انجام می‌دهد و در هر تکرار بیشترین کاهش را برای تابع هدف ایجاد می‌کند. اگر نرخ یادگیری به اندازه کافی کوچک اختیار شود و تابع هدف دارای یک می‌نیمم منحصربفرد باشد آنگاه این روش نیز همگرا خواهد بود [۶].

۳ نتایج عددی

در این بخش، نتایج کار را در مثال‌های متنوعی بکارگرفته و تاثیر روش را نشان می‌دهیم.

مثال ۱.۳. تابع زیر دارای رفتار نوسانی و میرا است.

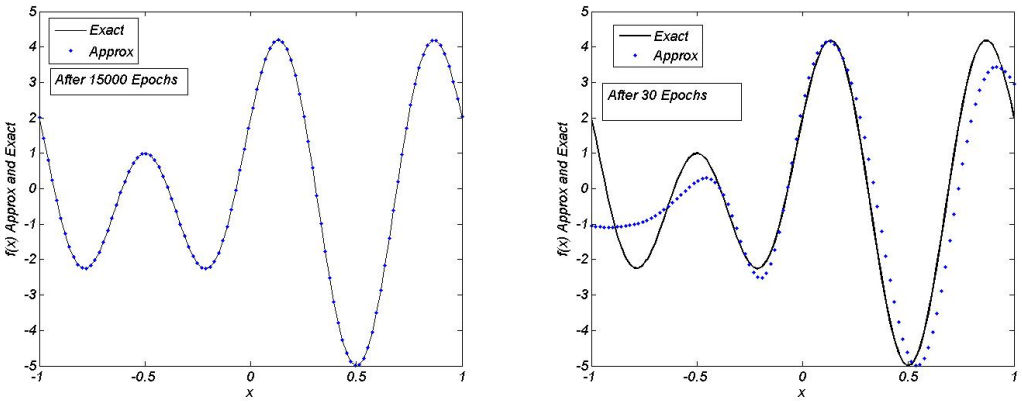
$$f(x) = ۰/۸e^{-۰/۲x}\sin(۱۰x), \quad x \in [۰, ۲]. \tag{۱.۳}$$

این تابع مثال خوبی برای تقریب توابع محسوب می‌شود. در [۳] این تابع با استفاده از روش شبکه عصبی RBF و الگوریتم رشد و هرس تقریب زده شده است، که در آن شبکه به تعداد ۴۳ نرون برای عمل تقریب بکار رفته است. با استفاده از شبکه عصبی موجک در این مقاله، پس از ۳۰۰۰۰ سیکل، استفاده از ۴۰ نرون و نرخ یادگیری ۰/۰۰۵ عمل تقریب انجام شده است.

مثال ۲.۳. تابع مثلثاتی زیر را در نظر می‌گیریم:

$$f(x) = ۳\sin(۳\pi x) + ۲\cos(۲\pi x), \quad x \in [-۱, ۱]. \tag{۲.۳}$$

این تابع نیز نوسانی است. با استفاده از شبکه موجک شامل $s = ۱۰$ نرون و ضریب یادگیری ۰/۰۰۵ بعد از ۱۵۰۰۰ سیکل، عمل یادگیری کامل شده است. نمودارهای دقیق و تقریبی نیز پس از ۳۰ و ۱۵۰۰۰ سیکل در شکل ۲ آمده است. با تقسیم بازه $[-۱, ۱]$ به ۱۰۰ قسمت بیشترین خطا ۰/۰۳۵۹۳ و کمترین مقدار آن $۱/۸۵۳ \times ۱۰^{-۴}$ است.



شکل ۲: نمودارهای تقریبی و دقیق در طی سیکل‌های متفاوت برای مثال ۲.۳.

مراجع

[۱] اژدر سلیمانپور باکفایت، شبکه‌های عصبی مصنوعی در علوم پایه . ناشر مولف، چاپ اول، ارومیه ۱۳۹۳.

[۲] لوران فاست، شبکه‌های عصبی . ترجمه هادی ویسی، کبری مفاخری و سعید باقری شورکی، انتشارات نص، تهران ۱۳۸۸.

[3] H. Han, Q. Chen and J. Qiao, Research on an online self-organizing radial basis function neural network, *Neural Comput and Applications* **19**(2010), 667-676.

[4] S. He, K. Reif, R. Unbehauen, Multilayer neural networks for solving a class of partial differential equations, *Neural Networks* **13**(2000), 385-396.

[5] D. S. Huang, H. S. Ip Horace, C. K. Law Ken, Zheru Chi, H. S. Wong, A new Partitioning neural network model for recursively finding arbitrary roots of higher order arbitrary polynomials, *Applied Mathematics and Computation* **162**(2005), 1183-1200.

[6] D. G. Luenberger, *Introduction to linear and nonlinear programming*, Reading, MA : addision-Wesley 1984.

[7] D. K. Ruch, P. J. Van Fleet, Wavelet Theory, *John Wiley & Sons, Inc, Hobokeu, New Jersey*, (2009), 1696-1705.