بسم الله الرحمن الرحيم

دانشکده: مهندسی برق و رباتیک

گروه: کنترل

شناسایی ربات فانتوم توسط شبکه‌های عصبی

دانشجو :

حسین صفاری

استاد راهنما :

دکتر حیدر طوسیان شاندیز

پایان‌نامه ارشد جهت اخذ درجه کارشناسی ارشد

ماه و سال انتشار : بهمن ۹۱

دانشگاه صنعتی شاهرود

دانشکده: برق و رباتیک

گروه:کنترل

پایان‌نامه کارشناسی ارشد آقای حسین صفاری

تحت عنوان: شناسایی ربات فانتوم توسط شبکه عصبی

در تاریخ ..... توسط کمیته تخصصی زیر جهت اخذ مدرک کارشناسی ارشد مورد ارزیابی و با درجه ..... مورد پذیرش قرار گرفت.

اساتید راهنما	امضاء	اساتید مشاور	امضاء
نام و نام خانوادگی :		نام و نام خانوادگی :	
نام و نام خانوادگی :		نام و نام خانوادگی :	

اساتید داور	امضاء	نماینده تحصیلات تکمیلی	امضاء
نام و نام خانوادگی :		نام و نام خانوادگی :	
نام و نام خانوادگی :			
نام و نام خانوادگی :			
نام و نام خانوادگی :			

تقدیم به:

پدر و مادر عزیزم

تشکر و قدردانی:

مراتب قدردانی خود را از استاد راهنمای این پروژه، جناب آقای دکتر طوسیان شاندیز، که با صرف وقت و همراهی همه جانبه موجب سرانجام رسیدن و موفقیت این تحقیق بودند، اعلام می‌دارم. همچنین از پژوهشکده رباتیک و مکاترونیک دانشگاه صنعتی امیرکبیر و ریاست این پژوهشکده، آقای دکتر زارعی نژاد که جهت انجام آزمایشات با اینجانب همکاری نمودند، کمال تشکر را دارم.

تعهد نامه

اینجانب ..... دانشجوی دوره کارشناسی ارشد رشته ..... دانشکده

..... دانشگاه صنعتی شاهرود نویسنده پایان نامه .....

.....تحت راهنمائی.....متعهد می شوم .

- تحقیقات در این پایان نامه توسط اینجانب انجام شده است و از صحت و اصالت برخوردار است .
- در استفاده از نتایج پژوهشهای محققان دیگر به مرجع مورد استفاده استناد شده است .
- مطالب مندرج در پایان نامه تاکنون توسط خود یا فرد دیگری برای دریافت هیچ نوع مدرک یا امتیازی در هیچ جا ارائه نشده است .
- کلیه حقوق معنوی این اثر متعلق به دانشگاه صنعتی شاهرود می باشد و مقالات مستخرج با نام « دانشگاه صنعتی شاهرود » و یا « Shahrood University of Technology » به چاپ خواهد رسید .
- حقوق معنوی تمام افرادی که در به دست آمدن نتایج اصلی پایان نامه تأثیرگذار بوده اند در مقالات مستخرج از پایان نامه رعایت می گردد.
- در کلیه مراحل انجام این پایان نامه ، در مواردی که از موجود زنده (یا بافتهای آنها) استفاده شده است ضوابط و اصول اخلاقی رعایت شده است .
- در کلیه مراحل انجام این پایان نامه، در مواردی که به حوزه اطلاعات شخصی افراد دسترسی یافته یا استفاده شده است اصل رازداری ، ضوابط و اصول اخلاق انسانی رعایت شده است .

تاریخ

امضای دانشجو

مالکیت نتایج و حق نشر

- کلیه حقوق معنوی این اثر و محصولات آن (مقالات مستخرج ، کتاب ، برنامه‌های رایانه ای ، نرم افزار ها و تجهیزات ساخته شده است) متعلق به دانشگاه صنعتی شاهرود می باشد . این مطلب باید به نحو مقتضی در تولیدات علمی مربوطه ذکر شود .
- استفاده از اطلاعات و نتایج موجود در پایان نامه بدون ذکر مرجع مجاز نمی باشد.

چکیده :

برای مدلسازی یک سیستم روش‌های مختلفی وجود دارد. مدلسازی معمولاً به کمک روابط فیزیکی حاکم بر سیستم یا یک روش شناسایی یا ترکیبی از هر دو صورت می‌گیرد. این مدل در حالت کلی ممکن است استاتیک یا دینامیک، خطی یا غیرخطی، تغییرپذیر یا تغییرناپذیر با زمان، زمان پیوسته یا زمان گسسته، قطعی یا تصادفی باشد. در یکی از این روش‌ها که مدلسازی جعبه سیاه نام دارد، ابتدا آزمایش روی سیستم واقعی انجام می‌شود و داده‌های ورودی، خروجی ثبت می‌شوند و سپس از این داده‌ها برای شناسایی و مدلسازی سیستم استفاده می‌شود.

در این پایان‌نامه به شناسایی ربات فانتوم توسط شبکه عصبی پرداخته شده است. جهت بررسی عملکرد روش‌های استفاده شده برای شناسایی، ربات فانتوم را انتخاب کرده و با داده‌های واقعی استخراج شده از ربات، به شناسایی این ربات می‌پردازیم. ربات فانتوم دارای سه رابط می‌باشد و به طور گسترده در کاربردهای لامسه‌ای و انجام فرامین از راه دور استفاده می‌شود. مدلسازی این ربات، برای کاربردهای پیش‌بینی، شبیه‌سازی، کنترل، و تشخیص خطا اهمیت دارد. نتایج حاصله نشان می‌دهد که شبکه‌های عصبی قادرند با دقت بالا ربات فانتوم را شناسایی نمایند.

کلمات کلیدی: ربات فانتوم، شبکه عصبی، شناسایی سیستم‌ها، الگوریتم‌های بهینه‌سازی هوشمند

لیست مقالات پذیرفته شده از پایان‌نامه:

Experimental Nonlinear Identification Of Phantom Haptic Robot Using Neural Networks

Hossein Saffari, Heydar Toossian Shandiz, Amin Rigi
Department of Electrical and Robotics Engineering, Shahrood University Of Technology
Email: h.saffari@shahroodut.ac.ir , htshandiz@shahroodut.ac.ir

فهرست

پیشگفتار

انسان‌ها همواره تلاش کرده‌اند که با دنیای اطراف خود، ارتباط برقرار کنند. در این راه، اولین گام شناخت دنیای پیرامون می‌باشد. پدیده‌های اطراف ما، با هم رابطه علت و معلولی دارند و اگر بخواهیم روی پدیده مورد نظر کنترل و نظارت داشته باشیم، ابتدا باید بدانیم که رابطه علت و معلول چیست. به این کار یعنی استخراج رابطه علت و معلولی مدلسازی می‌گویند. با توجه به اینکه فرآیند مورد نظر، خطی یا غیر خطی باشد می‌توان از روش‌های مختلفی برای شناسایی سیستم مورد نظر استفاده نمود. یکی از روش‌های شناسایی سیستم‌های غیرخطی، شبکه‌های عصبی می‌باشند که در این پروژه به آن پرداخته خواهد شد.

شبکه‌های عصبی دارای ویژگی‌های مهمی هستند که آنها را برای شناسایی و کنترل سیستم‌های غیرخطی، مناسب می‌سازد. این ویژگی‌ها شامل موارد زیر می‌باشد:

- عدم نیاز به شناخت ارتباط بین داده‌ها
- توانایی یادگیری و خود فراگیر بودن
- توانایی خود تنظیم بودن
- قادر به مدلسازی سیستم‌های گوناگون
- توانایی در تقریب توابع غیرخطی و ساختار موازی پردازشی.

علاوه بر این، شبکه‌های عصبی شامل عناصر غیرخطی می‌باشند و این شبکه‌ها را قادر می‌سازد سیستم‌های غیرخطی پیچیده را شناسایی نمایند.

شناخت دینامیک یک سیستم برای تعیین ساختار شبکه عصبی، ورودی‌ها، خروجی‌ها و نحوه آموزش شبکه برای شناسایی یک سیستم مفید است. سیستم دینامیکی، سیستمی است که خروجی آن در هر لحظه، به ورودی‌ها و خروجی‌های فعلی و قبلی وابسته است. اغلب یک سیستم دینامیکی توسط معادله دیفرانسیل یا معادله تفاضلی تعریف می‌شود، به طوریکه سرعت تغییرات تابعی از زمان یا سایر پارامترها می‌باشد. اساساً اکثر سیستم‌های واقعی، سیستم‌های دینامیکی هستند و بیشتر سیستم‌های دینامیکی، غیرخطی هستند.

محورهای کلی این پایان نامه برای شناسایی ربات فانتوم عبارتند از:

- تولید داده‌های تجربی

- طراحی و پیاده‌سازی یک شبکه عصبی مناسب به طوری که داده‌های تولید شده را به درستی فرا گیرد.

بر اساس این دو محور اساسی، فصل‌های این پایان‌نامه شکل می‌گیرد.

در فصل اول که مقدمه می‌باشد، اهمیت شناسایی سیستم‌ها و کارهای پیشین در زمینه شناسایی ربات فانتوم معرفی می‌شوند. در فصل دوم شناسایی سیستم‌ها برای سیستم‌های خطی و غیرخطی معرفی می‌شوند. در فصل سوم الگوریتم‌های مختلف که در بحث شناسایی سیستم‌ها اهمیت دارد، ارائه می‌شود. ساختار شبکه عصبی، الگوریتم‌های یادگیری و قوانین یادگیری در فصل چهارم مورد بررسی قرار می‌گیرند. در فصل پنجم معرفی مختصر ربات فانتوم و در فصل ششم نتایج شبیه‌سازی‌ها ارائه می‌شود. در نهایت در فصل هفتم نتیجه‌گیری و پیشنهادات برای ادامه کار مطرح شده است.

فصل اول:

مقدمه

۱–۱ – مقدمه

شناسایی سیستم‌ها برای کارهایی که بر مبنای مدل سیستم هستند از قبیل کنترل، شبیه‌سازی، پیش‌بینی و تشخیص خطا کاربرد دارند. برای بسیاری از سیستم‌های دینامیکی، تاثیرات غیرخطی مهم است. مدلسازی بر اساس اصول اولیه فیزیک و روابط ریاضی کاری وقت گیر، هزینه‌بر و در برگیرنده بسیاری پارامترهای نامعلوم است. بنابراین روش‌های مدلسازی که بر اساس داده‌های اندازه‌گیری شده می‌باشند و نیاز به دانش قبلی کمتری دارند از مقبولیت خاصی برخوردار هستند.

هدف اصلی در این پروژه بررسی روش‌های مختلف شناسایی سیستم‌ها و به طور خاص شناسایی سیستم‌ها توسط شبکه عصبی می‌باشد. هم چنین جهت بررسی کارایی این روش‌ها، به شناسایی تجربی ربات فانتوم پرداخته شده است که نتایج نشان می‌دهد شبکه‌های عصبی استفاده شده قادرند رفتار ربات مورد نظر را با دقت بالایی تقریب بزنند. بنابراین با توجه به توضیحات ارائه شده، محور اصلی در این پروژه، شناسایی غیرخطی ربات فانتوم به کمک شبکه‌های عصبی توسط داده‌های اندازه‌گیری شده در آزمایشگاه می‌باشد. مدل‌هایی که بر اساس داده‌های اندازه‌گیری شده به دست می‌آیند به دو بخش تقسیم می‌شوند:

۱- مدل‌های پارامتری
۲- مدل‌های غیرپارامتری

در مدل‌های پارامتری، معادلات سیستم از قوانین فیزیکی مشخص است و فقط پارامترهای آن نامعلوم بوده و باید تعیین شود. به عبارت دیگر، ساختار مدل معلوم ولی تعدادی پارامتر نامعلوم وجود دارد. به عنوان مثال، اگر معادلات دینامیکی سیستم پاندول معکوس موجود باشد و پارامترهای آن نامعلوم باشد با مدلسازی پارامتری مواجه هستیم. پارامترهای این معادلات به کمک روش‌های بهینه‌سازی قابل حصول است.

مدل‌های غیر پارامتری نیاز به هیچ فرضی در مورد مدل ندارد و ساختار مدل مشخص نمی‌باشد. مدل غیرپارامتری به این معنی نیست که هیچ پارامتری ندارد. مسلماً هر مدلی دارای پارامتر می‌باشد که باید تخمین زده شوند، اما در مورد این مدل‌ها، پارامترها دارای مفهوم فیزیکی نیستند و مخصوص مدل می‌باشد. مهمترین مزیت این مدل‌ها این است که هیچ نیازی به داشتن دید مستقیم به سیستم دینامیکی نیست و لزومی به ارائه یک مدل فیزیکی نمی‌باشد. در این صورت فقط با داشتن داده‌های اندازه‌گیری شده می‌توان مدلی از سیستم به دست آورد. در این مدلسازی چون فرضی در مورد

ساختار مدل نمی‌شود اصطلاحاً مدلسازی جعبه سیاه اطلاق می‌شود. از مهم‌ترین و کارایی ترین مدل‌های غیرپارامتری می‌توان شبکه‌های عصبی را نام برد که در این پایان‌نامه بررسی می‌شود.

۱-۲- مروری برکارهای پیشین شناسایی ربات فانتوم

ربات فانتوم یک ربات لامسه‌ای[۱] است که توسط شرکت سنسیبل تکنولوژی[۲] ساخته شده است و به صورت گسترده در کاربردهای آزمایشگاهی استفاده می‌شود. ربات فانتوم یک ربات با سه درجه آزادی می‌باشد و در کاربردهای حسی و از راه دور استفاده می‌شود. بدست آوردن مدل‌های دقیق برای این ربات، برای کنترل، شبیه‌سازی، تخمین اهمیت دارد.

دو رویکرد برای شناسایی این ربات مورد توجه است:

۱) شناسایی قطعه به قطعه[۳]
۲) شناسایی تجربی[۴]

در شناسایی قطعه به قطعه مشخصات مکانیکی اجزا مختلف بازوی ربات از جمله جرم و اینرسی لینک‌ها، و سختی سیستم انتقال مستقلا اندازه گیری می‌شود[۱]. یکی از مشکلات پیش رو برای استفاده از این روش، نیاز به جداسازی اجزای مختلف ربات می‌باشد که در برخی موارد جداسازی اجزای ربات امکان پذیر نیست. هم چنین، به کمک طراحی رایانه‌ای[۵]، می‌توان مدلی را ارائه نمود که ویژگی‌های مکانیکی ربات را بر اساس روابط ریاضی و اجزاء آن تخمین می‌زند. مشکلات استفاده از روش طراحی رایانه‌ای عبارتند از: ساده‌سازی ریاضی، در نظر نگرفتن اجزا کوچک، فرض چگالی یکنواخت که باعث می‌شود مدل بدست آمده از این روش دقیق نباشد.

در حالیکه روش شناسایی قطعه به قطعه می‌تواند مشخصات مکانیکی اجزای مختلف را شناسایی کند، اما نمی‌تواند اثرات دینامیک پیچیده همانند اصطکاک مفاصل را مدل کند.

در روش شناسایی تجربی، یک مدل صریح یا تلویحی از دینامیک‌های بازوی ربات بر اساس آزمایش‌های انجام گرفته، ارائه می‌شود. یک مثال از شناسایی سیستم تلویحی[۶]، آموزش یک شبکه

[1] Haptic
[2] SensAble Technologies
[3] piece – wise
[4] Experimental
[5] Computer Aided Design(CAD)
[6] implicit

عصبی برای ارائه دینامیک‌های ربات می‌باشد و مدل ربات را به صورت یکجا ارائه می‌دهد و توصیف صریحی از پارامترهای ربات به صورت انفرادی ارائه نمی‌دهد[۲]. در روش شناسایی صریح[1] که معمولاً برای شناسایی بازوی ربات صنعتی استفاده می‌شود، دینامیک‌های بازوی ربات بر اساس آزمایش‌های انجام گرفته، به صورت چندین عبارت پارامتریک بیان می‌شود.

در واقع پارامترهای ربات که باید شناسایی شوند به صورت عبارات صریح ارائه می‌گردد. هم چنین اثبات شده است که دینامیک‌های ربات می‌تواند به صورت خطی پارامتریک شده و توسط روش کمترین مربعات خطا شناسایی شود[۲].

دقت روش های شناسایی تجربی، به جامع بودن دینامیک های مدل، دقت اطلاعات ثبت شده، و قوت خط سیر[2] ربات بستگی دارد.

برخلاف روش شناسایی قطعه به قطعه، روش شناسایی صریح می‌تواند مجموعه ای از پارامترها را که ترکیبی از ویژگی‌های انفرادی اجزای مختلف است را شناسایی نماید. مهمترین مزیت روش شناسایی تجربی بر روش قطعه به قطعه توانایی آنها در بررسی اثر اصطکاک، در نظرگرفتن هر نوع تغییر در بازو براثر پیکربندی‌های مختلف یا اضافه شدن اجزای جدید می‌باشد.

در مرجع [۱] مدل دینامیکی ربات فانتوم استخراج شده است که در این روش برخی فرضیات ساده‌سازی در نظر گرفته شده است.

هم چنین در [۳] مدل پارامتریک ربات فانتوم برای پیکربندی و حالت های مختلف بررسی شده است. در این مدل از برخی فرضیات ساده‌سازی، جلوگیری کرده و در نتیجه این مدل می‌تواند جایگزین سری‌های دیگر ربات فانتوم شود. در واقع این مدل می‌تواند در صورتیکه دینامیک ربات تغییر یابد، نیز استفاده شود. این حالت متناوباً اتفاق می‌افتد، به عنوان مثال در مواقعی که محققان از این ربات در پیکره بندی‌های مختلف استفاده می‌کنند یا با سنسورها یا اجزا مختلف که باعث تغییر در جرم، اینرسی، یا طول اجزا مختلف ربات می‌شود. در این مواردکه تغییر در دینامیک یا اجزا ربات داریم، روش قطعه به قطعه زمان بر بوده و معمولا دقیق نیستند. در مقاله [۲] شناسایی تجربی این ربات صورت گرفته است و نتایج نشان می‌دهد که شناسایی با دقت بالای ۹۵ درصد انجام گرفته است.

این ربات به دلیل کاربرد وسیعی که در کاربردهای رباتیک از راه دور و لامسه‌ای دارد، مورد توجه محققین حوزه رباتیک می‌باشد.

[1] explicit
[2] trajectory

۱–۳– مروری بر کارهای پیشین شناسایی توسط شبکه عصبی

در مورد شناسایی سیستم‌های دینامیکی توسط شبکه‌های عصبی کارهای زیادی انجام شده است که از آن جمله می‌توان به مراجع [۴] و [۵] و [۶] اشاره نمود. در مرجع [۴] از شبکه‌های عصبی بازگشتی جهت شناسایی رفتار دینامیکی سیستم سرو هیدرولیکی استفاده شده است و شبکه استفاده شده در این مقاله، سیستم مورد نظر را با دقت بالایی شناسایی می‌کند. مرجع [۵] سه الگوریتم مختلف آموزش شبکه‌های عصبی بازگشتی برای شناسایی سیستم‌های دینامیکی را پیشنهاد می‌دهد و از میانگین مربعات خطا برای مقایسه سه الگوریتم استفاده می‌نماید. سه الگوریتم پیشنهادی شامل پس‌انتشار خطا زمانی[۱]، آموزش بازگشتی زمان حقیقی[۲]، و فیلتر کالمن توسعه یافته[۳] می‌باشد که ازبین الگوریتم‌های پیشنهادی، فیلتر کالمن بیشترین دقت را داشته، ولی محاسبات بسیاری را در پی دارد. مرجع [۶] کارآیی روش NARX که یک ساختار شبکه عصبی بازگشتی می‌باشد، در شناسایی عملی سیستم‌های دینامیکی اثبات شده است. شناسایی سیستم‌ها توسط شبکه عصبی برای سیستم‌های غیرخطی از اهمیت ویژه‌ای برخوردار است. در [۷] شبکه‌های عصبی که در بحث کنترل و تقریب توابع استفاده می‌شوند، معرفی شده است. هم‌چنین سه ساختار کنترلی کنترل تطبیقی مدل مرجع[۴]، کنترل پیش‌بین مدل[۵]، و خطی‌سازی پس‌خورد[۶] ارائه گردیده است. هورنیک در مرجع [۸] نشان داده است که شبکه عصبی با دو لایه میانی با تعداد کافی نرون و توابع فعالساز مناسب می‌تواند توابع پیوسته مورد نظر را تقریب بزند. با توجه به اینکه نرون‌ها که تشکیل دهنده یک شبکه عصبی می‌باشند دارای ماهیت غیرخطی می‌باشند، آنها را برای شناسایی سیستم‌های غیرخطی مناسب می‌سازد. شبکه‌های عصبی نیز به دو دسته اصلی تقسیم بندی می‌شوند. شبکه‌های عصبی استاتیک که ساختار آن به صورت پیش‌خور می‌باشد. شبکه‌های عصبی دینامیک که برای سیستم‌هایی استفاده می‌شود که خروجی سیستم در هر لحظه به ورودی‌ها و خروجی‌های لحظات قبل وابسته است. اکثر سیستم‌های عملی که با آنها سروکار داریم سیستم‌های دینامیکی می‌باشند.

[1] back-propagation through time (BPTT)
[2] real time recurrent learning (RTRL)
[3] extended Kalman filtering (EKF)
[4] model reference adaptive control
[5] model predictive control
[6] feedback linearization control

۱–۴– استفاده از ربات فانتوم برای استخراج داده‌ها

برای استخراج داده‌ها، ابتدا سیگنال‌های تحریک مناسب برای ورودی انتخاب گردید. سپس از ربات فانتوم در پژوهشکده مکاترونیک دانشگاه صنعتی امیرکبیر جهت ثبت داده‌ها استفاده گردید. ورودی اعمالی به ربات مذکور، گشتاور بوده که به سه رابط اعمال و خروجی هر رابط که زاویه نسبت به مبنای اندازه‌گیری خود می‌باشد، ثبت گردید. همان طور که در فصل دوم اشاره خواهد شد برای شناسایی یک سیستم دینامیکی باید ورودی‌های خاصی را به آن اعمال نمود.

فصل دوم:

شناسایی سیستم‌های دینامیکی

۲-۱-مقدمه

شناسایی سیستم بهترین راه برای استخراج مشخصه‌های یک سیستم از روی داده‌های اندازه‌گیری شده می‌باشد. در بسیاری از کاربردهای مهندسی به علت نیاز به مدلسازی، گرایش به سمت شناسایی سیستم پدید آمده است. مدل‌های بدست آمده از شناسایی سیستم برای کاربردهای کنترل، شبیه‌سازی[۱]، پیش‌بینی[۲]، و تشخیص خطا[۳] نیاز می‌باشد. شناسایی سیستم‌های دینامیک غیرخطی از مباحث مهم در بحث شناسایی سیستم‌ها می‌باشد. مدلسازی غیرخطی بر اساس اصول فیزیکی کاری وقت‌گیر و نیاز به دانش قبلی از سیستم و دربرگیرنده بسیاری پارامترهای نامعلوم است. بنابراین روش‌هایی که بر اساس داده‌های اندازه‌گیری شده می‌باشد و نیاز به دانش قبلی کمتری دارد از مقبولیت خاصی برخوردار هستند.

از تابع تبدیل داده شده برای یک سیستم خطی تغییرناپذیر بازمان می‌توان پاسخ خروجی آن را بدست آورد. عکس این فرآیند یعنی استخراج تابع تبدیل از پاسخ اندازه گیری شده، "شناسایی سیستم" نام دارد. شناسایی اساسا فرآیند نمونه برداری از سیگنال‌های ورودی و خروجی یک سیستم و نهایتاً استفاده از داده‌های ترتیبی برای مدلسازی سیستم می‌باشد.

تعریف دقیق شناسایی سیستم را می‌توان در عبارت زیر خلاصه کرد:

"شناسایی به فرآیند ساخت مدل برای سیستم‌های دینامیکی از روی مشاهدات و دانش قبلی اطلاق می‌شود.[۹]"

شناسایی بر نظر Ljung [۹] به سه مرحله تقسیم بندی می‌شود. مرحله آماده‌سازی، مرحله تخمین، و مرحله آزمایش.

در مرحله اول، مجموعه‌ای از داده‌های ورودی، خروجی ثبت می‌شوند. سپس در مرحله بعد به تعیین ساختار مدل و تخمین پارامترهای مدل می‌پردازیم. سپس آزمایش و اعتبارسنجی مدل انجام می‌گیرد.

در یکی از روش‌های راستی‌آزمایی مدل، در مرحله اعتبارسنجی مدل، خروجی سیستم واقعی با خروجی مدل مقایسه می‌گردد. اگر خطای بین داده‌های سیستم واقعی و مدل از حد مشخصی کمتر

[1] Simulation
[2] Prediction
[3] Fault diagnosis

بود مدل مربوطه مناسب می‌باشد، در غیر این صورت به تصحیح و اصلاح مدل بدست آمده می‌پردازیم.

معمولاً داده‌ها را به دو دسته تقسیم‌بندی می‌کنند. یکی برای تخمین پارامترها و بدست آوردن مدل و دیگری برای آزمایش و اعتبارسنجی مدل.

۲-۲- تقسیم بندی روش های کلی شناسایی

۲-۲-۱- شناسایی غیربرخط[1]

در این نوع شناسایی ابتدا کل داده‌ها ثبت و جمع‌آوری می‌شوند و سپس مرحله شناسایی صورت می‌پذیرد.

۲-۲-۲- شناسایی برخط[2]

در این روش شناسایی هم‌زمان با ثبت داده‌ها، عمل اصلاح پارامترهای سیستم صورت می‌پذیرد.

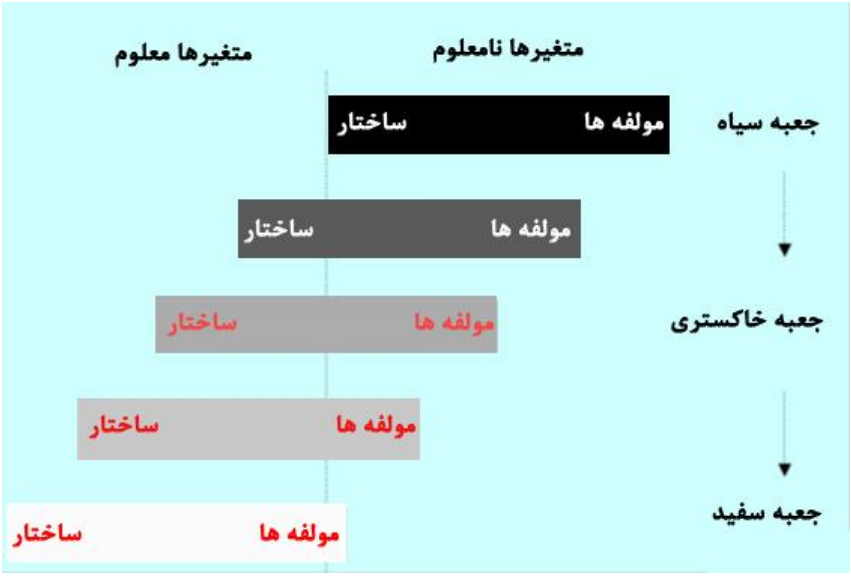
روش‌های کلی برای شناسایی سیستم‌های غیرخطی را می‌توان به صورت زیر دسته‌بندی نمود:

- شناسایی جعبه سیاه و جعبه خاکستری:

در شناسایی جعبه سیاه هیچ اطلاعاتی از دینامیک‌ها و ساختار داخلی سیستم نداریم و تنها با داشتن تعدادی ورودی و خروجی سیستم واقعی به شناسایی مدل می‌پردازیم. از آن جمله می‌توان به شبکه‌های عصبی، موجک[3]، نروفازی[4] اشاره نمود.

در شناسایی جعبه خاکستری، یک دید کلی از ساختار سیستم داریم و با توجه به این اطلاعات سعی می‌شود مدل کاملتر و دقیق‌تری از سیستم ارائه گردد. ساختار کلی جعبه سیاه و جعبه خاکستری در شکل (۲-۱) نشان داده شده است.

[1] Offline
[2] Online
[3] Wavelet
[4] Neurofuzzy



شکل(۲-۱): ساختار کلی جعبه سیاه و جعبه خاکستری

۲-۳- تقسیم بندی مدل‌ها در شناسایی سیستم

۲-۳-۱- مدل استاتیک: در این مدل، خروجی در هر زمان تابع ورودی در همان زمان است و نه زمان‌های قبلی که این مسئله در اکثر مدل‌های واقعی صادق نیست.

۲-۳-۲- مدل دینامیک: در مدل دینامیک، خروجی در هر زمان، به ورودی در آن زمان و زمان های گذشته وابسته است. ترتیب داده‌ها در این روش مهم است به طوریکه خروجی در هر لحظه، به ورودی‌ها و خروجی‌های لحظات قبل وابسته‌اند.

مدل‌هایی که بر اساس داده‌های اندازه گیری شده می‌باشند خود به دو دسته تقسیم می‌شوند:

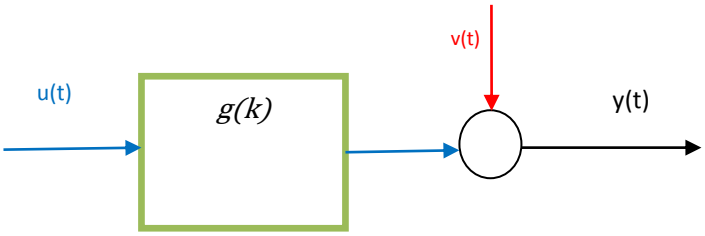
۲-۳-۳ – مدل‌های پارامتری: در این مدل‌ها ساختار مدل معلوم بوده و تعداد محدودی پارامتر برای تخمین وجود دارد. در این مدل‌ها، معادلات سیستم، از قوانین فیزیکی مشخص است و فقط پارامترهای آن است که باید تعیین شوند. مدل‌های پارامتری می‌تواند خطی یا غیر خطی باشند. شناسایی پارامتری به روند مقایسه و کمینه‌سازی اختلاف پاسخ مدل و پاسخ سیستم واقعی از طریق تنظیم ضرایب مدل به صورت سعی و خطا اطلاق می‌شود.

۲-۳-۴- مدل‌های غیرپارامتری: در این مدل، ساختار مدل مشخص نمی‌باشد و ما فقط توصیفی از مدل داریم. مهم‌ترین مزیت این مدل‌ها این است که هیچ نیازی به داشتن دید مستقیم به سیستم

و معادلات دینامیکی نیست و لزومی به ارائه یک مدل فیزیکی نیست. در این صورت فقط با داشتن داده‌های اندازه‌گیری شده می‌توان مدلی از سیستم بدست آورد. از مهم‌ترین و کاراترین مدل‌های غیرپارامتری می‌توان به شبکه‌های عصبی اشاره نمود. در این نوع مدلسازی چون فرضی در مورد ساختار مدل نمی‌شود اصطلاحاً مدلسازی جعبه سیاه اطلاق می‌شود. هدف اصلی در این پایان‌نامه، نیز شناسایی غیرپارامتریک می‌باشد.

۲-۴- مدلسازی سیستم‌های خطی و تغییرناپذیر با زمان گسسته

توصیف کلی ما از یک مدل LTI با وجود اغتشاش خارجی به صورت شکل (۲-۲) و معادله (۲-۱) است [۹]:



شکل (۲-۲): توصیف کلی مدل LTI

در این پایان‌نامه، ما با مشاهده ورودی و خروجی در زمان‌های گسسته روبرو هستیم و از ورودی و خروجی با زمان نمونه‌برداری T نمونه‌برداری می کنیم. برای سادگی از نمایش زمان نمونه‌برداری در نوشتار خودداری کرده و از t برای شمردن نمونه‌ها استفاده می‌کنیم به طوریکه: t=0,1,2, … ، در واقع t شماره نمونه‌ها در زمان‌های نمونه‌برداری می‌باشد. اگر قضیه کانولوشن را برای شکل (۲-۲) بنویسیم، خواهیم داشت:

$$y(t) = \sum_{k=1}^{\infty} g(k)\, u(t-k) + v(t) \qquad (۱-۲)$$

که k متغیر کانولوشن می‌باشد. e(t) نویز سفید با میانگین صفر و انحراف معیار یک می‌باشد. فرض می‌کنیم v(t) به صورت زیر تعریف شود:

$$v(t) = \sum_{k=0}^{\infty} h(k)\, e(t-k) \qquad (۲-۲)$$

اگر G(q) و H(q) را به صورت زیر تعریف کنیم:

$$G(q) = \sum_{k=1}^{\infty} g(k)\, q^{-k} \quad (۳-۲)$$

$$H(q) = \sum_{k=0}^{\infty} h(k)\, q^{-k} \quad (۴-۲)$$

که در آن q عملگر انتقال می‌باشد. با جایگزینی روابط فوق رابطه (۲-۱) به شکل زیر در می‌آید:

$$y(t) = G(q)u(t) + H(q)e(t) \quad (۵-۲)$$

اگر H(q) مونیک[1] باشد، یعنی در لحظه صفر داشته باشیم $h(0) = 1$، H(q) به صورت زیر خواهد بود:

$$H(q) = 1 + \sum_{k=1}^{\infty} h(k)\, q^{-k} \quad (۶-۲)$$

۲-۵- معرفی مدل‌های خطی تغییر ناپذیر با زمان

۲-۵-۱- مدل ARX

ساده ترین ساختاری که رابطه بین ورودی‌ها و خروجی‌های یک سیستم را تعریف می‌کند، به صورت زیر می‌باشد:

$$y(t) + a_1 y(t-1) + \ldots + a_{n_a} y(t-n_a) = b_1 u(t-1) + \cdots + b_{n_b} u(t-n_b) + e(t) \quad (۷-۲)$$

از آنجایی که نویز سفید e(t) باعث ایجاد خطای مستقیم در معادله تفاضلی فوق می‌شود، مدل فوق اغلب با نام مدل خطای معادله[2] شناخته می‌شود.

پارامترهایی که در مدل فوق باید تنظیم شوند عبارتند از :

$$\theta = [\, a_1\, a_2\, \ldots\, a_{n_a} \quad b_1\, \ldots\, b_{n_b}\,]^T \quad (۸-۲)$$

اگر A(q) و B(q) را به صورت زیر تعریف کنیم،

$$A(q) = 1 + a_1 q^{-1} + \cdots + a_{n_a} q^{-n_a} \quad (۹-۲)$$

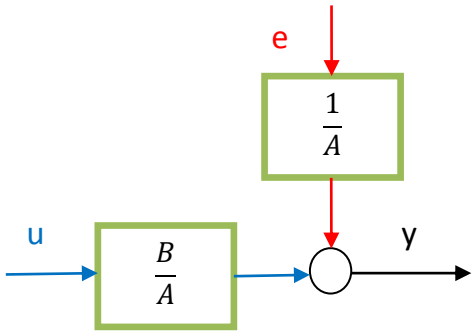
[1] monic
[2] equation error

$$B(q) = b_1 q^{-1} + \cdots + b_{n_b} q^{-n_b} \qquad (۱۰-۲)$$

طبق رابطه (۲-۵) خواهیم داشت:

$$G(q) = \frac{B(q)}{A(q)} \ , \ H(q) = \frac{1}{A(q)} \qquad (۱۱-۲)$$

همچنین مدل (۲-۷) را ARX نیز می‌نامیم. به طوریکه AR که مخفف عبارت خودبازگشتی[1] است، به بخش A(q) y(t) که به مقادیر گذشته خروجی اشاره دارد. X نیز به ورودی خارجی[2] اشاره می‌نماید. اگر در معادله (۲-۹) $n_a = 0$ باشد، رابطه (۲-۷) به مدل FIR[3] تبدیل می‌شود. شکل (۲-۳) مدل ARX را نمایش می‌دهد.



شکل (۲-۳): ساختار مدل ARX

$$y(t) = \frac{B(q)}{A(q)} u(t) + \frac{1}{A(q)} e(t) \qquad (۱۲-۲)$$

۲-۵-۲- مدل ARMAX

عیب اساسی مدل ARX عدم درجات آزادی کافی برای توصیف ویژگی‌های نویز اعمالی به سیستم است. می‌توان با میانگیری روی نویز سفید، انعطاف پذیری لازم را به مدل اضافه نمود. مدل به دست آمده به صورت زیر خواهد بود:

$$y(t) + a_1 y(t-1) + \ \ldots + \ a_{n_a} y(t-n_a) = \qquad (۱۳-۲)$$
$$b_1 u(t-1) + \cdots + b_{n_b} u(t-n_b) + c_1 e(t-1) + \cdots + c_{n_c} e(t-n_c)$$

$$C(q) = c_1 q^{-1} + \cdots + \ c_{n_c} q^{-n_c} \qquad (۱۴-۲)$$

$$G(q) = \frac{B(q)}{A(q)} \ , \ H(q) = \frac{C(q)}{A(q)} \qquad (۱۵-۲)$$

[1] Autoregressive
[2] Exogenous Output
[3] Finite Impulse Response

$$\theta = [\, a_1\, a_2 \ \dots\ a_{n_a} \qquad b_1 \ \dots\ b_{n_b} \quad c_1 \ \dots\ c_{n_c}\,]^T \tag{۱۶-۲}$$

با توجه به اعمال میانگین متحرک[1] روی e(t) توسط C(q)، مدل فوق ARMAX نامیده می‌شود.

۲-۵-۳- مدل ARARX

به جای استفاده از میانگین متحرک در معادله (۲-۱۳)، می‌توان از عبارت خودبرگشتی روی e(t) استفاده نمود و معادله زیر حاصل شود:

$$A(q)y(t) = B(q)\,u(t) + \frac{1}{D(q)}\,e(t) \tag{۱۷-۲}$$

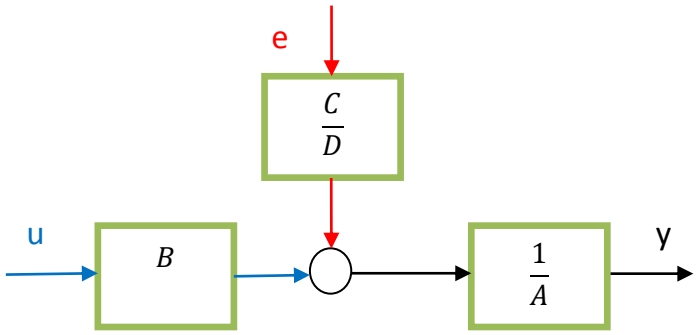
$$D(q) = d_1 q^{-1} + \cdots + d_{n_d} q^{-n_d} \tag{۱۸-۲}$$

۲-۵-۴- مدل ARMAMAX

در حالت کلی تر، معادله ARARMAX را داریم که مدل آن به صورت زیر است:

$$A(q)y(t) = B(q)\,u(t) + \frac{C(q)}{D(q)}\,e(t) \tag{۱۹-۲}$$

در این حالت کلی، ساختار مدل ARMAMAX به صورت شکل (۲-۴) می‌باشد:



شکل (۲-۴): ساختار مدل ARMAMAX

۲-۵-۵- مدل خطای خروجی[2]

در مدل‌های خطای معادله که در بخش۲-۵-۱ معرفی شد، عبارت A(q) عامل مشترک تمام مدل‌ها می‌باشد. در واقع خروجی واقعی و مقادیر قبلی آن در مدل موجود هستند.

[1] Moving Average
[2] Output Error Model

اگر فرض کنیم رابطه بین ورودی و خروجی w ((y(t) = w(t) + e(t)) به صورت یک معادله تفاضلی نوشته شود، معادله زیر حاصل می‌شود که با مدل خطای خروجی شناخته می‌شود:

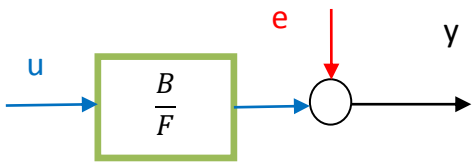
$$w(t) + f_1 w(t-1) + \ldots + f_{n_f} w(t-n_f) = b_1 u(t-1) + \cdots + b_{n_b} u(t-n_b) \quad (۲۰-۲)$$

$$y(t) = w(t) + e(t) \quad (۲۱-۲)$$

$$F(q) = f_1 q^{-1} + \cdots + f_{n_f} q^{-n_f} \quad (۲۲-۲)$$

$$y(t) = \frac{B(q)}{F(q)} u(t) + e(t) \quad (۲۳-۲)$$

$$\theta = [\, b_1 \ \ldots \ b_{n_b} \quad f_1 \ \ldots \ f_{n_f} \,]^T \quad (۲۴-۲)$$

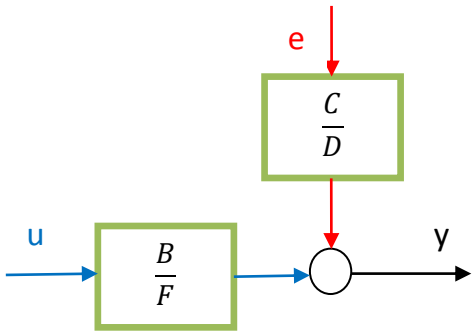


شکل (۲-۵): ساختار مدل OE

۲-۵-۶- مدل Box- Jenkins

با توسعه مدل OE و تلفیق آن با مدل ARMA، مدل Box – Jenkins حاصل می‌شود که معادله آن به صورت زیر می‌باشد:

$$y(t) = \frac{B(q)}{F(q)} u(t) + \frac{C(q)}{D(q)} e(t) \quad (۲۵-۲)$$



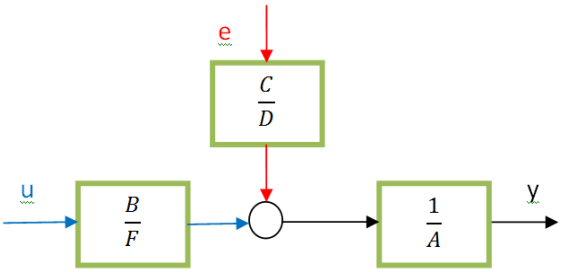
شکل (۲-۶): مدل Box - Jenkins

همان طور که ملاحظه می‌شود، عبارت A(q) در معادله (۲-۲۵) فوق وجود ندارد.

ساختار کلی مدل‌ها:

بر اساس اینکه کدامیک از عبارت A,B,C,D,F در معادله (۲-۲۶) استفاده شود، حالت‌های مختلف مطرح شده در فوق قابل حصول است.

$$A(q)y(t) = \frac{B(q)}{F(q)}\,u(t) + \frac{C(q)}{D(q)}\,e(t) \qquad (۲۶-۲)$$



شکل (۲-۷): مدل کلی معادله ۲-۶۲

در جدول ۲-۱ مدل‌های مختلف بر اساس چندجمله‌ای بکار رفته در معادله (۲-۲۶) قابل مشاهده است.

۲-۶- شناسایی دینامیکی سیستم‌های غیرخطی

در ابتدا یک سیستم دینامیکی ساده با یک ورودی و یک خروجی را در نظر می‌گیریم. ورودی و خروجی‌های فیزیکی این سیستم در لحظه k را به ترتیب با u(k) و y(k) نشان می‌دهیم. شناسایی چنین سیستمی با درجه m برای u و درجه n برای y، معادل تقریب f مطابق روابط زیر است[۲۴]:

$$y(k) = f(X(k)) \qquad (۲۷-۲)$$

$$X(k) = [u(k-1), \dots, u(k-m), y(k-1), \dots, y(k-n)] \qquad (۲۸-۲)$$

تابع f که یک تقریب‌زن[1] است، می‌تواند چندجمله‌ای، شبکه عصبی، شبکه فازی باشد. در این پایان‌نامه هدف استفاده از شبکه عصبی برای تابع f است. به بردار X(K) بردار رگرسور می‌گویند. برای مشخص کردن ورودی تابع f که همان بردار X(K) است، علاوه بر مشخص بودن متغیرهای ورودی فیزیکی، باید درجه سیستم دینامیکی m و n معلوم بوده و یا تخمین زده شود.

جدول ۲-۱: ساختار مدل‌های خطی تغییرناپذیر مختلف

چندجمله‌ای بکار رفته در معادله ۲-۲۶	نام ساختار مدل
B	FIR
AB	ARX
ABC	ARMAX
AC	ARMA
ABD	ARARX
ABCD	ARARMAX
BF	OE
BFCD	BJ

معمولاً در بردار ورودی سیستم‌های دینامیکی (رگرسور) از خروجی مدل یا سیستم واقعی استفاده می‌شود. سیستم واقعی که با نام فرآیند هم از آن یاد می‌کنیم، سیستمی در عالم واقعیت است که هدف ما پیدا کردن یک مدل ریاضی برای آن می‌باشد. اما مدل ریاضی یک رابطه ریاضی است که قرار است خروجی سیستم واقعی را شبیه‌سازی یا تخمین بزند.

همان طور که در رابطه (۲-۲۸) ملاحظه گردید، در بردار رگرسور از گذشته خروجی (مدل یا سیستم واقعی) برای پیش‌بینی خروجی در زمان های بعد استفاده می‌شود. به این نوع بردار رگرسور

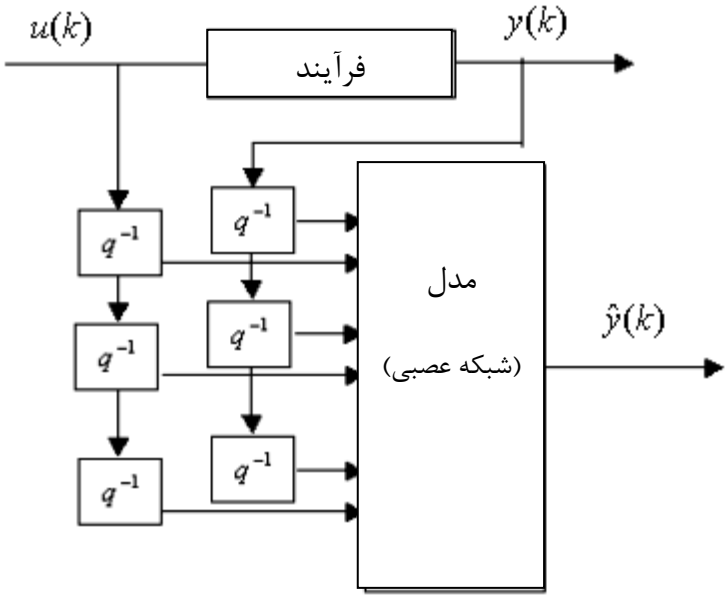
[1] Estimator

که شامل خروجی نیز است، خودبرگشت گفته می‌شود. حال بنا به اینکه از خروجی سیستم واقعی (y) یا خروجی مدل ($\hat{y}$) در بردار رگرسور استفاده شود، دو مدل بسیار مهم بدست می‌آید:

۲-۶-۱- مدل غیرخطی Equation Error

در این مدل از خروجی واقعی سیستم دینامیکی در بردار رگرسور استفاده می‌شود. چون در این نوع مدل، از خروجی سیستم دینامیکی استفاده می‌شود، مدل به صورت سری با سیستم واقعی قرار می-گیرد (شکل(۸-۲)). بنابراین به این مدل‌ها مدل سری – موازی نیز گفته می‌شود. بردار رگرسور این مدل عبارتست از:

$$X(k) = [u(k-1), \dots, u(k-m), y(k-1), \dots, y(k-n)] \qquad (۲۹-۲)$$



شکل (۲-۸): مدل سری- موازی

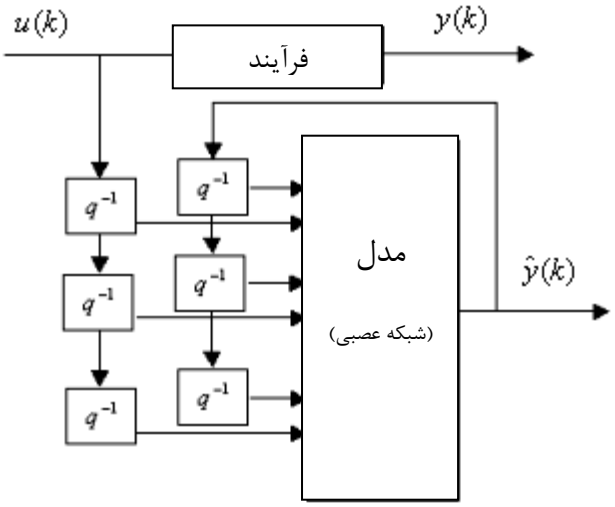
این مدل غیرخطی به NARX[1] نیز معروف است. عبارت خودبرگشت بخاطر این است که از خروجی سیستم واقعی در بردار رگرسور استفاده شده است. ورودی u(k) که در بردار رگرسور استفاده شده است به معنای ورودی خارجی از محیط است و در اصطلاح شناسایی سیستم برون‌زاد[2] گفته می‌شود. عبارت غیرخطی نیز به این مسئله اشاره دارد که ساختار مدل که برای شناسایی استفاده می‌شود یا

[1] Nonlinear Autoregressive with Exogenous
[2] Exogenous

همان تابع تخمین‌زن f، دارای ساختار غیرخطی است. حال اگر از شبکه‌های عصبی به عنوان مدل یا همان تابع f استفاده شود، مدل NARX به NNARX[1] تبدیل می‌شود.

۲-۶-۲- مدل غیرخطی Output Error

در این مدل از خروجی خود مدل در بردار رگرسور استفاده می‌شود. چون در این نوع مدل از خروجی مدل در بردار رگرسور استفاده می‌شود، مدل به صورت مطلقاً موازی با سیستم واقعی قرار می‌گیرد.



شکل (۲-۹): مدل موازی

که بردار رگرسور به صورت زیر است:

$$X(k) = [u(k-1), \dots, u(k-m), \hat{y}(k-1), \dots, \hat{y}(k-n)] \qquad (۳۰-۲)$$

که $\hat{y}$ خروجی مدل می‌باشد. به این مدل اصطلاحاً مدل NOE که اختصار خطای خروجی غیرخطی[2] است، گفته می‌شود. علت این نامگذاری این است که در این مدل خطای بین خروجی واقعی و خروجی مدل یعنی $e = y - \hat{y}$ کمینه می‌شود.

در صورتیکه به جای تابع تخمین گر f از شبکه عصبی استفاده شود، شبکه بدست آمده را اصطلاحاً NNOE[3] می‌گویند.

[1] Neural Network Autoregressive with Exogenous
[2] Nonlinear Output Error
[3] Neural Network Output Error

با توجه به اینکه برای مدل سری – موازی (NARX)، خروجی فرآیند در بردار رگرسور استفاده می‌شود، و در مدل موازی (NOE)، خروجی مدل پس‌خورد می‌شود تفاوت مهمی را در کاربرد آنها ایجاد می‌کند. دو کاربرد مهم در شناسایی شبیه‌سازی و پیش‌بینی می‌باشد. در شبیه‌سازی می‌خواهیم با داشتن ورودی و بدون داشتن خروجی سیستم واقعی، خروجی سیستم را پیش‌بینی کنیم. یعنی ساختاری مشابه شکل (۲-۹) می‌خواهیم. به طور مثال در ساخت شبیه‌ساز خودرو فقط ورودی‌هایی که راننده اعمال می‌کند در دسترس است. در این حالت رانندگی واقعی صورت نمی‌گیرد تا بتوانیم خروجی‌های واقعی را داشته باشیم. بنابراین فقط می‌توان از خروجی‌های مدل برای پیش‌بینی استفاده نمود.

بنابراین برای اهداف شبیه‌سازی فقط می‌توان از مدل OE استفاده کرد. اما برای پیش‌بینی که برای اهداف کنترلی استفاده می‌شود هم می‌توان از مدل موازی و هم از مدل سری – موازی استفاده نمود. البته برای مدل NARX خروجی سیستم واقعی باید در دسترس باشد. اغلب به دلیل اینکه NARX دارای پسخورد نمی‌باشد و پایداری و بهینه‌سازی آن آسان می‌باشد از این مدل برای کاربردهای پیش‌بینی استفاده می‌شود.

منظور از ساخت یک مدل در شبکه‌های عصبی، همان آموزش شبکه است. و منظور از پیاده‌سازی شبکه، این است که مدل یا شبکه ساخته شده را مورد استفاده قرار داده و به ازای ورودی‌های مختلف، خروجی‌های مطلوب را دریافت نمود. آموزش شبکه با توجه به اینکه از خروجی‌های واقعی استفاده شود یا از خروجی مدل، به دو بخش تقسیم می‌شود:

۱) ساخت مدل سری– موازی: در این حالت خروجی‌های سیستم واقعی و ورودی‌ها در دسترس است و با کمک آنها می‌توان خروجی را در زمان های آینده پیش‌بینی کرد.
۲) ساخت مدل موازی: در این حالت خروجی‌های سیستم واقعی در دسترس نیست و فقط ورودی‌های کنترلی موجود است. یکی از کاربردهای این مدل ساخت شبیه‌ساز ها می‌باشد.

۲–۷– طراحی سیگنال تحریک برای سیستم‌های خطی و غیرخطی

یکی از مهمترین مسائل در بحث شناسایی سیستم‌ها، طراحی مناسب سیگنال تحریک پایا[1] برای جمع‌آوری داده‌ها می‌باشد. این مرحله برای سیستم‌های غیرخطی باید دقیق‌تر انجام شد، زیرا مدلها با

[1] Persistent Exciting

دینامیک‌های غیرخطی، دارای شرایط پیچیده‌اند و بنابراین داده‌ها برای تحریک سیستم باید دارای اطلاعات بیشتری باشند. در سیستم‌های واقعی معمولاً محدودیت سیگنال ورودی وجود دارد، لذا ورودی باید بین یک مقدار کمینه و بیشینه محدود شود. یعنی داشته باشیم:

$$u \in [\, u_{min} \quad u_{max} \,]$$

برای هر سیستم دینامیکی با توجه به مشخصات ویژه آن باید سیگنال مخصوصی طراحی شود و باید طوری باشد که مدهای دینامیکی سیستم برانگیخته شود. تجربه نشان داده است که شکل سیگنال ورودی می‌تواند روی دقت تقریب یک پارامتر تاثیر زیادی داشته باشد.

در زیر برخی از سیگنال‌هایی که در شناسایی سیستم‌ها استفاده می‌شود، به اختصار ارائه شده است:

سیگنال ثابت:

این سیگنال برای سیستم‌های خطی مناسب است که دارای یک پارامتر مجهول می‌باشند. این سیگنال برای شناسایی مناسب نیست، زیرا نمی‌تواند دینامیک‌های سیستم را تحریک نماید.

سیگنال پله:

این سیگنال نیز می‌تواند برای سیستم‌های خطی بکار رود که فرکانس کاری پایین دارند و تعداد دو پارامتر مجهول دارند.

سیگنال پالس:

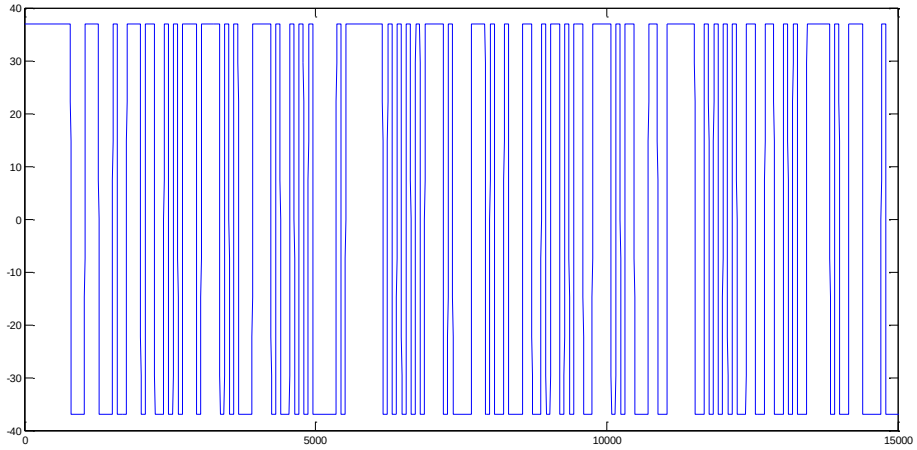
این سیگنال برای شناسایی سیستم‌ها مناسب می‌باشد. با توجه به فرکانس سیگنال مربعی، می‌توان یک محدوده فرکانسی مطلوب از سیستم را مورد تاکید قرار داد.

سیگنال PRBS[1]:

سیگنال PRBS یک سیگنال متناوب، قطعی، و با ویژگی‌های نویز سفید است. در واقع این سیگنال، یک سیگنال متناوب با فرکانس‌های مختلف است تا بتواند تمام مودهای سیستم را تحریک نماید. در شکل(۲-۱۰) یک نمونه سیگنال PRBS مشاهده می‌شود. برای سیستم‌های غیرخطی، در کنار ویژگی‌های فرکانسی سیگنال PRBS، دامنه سیگنال تحریک نیز باید تمام شرایط کاری را شامل شود. یک راه حل برای این موضوع، توسعه سیگنال PRBS به دامنه‌های مختلف است، به طوریکه در

[1] Pseudo Random Binary Signal

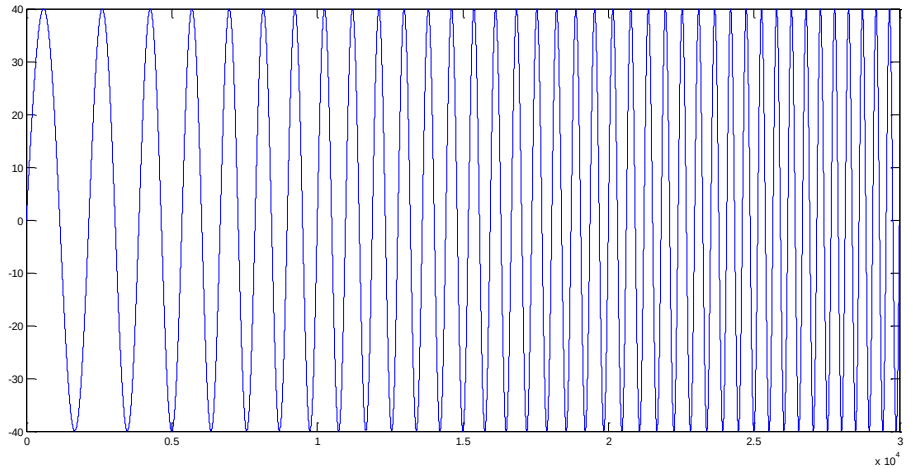
هر گام، به سیگنال PRBS یک دامنه خاص اختصاص دهیم و در این حالت، سیگنال APRBS[1] را داریم که در واقع یک سیگنال باینری شبه تصادفی با مدولاسیون دامنه می‌باشد.



شکل (۲-۱۰): سیگنال PRBS

سیگنال سینوسی جاروب شده [2] :

سیگنال دیگری که جهت شناسایی سیستم‌ها رایج است، سیگنال سینوسی با فرکانس‌های مختلف است. این سیگنال، یک سیگنال سینوسی با جاروب فرکانسی است. به عبارت دیگر، فرکانس سیگنال سینوسی، به صورت خطی با زمان افزایش می‌یابد.



شکل (۲-۱۱): سیگنال Chirp

[1] Amplitude Modulated PRBS

[2] Chirp Signal

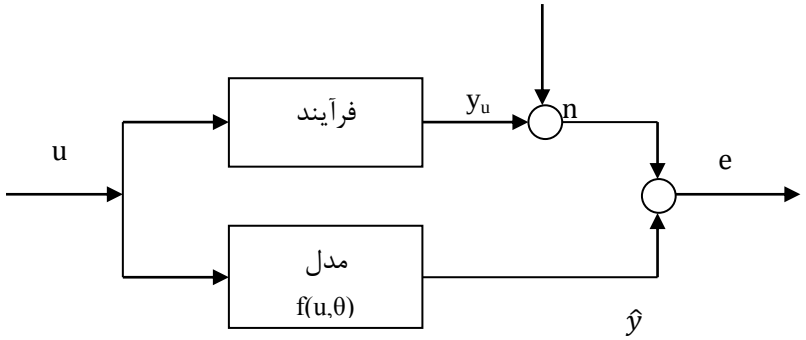
فصل سوم :

مروری بر روش‌های بهینه‌سازی و تخمین پارامتر

۳-۱- مقدمه

همان طور که در فصل ۲ اشاره شد هدف شناسایی این است که خروجی یک مدل را به خروجی اندازه‌گیری شده که مطلوب است، نزدیک کند. یا بعبارتی خطای نشان داده شده در شکل (۳-۱) را کاهش دهد. به همین خاطر در تمام روش‌های شناسایی سیستم، یک تابع هزینه بر اساس اختلاف خروجی مدل از خروجی اندازه گیری شده تعریف می‌شود که باید با روشی مناسب آن را کمینه نمود. کمینه کردن این تابع هزینه با انتخاب بهینه‌ترین پارامترها محقق می‌شود. بنابراین یکی از مراحل مهمی که در شناسایی با آن مواجه هستیم، مسئله بهینه‌سازی است. در مورد بهینه‌سازی مباحث بسیار متنوع و مختلفی وجود دارد که بررسی مسائل آن در حد یک کتاب می‌باشد. در این فصل با توجه به کاربرد خاصی که در این پروژه پیگیری می‌شود، ابتدا مبحث مورد نیاز در بهینه‌سازی را مشخص کرده و سپس به آن خواهیم پرداخت.

در این فصل مقدمه‌ای بر بهینه‌سازی با رویکرد مدلسازی و شناسایی ارائه می‌دهیم. بیشتر روش‌هایی که در این فصل ارائه می‌شود، روش‌های بهینه‌سازی پارامتریک[1] است، بدین معنا که پارامترهای موجود در یک معادله بهینه‌سازی را بهینه می‌کنند. در شکل (۳-۱) مفهوم اساسی بهینه‌سازی با رویکرد مدلسازی نمایش داده شده است.



شکل (۳-۱): فرآیند و مدل

در مدل، $f(.)$ ورودی‌های بردار u را به خروجی اسکالر y نگاشت می‌کند. پارامترهای مدل$f(.)$ در بردار پارامتر θ ثبت شده اند به طوریکه،

$$\hat{y} = f(u, \theta) \qquad (۳-۱)$$

هدف از بهینه‌سازی پارامتریک، یافتن بهترین مقادیر برای پارامترهای θ است، به‌طوریکه خروجی مدل $\hat{y}$ به خروجی y منطبق شود.

[1] Parametric Optimization

در کنار بحث بهینه‌سازی پارامتریک، بهینه‌سازی ساختار[1] را داریم که به مسئله جستجوی ساختار بهینه برای یک مدل می‌پردازد. به عنوان مثال در این روش به دنبال ساختار بهینه برای تابع f و تعداد پارامترها می‌باشیم.

یک دید کلی:

در آموزش شبکه‌های عصبی یک روش بسیار مطرح به نام پس انتشار خطا وجود دارد که در فصل بعد به آن اشاره خواهد شد. این روش در حقیقت گرادیان تابع هزینه را نسبت به پارامترهای مدل که همان وزن‌ها هستند را به دست می‌دهد. حال سوال اصلی این است که از گرادیان چگونه می‌توان برای پیدا کردن وزن‌های بهینه استفاده نمود. چگونگی استفاده از این گرادیان، به روش بهینه‌سازی انتخاب شده برمی‌گردد. یکی از روش‌های بسیار مناسب، روش لونبرگ-مارکوارت[2] است. چون این روش، بر اساس روش های قدیمی‌تر بهینه‌سازی مثل روش گاوس-نیوتن بنیان گذاشته شده است، لازم است که در ابتدا یک نگاه کلی به دیگر روش های بهینه‌سازی انداخته شود و سپس به توضیح این روش، پرداخته شده است.

۳-۲- بهینه‌سازی آزاد یا غیر مقید

در این نوع بهینه‌سازی خود تابع هزینه کمینه می‌شود و نیاز به رعایت قیود دیگر از جمله نامساویها نمی‌باشد. در این حوزه روش‌های مختلفی وجود دارد:

۱) روش‌های جستجو: در این روش از اطلاعات مشتق استفاده نمی‌شود و برای مسائلی با خواص غیرخطی بالا و یا دارای گسستگی مناسب می‌باشند.
۲) روش گرادیان: در این روش از اطلاعات مشتق تابع مورد نظر استفاده می‌شود و این روش برای زمانی مناسب می‌باشد که تابعی که کمینه می‌شود نسبت به مشتق اول خود پیوسته باشد.
۳) روش گرادیان درجه بالا: روش های گرادیان با درجه بالا مانند روش نیوتون، هنگامی استفاده می‌شود که اطلاعات درجه دوم تابع، آماده و به سادگی قابل محاسبه باشد. محاسبه اطلاعات درجه دو با استفاده از مشتق‌گیری عددی از لحاظ محاسباتی پر هزینه است.

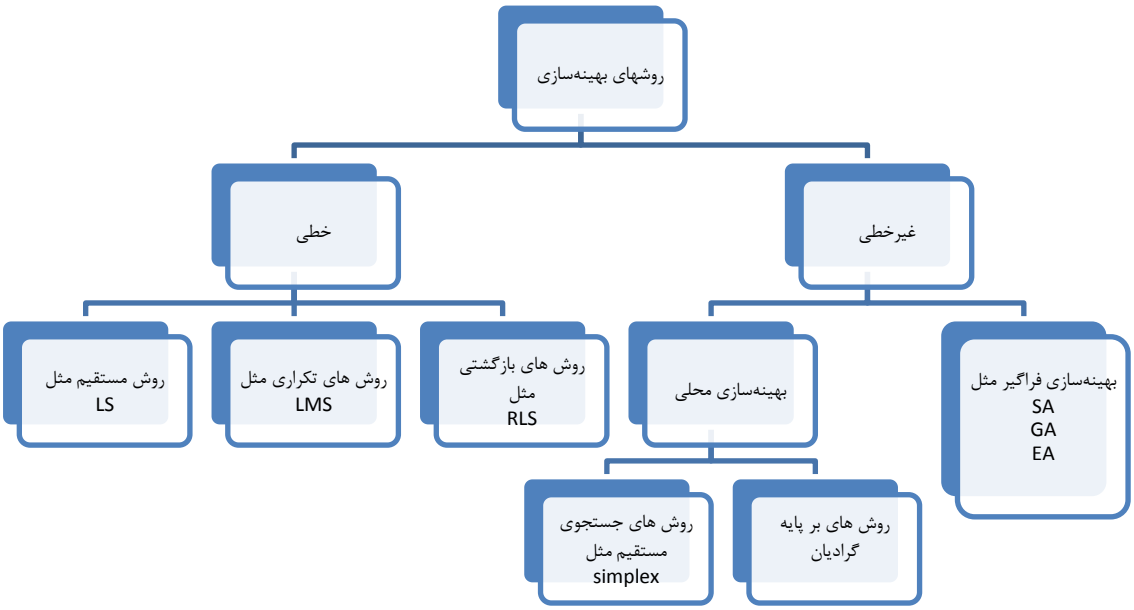
۳-۳- مروری بر روش های بهینه‌سازی پارامتریک

بهینه‌سازی روش های آموزش با ناظر به سه روش تقسیم بندی می‌شوند:

[1] Structure Optimization
[2] levenberg - marquardt

- بهینه‌سازی خطی
- بهینه‌سازی غیرخطی محلی
- بهینه‌سازی غیرخطی فراگیر

در شکل (۳-۲) روش‌های بهینه‌سازی پارامتریک به طور خلاصه نمایش داده شده است.



شکل (۳-۲): مروری بر روش های بهینه‌سازی پارامتریک

از آنجا که روشی که برای شناسایی در این پروژه استفاده می‌شود (شبکه عصبی)، یک مدل غیرخطی است، باید از روش‌های بهینه‌سازی غیرخطی استفاده نماییم.

قبل از شروع معرفی الگوریتم‌های بهینه‌سازی، معیاری که بر طبق رابطه ریاضی دقیق تعریف می‌شود، نیاز است تا بهینه گردد. در یادگیری با ناظر خطای e(i) معمولا محاسبه می‌گردد که به صورت اختلاف خروجی فرآیند y(i) و خروجی مدل $\hat{y}(i)$ تعریف می‌گردد. معمولاً خروجی واقعی با نویز همراه می‌باشد در نتیجه خروجی واقعی مطلوب نامعین است. یک معیار انتخاب برای تابع هزینه میانگین مربعات خطا[1] می‌باشد که به صورت زیر تعریف می‌گردد:

$$e(i) = y(i) - \hat{y}(i) \qquad (۳-۲)$$

$$I(\theta) = MSE = \frac{1}{N}\sum_{i=1}^{N} e^{\,2}(i) \qquad (۳-۳)$$

[1] Mean Square Error

از آنجایی که هدف نهایی کمینه‌سازی تابع $I(\theta)$ می‌باشد، به تابع فوق، تابع هزینه می‌گویند.

مسائل بهینه‌سازی خطی و غیرخطی، این نوع از تابع هزینه را برای کمینه‌سازی بکار می‌برند که با نام مسائل حداقل مربعات[1] و حداقل مربعات غیرخطی[2] از آنها یاد می‌شود. علت محبوبیت این تابع هزینه بدین خاطر است که برای رسیدن به نقطه کمینه تابع هزینه، گرادیان تابع هزینه باید برابر صفر باشد. با کمک رابطه (۳-۳) اگر خطا نسبت به پارامترها خطی باشد معادله به یک معادله خطی تبدیل خواهد شد که به راحتی قابل حل است.

اگر گرادیان تابع هزینه نسبت به پارامترهای θ غیرخطی باشد، یک روش بهینه‌سازی غیرخطی در فضای جستجو بکارگرفته شود تا به مقدار بهینه θ دست پیدا کنیم. به عنوان مثال، وزن های لایه میانی در شبکه عصبی یا مکان توابع عضویت و عرض آنها در شبکه‌های فازی، پارامترهای غیرخطی هستند. حتی در بحث شناسایی سیستم‌های خطی نیز ممکن است پارامترهای غیرخطی بوجود آید، مثلاً به هنگام کمینه کردن خطای خروجی یک مدل دینامیکی پارامترهای غیرخطی ایجاد می‌شود.

مسائل بهینه‌سازی غیرخطی عموماً دارای ویژگی‌های زیر می‌باشند:

- تعدادی کمینه محلی وجود دارد.
- حل تحلیلی برای مسئله بهینه‌سازی وجود ندارد.
- یک الگوریتم تکراری مورد نیاز است.
- به سختی می‌توانند در روش های برخط استفاده شوند.

روش های بهینه‌سازی محلی و فراگیر، فقط در سیستم‌های غیرخطی بکار می‌آید، زیرا مسائل خطی همیشه یک نقطه بهینه دارند[۱۰]. در روش‌های بهینه‌سازی محلی، از یک نقطه در فضای پارامترها آغاز می‌شود و در همسایگی این نقطه به سمت نقطه بهینه حرکت می‌نماید. واضح است که در این روش، نقطه بهینه به نقطه آغاز در فضای جستجو نزدیک است و در حالت کلی، نقطه بهینه عمومی نیست. در بهینه‌سازی عمومی، کل فضای جستجو برای یافتن نقطه بهینه جاروب می‌شود. در بهینه‌سازی محلی، یک رویکرد مناسب برای یافتن نقطه بهینه مطلوب، شروع از چندین نقطه[3] است. در این روش با استفاده از یک الگوریتم بهینه‌سازی محلی، از چندین نقطه اولیه به جستجوی نقطه

[1] Least Square
[2] Nonlinear Least Square
[3] Multi – Start Method

بهینه محلی می‌پردازیم و بهترین نقطه را به عنوان جواب نهایی انتخاب می‌کینم. بزرگترین اشکال این روش، چگونگی انتخاب پارامترهای اولیه است.

رویکرد دیگر برای توسعه روش بهینه‌سازی محلی به حالت فراگیر، اضافه نمودن نویز به پارامترهای به روز رسانی می‌باشد.

اضافه نمودن نویز سرعت همگرایی الگوریتم را کاهش می‌دهد، اما این عمل می‌تواند پارامترها را از افتادن در نقاط بهینه محلی خارج کند.

۳-۴- بهینه‌سازی محلی

۳-۴-۱- حالت کلی الگوریتم بهینه‌سازی گرادیان

الگوریتم‌های بهینه‌سازی بر پایه گرادیان، از رایج‌ترین و مهم‌ترین تکنیک‌های بهینه‌سازی محلی می‌باشند. در معادله (۳-۴) فرض می‌شود که گرادیان تابع هزینه نسبت به پارامتر θ به صورت محاسبات تحلیلی یا روش‌های تقریبی تفاضلی مشخص باشد.

در این روش بردار پارامترهای θ_{k-1}، با گام η_{k-1} و در جهت p_{k-1} (جهت خلاف گرادیان g) تغییر می‌کند.

$$\theta_k = \theta_{k-1} - \eta_{k-1} p_{k-1} \qquad (۳-۴)$$

$$p_{k-1} = R_{k-1} g_{k-1} \qquad (۳-۵)$$

$g = \frac{\partial I(\theta)}{\partial \theta}$ در تابع هزینه، گرادیان تابع هزینه نسبت به بردار پارامتر θ بوده و R_{k-1} ماتریس جهت مقیاس شده می‌باشد. واضح است که هدف بهینه‌سازی این است که در هر تکرار تابع هزینه کاهش یابد. $I(\theta) \leq I(\theta - 1)$

ساده‌ترین انتخاب برای ماتریس R ماتریس واحد می‌باشد یعنی $R=I$، که باعث جهت بیشترین نزول و خلاف جهت گرادیان g می‌باشد[۱۰].

۳-۴-۲- روش بهینه‌سازی بیشترین تنزل[1]

[1] Steepest Descent Gradient

روش بهینه‌سازی بیشترین تنزل، ساده‌ترین روش معادله (۳-۴) است. به طوریکه ماتریس R برابر ماتریس واحد قرار داده شده است.

$$\theta_k = \theta_{k-1} - \eta_{k-1} g_{k-1} \qquad (۳-۶)$$

بنابراین جهت جستجو در خلاف جهت گرادیان می‌باشد و بدین‌صورت کاهش تابع هزینه در هر تکرار را شاهد هستیم. اندازه کوچک پارامتر η باعث کندی الگوریتم و مقادیر زیاد برای η باعث واگرایی الگوریتم می‌گردد.

به طور خلاصه، این روش دارای ویژگی‌های زیر می‌باشد:

- عدم نیاز به مشتق مرتبه دوم
- فهم ساده
- سادگی پیاده‌سازی
- محاسبات خطی می‌باشد.
- حافظه کمی نیاز است.
- سرعت همگرایی پایین
- نیاز به تکرار زیاد

این نکته جالب است که بدانیم روش‌های بهینه‌سازی غیرخطی چگونه برای آموزش شبکه‌های عصبی چند لایه بکار گرفته می‌شوند. وربوس[1] الگوریتم پس انتشار خطا برای آموزش شبکه‌های عصبی چندلایه پرسپترون را در سال ۱۹۷۴ ارائه داد. در واقع این الگوریتم، محاسبه گرادیان است به‌طوریکه قانون چین[2] را به یک شبکه عصبی چند لایه اعمال نماییم. الگوریتم پس انتشار خطا، مدل ساده ای از روش بهینه‌سازی بیشترین تنزل است.

پارامترهای اولیه θ_0 :

از آنجایی که روش‌های بهینه‌سازی غیرخطی، روش تکراری هستند، در تکرار اول $k=1$ بردار پارامتر اولیه θ_0 باید تعیین گردد. واضح است که انتخاب مناسب پارامتر اولیه باعث افزایش سرعت همگرایی الگوریتم و افزایش احتمال دستیابی به نقطه بهینه عمومی می‌گردد. اگر پارامتر θ نماینده یک پارامتر

[1] Werbos
[2] Chain

فیزیکی باشد، معمولا داشتن دانش قبلی درباره سیستم می‌تواند باعث انتخاب یک مقدار معقول برای پارامتر θ_0 گردد.

۳-۴-۳- روش بهینه‌سازی نیوتون

در روش بهینه‌سازی نیوتون، ماتریس جهت R_{k-1} در معادله (۳-۴) و (۳-۵) به صورت معکوس ماتریس هشین[1] انتخاب می‌گردد.

$$\theta_k = \theta_{k-1} - \eta_{k-1} H_{k-1}^{-1} g_{k-1} \quad (۳-۷)$$

درعلم ریاضیات، و در بحث ماتریس ها و توابع چند متغیره، ماتریس هشین عبارت است از ماتریس مربعی که شامل مشتقات جزئی مرتبه دوم تابعی می‌باشد. این ماتریس در واقع بیان‌گر میزان انحنای موضعی تابع مورد نظر به ازای متغیرهای آن هست. چنین ماتریسی در قرن نوزدهم میلادی توسط ریاضیدان آلمانی آن مطرح و به نام او، نام گذاری شد.

تعریف:

فرض می‌شود تابع چند متغیره ای با دامنه اعداد حقیقی، وجود داشته باشد:

$$f(x_1, x_2, \dots, x_n) \quad (۳-۸)$$

اگر مشتقات جزئی تابع ذکر شده موجود باشد، آنگاه ماتریس Hessian تابع f عبارت است از:

(۳-۹)

$$H(f) = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2^2} & \cdots & \frac{\partial^2 f}{\partial x_2 \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \frac{\partial^2 f}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_n^2} \end{bmatrix}.$$

بنابراین در روش بهینه‌سازی نیوتون، تمام مشتق‌های مرتبه دوم تابع هزینه باید موجود باشند.

[1] Hessian

مسئله مهمی که درباره ماتریس هشین وجود دارد این است که در معادله (۳-۷) تابع هزینه در صورتی کاهش می‌یابد که ماتریس هشین مثبت معین[1] باشد.

ویژگی‌های این روش را به صورت زیر می‌توان خلاصه نمود:

- نیاز به محاسبه مشتق مرتبه دوم در محاسبات
- محاسبات پیچیده به علت وجود مشتق مرتبه دوم و محاسبه ماتریس معکوس
- سریعترین الگوریتم بهینه‌سازی غیرخطی
- مناسب برای مسائل ساده
- نیاز به یک تکرار در مسائل بهینه‌سازی خطی با انتخاب $\eta = 1$

۳-۵- بهینه‌سازی محلی حداقل مربعات غیرخطی

در بخش های قبل، روش بهینه‌سازی غیرخطی براساس روش کلی گرادیان مطرح شد. در این روش ها هیچ فرضی در مورد ساختار تابع هزینه بجز پیوستگی آن نشده بود. اما در عمل تابع هزینه زیر بسیار مرسوم است:

$$I(\theta) = \sum_{i=1}^{N} q_i e^2\,(i,\theta) \qquad (۱۰-۳)$$

که مجموع مربعات وزن دار خطا می‌باشد. برای مسائل بهینه‌سازی غیرخطی، می‌توان تابع هزینه کلی‌تر زیر را در نظر گرفت.

$$I(\theta) = \sum_{i=1}^{N} f^2\,(i,\theta) \qquad (۱۱-۳)$$

بهینه‌سازی رابطه (۳-۱۱) معروف به مسئله "حداقل مربعات غیر خطی" می‌باشد. در این بخش، روش‌های موثری که از اطلاعات مربوط به ساختار هزینه استفاده می‌کند ارائه خواهد شد. در شکل برداری رابطه (۳-۱۱) به شکل زیر در خواهد آمد:

$$I(\theta) = f^T f \qquad (۱۲-۳)$$

به طوریکه،

$$f = [\, f(1,\theta)\, f(2,\theta) \ldots f(N,\theta)\,]^T$$

در ادامه برای راحتی کار θ را حذف می‌نماییم.

[1] Positive Definite

عضو j ام گرادیان به صورت زیر محاسبه شده است:

$$g_j = \frac{\partial I(\theta)}{\partial \theta_j} = 2\sum_{i=1}^{N} f\,(i)\,\frac{\partial f(i)}{\partial \theta_j} \qquad (۱۳-۳)$$

بنابراین ژاکوبین به صورت زیر خواهد بود:

$$J = \begin{bmatrix} \frac{\partial f(1)}{\partial \theta_1} & \cdots & \frac{\partial f(1)}{\partial \theta_n} \\ . & . & . \\ \frac{\partial f(N)}{\partial \theta_1} & \cdots & \frac{\partial f(N)}{\partial \theta_n} \end{bmatrix} \qquad (۱۴-۳)$$

گرادیان به صورت زیر خواهد بود:

$$g = 2\,J^T f \qquad (۱۵-۳)$$

و ماتریس هشین نیز به صورت زیر محاسبه می‌شود:

$$H_{ij} = \frac{\partial^2 I(\theta)}{\partial \theta_j \partial \theta_i} = 2\sum_{i=1}^{N}\left(\frac{\partial f(i)}{\partial \theta_i}\frac{\partial f(i)}{\partial \theta_j} + f(i)\,\frac{\partial^2 f(i)}{\partial \theta_i \partial \theta_j}\right) \qquad (۱۶-۳)$$

با تعریف $T_{ij} = \frac{\partial^2 f(i)}{\partial \theta_i \partial \theta_j}$ هشین تابع هزینه در (۳-۱۶) به صورت زیر خواهد بود:

$$H_{ij} = 2\,J^T J + 2\sum_{i=1}^{N} f(i)\,T(i) \qquad (۱۷-۳)$$

اگر عبارت دوم در رابطه فوق را با S جایگزین داشته باشیم، خواهیم داشت:

$$H_{ij} = 2\,J^T J + 2S \qquad (۱۸-۳)$$

با تقریب عبارت فوق، رابطه (۳-۱۸) به صورت زیر تبدیل می‌شود:

$$H \approx J^T J \qquad (۱۹-۳)$$

۳-۵-۱ - روش گاوس – نیوتن

این روش، نسخه حداقل مربعات غیرخطی روش نیوتن در رابطه (۳-۷) است.

$$\theta_k = \theta_{k-1} - \eta_{k-1}\,(J_{k-1}^T J_{k-1})^{-1}\,J_{k-1}^T f_{k-1} \qquad (۲۰-۳)$$

در عمل، معکوس ماتریس رابطه (۳-۲۰) به طور صریح بیان نمی‌شود. در عوض، معادلات خطی n بعدی زیر حل می‌شود تا جهت جستجوی p_{k-1} بدست آید:

$$(J_{k-1}^T J_{k-1})\, p_{k-1} = J_{k-1}^T f_{k-1} \qquad (۲۱-۳)$$

زیرا داریم طبق رابطه (۳-۴) داریم: $p_{k-1} = (J_{k-1}^T J_{k-1})^{-1} J_{k-1}^T f_{k-1}$.

۳-۵-۲- روش لونبرگ- مارکوارت

این روش یک نسخه از روش گاوس – نیوتن است.

$$\theta_k = \theta_{k-1} - \eta_{k-1} (J_{k-1}^T J_{k-1} + \alpha_{k-1} I)^{-1} J_{k-1}^T f_{k-1} \qquad (۲۲-۳)$$

به طوریکه معکوس ماتریس به صورت صریح قابل محاسبه نیست، و در عوض معادله زیر را حل می‌کینم:

$$(J_{k-1}^T J_{k-1} + \alpha_{k-1} I)\, p_{k-1} = J_{k-1}^T f_{k-1} \qquad (۲۳-۳)$$

اضافه کردن عبارت $\alpha_{k-1} I$ در رابطه (۳-۲۲) برای تقریب ماتریس هشین $J_{k-1}^T J_{k-1}$ در حقیقت یک روش تعمیم‌دهی است و باعث حل مسئله نامساعد ماتریس $J_{k-1}^T J_{k-1}$ می‌شود. روش LM را می‌توان به این صورت توضیح داد که به ازای مقادیر کم α_{k-1}، به الگوریتم گاوس– نیوتن و به ازای مقادیر بزرگ α_{k-1} به روش بیشترین تنزل نزدیک می‌شود.

بهترین راهبرد برای انتخاب α_{k-1} به صورت زیر خواهد بود:

در ابتدا یک مقدار مثبت α_{k-1} انتخاب می‌شود. در هر تکرار این مقدار با یک ضریب مناسب کاهش یافته و از آنجا که فرض می‌شود پارامترها به مقدار بهینه خود نزدیک می‌شوند، روش گاوس– نیوتن جوابگو خواهد بود. اگر کاهش α_{k-1} منجر به جهت جستجوی نامناسب شد، (افزایش تابع هزینه)، آنگاه α_{k-1} دوباره با یک ضریب مناسب افزایش خواهد یافت تا یک جهت سراشیبی نتیجه شود.

دقت شود که روش LM یک روش حداقل مربعات غیرخطی است و از آنجا که همگرایی روش سریع بوده و چون یک روش مقاوم است، در بیشتر بهینه‌سازی‌های محلی استفاده می‌شود.

۳-۶- بهینه‌سازی فراگیر

۳-۶-۱-الگوریتم بهینه‌سازی ذرات

روش بهینه‌سازی ازدحامی ذرات[1] PSO در سال ۱۹۹۵ توسط جیمزکندی و راسل ابرهارت معرفی گردید. این روش بهینه‌سازی از عملکرد دسته جمعی حیوانات مانند پرندگان و ماهی‌ها الهام گرفته شده است. حیواناتی مانند پرندگان و ماهی‌ها که به صورت دسته‌جمعی زندگی می‌کنند، معمولاً به صورت گروهی تصمیم‌گیری می‌کنند. ایده اصلی به کار رفته در این الگوریتم به این صورت است که هر عضو از جمعیت، طبق منطق خود عملی که درست است را انجام می‌دهد. اما برای تصمیم‌گیری، نتیجه تصمیم‌گیری‌های قبلی خود و دیگران را در نظر می‌گیرد. بدین‌ترتیب، یک جریان اطلاعاتی بین اعضای جامعه به وجود می‌آید که در کل باعث می‌شود که تصمیم‌های معقول‌تری از طرف اعضای جامعه اتخاذ گردد[۱۱].

در واقع دو اصل مهم در این الگوریتم مورد توجه قرار می‌گیرند که عبارتند از:

- خود ترتیبی هر یک از اعضای جامعه: هر یک از اعضای جامعه بهترین عمل را طبق منطق خود انجام می‌دهد.
- جریان اطلاعاتی بین اعضای جمعیت: تبادل اطلاعات بین اعضای جمعیت

یکی از مشخصه‌های اصلی روش PSO تاکید بر جمعی بودن تفکر و هوش است. تفاوت عمده‌ای که این روش با الگوریتم‌های دیگر از جمله الگوریتم ژنتیک دارد، این است که اعضای جامعه از وضعیت سایر اعضا و یا بهترین عضو جامعه باخبر هستند و نتیجه بدست آمده توسط آنها را در تصمیم‌گیری‌های خود لحاظ می‌کنند. هم چنین اعضای جامعه، بهترین نتیجه‌ی خود را در طی اجرای الگوریتم به یاد داشته و همواره سعی می‌کنند تا آن را نیز در تصمیمات خود دخالت دهند.

به همین دلیل، در صورتی که یکی از ذرات دچار اشتباه در یافتن جواب بهینه شود، الگوریتم به زودی آن را جبران می‌کند. به این صورت، اعضای جامعه می‌توانند به راحتی محدوده اطراف خود را جستجو کنند، بدون آنکه نگران خراب‌تر شدن نتیجه باشند. اگر تصمیمات جدید خوب باشند، پذیرفته می‌شوند و اگر بد باشند، الگوریتم، قابلیت جبران اشتباهات را با توجه به وضعیت بهترین عضو جامعه خواهد داشت.

[3] Particle Swarm Intelligence

تاثیرپذیری افراد از سایر افراد جامعه، توسط ضرایبی موسوم به ضرایب یادگیری تعیین می‌شود.

روش بهینه‌سازی PSO :

در روش بهینه‌سازی PSO، هر جواب پیشنهادی برای مسئله مورد بررسی، که یک ذره نامیده می‌شود، به عنوان یک نقطه در فضای جستجو در نظر گرفته می‌شود.

بین نقاط فضا و بردارهایی که از مبدأ مختصات شروع می‌شوند، یک رابطه یک به یک وجود دارد . لذا، موقعیت ذرات را به صورت بردار، نمایش می‌دهیم .در هر تکرار، مجموعه جواب‌های به دست آمده تغییر می‌کنند و تلاش می‌کنیم که جوابهای بهتری برای مسأله مورد نظر به دست بیاوریم.
فرض می‌کنیم جمعیت جامعه N_{pop} باشد. به این ترتیب مجموعه جواب‌هایی که در تکرار j ام در جامعه موجود هستند، عبارتند از:

$$S^j = \{ x_i^j | x_i^j \in R^n, 1 \le i \le N_{pop}, i \in Z\} \qquad (۲۴-۳)$$

که درآن n ابعاد فضای جستجو است. اطلاعات اولیه جامعه، به صورت کاملاً تصادفی ایجاد می‌شوند. سپس اطلاعات هر تکرار با استفاده از تکرار قبلی به وجود می‌آید. هر ذره علاوه بر موقعیت، یک خاصیت اساسی دیگر به اسم سرعت دارد. سرعت هر ذره که با یک بردار نشان داده می‌شود، جهت حرکت و میزان حرکت ذره را در هر تکرار مشخص می‌کند. موقعیت ذرات در تکرار $j+1$ ام به این صورت محاسبه می‌شود:

$$x_i^{j+1} = x_i^j + v_i^{j+1} \qquad (۲۵-۳)$$

که در آن v_i^j بردار سرعت ذره i ام در تکرار j ام است.

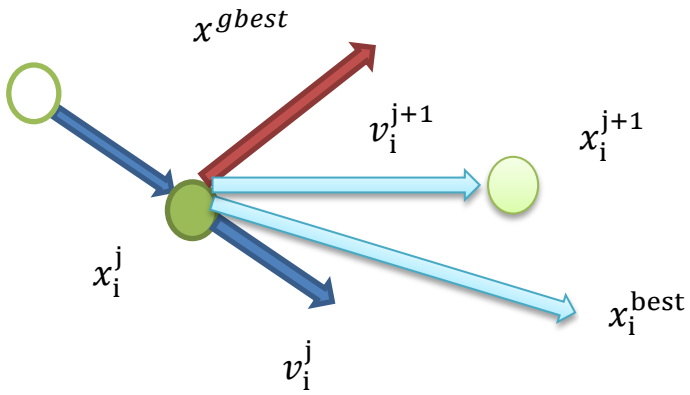
این قانون نشان‌دهنده طبیعت هر ذره است. هر ذره در هر تکرار، به اندازه بردار سرعتش در تکرار قبلی جابجا می‌شود. تصمیمی که هر ذره برای تکرارهای بعدی می‌گیرد، روی سرعت آن ذره تاثیر می‌گذارد. به این ترتیب سرعت هر ذره در تکرار $j+1$ ام، از رابطه زیر قابل محاسبه است:

$$v_i^{j+1} = w.v_i^j + r1.c1\,(x_i^{best} - x_i^j) + r2.c2\,(x^{gbest} - x_i^j) \qquad (۲۶-۳)$$

که در آن w ضریب اینرسی است و بسته به نیاز می‌تواند ثابت یا متغیر باشد.

وجود ضریب اینرسی تضمین می‌کند که ذراتی که بهترین جواب جامعه را به دست می‌آورند متوقف نشوند و همچنان به حرکت خود در مسیر قبلی ادامه دهند. هم چنین این ضریب، عملکردی مشابه جهش در الگوریتم ژنتیک دارد.

شکل (۳-۳) شمای کلی الگوریتم بهینه‌سازی ذرات را نشان می‌دهد.



شکل (۳-۳): نمای برداری الگوریتم بهینه‌سازی ذرات

در شکل (۳-۳) مشاهده می‌گردد که موقعیت بعدی در الگوریتم، با توجه به ضریب اینرسی به نسبتی در جهت سرعت قبلی v_i^j ، با توجه به بهترین تجربه هر ذره کمی در جهت x_i^{best} ، و با توجه به بهترین تجربه جمعیت مقداری هم در جهت x^{gbest} حرکت می‌نماید.

ضرایب c1 و c2 ضرایب یادگیری می‌باشند و معمولاً در بازه [۰و۲] انتخاب می‌شوند. r1 و r2 اعداد تصادفی هستند که معمولاً با توزیع یکنواخت و در بازه [۰و۱] اختیار می‌شوند.

x_i^{best} بهترین جوابی است که تا به حال توسط ذره i ام پیدا شده است.

x^{gbest} نیز بهترین جوابی است که توسط کل ذرات موجود در جامعه، پیشنهاد شده است.

c1 و c2 به ترتیب میزان تاثیر تجارب شخصی و تجربه جمعی را در تصمیم‌گیری ذره نشان می‌دهند.

دو مشخصه مهم که در جواب الگوریتم PSO تاثیرگذار است، تعداد جمعیت ذرات و ضرایب یادگیری می‌باشند. در ادامه به تاثیر این دو پارامتر در جواب‌های نهایی مسئله می‌پردازیم.

تاثیر جمعیت ذرات: N_{pop}

با افزایش تعداد اعضای جامعه، الگوریتم سریع‌تر به جواب می‌رسد، ولی حجم محاسبات افزایش چشم‌گیری دارد.

ما همیشه به دنبال دستیابی به بهترین کیفیت جواب هستیم. کیفیت جواب‌ها با تابع هزینه مشخص می‌شود و هدف دست‌یابی به هزینه نهایی کمتر می‌باشد. بیشتر شدن تعداد ذرات، تاثیر زیادی بر کیفیت جواب بدست آمده دارد و ما را ترغیب به انتخاب بیشتر ذرات می‌کند.

تاثیر ضریب یادگیری c1 : (ضریب خود اتکایی ذره)

افزایش ضریب یادگیری c1 باعث کاهش سرعت همگرایی می‌شود. ضریب c1 در واقع نشان‌دهنده میزان اتکای فرد بر تجارب شخصی است. بنابراین طبیعی است که افزایش این ضریب باعث کندتر شدن الگوریتم خواهد شد.

همچنین افزایش ضریب یادگیری c1 باعث بهبود نسبی کیفیت پاسخ نهایی می‌شود. بهتر شدن جواب‌ها با افزایش c1 را می‌توان به این نحو توجیه کرد که در هر حال c1 یک فاکتور یادگیری است و بیشتر شدن آن باعث تقویت جریان اطلاعاتی بین فرد و تجارب گذشته‌اش می‌شود. به همین ترتیب، هر چند بهترین تصمیم ممکن توسط فرد اتخاذ نمی‌شود، اما تصمیمی که با در نظر گرفتن تجارب قبلی گرفته شده است، قطعاً تصمیم نامناسبی نخواهد بود.

مقادیر بین ۰/۷۵ و ۱ می‌توانند مقادیر مناسبی برای این الگوریتم باشند[۱۱].

یکی از ویژگی‌های ضریب c_1 این است که به ذرات قابلیت جبران خطاهایشان را می‌دهد و اگر حرکت اشتباهی در فضای جستجو انجام دهند، می‌توانند آن را جبران کنند. هم چنین این ضریب، ذرات را در مقابل تغییراتی که در جامعه رخ می‌دهد، مقاوم‌تر می‌کنند و اگر مقدار ضریب c_1 از حدی بیشتر شود، ذرات حساسیت خود را نسبت به وجود ذرات دیگر از دست خواهند داد.

تاثیر ضریب یادگیری c_2 : (ضریب بهترین تصمیم جمعی)

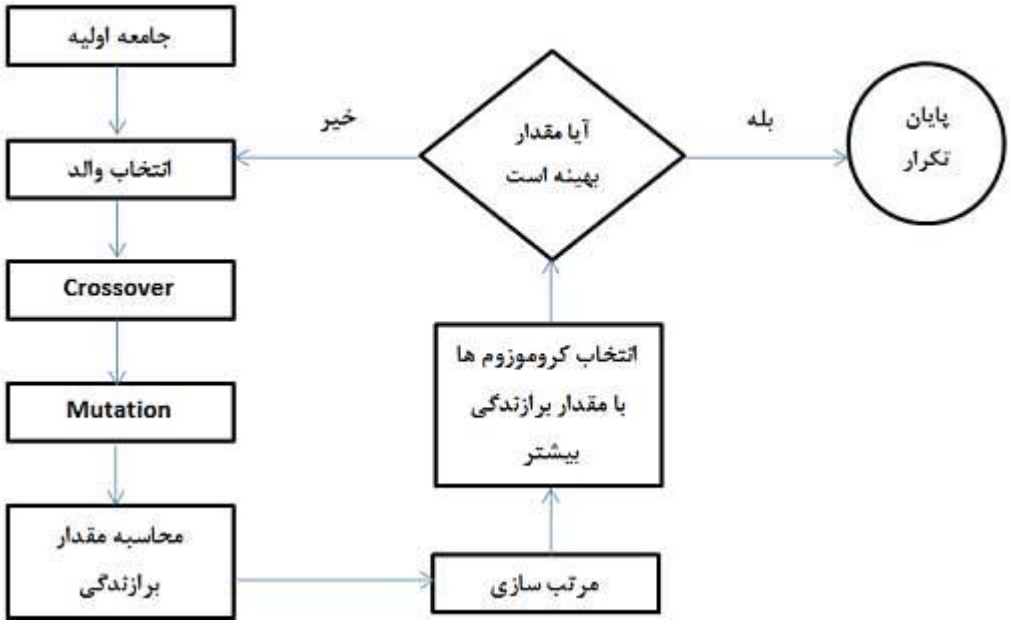
افزایش این ضریب هم باعث کندتر شدن الگوریتم می‌شود. لازم به ذکر است که منظور از همگرایی، الزاماً همگرایی به یک نقطه مطلوب نیست. یعنی نباید از ازدیاد ضرایب یادگیری، نگران شد. چون بیشتر شدن مقدار ضرایب یادگیری، الگوریتم را از افتادن در مینیمم محلی باز می‌دارد.

تاثیر ضریب یادگیری c_2 در بهتر شدن جواب نهایی بسیار بیشتر از ضریب c_1 است. ضمناً اثری که c_2 بر کندشدن الگوریتم می‌گذارد، کمتر از ضریب c_1 می‌باشد. در واقع با تبعیت ذرات از یک ذره، نوعی همگرایی کلی بین ذرات حاکم می‌شود.

مقادیر بین ۰/۲۵ و ۰/۷۵ می‌توانند مقادیر مناسبی برای c_2 باشند[۱۱].

۳-۶-۲- الگوریتم ژنتیک

الگوریتم ژنتیک یکی از الگوریتم‌های تکاملی است که اساساً روش‌های بهینه‌سازی تصادفی هستند و ایده اصلی آن از نظریه تکاملی طبیعت گرفته شده است. در محیط طبیعی بیشتر خصوصیات هر نسل از نسل قبل به ارث می‌رسد و البته ویژگی‌های جدیدی نیز بروز می‌کند. اعضای جمعیت در هر نسل با هم به رقابت می‌پردازند و آنهایی که بر حسب خصوصیات برترشان، قدرت و شایستگی بیشتری برای بقا داشته باشند زنده می‌مانند و به تولید مثل می‌پردازند. بنابراین با گذشت زمان نسل‌های آینده به سوی تکامل پیش می‌روند. نمایی از الگوریتم ژنتیک در شکل (۳-۴) نشان داده شده است.



شکل (۳-۴): الگوریتم ژنتیک

در ادامه هر یک از بخش‌های الگوریتم توضیح داده خواهد شد.

- تشکیل جمعیت اولیه:

الگوریتم ژنتیک روشی تصادفی برای بهینه‌سازی است. تشکیل جمعیت اولیه به صورت تصادفی است و اپراتورهایی که نسل‌های جدید را می‌سازند دارای عناصر تصادفی هستند. این ویژگی باعث می‌شود که الگوریتم ژنتیک قادر به بهینه‌سازی سراسری شود.

هر نسل در الگوریتم ژنتیک جمعیتی از اعضا است و هر عضو با یک کروموزوم که ویژگی‌های آن را به صورت کد شده دربردارد، مشخص می‌شود. در شبکه‌های عصبی، کروموزوم آرایه‌ای یک سطری شامل ژن‌ها (بیت ها) است که تمام وزن‌ها و بایاس‌ها را دربردارد. کافیست تمام وزن‌ها و بایاس‌های شبکه در یک رشته به دنبال هم ذخیره شوند.

مقادیر تصادفی وزن‌ها و بایاس‌ها را می‌توان از یک توزیع نرمال استاندارد با میانگین صفر و انحراف معیارσ به دست آورد.

- ارزیابی شایستگی:

از آنجایی که هدف کمینه کردن خطا می‌باشد، میانگین مربعات خطا، معیار مناسبی به عنوان تابع هزینه است. عکس این مقدار را با عنوان تابع شایستگی تعریف می‌کنیم.

$$Fitness = 1/MSE \qquad (۲۷-۳)$$

- گزینش والدین:

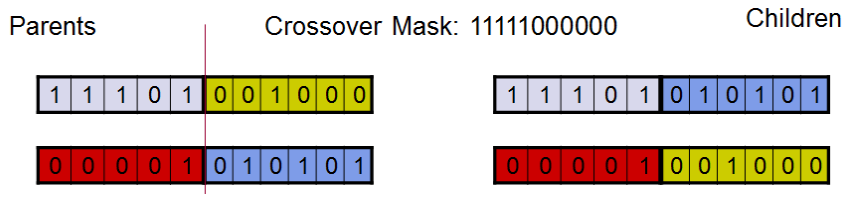
یکی از روش‌های مناسب برای انتخاب والدین، استفاده از چرخ رولت است که در آن سطح چرخ به بخش‌هایی به تعداد اعضای جمعیت با سطح‌هایی متناسب با شایستگی آنها تقسیم می‌کنیم. سپس چرخ به گردش در می‌آید و در نقطه‌ای متوقف می‌شود و عضوی از جمعیت مشخص و انتخاب می‌شود. بدین ترتیب احتمال انتخاب اعضای شایسته‌تر به عنوان والدین بیشتر است. بر این اساس ابتدا شایستگی نسبی هر عضو طبق رابطه (۳-۲۸) محاسبه می‌شود:

$$P_i^{selection} = \frac{Fit_i}{\sum Fit_i} \qquad (۲۸-۳)$$

آنگاه برای هر یک از اعضا عددی تصادفی بین صفر و یک انتخاب می‌شود. تعداد والدین معمولاً حدود ۵۰ تا ۸۰ درصد تعداد اعضای نسل انتخاب می‌شود.

- تولید فرزند (ترکیب[1] و جهش[2]) :

از هر والد، دو فرزند با عملکردهای ترکیب و جهش تولید می‌شود. عملکرد ترکیب یا جابجایی از نمونه طبیعی آن که کروموزوم‌های هر فرزند نیمی از ژن‌های خود را از هر والد به ارث می‌برد، تقلید می‌کند. در این عملگر دو کروموزوم والد از چند نقطه تصادفی شکسته می‌شوند و جابجایی روی ژن‌ها صورت می‌گیرد. در شکل (۳-۵) عملگر ترکیب تک نقطه‌ای نمایش داده شده است.



شکل (۳-۵): عملگر ترکیب در الگوریتم ژنتیک

جهش ژنتیکی، به این صورت است که یکی از بیت‌های آرایه به صورت تصادفی از ۱ به صفر یا بلعکس تغییر می‌کند. با استفاده از یک توزیع یکنواخت یک بیت بصورت تصادفی انتخاب و مقدار آن تغییر پیدا می‌کند. معمولاً عملگر جهش بعد از انجام ترکیب اعمال می‌شود.

شکل (۳-۶): عملگر جهش در الگوریتم ژنتیک

- انتخاب نسل جدید:

درصد نسبت تعداد والدین (که برابر تعداد فرزندان می‌شود) به تعداد کل جمعیت را شکاف نسل‌ها می‌گویند. بدین ترتیب، نسل جدید به اندازه این درصد از فرزندان تشکیل می‌شوند. باقی‌مانده اعضای نسل جدید به دو طریق انتخاب شایسته‌ترین اعضا (نخبه گزینی) و انتخاب تصادفی از میان اعضای نسل قبل انتخاب شده، مستقیماً به نسل بعد راه پیدا می‌کنند. نخبه‌گزینی مانع از ضایع شدن انتخاب اعضای شایسته می‌شود و انتخاب تصادفی نیز الگوریتم را از ایستایی در نقاط بهینه محلی باز می‌دارد.

[1] Crossover
[2] Mutation

بنابراین پارامترهای الگوریتم ژنتیک به صورت زیر می‌باشد:

- تعداد جمعیت
- شکاف نسل‌ها
- احتمال جهش
- آهنگ جهش

در این فصل خلاصه‌ای از روش‌های بهینه‌سازی که در بحث شناسایی سیستم‌ها و تخمین پارامترها کاربرد دارند، ارائه گردید. برای آموزش شبکه‌های عصبی در حالتی که تعداد وزن‌ها و بایاس‌ها کم باشند و ساختار شبکه عصبی پیچیده نباشد، معمولاً روش‌های بر پایه گرادیان پاسخگو بوده و به جواب بهینه سراسری می‌رسیم. اما در صورتیکه ساختار شبکه پیچیده باشد، این الگوریتم‌ها برای یافتن جواب عمومی مناسب نمی‌باشند و از روش‌های بهینه سراسری استفاده می‌کنیم.

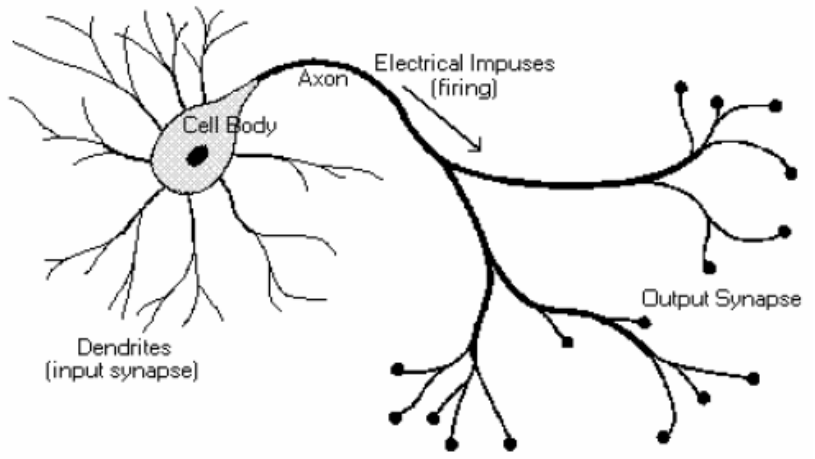
فصل چهارم:

شبکه‌های عصبی

۴-۱- مقدمه

زمینه شبکه‌های عصبی دارای تاریخچه‌ای در حدود پنج دهه می‌باشد. اما استفاده جدی از آن در حدود بیست سال گذشته بوده است و هنوز این استفاده در حال افزایش است.

شبکه عصبی، الگوی پردازش اطلاعات است که از ساختار مغز انسان الهام گرفته است[۱۳]. این شبکه‌ها از تعداد زیادی عناصر پردازشی موازی به نام نرون‌ها تشکیل شده اندکه برای حل یک مسئله خاص کار می‌کنند. شبکه‌های عصبی مصنوعی از محیط اطرافشان آموزش می‌بینند و از داده‌های ورودی، خروجی مشخص برای تنظیم ارتباطات سیناپسی که بین دو نرون وجود دارد، استفاده می‌کنند. آموزش در شبکه عصبی معادل تنظیم وزن ارتباطات سیناپسی (وزن‌ها) می‌باشد که بین نرون‌ها وجود دارد. این ارتباطات دانش لازم برای حل یک مساله خاص را ذخیره می‌کنند.



شکل (۴-۱) : ساختار نرون زیستی [۱۳]

هر نرونی از سه قسمت اصلی تشکیل شده اند :

۱- بدنه سلول که شامل هسته و قسمت های حفاظتی دیگر می‌باشد.
۲- دندریت ها که سیگنال‌های الکتریکی را دریافت می‌کنند.
۳- اکسون که نقش کانال ارتباطی را بر عهده دارد.

محل تلاقی یک اکسون از یک سلول به دندریت‌های سلول دیگر را سیناپس می‌گویند. پیام‌های عصبی در شبکه عصبی به صورت یک طرفه حرکت می‌کند : از دندریت‌ها به بدنه سلول و سپس به اکسون.

یکی از جالب ترین ویژگی‌های شبکه‌های عصبی، توانایی یادگیری این شبکه‌ها می‌باشد. این کار از طریق مثال‌های مختلف برای آموزش و استفاده از الگوریتم‌های آموزش انجام می‌پذیرد، که وزن‌ها یا پارامترهای توابع تحریک را تغییر می‌دهد. یکی از مشکلات پیش‌رو، تعیین زمانی است که مطمئن شویم شبکه به اندازه کافی، آموزش دیده است.

۴-۲- انواع یادگیری

در حالت کلی سه نوع یادگیری موجود است: یادگیری با ناظر، یادگیری تشدیدی، بدون ناظر

- در یادگیری بدون ناظر فرض بر این است که در هر مرحله از تکرار الگوریتم یادگیری، جواب سیستم یادگیرنده از قبل آماده است و به عبارتی الگوریتم یادگیری به جواب واقعی مسئله دسترسی خواهد داشت.

- در یادگیری تشدیدی که شکل خاصی از یادگیری با ناظر است، سیگنال برگشتی از محیط که به سیگنال تشدیدی موسوم است درباره شبکه نقادی می‌کند. مثلاً در یادگیری با ناظر می‌گوییم جواب مطلوب برای ورودی p بایستی t باشد، درحالیکه در یادگیری تشدیدی می‌گوییم چقدر خوب یا بد شبکه به ورودی p پاسخ داده است.

- در یادگیری بدون ناظر، جواب مطلوب برای سیستم یادگیرنده موجود نیست و شبکه عصبی به خطای یادگیری جهت بهبود رفتار سیستم یادگیرنده دسترسی نداریم. در این نوع یادگیری، پارامترهای شبکه عصبی به صورت خود سازمانده می‌باشند و توسط پاسخ سیستم اصلاح و تنظیم می‌شوند.

در آموزش شبکه‌های عصبی دو استراتژی وجود دارد:

۱- آموزش دسته‌ای[1]: در این آموزش تغییرات وزن‌های شبکه عصبی بعد از ارائه یک مجموعه داده‌ها (دسته‌ای از داده‌ها) به شبکه انجام می‌گیرد.
۲- آموزش افزایشی[2]: در این نوع آموزش تغییر وزن‌ها بعد از ارائه هر داده خاص انجام می‌گیرد. این روش با عناوین دیگری از جمله آموزش برخط و آموزش تطبیقی نیز شناخته می‌شود.

شبکه‌های عصبی برای کاربردهای مختلفی استفاده می‌شود:

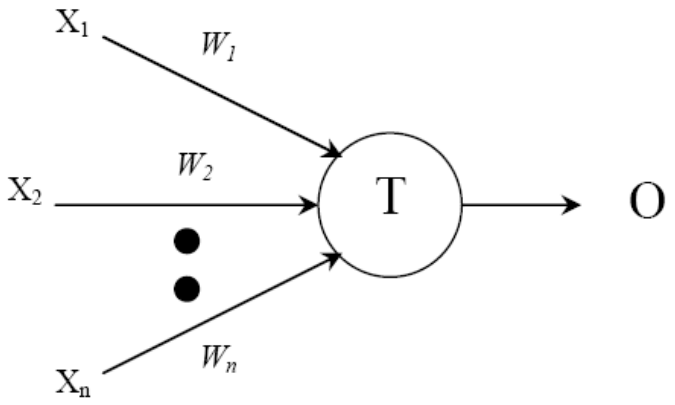
۱- طبقه‌بندی
۲- تقریب یک سیستم استاتیک
۳- شناسایی یک سیستم دینامیکی

برای مورد اول شبکه طوری پیاده‌سازی می‌شود تا طبق یک سری ویژگی‌های خاص، داده‌ها را طبقه‌بندی نماید. دو مورد ۲ و ۳ در حقیقت یک نگاشت هستند که یک سری داده‌ها را از ورودی به خروجی مرتبط می‌کند. اما این دو مورد یک تفاوت عمده با هم دارند. در مورد ۲، داده‌های بدست آمده از یک سیستم استاتیک می‌باشد. در مورد این داده‌ها ترتیب مهم نیست و برای این سیستم‌ها یک شبکه عصبی استاتیک جوابگوی ما خواهد بود. اما اگر داده‌های اندازه‌گیری معرف یک سیستم دینامیکی باشد، یعنی ترتیب داده‌ها مهم باشد و خروجی در هر لحظه، به ورودی‌ها و خروجی‌های لحظات قبل وابسته باشد، در این صورت شبکه عصبی مورد نظر باید دینامیک بوده تا بتواند رفتار سیستم مورد نظر را مدل کند.

۴-۳- مدل نرون مصنوعی

در شبکه‌های عصبی مصنوعی، ورودی‌ها به صورت خطی در وزن‌ها ضرب می‌شوند و با هم جمع می‌شوند. والترپیتس[3] اولین مدل یک نرون محاسباتی را ارائه داد.

[1] Batch
[2] Incremental
[3] Walter Pitts



شکل (۴-۲) : ساختار نرون مصنوعی

هر نرون شامل یک عنصر پردازشی با تعدادی ورودی و یک خروجی می‌باشد. در گام اول، ورودی‌های $x_1 , x_2 , x_3 \ldots\ldots\ldots , x_n$ در وزن‌هایشان ضرب می‌شوند و سپس با هم جمع می‌شوند. عمل جمع شدن توسط رابطه زیر ارائه می‌گردد:

$$net = (\sum_{i=1}^{n} w_i . x_i) \qquad (۱-۴)$$

علاوه بر این، ممکن است یک مقدار آستانه یا بایاس به رابطه فوق اضافه گردد :

$$net = (\sum_{i=1}^{n} w_i . x_i) + b \qquad (۲-۴)$$

در این بخش به بررسی شبکه‌های عصبی مصنوعی برای مدلسازی می‌پردازیم. مهمترین ویژگی‌های شبکه‌های عصبی عبارتند از :

- متشکل از تعداد زیادی اجزای ساده به نام نرون‌ها
- پردازش موازی نرون‌ها
- اجزای به هم مرتبط
- مقاوم در برابر نقص در یکی از اجزا
- یادگیری از داده‌ها

این ویژگی‌ها، یک شبکه عصبی مصنوعی را برای پیاده‌سازی سخت‌افزاری مناسب می‌سازد.

دو رویکرد تحقیقاتی برای شبکه‌های عصبی وجود دارد. علوم پزشکی، زیست‌شناسی و روانشناسی علومی هستند که علاقه‌مند به یادگیری جزئیات عملکرد مغز انسان و مدلسازی آن می‌باشند. از سوی

دیگر، مهندسین علاقه‌مند به توسعه یک ابزار عمومی برای حل مسائل مختلف با الهام از شبکه‌های عصبی زیستی می‌باشند. در این بخش ما به بخش دوم یعنی استفاده از شبکه‌های عصبی در مهندسی و کاربرد آن در ریاضیات، آمار و بهینه‌سازی می‌پردازیم.

۴-۴- تقسیم بندی کلی شبکه‌های عصبی

یک تقسیم بندی کلی برای شبکه‌های عصبی به صورت زیر می‌باشد:

۴-۴-۱- شبکه‌های استاتیک یا بدون حافظه

این شبکه‌ها به صورت پیش‌خور بوده و رابطه ورودی – خروجی عملاً چیزی جز یک تابع استاتیک نیست. در این نوع شبکه‌ها هیچ مدار پسخوردی موجود نیست.

این شبکه‌ها نیز خود به شبکه‌های تک لایه و چند لایه تقسیم می‌شوند.

۴-۴-۲- شبکه‌های دینامیک یا حافظه‌دار

در این شبکه‌ها نرون دارای پس‌خورد بوده، بدین مفهوم که خروجی نرون در لحظه حال نه تنها به ورودی در آن لحظه بلکه به مقدار خروجی خود نرون در لحظه گذشته نیز وابسته است. در شبکه‌های پسخور، حداقل یک سیگنال برگشتی از یک نرون به همان نرون یا نرون‌های همان لایه و یا لایه قبل وجود دارد.

از آنجا که ربات مورد نظر در این پروژه، یک سیستم دینامیکی است، شناسایی آن توسط شبکه‌های عصبی دینامیک مقدور است. بنابراین کار اصلی بر روی این شبکه‌ها خواهد بود که این موضوع در بخش ۴-۷ مورد بررسی قرار خواهد گرفت. اما برای شناخت و استفاده از این شبکه‌ها، نیاز به پیش‌زمینه‌ای درباره شبکه‌های عصبی استاتیک می‌باشد.

دو ساختار شبکه عصبی رایج شبکه‌های عصبی چند لایه پرسپترون[1] و شبکه‌های عصبی توابع پایه شعاعی[2] می‌باشند که در بخش ۴-۵ و ۴-۶ معرفی می‌گردند.

[1] Multi Layer Perceptron
[2] Radial Basis Function

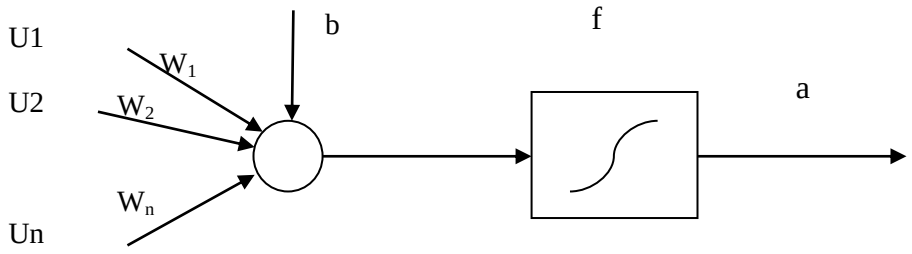
۴-۵- شبکه‌های عصبی چندلایه پرسپترون(MLP)

شبکه‌های عصبی چند لایه پیش‌خور به طور وسیعی در زمینه‌های متنوعی از قبیل طبقه‌بندی الگوها، پردازش تصاویر، تقریب توابع و ... مورد استفاده قرار گرفته است.

شبکه عصبی MLP رایج‌ترین ساختار شبکه عصبی مورد استفاده می‌باشد. در این بخش ابتدا نرون MLP معرفی می‌گردد و عملکرد آن نشان داده خواهد شد. سپس ساختار شبکه MLP ارائه می‌گردد. در بخش ۴-۵-۴ الگوریتم پس انتشار خطا معرفی و برای آموزش شبکه عصبی توسعه داده می‌شود.

۴-۵-۱- نرون MLP

شکل (۴-۳) مدل یک نرون MLP را نشان می‌دهد. این تک نرون، پرسپترون نامیده می‌شود. عملکرد این نرون به دو بخش تقسیم می‌شود. ابتدا داده‌های ورودی در وزن‌های نرون ضرب شده و سپس تابع فعال‌ساز غیرخطی f(x) روی داده‌ها اعمال می‌شود.



شکل (۴-۳) : ساختار یک نرون پرسپترون

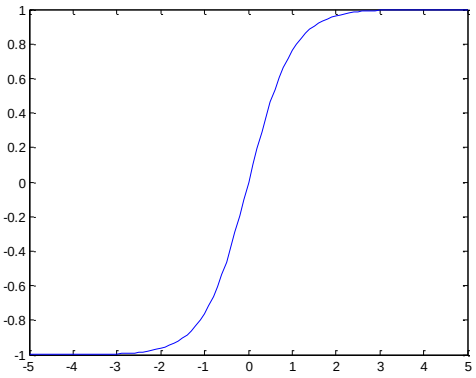
در شبکه‌های تک لایه، بردار ورودی u توسط نرون های لایه اول، طبق رابطه زیر به خروجی مرتبط می‌شوند:

$$a = f(wp + b) \qquad (۴-۳)$$

این شبکه شکل ساده‌ای از شبکه‌های پیش‌خور است.

۴-۵-۲-توابع محرک در شبکه‌های عصبی

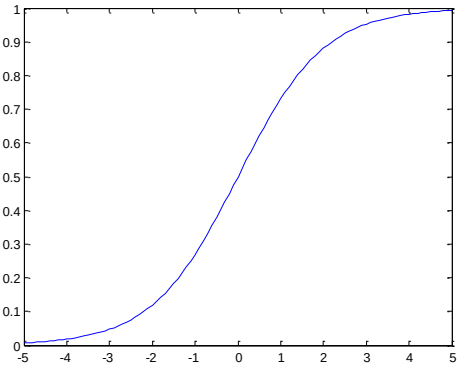
تابع محرک f می‌تواند خطی یا غیر خطی باشد. یک تابع محرک بر اساس نیاز خاص حل یک مسئله که قرارست توسط شبکه عصبی حل شود انتخاب می‌شود.در ادامه توابع فعالساز که به طور معمول در شبکه‌های عصبی استفاده می‌شوند، معرفی می‌گردد:



شکل (۴-۳) : تابع فعالساز تانژانت هایپربولیک

که معادله آن به صورت زیر می‌باشد:

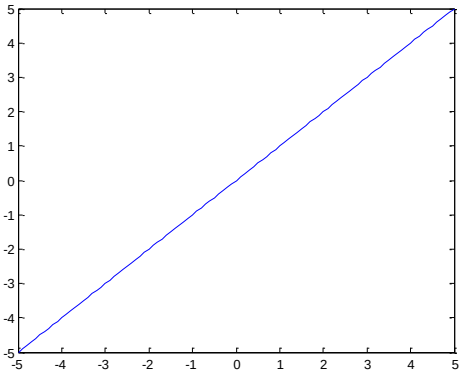
$$a = tansig(n) = 2/(1+exp(-2*n))-1 \qquad (۴-۴)$$



شکل (۴-۴) : تابع فعالساز لگاریتمی

و معادله این تابع به صورت زیر می‌باشد:

$$a=logsig(n) = 1 / (1 + exp(-n)) \qquad (۴-۵)$$



شکل (۴-۵) : تابع فعالساز خطی

۴-۵-۳- معیار های اندازه گیری (کارایی)

۴-۵-۳-۱- نرم یک MAE[1]

در این روش خطا توسط معیار متوسط قدر مطلق خطا اندازه گیری می‌شود.

$$MAE = \frac{1}{N}\sum_{i=0}^{N}|e_i| \qquad (۴-۶)$$

۴-۵-۳-۲- نرم دو SSE[2]

در این روش خطا توسط معیار جمع مربع خطا اندازه گیری می‌شود.

$$SSE = \sum_{i=0}^{N} {e_i}^2 \qquad (۴-۷)$$

۴-۵-۳-۳ MSE[3]

در این روش خطا توسط معیار متوسط مربع خطا اندازه گیری می‌شود.

$$MSE = \frac{1}{N}\sum_{i=0}^{N} {e_i}^2 \qquad (۴-۸)$$

1.Mean absolute error
2. Sum squared error
3.Mean squared error

۴-۵-۴- الگوریتم پس انتشار خطا [1]

الگوریتم پس انتشار خطا، یکی از رایج‌ترین الگوریتم‌ها جهت آموزش شبکه‌های عصبی چند لایه پیش‌خور می‌باشد. این الگوریتم تقریبی از الگوریتم بیشترین تنزل است. علی‌رغم موفقیت‌های این الگوریتم در آموزش شبکه‌های عصبی، این روش با چند مشکل اصلی روبرو است:

- الگوریتم پس انتشار خطا ممکن است به نقاط کمینه محلی همگرا شود. بنابراین زمانی که این الگوریتم همگرا می‌شود، نمی‌توان مطمئن بود که به جواب بهینه رسیده باشیم.
- سرعت همگرایی الگوریتم BP پایین است.
- همگرایی الگوریتم BP به انتخاب مقادیر اولیه وزن‌ها و بایاس‌ها وابسته است.

در شبکه‌های چندلایه پیش‌خور پرسپترون MLP از الگوریتم BP برای آموزش و تنظیم وزن‌ها و بایاس‌های شبکه استفاده می‌شود. در شبکه‌های MLP، هر نرون دارای یک تابع تحریک غیرخطی است که از ویژگی مشتق‌پذیری برخوردار است. در این حالت، ارتباط بین پارامترهای شبکه و سیگنال خطا، کاملا پیچیده و غیرخطی است. بنابراین مشتقات جزئی نسبت به پارامترهای شبکه به راحتی قابل محاسبه نیستند. جهت محاسبات مشتق از قانون زنجیره‌ای معمول در جبر استفاده می‌شود.

۴-۵-۵- فرمول بندی الگوریتم BP

الگوریتم پس انتشار خطا با معادلات زیر توصیف می‌شود:

$$W(k+1) = W(k) - \propto \frac{\partial I}{\partial W(k)} \qquad (۱۱-۴)$$

$$b(k+1) = b(k) - \propto \frac{\partial I}{\partial b(k)} \qquad (۱۲-۴)$$

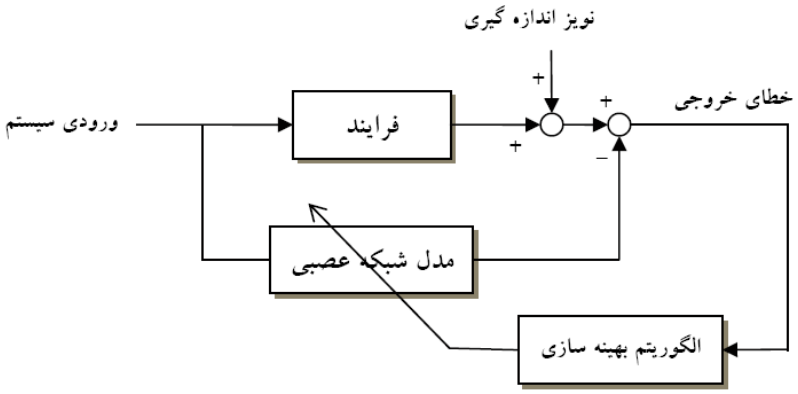
به طوریکه W پارامتر وزن شبکه، b پارامتر بایاس شبکه، α نرخ یادگیری[2] شبکه، و I تابع هزینه می‌باشد که به صورت زیر تعریف می‌گردد:

$$e = y - \hat{y} \qquad (۱۳-۴)$$

$$I = MSE = \frac{1}{N} \sum_{i=1}^{N} e_i^2 \qquad (۱۴-۴)$$

[1] Back Propagation Algorithm
[2] Learning Rate

بنابراین آموزش شبکه عصبی، یک نوع بهینه‌سازی آزاد است که در آن باید یک تابع هزینه کمینه شود و قیدی برای بهینه‌سازی وجود ندارد. ساده‌ترین روش بهینه‌سازی روش گرادیان نزولی است که به دلیل سادگی پیاده‌سازی و توانایی آموزش داده‌های آموزشی زیاد برای شبکه‌های عصبی ارجحیت دارد. در شبکه عصبی چندلایه، هر نرون در لایه مخفی، مجموع حاصلضرب اطلاعات ورودی و وزن‌های ارتباطی را محاسبه می‌کند و سپس این حاصل را با استفاده از یک تابع فعالسازی به نرون لایه بعد می‌فرستد. مقادیر محاسبه شده خروجی با مقادیر واقعی آنها مقایسه می‌شود. چنانچه مقدار خطا از خطای مطلوب که از قبل در نظر گرفته شده، متفاوت باشد به عقب بازگشته و با تغییر ضرایب ارتباطی و تکرار مراحل قبلی مجدداً خروجی‌های جدید محاسبه می‌گردد.



شکل (۴-۶) : ساختار کلی بهینه‌سازی وزن‌های شبکه عصبی

با این حال، برای بهبود روش آموزش شبکه‌های می‌توان از ضریب مومنتوم[1] در رابطه (۴-۱۱) استفاده نمود:

$$W(k+1) = m_c W(k) - \propto * (1 - m_c) \frac{\partial I}{\partial W(k)} \qquad (۱۵-۴)$$

هم چنین از روش‌های بهینه‌سازی با درجه بالاتر نیز می‌توان استفاده نمود. یکی از این روش‌ها که به وفور برای بهینه‌سازی شبکه‌های عصبی استفاده می‌شود، روش LM است که در فصل قبل ارائه گردید. بعضی از مزیت های این روش عبارتند از:

۱- سرعت بالا
۲- همگرایی مناسب
۳- مقاومت عددی

[1] Momentum constant (m_c)

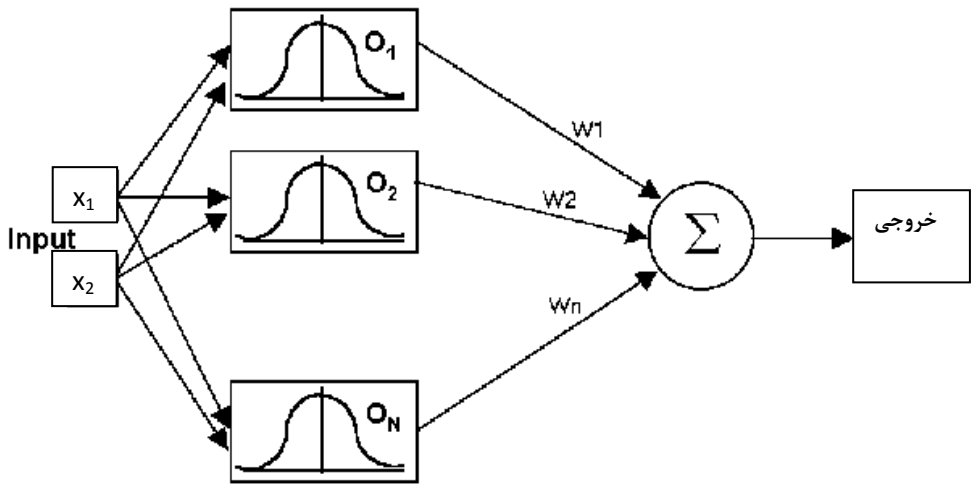
روش LM نسخه‌های متفاوتی دارد. یکی از این نسخه‌ها که در مرجع [۱۵] ارائه شده است، به صورت زیر می‌باشد:

$$\theta_{k+1} = \theta_k + \mu_k g_k \qquad (۱۶-۴)$$

از نقطه سعی و خطای جدید θ_k، نقطه جدید دیگر با حرکت به اندازه μ_k در جهت g_k بدست می‌آید.

۴-۶- شبکه‌های عصبی پایه شعاعی RBF

یکی از قدرتمندترین شبکه‌های عصبی مورد استفاده در مسائل تخمین توابع، شبکه‌های عصبی پایه شعاعی[1] RBF می‌باشند. برخلاف شبکه‌های عصبی پرسپترون که دارای لایه‌های متوالی متعدد می‌باشد، شبکه RBF از سه لایه ثابت تشکیل شده است. لایه ورودی که محل تزریق سیگنال‌های ورودی به شبکه است. لایه میانی یا لایه RBF که شامل توابع RBF می‌شود و لایه خروجی که ترکیبی خطی از کلیه خروجی‌های طبقه RBF را می‌سازد. در اکثر مواقع از توابع گوسی در لایه RBF استفاده می‌شود که این توابع دارای دو پارامتر مرکز تابع گوسی و واریانس یا میزان گستردگی تابع گوسی می‌باشد. ساختار یک شبکه RBF در شکل (۴-۷) نمایش داده شده است.



شکل (۴-۷) : ساختار کلی شبکه عصبی RBF

$$y = W^T . \emptyset\,(x) = \sum_{i=1}^{n} w_i \varphi_i(\,||x - x_i||\,) \qquad (۱۷-۴)$$

[1] Radial Basis Function

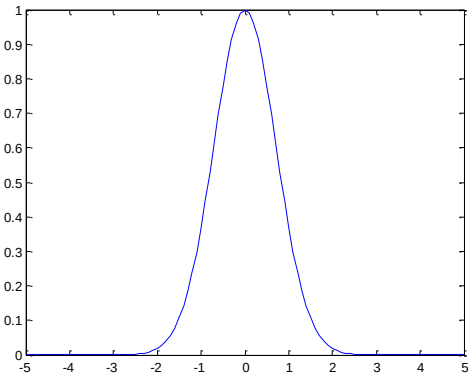
که معادله فوق، مجموع وزن‌دار توابع گوسیRBF را نشان می‌دهد. در واقع، ما خروجی y را بر اساس مقادیر توابع RBF تقریب زده‌ایم.

در معادله فوق تابع گوسی RBF به صورت زیر تعریف می‌شود:

$$\varphi_i(r) = \exp(-\frac{(r-xi)^2}{2\sigma^2}) \qquad (۱۸-۴)$$

تابع گوسی فوق توسط دو پارامتر تعیین می‌شود. مرکز تابع گوسی و واریانس یا همان گستردگی تابع گوسی. در شکل (۴-۸) تابع گوسی با مرکز صفر و انحراف معیار یک نمایش داده شده است.

$$\varphi(r) = \exp(-\frac{r^2}{2}) \qquad (۱۹-۴)$$



شکل (۴-۸) : تابع گوسی با میانگین صفر و انحراف معیار یک

در شبکه‌های عصبی RBF، سه پارامتر شبکه عصبی که باید تنظیم شوند عبارتند از:

w_i : ضرایب وزنی شبکه عصبی RBF

σ : واریانس یا پراکندگی تابع گوسی

x_i : مراکز تابع گوسین

۴-۷-شبکه‌های عصبی دینامیک

همانطور که در مقدمه عنوان گردید، برای نگاشت سیستم‌های دینامیکی، شبکه‌های استاتیک دیگر جوابگو نخواهد بود. در داده‌های اندازه‌گیری شده از یک سیستم دینامیکی ترتیب داده‌ها مهم خواهند بود، یعنی مقدار کنونی یک کمیت به مقدارهای قبلی نیز وابسته است. بنابراین در شبکه عصبی باید

تغییراتی اعمال گردد تا این وابستگی به گونه‌ای مدل شود. ایده اصلی در این مورد، تاخیرهای زمانی و پسخورد است. با استفاده از عناصر تاخیر زمانی و پسخورد در شبکه می‌توان وابستگی زمانی موجود در داده‌ها را مدل کرد و شبکه عصبی ساختار دینامیکی داشته باشد.

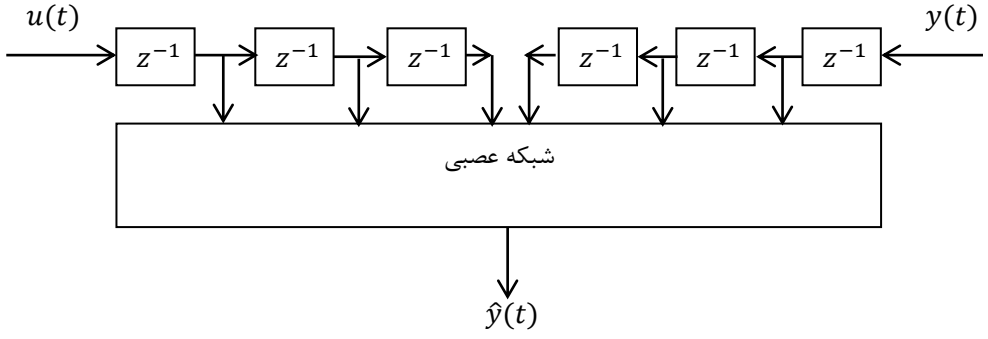
استفاده از تاخیر زمانی در شبکه عصبی به دو گونه است:

۱- استفاده از تاخیر زمانی در ساختار داخلی شبکه که منجر به شبکه‌های دینامیک داخلی می‌شود. علاوه بر این در ساختار داخلی ممکن است از پسخوردهایی نیز استفاده شود که منجر به ساختار بازگشتی می‌شود.
۲- استفاده از تاخیرهای زمانی در ساختار خارجی شبکه عصبی یا پسخورد از خروجی شبکه عصبی که شبکه‌های عصبی دینامیک خارجی را ایجاد می‌نماید.

اساس این پایان‌نامه، استفاده از شبکه‌های دینامیک خارجی می‌باشد که ساختار ساده تری نسبت به شبکه‌های دینامیک داخلی دارند.

۴-۷-۱-شبکه‌های عصبی تاخیر زمانی[1](TDNN)

در مورد مدلسازی سیستم‌های دینامیکی، ایده‌های مختلفی مطرح است. یکی از این ایده‌ها استفاده از یک سری تاخیر زمانی در ورودی شبکه است. در حقیقت، دینامیک‌های سیستم را با این تاخیرهای زمانی مدل می‌کنند. یعنی خروجی شبکه در زمان حال، تابعی از خروجی و ورودی در زمان‌های گذشته است. شبکه‌ای که به این صورت بدست می‌آید، به شبکه عصبی تاخیر زمانی معروف است که با نام TDNN بکار برده می‌شود. در شکل (۴-۹) ساختار کلی شبکه تاخیر زمانی نمایش داده شده است.



شکل (۴-۹) : شبکه عصبی تاخیر زمانی

[1] Time Delay Neural Network

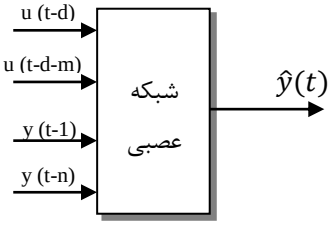
۴-۷-۲- شبکه‌های عصبی بازگشتی داخلی[1] (RNN)

ایده دیگری که مطرح است، استفاده از پسخورد در درون ساختار شبکه عصبی است. این شبکه‌ها با عنوان شبکه‌های عصبی بازگشتی داخلی معروف هستند. این شبکه‌ها گرچه دارای پیچیدگی‌هایی در آموزش می‌باشند ولی دارای مزایای قابل توجهی نسبت به شبکه‌های TDNN می‌باشند. از جمله این مزایا این است که چون تاخیر زمانی در ورودی شبکه استفاده نمی‌شود، ابعاد فضای ورودی کوچکتر می‌شود و همچنین نیاز به دانستن درجه سیستم نمی‌باشد. شبکه‌های RNN به طوری عمل می‌کنند که خروجی شبکه به ورودی‌ها و خروجی‌های قبلی شبکه وابسته است و این همان ایده اساسی در مدلسازی سیستم‌های دینامیکی است.

در شبکه‌های TDNN و RNN بسته به اینکه از چه متغیرهایی به عنوان ورودی استفاده کینم، ساختارهای مختلفی بدست می‌آید که با عناوین مختلفی شناخته می‌شود.

- شبکه NNARX: بردار رگرسور ورودی‌ها در این مدل شبکه عصبی به صورت (۴-۱۹) است و ساختار این شبکه در شکل (۴-۱۰) نمایش داده شده است.

$$X(k) = [u(k-1), \dots, u(k-m),\ y(k-1), \dots, y(k-n)] \qquad (۴-۲۰)$$

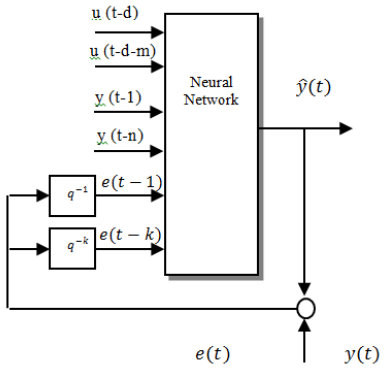


شکل (۴-۱۰) : شبکه عصبی NNARX

- شبکه NNARMAX: بردار رگرسور ورودی‌ها در این مدل شبکه عصبی به صورت (۴-۲۰) است و ساختار این شبکه در شکل (۴-۱۱) نمایش داده شده است.

$$X(k) = [u(k-1), \dots,\ u(k-m), y(k-1), \dots, y(k-n), e(t-1), \dots, e(t-c)] \qquad (۴-۲۱)$$

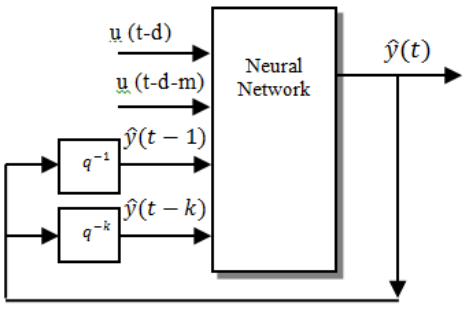
[1] Recurrent neural network



شکل (۴-۱۱) : شبکه عصبی NNARMAX

- شبکه NNOE: بردار رگرسور ورودی‌ها در این مدل شبکه عصبی به صورت (۴-۲۱) است و ساختار این شبکه در شکل (۴-۱۲) نمایش داده شده است. همان‌طور که مشاهده می‌شود از خروجی مدل پسخورد گرفته می‌شود و به ورودی اعمال می‌گردد.

$$X(k) = [u(k-1), \dots, u(k-m), \hat{y}(k-1), \dots, \hat{y}(k-n)] \qquad (۴-۲۲)$$



شکل (۴-۱۲) : شبکه عصبی NNOE

نکته: شبکه NNARX به خاطر وجود عناصر تاخیر و نبود پسخورد در ساختار این شبکه، جز شبکه‌های تاخیر زمانی و NNARMAX و NNOE به خاطر پسخورد گرفتن از خروجی خود مدل، جز شبکه‌های بازگشتی محسوب می‌شود.

بر اساس نوع اتصالات داخلی، شبکه‌های عصبی بازگشتی داخلی به سه نوع تقسیم می‌شوند:

۴-۷-۲-۱-شبکه‌های عصبی کاملاً بازگشتی

در این نوع شبکه ارتباط بین هر دو نرون معرف یک حالت داخلی شبکه است. به عبارت دیگر، علاوه بر ارتباط‌های پیش‌خور، تمام نرون‌های لایه پنهان و خروجی با ارتباط‌های تاخیر زمانی نیز به هم

متصلند. این ساختار از لحاظ پیاده‌سازی عملی دارای مشکلات فراوانی می‌باشد و در عمل از این نوع شبکه در شناسایی سیستم‌های غیرخطی کمتر استفاده می‌شود.

۴-۷-۲-۲-شبکه‌های عصبی به طور جزئی بازگشتی

این شبکه‌ها بر اساس شبکه‌های MLP و با یک لایه اضافی به نام لایه زمینه[1] می‌باشد. نرون‌های لایه زمینه به عنوان حافظه داخلی عمل می‌کنند. شبکه المن[2] و جردن[3] به عنوان شبکه‌های بازگشتی جزئی شامل پسخورد از لایه پنهان و خروجی به نرون‌های لایه زمینه می‌باشد. همان‌طور که ذکر گردید، عناصر تاخیر q^{-1} با مربوط کردن خروجی نرون به حالت‌های آن در زمان‌های گذشته باعث ایجاد ساختار بازگشتی می‌شود.

۴-۷-۲-۳- شبکه‌های به طور محلی بازگشتی و به طور سراسری پیش خوران

این شبکه‌ها بر اساس شبکه‌های استاتیک پیش خور بنا گذاشته شده است و با بازگشت‌پذیری‌های محلی گسترش یافته است. به این معنا که هیچ پسخورد جانبی بین نرون های یک لایه و هم چنین نرون‌های لایه‌های مختلف وجود ندارد. این بازگشتی‌ها صرفاً به سیناپس‌ها و هم‌چنین خود نرون‌ها مربوط می‌شود.

۴-۸- آموزش شبکه عصبی توسط الگوریتم ژنتیک

تحقیقات در زمینه الگوریتم ژنتیک و شبکه‌های عصبی در اواسط دهه ۸۰ میلادی آغاز شد و اکنون می‌توان بیش از ۳۲۰ مرجع یافت[۱۷]. اغلب روش های آموزش شبکه‌های عصبی بر پایه الگوریتم‌های گرادیانی و پس انتشار خطا برای یافتن پارامترهای شبکه عصبی از جمله وزن‌ها و بایاس‌ها هستند. متاسفانه پس انتشار خطا یک الگوریتم بهینه‌سازی محلی است و احتمال به دام افتادن در کمینه محلی زیاد است.

محققین برای رفع ایراد های این روش راهکارهای مختلفی را پیشنهاد کرده‌اند:

[1] Contex layer
[2] Elman
[3] Jordan

- تنظیم پارامترهای الگوریتم به گونه‌ای که مومنتم[1] جستجو باعث گذر از مینیمم‌های محلی گردد.
- شروع مجدد جستجو از نقاط متعدد تصادفی
- روش بهینه‌سازی تبرید شبیه‌سازی شده[2]

تبرید فرآیندی است برای رساندن فلزات به کمترین سطح انرژی و در نتیجه مقاومت بالا که طی آن ابتدا فلز تا دمای بالایی نزدیک به نقطه ذوب گرم می‌شود و آنگاه به آرامی و به تدریج خنک می‌شود. اکثر پیوندهای اتمی فلزات در دمای بالا از هم گسیخته می‌شود و در حین سرد شدن از آنجایی که این روند به آهستگی صورت می‌گیرد، ذرات فرصت کافی برای مرتب شدن و قرار گرفتن در مکان‌هایی با حداقل انرژی را دارند. توسعه مدل ریاضی این روش ابزار مناسبی برای مسئله بهینه‌سازی در آموزش شبکه عصبی است.

- الگوریتم ژنتیک

تاکنون نتایج مثبتی از بکارگیری الگوریتم ژنتیک در آموزش [۱۸و۱۷] و طراحی ساختار شبکه‌های عصبی [۱۹] بدست آمده است. در مقایسه ای بین روش‌های ژنتیک، تبرید و پس انتشار خطا برتری الگوریتم ژنتیک به اثبات رسیده است.

کاربردهایی که تا کنون برای الگوریتم ژنتیک در شبکه‌های عصبی ارائه شده است عبارتند از:

- جستجو برای بهینه‌سازی وزن‌ها و آموزش شبکه عصبی

با کد کردن پارامترهای وزنی و بایاس‌های شبکه به صورت باینری یا مقادیر اصلی و اعمال الگوریتم ژنتیک می‌توان به جواب بهینه سراسری رسید و شبکه را آموزش داد. از این روش برای شناسایی نیز استفاده شده است [۱۸]. ایراد این روش نیاز به تکرار زیاد برای همگرایی الگوریتم است که این روش برای شناسایی برخط مناسب نخواهد بود.

- جستجو برای بهینه‌سازی ساختار شبکه عصبی

الگوریتم ژنتیک با جمعیتی از شبکه‌های عصبی کامل با تعداد نرون‌های متفاوت در لایه پنهان ابعاد شبکه عصبی را بهینه می‌کند. این روش را می‌توان با روش های دیگر که وزن های اضافی را حذف و شبکه را هرس می‌کند ادغام نمود[۲۰].

- بهینه‌سازی همزمان پارامتر و ساختار شبکه [۱۷].
- استفاده از الگوریتم ژنتیک که الگوریتم پس انتشار خطا را بهینه می‌کند[۲۰].
- ترکیب الگوریتم ژنتیک و الگوریتم پس انتشار خطا به صورت سری

[1] momentum
[2] Simulated Annealing

روشی ترکیبی که از توانایی الگوریتم ژنتیک در پیدا کردن جوابی نزدیک به کمینه سراسری و قابلیت الگوریتم پس انتشار خطا در تنظیم دقیق و نهایی جواب استفاده می‌کند[۲۰]. این روش از سرعت بهتری نسبت به الگوریتم ژنتیک برخوردار است.

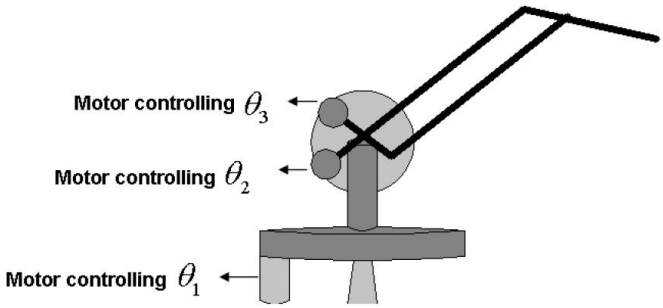
فصل پنجم:

معرفی ربات فانتوم

۵–۱–ربات فانتوم

ربات فانتوم، ربات دارای سه رابط با سه درجه آزادی[1] می‌باشد[۲۳]. شکل (۵-۱) و (۵-۲) طرح کلی ربات فانتوم را نشان می‌دهد که دارای سه موتور در سه مفصل می‌باشد.

شکل(۵-۱): ربات فانتوم



شکل(۵-۲): ربات فانتوم و زوایای مربوطه [۲]

این ربات توسط شرکت سنسیبل تکنولوژی[2] به صورت تجاری تولید شد. در این پروژه برای آزمایشات مربوطه و ثبت داده‌ها از ربات پژوهشگاه رباتیک دانشگاه امیرکبیر استفاده شد.

[1] three degrees-of-freedom (DOF)
[2] SensAble Technology

شکل(۵-۳): ربات فانتوم پژوهشگاه رباتیک دانشگاه امیرکبیر

این ربات به طور گسترده در کاربردهای لامسه‌ای[1] و عملیات از راه دور[2] استفاده می‌شود. به طور کلی واسط‌های لامسه‌ای ابزارهایی هستند که به کاربر اجازه می‌دهند تا بتواند با محیط مجازی و یا سیستم‌های عملکرد از راه دورارتباط برقرار کند. توانایی شناسایی یا درک لامسه‌ای در واقع یک عمل حسگری می‌باشد که می‌تواند خصوصیات و ویژگی‌های سطوح و احجام را برای انسان معین کند.

[1] Haptic
[2] Telerobotics

مدل‌های دینامیکی دقیق این ربات برای کنترل و شبیه‌سازی ربات اهمیت دارد. در [۱] با استفاده از روش اویلر-لاگرانژ، ساختار و معادلات مربوط به حرکت ربات فانتوم به صورت زیر ارائه شده است. معادلات دینامیکی و حرکت این ربات به صورت رابطه (۵-۱) و (۵-۲) می‌باشد.

$$\tau = M(\theta)\ddot{\theta} + C\left(\theta, \dot{\theta}\right)\dot{\theta} + N(\theta) \qquad (۵-۱)$$

$$\begin{bmatrix}\tau_1\\ \tau_2\\ \tau_3\end{bmatrix} = \begin{bmatrix}M_{11} & 0 & 0\\ 0 & M_{22} & M_{23}\\ 0 & M_{32} & M_{33}\end{bmatrix}\begin{bmatrix}\ddot{\theta_1}\\ \ddot{\theta_2}\\ \ddot{\theta_3}\end{bmatrix} + \begin{bmatrix}C_{11} & C_{12} & C_{13}\\ C_{12} & 0 & C_{23}\\ C_{13} & C_{32} & 0\end{bmatrix}\begin{bmatrix}\dot{\theta}_1\\ \dot{\theta}_2\\ \dot{\theta}_3\end{bmatrix} + \begin{bmatrix}0\\ N_2\\ N_3\end{bmatrix} \qquad (۵-۲)$$

به طوری که M و C و N به ترتیب ماتریس اینرسی[1]، ماتریس پیچشی و گریز از مرکز[2]، و بردار گرانشی[3] را نمایش می‌دهند.

در معادله $\tau = [\tau_1 \quad \tau_2 \quad \tau_3]^2$ بردار گشتاور ورودی بوده که به سه رابط ربات اعمال می‌شود و $\theta = [\theta_1 \quad \theta_2 \quad \theta_3]^2$ بردار زاویه خروجی بوده و توسط اینکدرهای نصب شده در مفصل‌ها اندازه‌گیری می‌شود.

[1] Inertial Matrix
[2] Coriolis and Centrifugal Matrix
[3] Gravitational Vector

فصل ششم:

شناسایی سیستم دینامیکی

ربات فانتوم

۶–۱– مقدمه

در این بخش به شناسایی، شبیه‌سازی و تحلیل ربات فانتوم می‌پردازیم. مراحل مختلف جهت انجام این پروژه به شرح زیر می‌باشد:

- مرحله آماده‌سازی و ثبت داده‌ها: در این مرحله، در پژوهشکده رباتیک و مکاترونیک دانشگاه صنعتی امیرکبیر آزمایشات مربوطه جهت ثبت داده‌های مورد نظر، روی ربات فانتوم انجام پذیرفت.

- مرحله طراحی ساختار شبکه عصبی: در این مرحله، با توجه به بررسی‌ها ومطالعات انجام پذیرفته روی ربات، ساختار شبکه عصبی مناسب برای شناسایی طراحی گردید. با توجه به اینکه ربات مذکور دارای رفتار دینامیکی غیرخطی می‌باشد، از شبکه‌های عصبی بازگشتی که در ساختار خود دارای تاخیر زمانی و پسخورد می‌باشند، استفاده شده است. همچنین شبکه عصبی طراحی شده دارای ۲ لایه میانی، که دارای توابع تحریک تانژانت هایپربولیک در لایه اول و تابع تحریک خطی در لایه پایانی می‌باشد، استفاده شده است.

- مرحله شبیه‌سازی و شناسایی در نرم افزار MATLAB : در این مرحله، به کمک داده‌های ثبت شده در آزمایشگاه، به شناسایی ربات پرداخته شده است. ساختارهای مختلف جهت شناسایی توسط شبکه عصبی که در بخش قبل طراحی شده بود، در این مرحله در نرم افزار MATLAB پیاده‌سازی شده و به مقایسه و تحلیل روش‌های پیشنهادی پرداخته شده است.

۶–۲– ورودی‌ها و خروجی‌های ربات فانتوم و شبکه عصبی پیشنهادی

ربات فانتوم، رباتی با سه درجه آزادی و با کاربردهای رباتیک از راه دور و عملیات حسی می‌باشد. ربات فانتوم را به عنوان یک جعبه سیاه در نظر گرفته و آزمایشات مورد نیاز جهت شناسایی را انجام می‌دهیم. لینک ارتباطی بین کامپیوتر و بازو، کارت IEEE1394 بوده و از محیط سیمولینک MATLAB برای شبیه‌سازی و ثبت نتایج استفاده شده است.

این ربات دارای سه رابط بوده، ورودی به هر رابط گشتاور بوده که به موتورهای هر مفصل اعمال می‌شود. در واقع گشتاور به صورت سیگنال تحریک پایا به هر رابط اعمال می‌گردد.

خروجی ربات نیز اندازه زاویه هر رابط نسبت به مبنای اندازه‌گیری خود می‌باشد. به طور خلاصه، $\tau = [\tau_1 \quad \tau_2 \quad \tau_3]^T$ بردار گشتاور ورودی بوده که به سه رابط ربات اعمال می‌شود و $\theta = [\theta_1 \quad \theta_2 \quad \theta_3]^T$ بردار زاویه خروجی بوده و توسط اینکدرهای نصب شده در مفصل‌ها اندازه‌گیری می‌شود.

در مرحله تولید داده‌ها حدود ۶۰/۰۰۰ نمونه تولید شد که ۷۰ درصد از داده‌ها برای آموزش و باقیمانده برای آزمایش و اعتبارسنجی شبکه استفاده می‌شود.

در مرحله بعد به طراحی ساختار شبکه عصبی می‌پردازیم. در طراحی ساختار و معماری شبکه عصبی، تعداد ورودی‌ها و خروجی‌ها و نوع آنها از سیستم مورد نظر مشخص می‌شود و با انتخاب طراح نیست. اما تعداد لایه‌های پنهان، تعداد نرون‌ها، نوع توابع فعالساز در دست طراح است و بنابراین یک طراحی بهینه برای ساختار شبکه عصبی ضروری می‌باشد. با مطالعه انجام گرفته، روش و فرمول دقیقی برای تعیین موارد ذکر شده وجود ندارد، با این وجود برخی قواعد کلی برای طراحی ساختار وجود دارد که به آن اشاره می‌شود. طبق [۲۲] یک شبکه عصبی با یک لایه پنهان و تعداد کافی نرون در لایه پنهان قادر است هر تابعی را با دقت دلخواه تقریب بزند. در عمل، شبکه‌های عصبی با یک یا در مواردی دو لایه پنهان به طور وسیعی بکار می‌روند و عملکرد بسیار خوبی دارند. بنابراین شبکه عصبی با یک یا دو لایه پنهان می‌تواند نتایج مطلوبی را ارائه دهد. اکثر مطالعات تجربی بیان می‌کنند که استفاده از شبکه‌هایی با بیش از دو لایه پنهان، بهبودی در نتایج ایجاد نخواهد کرد.

همچنین، با وجود اهمیت تعداد نرون‌های لایه پنهان، فرمول دقیقی برای تعیین آن نیست. با این وجود قانون‌های کلی و تقریبی برای تعیین تعداد نرون‌های لایه پنهان وجود دارد. یکی از این قواعد استفاده از رابطه $x = \sqrt{n \times m}$ می‌باشد که n تعداد نرون‌های ورودی و m تعداد نرون‌های خروجی می‌باشد. محدوده واقعی تعداد نرون‌ها می‌تواند از نصف مقدار x تا دو برابر آن باشد.

توابع فعالساز روابط ریاضی هستند که خروجی یک نرون را تعیین می‌کنند. اغلب شبکه‌های عصبی از توابع فعالساز تانژانت هایپربولیک، لگاریتمی و خطی استفاده می‌نمایند. برای انتخاب توابع فعالساز نیز باید به نکات زیر توجه نمود. درصورتیکه نگاشت بین ورودی و خروجی غیرخطی باشد، توابع فعالساز خطی در لایه‌های میانی مناسب نمی‌باشند. معمولاً در صورتیکه با حالت غیرخطی روبرو باشیم توابع فعالساز تانژانت هایپربولیک جواب مطلوبی خواهد داشت.

ما در این پروژه با توجه به قواعد موجود و شبیه‌سازی‌های مختلف، به ساختار شبکه عصبی مطلوب رسیدیم. شبکه عصبی که انتخاب گردید، شبکه با دو لایه، پنج نرون در لایه میانی با تابع فعالساز تانژانت، و یک نرون با تابع فعالساز خطی در لایه آخر می‌باشد. شبکه عصبی مورد نظر با داده‌های اندازه‌گیری شده در آزمایشگاه آموزش داده شد و با سه معیار ارزیابی، عملکرد روش‌های پیشنهادی مورد بررسی قرار گرفت که به شرح زیر می‌باشد:

- مجموع مربعات خطا[1] (SSE) که در بخش ۳-۵-۴ به آن اشاره شد.
- میانگین مربعات خطای نسبی[2] (RMSE): جهت محاسبه RMSE از رابطه (۶-۱) استفاده می‌نماییم:

$$RMSE = 100 * \frac{\sum_i (y_i - \hat{y}_i)^2}{\sqrt{\sum_i (y_i)^2}\sqrt{\sum_i (\hat{y}_i)^2}} \quad (۱-۶)$$

- همبستگی متقابل[3] بین خروجی واقعی و خروجی مدل (CC): همبستگی متقابل، معیاری است که میزان تشابه دو سیگنال را نشان می‌دهد. همبستگی متقابل نزدیک به ۱۰۰، تشابه بیشتر بین دو سیگنال را نشان می‌دهد[۱۴]. جهت محاسبه همبستگی متقابل از رابطه (۶-۲) استفاده می‌نماییم.

$$CC = 100 * \frac{\sum_i y_i * \hat{y}_i}{\sqrt{\sum_i (y_i)^2}\sqrt{\sum_i (\hat{y}_i)^2}} \quad (۲-۶)$$

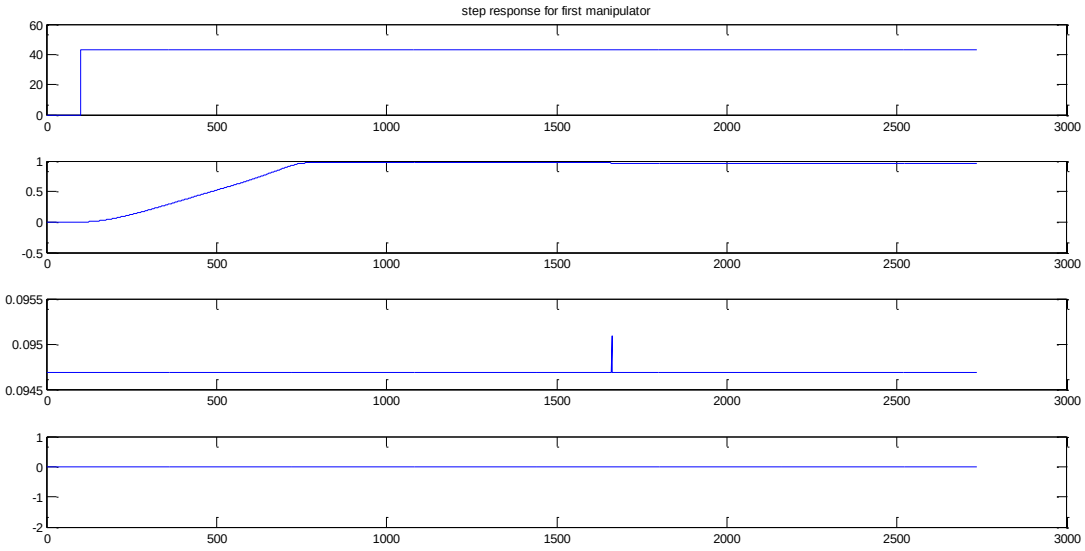
در مرحله نهایی به اعتبارسنجی شبکه آموزش دیده می‌پردازیم تا تحلیلی درباره دقت شبکه داشته باشیم.

در تمامی مدل‌های پیشنهادی معیار خاتمه الگوریتم تعداد ۱۰۰ تکرار می‌باشد. یعنی پس از ۱۰۰ تکرار، معیارهای ذکر شده در بالا محاسبه می‌شود.

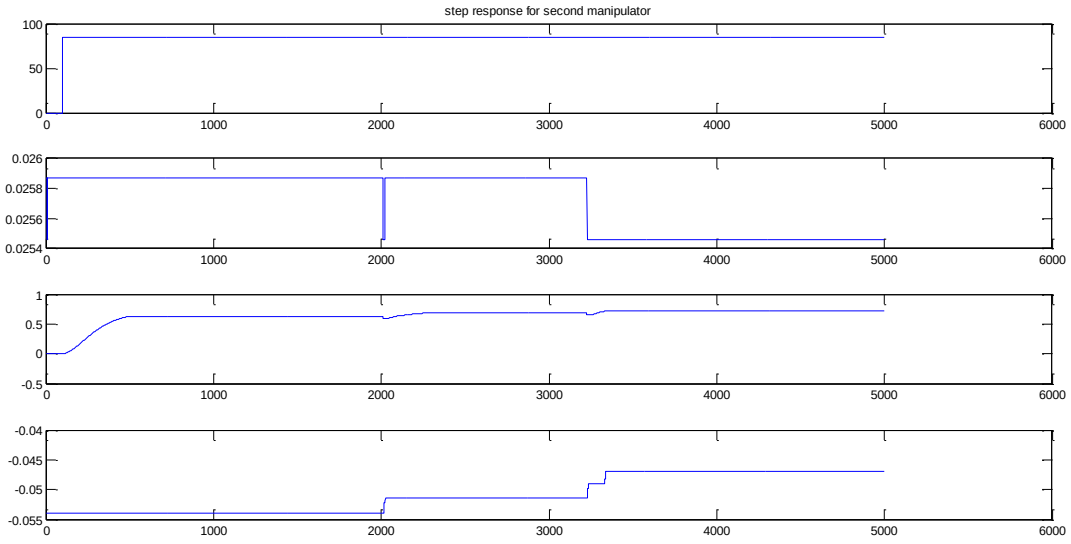
۶-۳- بررسی پاسخ پله هر یک از رابطها

در این بخش پاسخ هر یک از رابطها به ورودی پله ارائه شده است. به هر یک از رابطها به صورت مستقل، ورودی پله اعمال شده و خروجی‌های هر رابط ثبت می‌شوند. شکل‌های (۶-۱) و (۶-۲) و (۶-۳) پاسخ پله هر یک از رابطها را به صورت مستقل نشان می‌دهد. ورودی رابطها گشتاور بر حسب میلی نیوتون متر می‌باشد و خروجی زاویه هر یک از رابطها بر حسب رادیان می‌باشد.

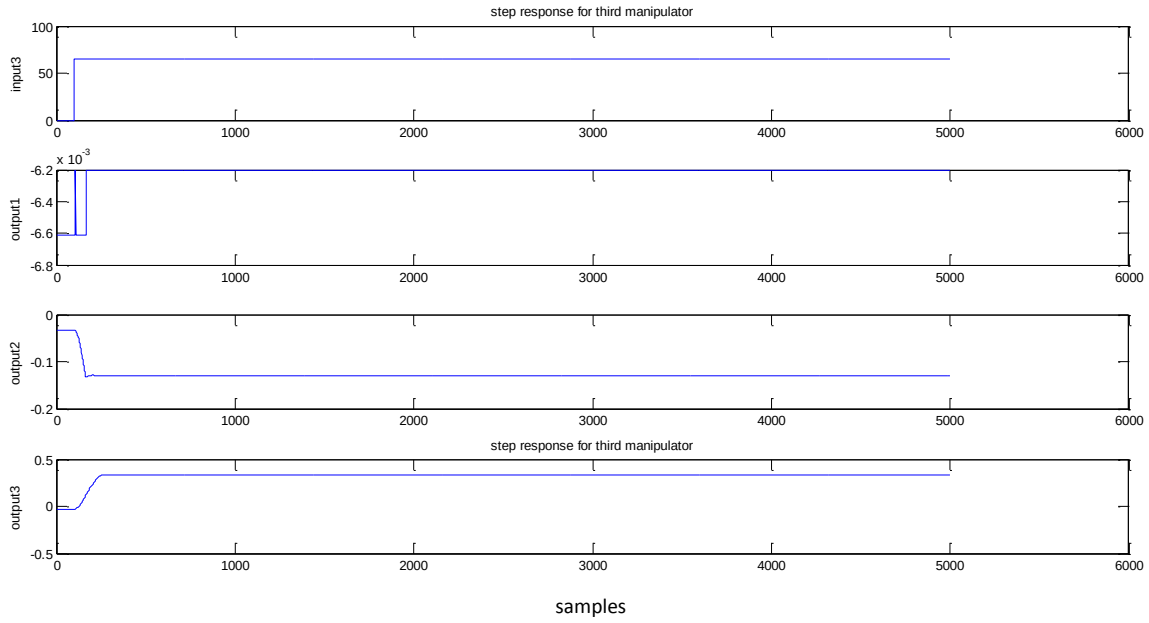
[1] Sum Square Error
[2] Relative Mean Square Error
[3] Cross correlation



شکل (۶-۱): بررسی پاسخ پله به رابط اول



شکل (۶-۲): بررسی پاسخ پله به رابط دوم



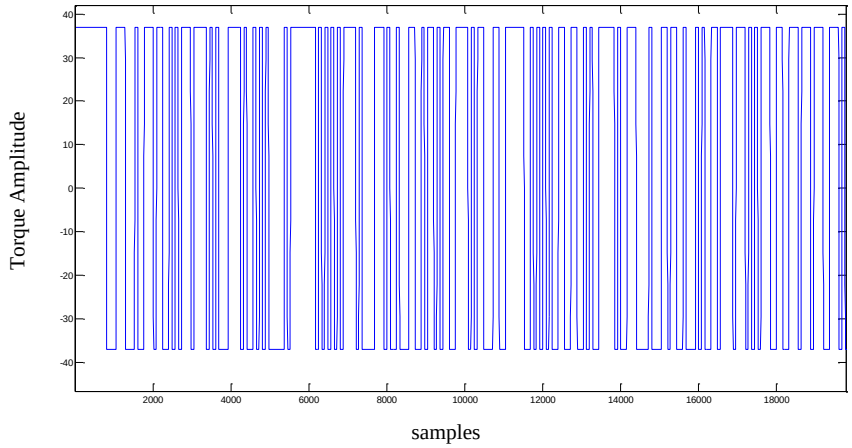
شکل (۶-۳): بررسی پاسخ پله به رابط سوم

۶-۴- سیگنال تحریک برای شناسایی ربات فانتوم

یکی از مراحل مهم در شناسایی سیستم، بحث طراحی سیگنال تحریک مناسب برای جمع آوری داده می‌باشد. این موضوع برای سیستم‌های غیرخطی نسبت به سیستم‌های خطی از درجه اهمیت بالاتری برخوردار است. زیرا این سیستم‌ها رفتار دینامیکی پیچیده‌ای دارند و بنابراین داده‌ها باید دارای اطلاعات بیشتری باشند. برای سیستم‌های خطی، سیگنال شبه تصادفی باینری[1] (PRBS)، مورد استفاده قرار می‌گیرد. این نوع سیگنال‌ها تمام فرکانس‌های کاری سیستم مورد نظر را در بردارند تا بتواند تمام مودهای سیستم را تحریک نماید. برای سیستم‌های غیرخطی، در کنار ویژگی‌های فرکانسی سیگنال تحریک، دامنه سیگنال نیز باید به درستی انتخاب گردد. یک راه حل برای این موضوع، توسعه سیگنال PRBS به دامنه‌های مختلف است، به طوریکه در هر گام، به سیگنال PRBS یک دامنه خاص اختصاص دهیم و در این حالت، سیگنال PRBS با مدولاسیون دامنه[2] داریم.

در ادامه سیگنال‌هایی که برای شناسایی ربات فانتوم در آزمایشگاه انتخاب گردید، اشاره می‌شود.

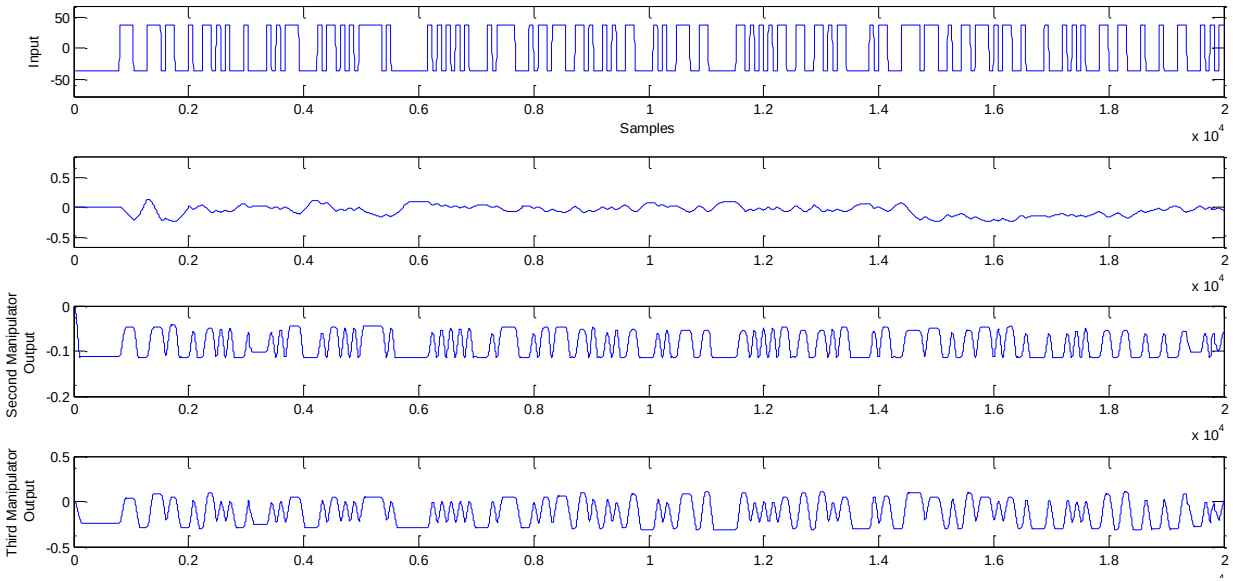
[1] Pseudo Random Binary Signal
[2] Amplitude Modulated PRBS



شکل (۶-۴): سیگنال PRBS اعمالی به ربات برای شناسایی

همان‌طور که در شکل (۶-۴) مشاهده می‌شود، سیگنال PRBS بین دوسطح خاص و با فرکانس‌های مختلف نوسان می‌کند.

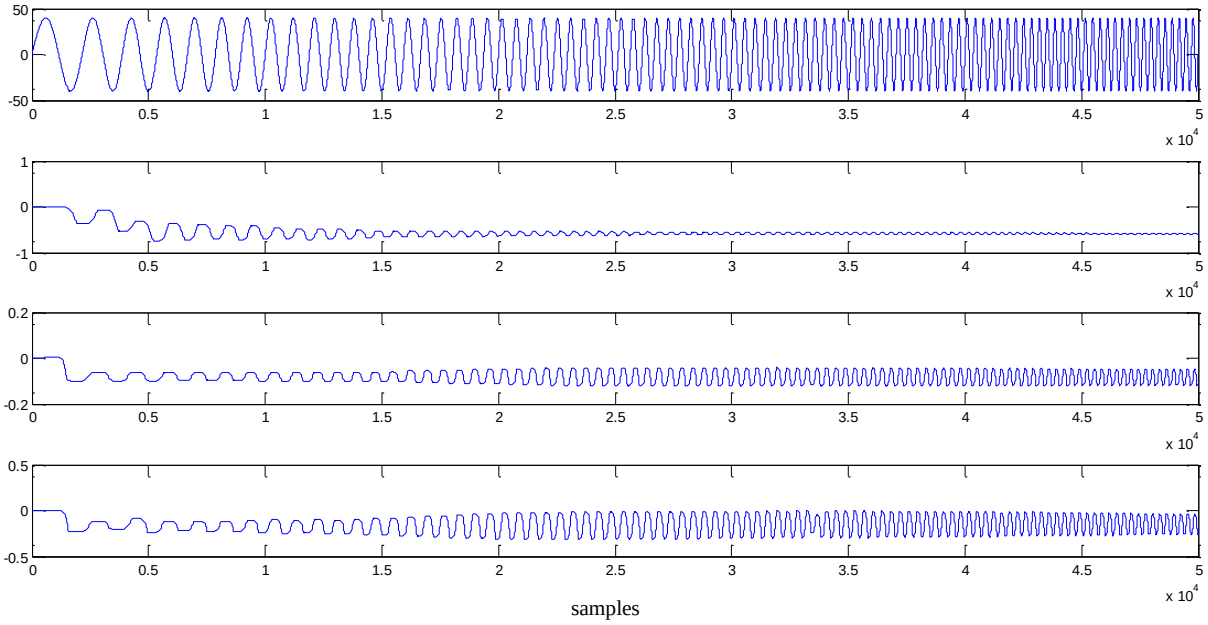
در شکل (۶-۵) ورودی به هر رابط و خروجی‌های هر رابط نمایش داده شده است.



شکل (۶-۵): ورودی و خروجی‌های سه رابط ربات

سیگنال دیگری که جهت شناسایی سیستم‌ها رایج است، سیگنال چیرپ[1] است. این سیگنال، یک سیگنال سینوسی با جاروب فرکانسی است. به عبارت دیگر، فرکانس سیگنال سینوسی، به صورت خطی با زمان افزایش می‌یابد.

[1] Chirp



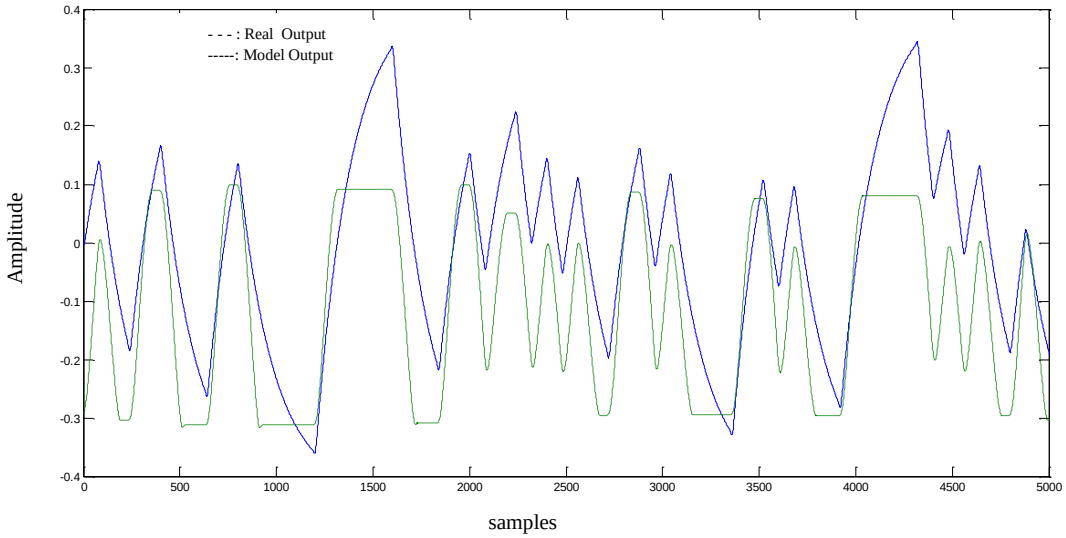
شکل (۶-۶): ورودی و خروجی‌های سه رابط ربات به ازای ورودی CHIRP

۶-۵- شبیه‌سازی ربات با روش های مختلف شناسایی

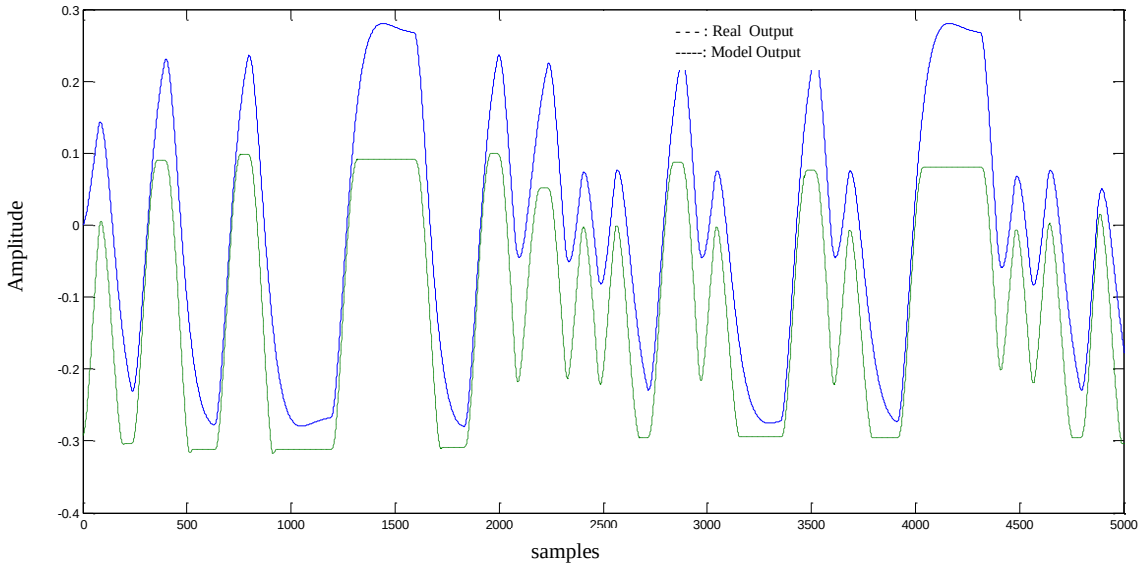
در این بخش به شناسایی ربات فانتوم توسط داده‌های اندازه‌گیری شده می‌پردازیم. روش کار بدین صورت است که در ابتدا از روش‌های معمول شناسایی ARX، ARMAX، OE استفاده می‌نماییم و کارایی آنها مورد بررسی قرار می‌گیرد. در ادامه، روش شناسایی توسط شبکه عصبی استاتیک و در نهایت شبکه‌های عصبی دینامیک مورد استفاده قرار می‌گیرد. نتایج جهت مقایسه در جدول (۶-۱) جمع‌آوری می‌گردد. لازم به توضیح است رابط سوم را جهت مقایسه در نظر گرفته و روش‌های مختلف را روی آن اعمال می‌کنیم.

۶-۵-۱- شناسایی به روش رایج

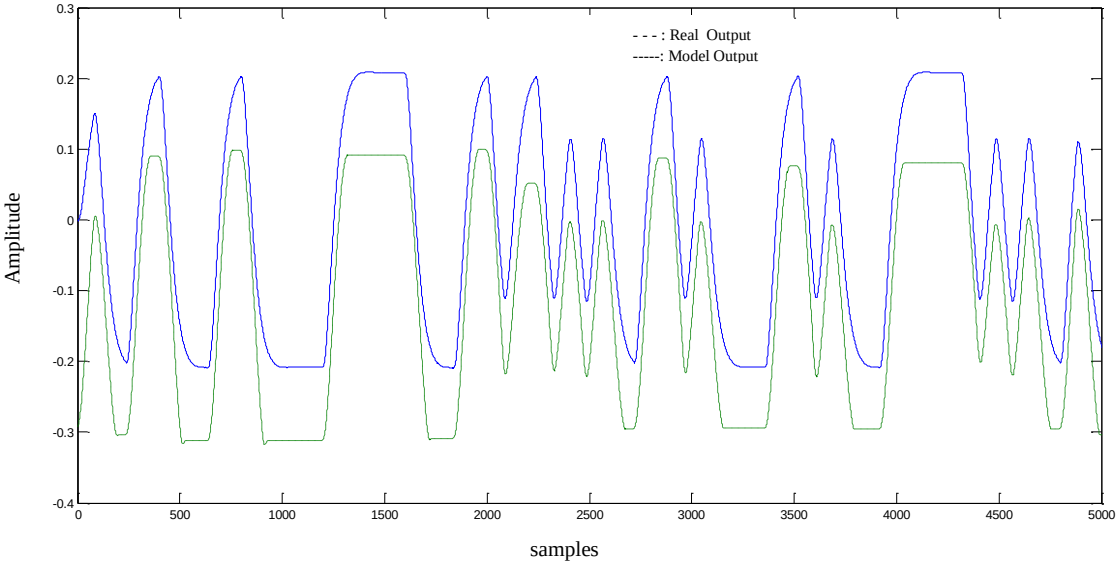
نتایج حاصل از روش های رایج شناسایی کلاسیک در شکل‌های زیر نمایش داده شده است و معیارهای MSE وRMSE و CC برای روش‌های مختلف در جدول (۶-۱) ارائه شده است.



شکل (۶-۷): شناسایی توسط روش ARX

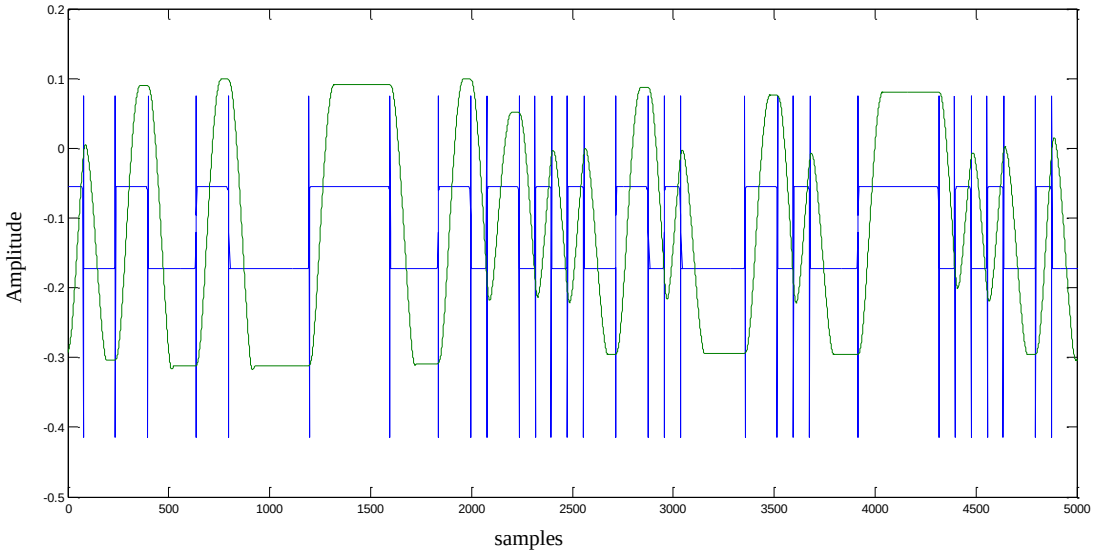


شکل (۶-۸): شناسایی توسط روش ARMAX



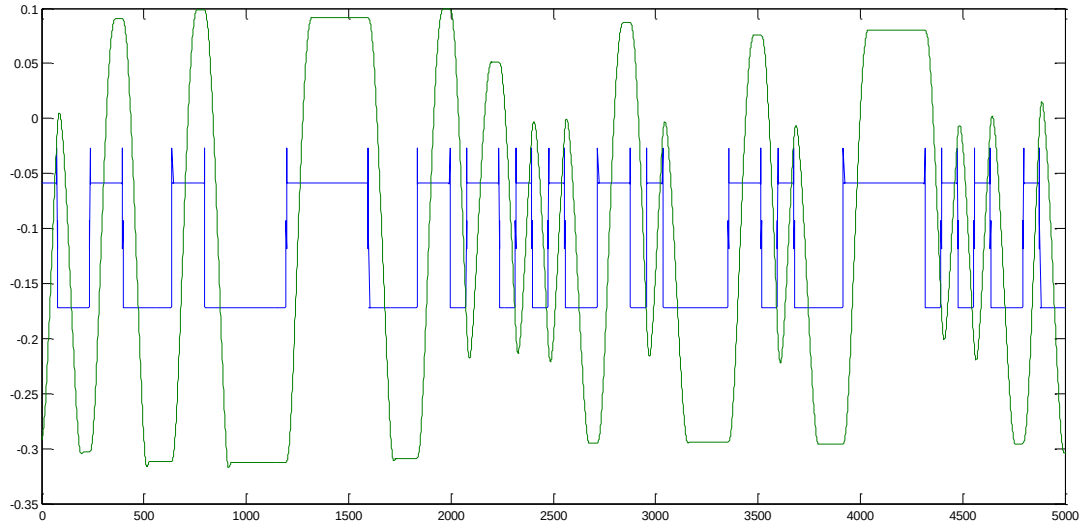
شکل (۶-۹): شناسایی توسط روش OE

همانطور که مشاهده می‌شود، روش‌های فوق توانایی دنبال کردن خروجی واقعی را دارند، اما دقت مناسبی ندارند. در ادامه توسط شبکه‌های عصبی استاتیک با یک لایه میانی که دارای پنج نرون با توابع فعالساز تانژانت هایپربولیک می‌باشد و لایه خروجی دارای تابع فعالساز خطی می‌باشد، ربات را شناسایی می‌نماییم. در ساختار این نوع شبکه، هیچ پسخورد بین لایه‌های شبکه وجود ندارد. یک مثال برای این شبکه‌ها، شبکه MLP می‌باشد.



شکل (۶-۱۰): شناسایی توسط شبکه عصبی استاتیک MLP

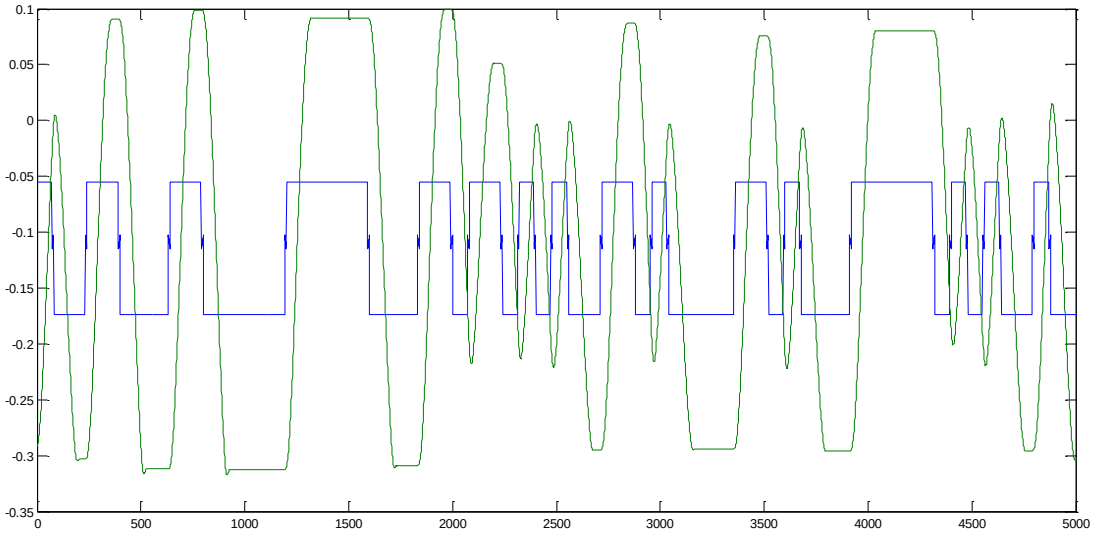
تعداد نرون‌های لایه میانی : ۵، نرخ آموزش: ۰.۰۵، روش آموزش : گرادیان نزولی، تعداد تکرار:۱۰۰



شکل (۶-۱۱): شناسایی توسط شبکه عصبی استاتیک MLP با ضریب مومنتوم

تعداد نرون‌های لایه میانی : ۵، نرخ آموزش: ۰.۵، ضریب مومنتوم: ۰.۵، روش آموزش : گرادیان نزولی با مومنتوم

تعداد تکرار:۳۰۰



شکل (۶-۱۲): شناسایی توسط شبکه عصبی استاتیک MLP با روش آموزش LM

تعداد نرون‌های لایه میانی : ۵، نرخ آموزش: ۰.۰۵، روش آموزش : LM، تعداد تکرار:100

جدول(۶-۱): نتایج شناسایی توسط شبکه عصبی با ورودی PRBS و روش‌های رایج شناسایی

	MSE	RMSE%	CC%
ARX	0.0215	68.0075	66.0477
ARMAX	0.0153	48.1379	75.9656
OE	0.0122	44.8251	79.2376
MLP with gradient	0.0185	78.5187	66.0621
MLP with momentum	0.0178	77.8209	67.3858
MLP with LM	0.0176	76.5640	67.9008
NNARX	2.5473e-005	0.0781	99.9610

۶-۶- شبیه‌سازی ربات با ورودی PRBS

در ادامه به ارائه شبیه‌سازی انجام گرفته توسط شبکه‌های عصبی بازگشتی در نرم افزار MATLAB می‌پردازیم.

شبکه‌های دینامیک:

در ساختار این نوع شبکه بین لایه‌های مختلف شبکه، تاخیر زمانی و پسخورد وجود دارد. همچنین نرون‌ها به صورت محلی دارای پسخورد می‌باشند. یک نمونه از این شبکه‌ها شبکه NNARX می‌باشد که از لحظات قبل خروجی و ورودی برای تنظیم مولفه‌های شبکه عصبی استفاده می‌کند.

در سیستم‌های دینامیکی، خروجی در هرلحظه، به ورودی در همان لحظه و تاریخچه ورودی و خروجی وابسته است. در حالت مدلسازی فیزیکی برای سیستم‌های دینامیکی یک معادله دیفرانسیل مرتبه اول پدید می‌آید.

تمرکز اصلی در این پایان‌نامه، استفاده از شبکه‌های عصبی دینامیک به عنوان یک ساختار کلی برای شناسایی سیستم‌های غیرخطی است.

۶-۶-۱-نتایج شبیه‌سازی توسط شبکه عصبی NNARX

رابطه بین ورودی و خروجی برای یک سیستم غیرخطی را به صورت زیر در نظر می‌گیریم:

$$y(k) = f(X(k)) \qquad (۶-۳)$$

که x(k) بردار رگرسور است. در بردار رگرسور از گذشته خروجی(مدل یا سیستم واقعی) برای پیش‌بینی خروجی در زمان های بعد استفاده می‌شود. f تابع تخمین است که می‌تواند یک چندجمله‌ای، شبکه عصبی و یا شبکه فازی باشد. در این پروژه برای تخمین تابع f از شبکه‌های عصبی دینامیکی استفاده می‌کنیم. برای شناسایی ربات توسط شبکه عصبی مصنوعی، از ساختارهای مختلفی استفاده می‌شود. سه ساختار شبکه عصبی که در این بخش برای شناسایی ربات فانتوم پیشنهاد شده است، به شرح زیر می‌باشد:

شبکه NNARX:

اگر بردار رگرسور معادله (۶-۳) به صورت زیر باشد، مدل شبکه عصبی NNARX حاصل می‌شود.

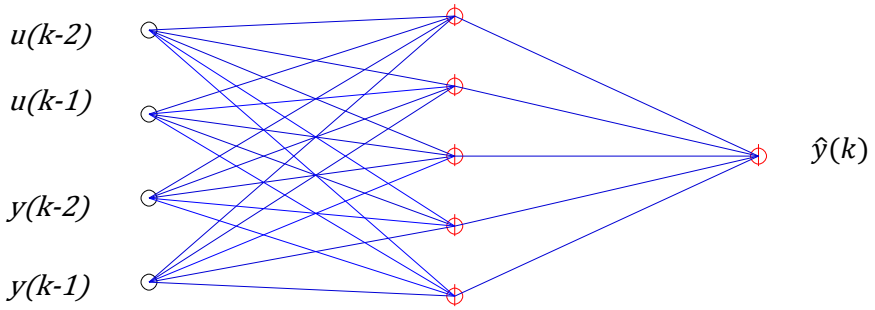
$$X(k) = [u(k-1), \dots, u(k-m), y(k-1), \dots, y(k-n)] \qquad (۶-۴)$$

در این مدل از مقادیر گذشته ورودی‌های کنترل و مقادیر گذشته خروجی‌های واقعی[1] استفاده می‌شود. شکل ساختار کلی شبکه NNARX را نشان می‌دهد.

در شبیه‌سازی‌های انجام گرفته از مقادیر ۲ لحظه قبل ورودی کنترل و خروجی استفاده شده است، یعنی بردار رگرسور به صورت زیر می‌باشد:

$$X(k) = [u(k-1), u(k-2), y(k-1), y(k-2)] \qquad (۶-۵)$$

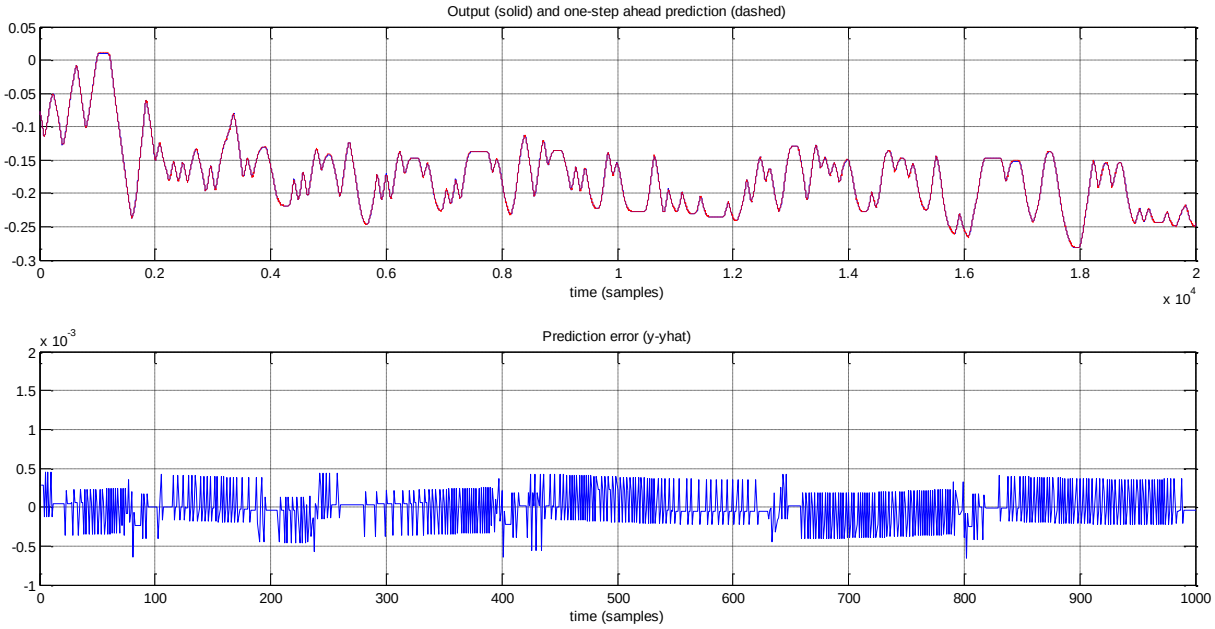
و شبکه عصبی طراحی شده دارای ۲ لایه بوده که لایه اول شامل ۵ نرون با توابع فعالساز تانژانت و لایه پایانی دارای یک نرون با تابع فعالساز خطی می‌باشد. شکل (۶-۱۳) ساختار کلی شبکه عصبی طراحی شده را نشان می‌دهد.



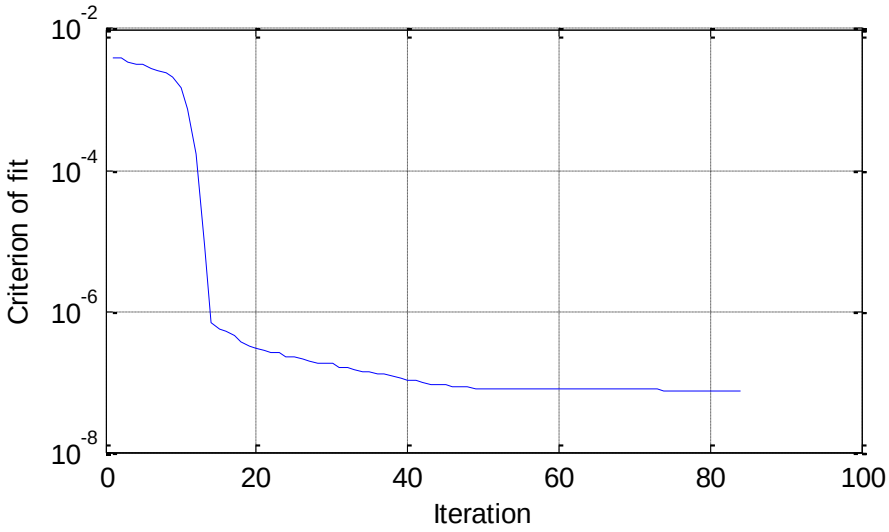
شکل (۶-۱۳): ساختار کلی شبکه NNARX استفاده شده برای شناسایی

[1] past control inputs and observed outputs

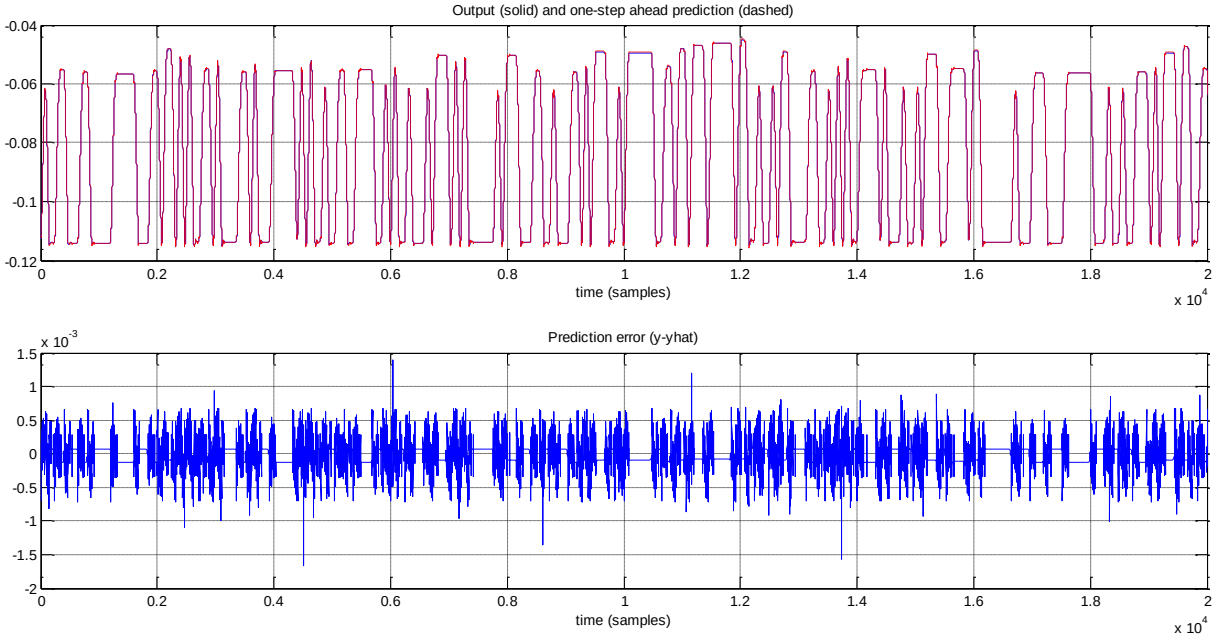
در ادامه نتایج حاصل از شناسایی توسط شبکه NNARX برای سه رابط ارائه می‌گردد و مقادیر SSE، RMSE و CC در جدول (۶-۲) خلاصه گردیده است.



شکل (۶-۱۴): الف: خروجی واقعی و پیش‌بینی یک گام به جلو
ب: خطای پیش‌بینی برای رابط اول شبکه NNARX

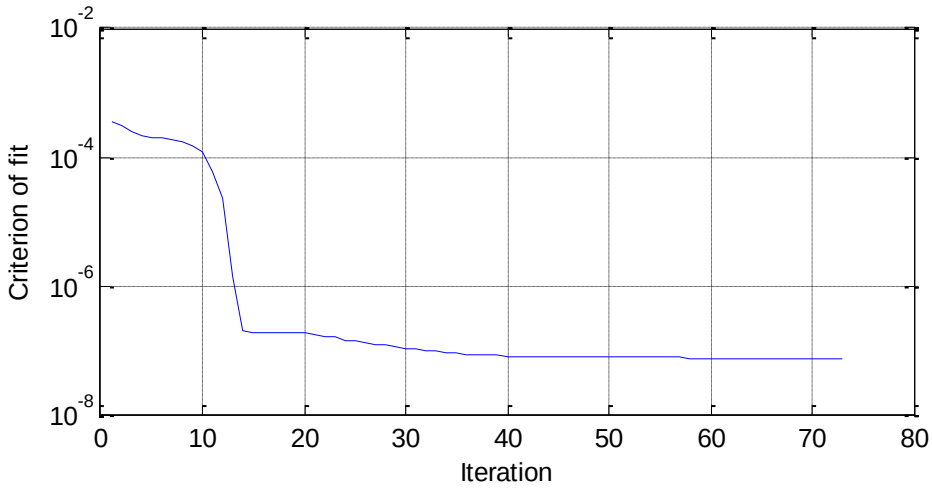


شکل (۶-۱۵): معیار SSE برای رابط اول در شبکه NNARX

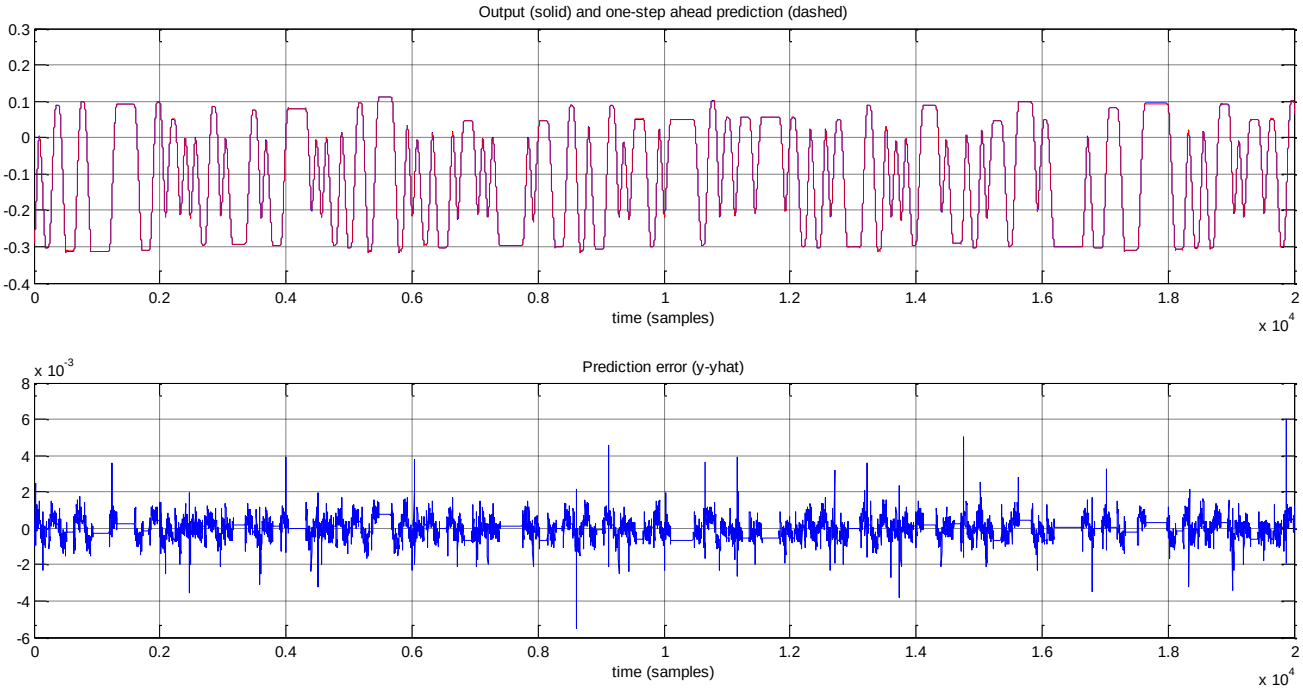


شکل (۶-۱۶): الف: خروجی واقعی و پیش‌بینی یک گام به جلو

ب: خطای پیش‌بینی برای رابط دوم شبکه NNARX

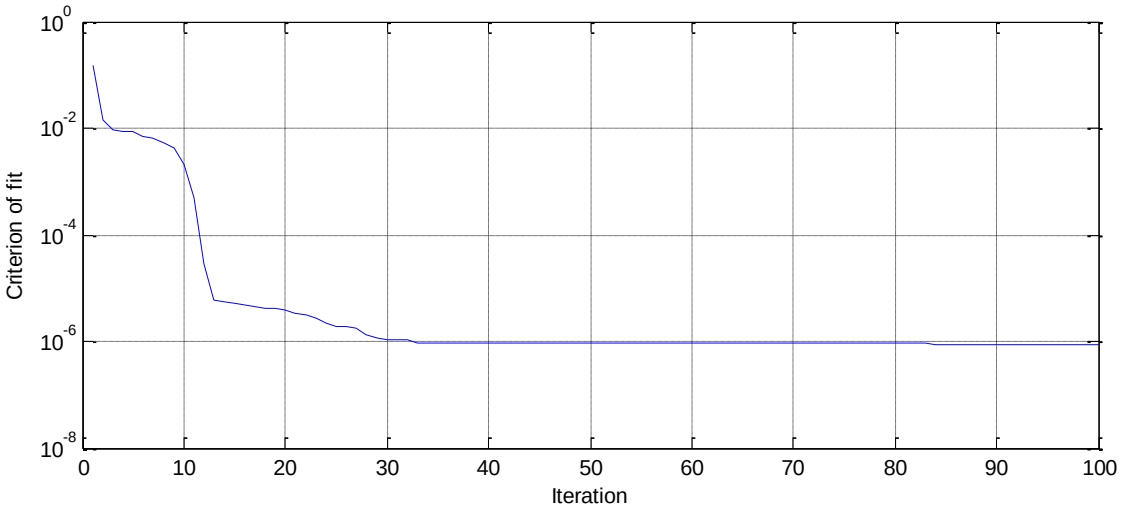


شکل (۶-۱۷): معیار SSE برای رابط دوم در شبکه NNARX



شکل (۶-۱۸): الف: خروجی واقعی و پیش‌بینی یک گام به جلو

ب: خطای پیش‌بینی برای رابط سوم شبکه NNARX



شکل (۶-۱۹): معیار SSE برای رابط سوم در شبکه NNARX

۶-۶-۲-نتایج شبیه‌سازی توسط شبکه عصبی NNARMAX

اگر بردار رگرسور معادله (۶-۳) به صورت زیر باشد، مدل NNARMAX حاصل می‌شود.

$$X(k) = [u(k-1), \dots, u(k-m), y(k-1), \dots, y(k-n), e(t-1), \dots, e(t-c)] \qquad (۶-۶)$$

به طوریکه خطای پیش‌بینی به صورت زیر تعریف می‌شود:

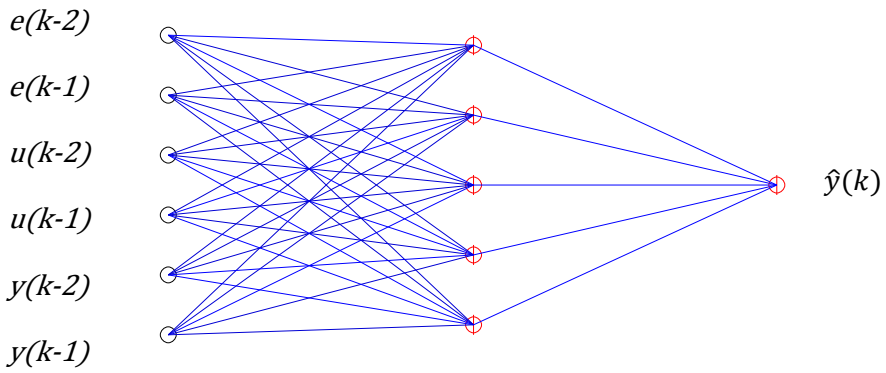
$$e(t) = y(t) - \hat{y}(t|\theta) \qquad (۶-۷)$$

که در آن $\hat{y}(t|\theta)$ خروجی مدل و $y(t)$ خروجی واقعی سیستم می‌باشد. در این مدل از مقادیر لحظات قبل ورودی کنترل، خروجی واقعی، و خطای لحظات قبل[1] استفاده می‌شود.

در شبیه‌سازی‌های انجام گرفته از مقادیر دو لحظه قبل ورودی کنترل و خروجی واقعی و خطا استفاده شده است، یعنی بردار رگرسور به صورت زیر می‌باشد:

$$X(k) = [u(k-1), u(k-2), y(k-1), y(k-2), e(t-1), e(t-2)] \qquad (۶-۸)$$

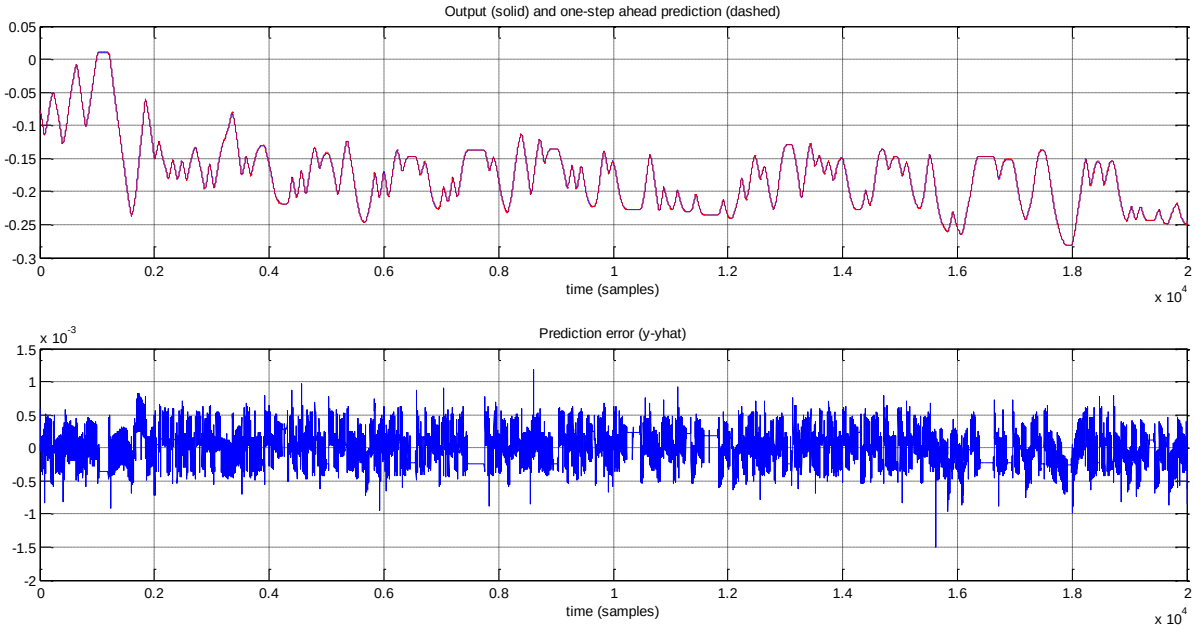
و شبکه عصبی طراحی شده دارای دو لایه بوده که لایه اول شامل پنج نرون با توابع فعالساز تانژانت و لایه پایانی دارای یک نرون با تابع فعالساز خطی می‌باشد. شکل (۶-۲۰) ساختار کلی شبکه عصبی طراحی شده را نشان می‌دهد.



شکل (۶-۲۰): ساختار کلی شبکه NNARMAX استفاده شده برای شناسایی

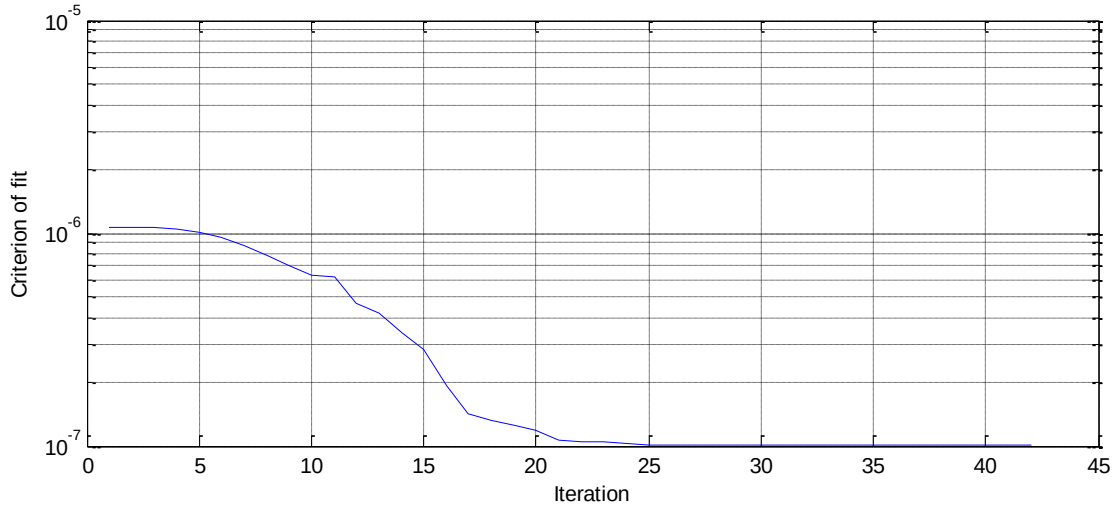
در ادامه نتایج حاصل از شناسایی توسط شبکه NNARMAX برای سه رابط ارائه می‌گردد و مقادیر SSE، RMSE و CC در جدول (۶-۲) خلاصه گردیده است.

[1] past control inputs, observed outputs and residuals

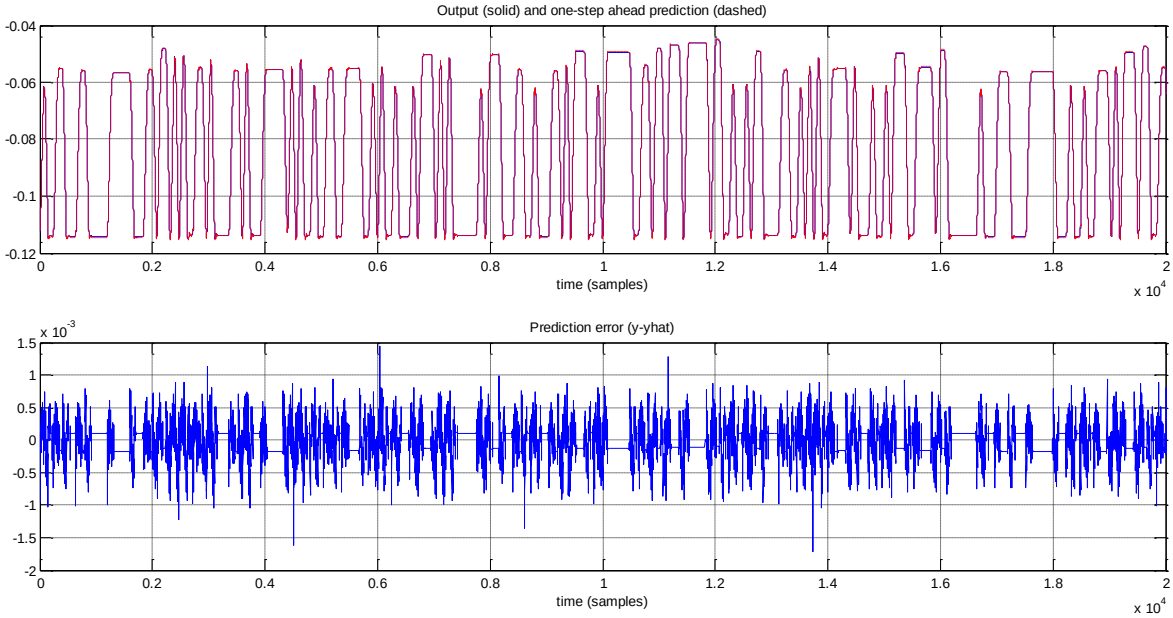


شکل (۶-۲۱): الف: خروجی واقعی و پیش‌بینی یک گام به جلو

ب: خطای پیش‌بینی برای رابط اول شبکه NNARMAX

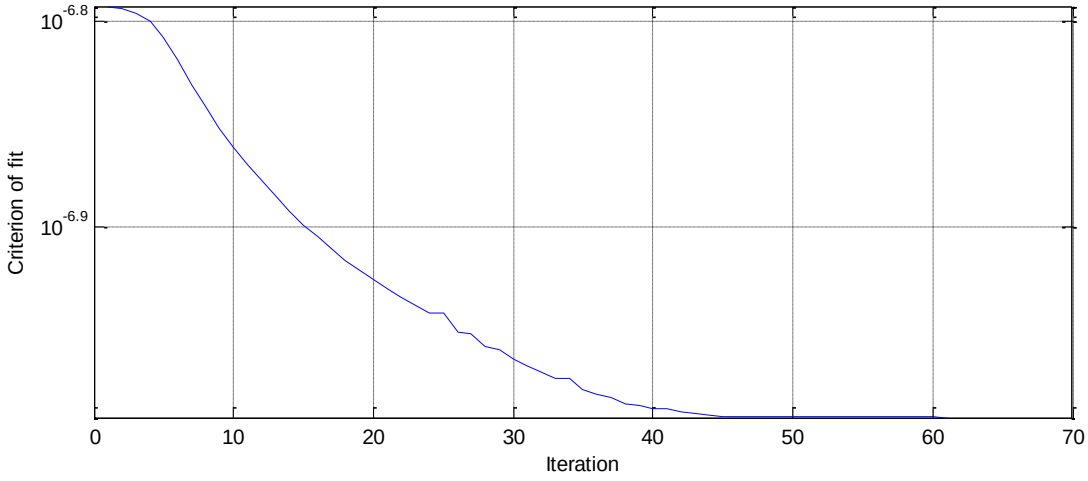


شکل (۶-۲۲): معیار SSE برای رابط اول در شبکه NNARMAX

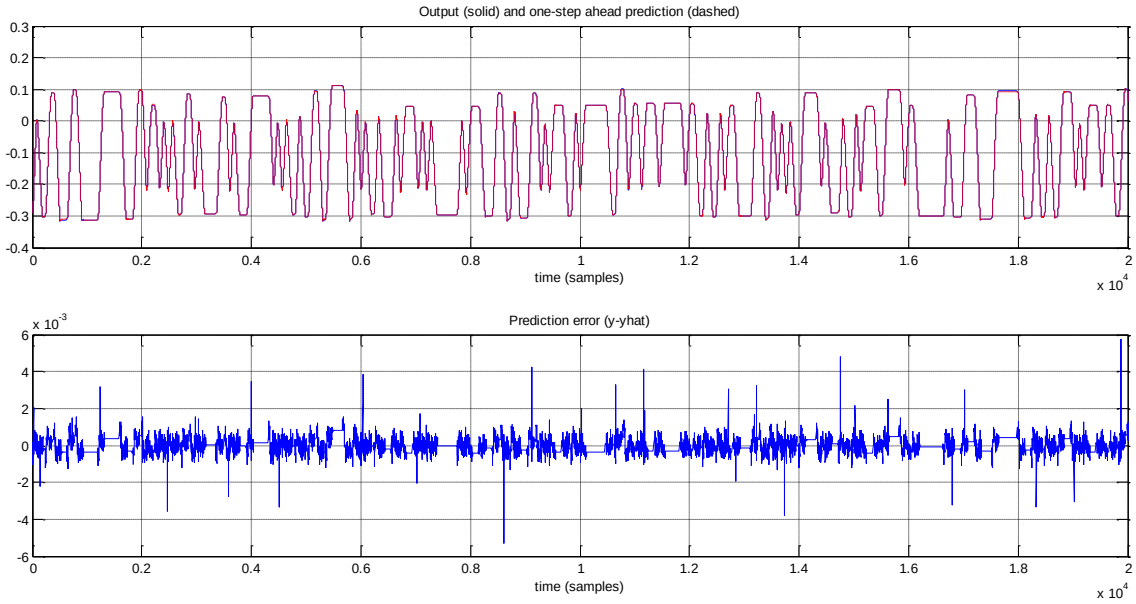


شکل (۶-۲۳): الف: خروجی واقعی و پیش‌بینی یک گام به جلو

ب: خطای پیش‌بینی برای رابط دوم شبکه NNARMAX

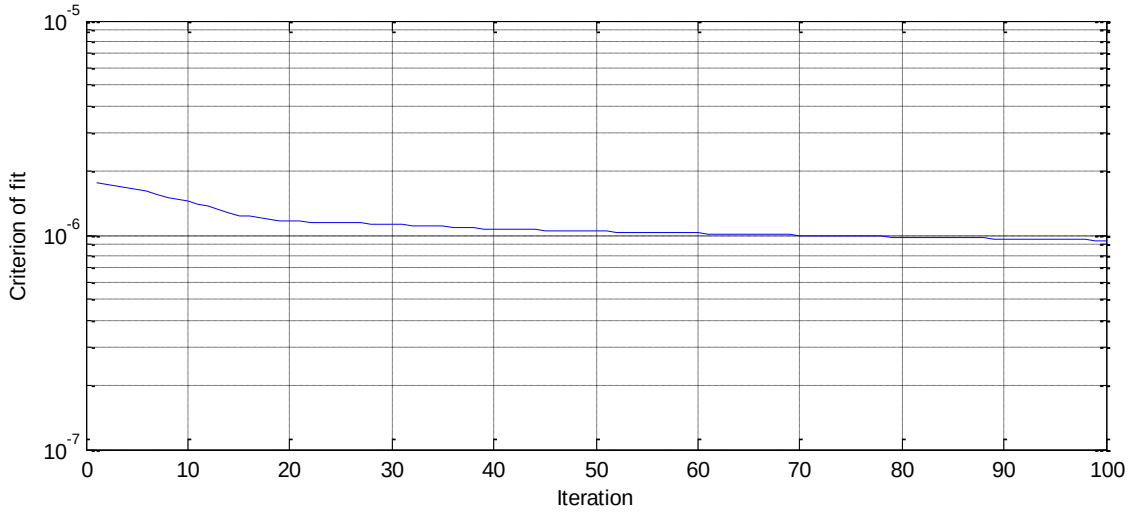


شکل (۶-۲۴): معیار SSE برای رابط اول در شبکه NNARMAX



شکل (۶-۲۵): الف: خروجی واقعی و پیش‌بینی یک گام به جلو

ب: خطای پیش‌بینی برای رابط سوم شبکه NNARMAX

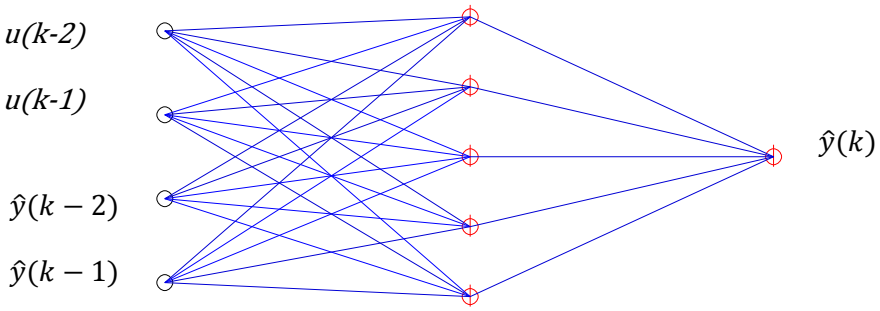


شکل (۶-۲۶): معیار SSE برای رابط سوم در شبکه NNARMAX

۶-۶-۳-نتایج شبیه‌سازی توسط شبکه عصبی NNOE

در این مدل از ورودی‌های کنترل و خروجی مدل[1] در بردار رگرسور استفاده می‌شود. بردار رگرسور و ساختار مدل NNOE به صورت زیر می‌باشد:

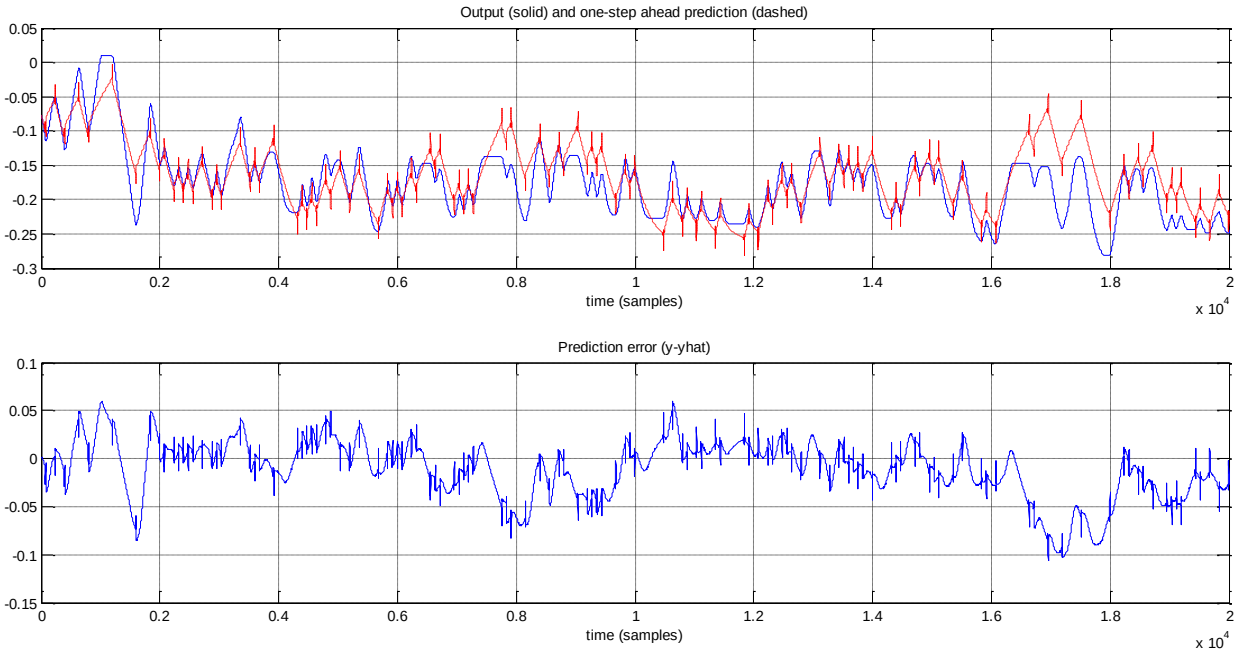
$$X(k) = [u(k-1), \dots, u(k-m), \hat{y}(k-1), \dots, \hat{y}(k-n)] \quad (۹-۶)$$



شکل (۶-۲۷): ساختار کلی شبکه NNOE استفاده شده برای شناسایی

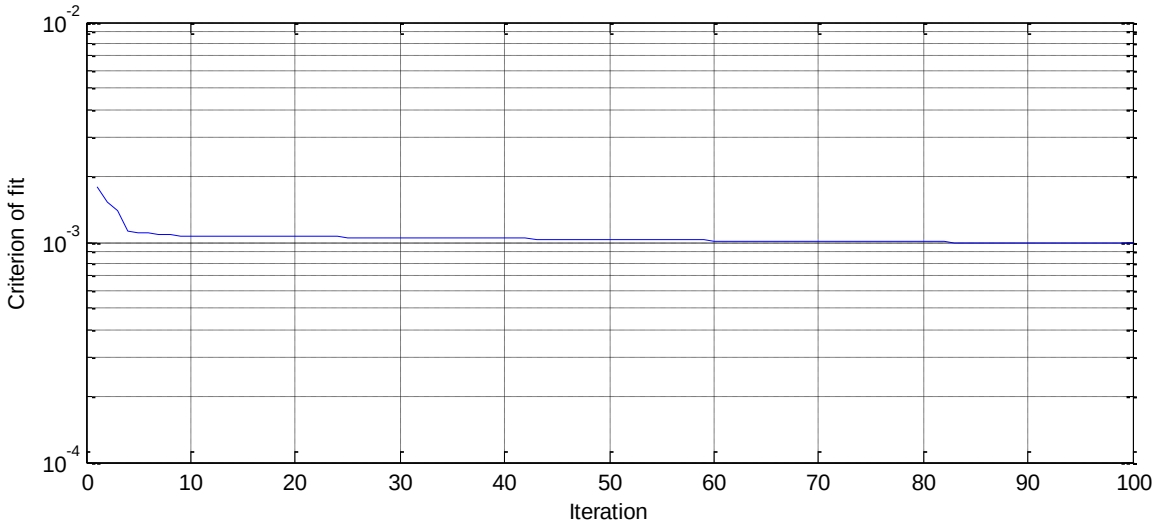
در ادامه نتایج حاصل از شناسایی توسط شبکه NNOE برای سه رابط ارائه می‌گردد و مقادیر SSE، RMSE و CC در جدول (۶-۲) خلاصه گردیده است.

[1] past control inputs, Model outputs

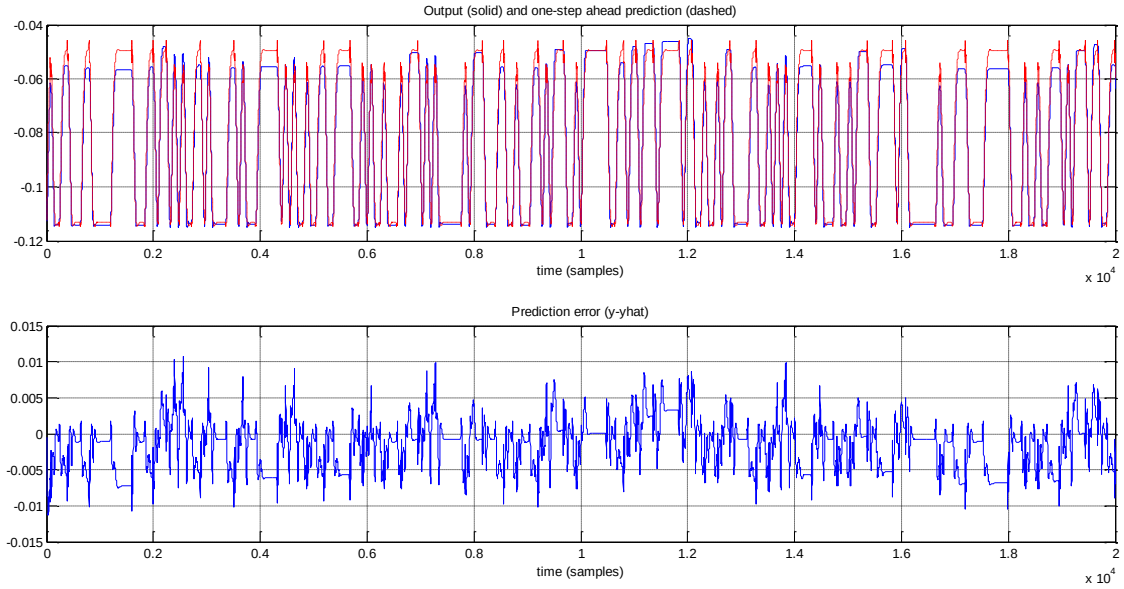


شکل (۶-۲۸): الف: خروجی واقعی و پیش‌بینی یک گام به جلو

ب: خطای پیش‌بینی برای رابط اول شبکه NNOE

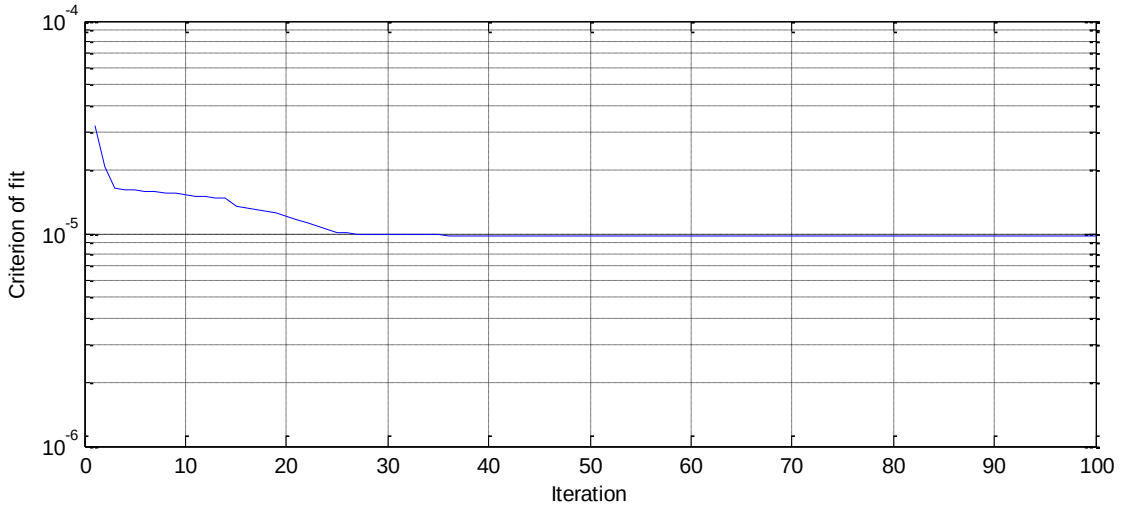


شکل (۶-۲۹): معیار SSE برای رابط اول در شبکه NNOE

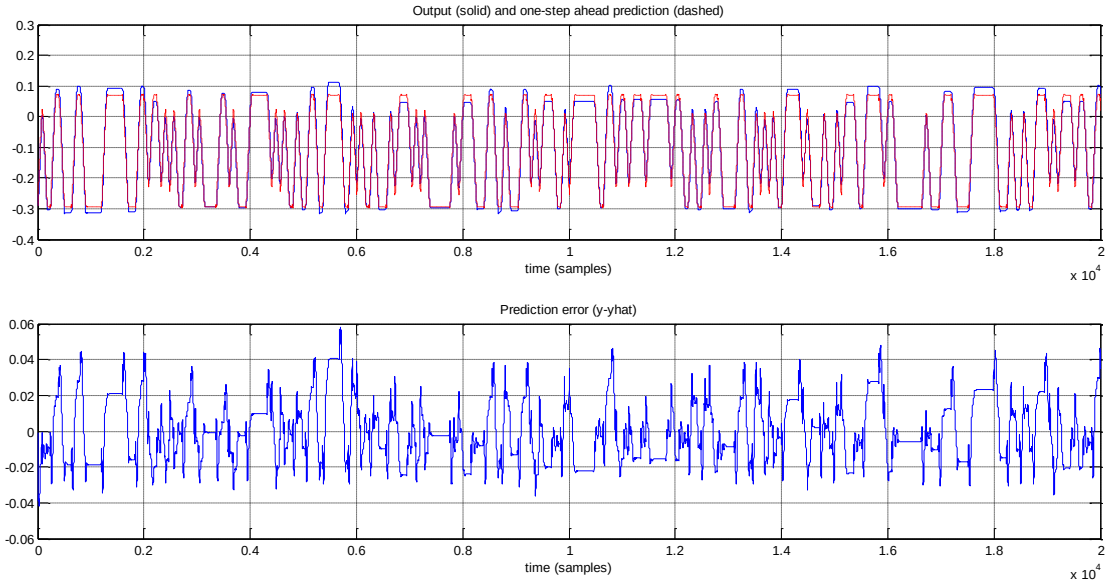


شکل (۶-۳۰): الف: خروجی واقعی و پیش‌بینی یک گام به جلو

ب: خطای پیش‌بینی برای رابط دوم شبکه NNOE

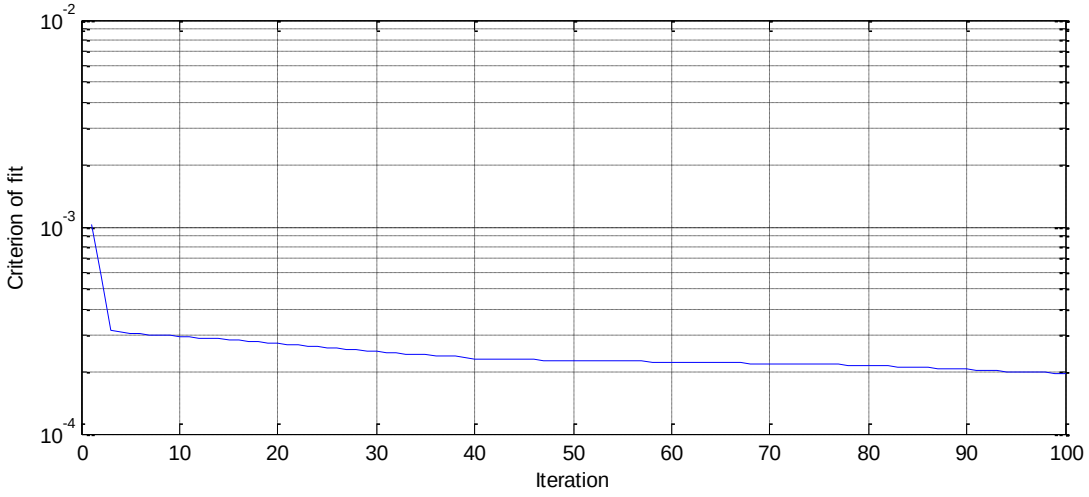


شکل (۶-۳۱): معیار SSE برای رابط دوم در شبکه NNOE



شکل (۶-۳۲): الف: خروجی واقعی و پیش‌بینی یک گام به جلو

ب: خطای پیش‌بینی برای رابط سوم شبکه NNOE



شکل (۶-۳۳): معیار SSE برای رابط سوم در شبکه NNOE

جدول(۶-۲): نتایج شناسایی توسط شبکه عصبی با ورودی PRBS

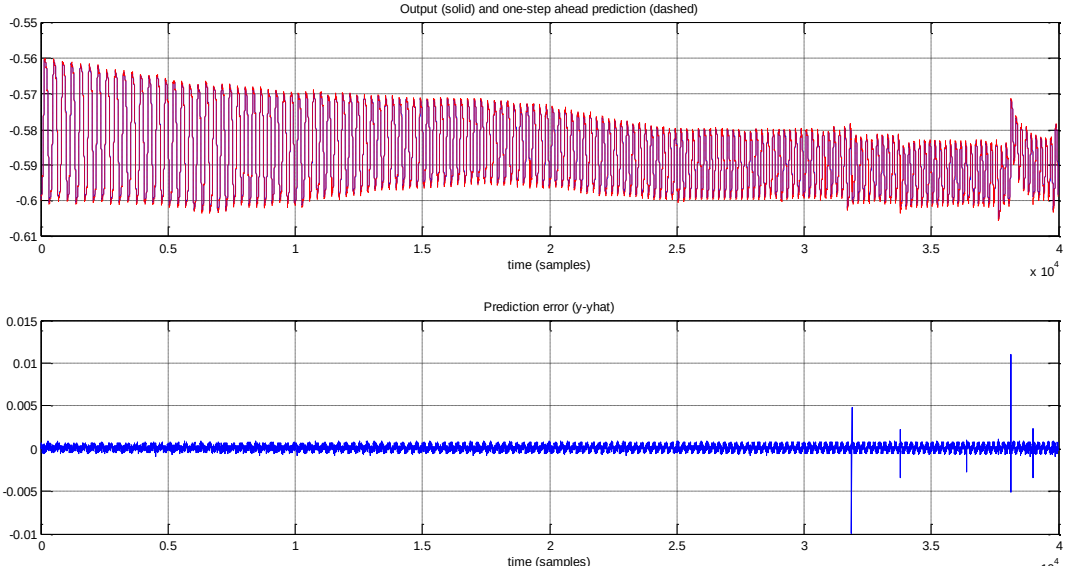
	SSE1	RMSE	CC1	SSE 2	RMSE2	CC2	SSE 3	RMSE3	CC3
NNARX	7.513e-008	0.0021	99.9990	7.503e-008	0.0153	99.9923	8.637e-007	0.0783	99.9609
NNARMAX	1.011e-007	0.0020	99.9990	9.918e-008	0.0148	99.9926	7.504e-007	0.0831	99.9585
NNOE	8.092e-004	14.5111	97.5409	6.173e-006	0.1889	99.9060	1.595e-004	1.1771	99.4198

همان‌طور که مشاهده می‌شود، شبکه‌های عصبی قادرند رفتار ربات را با دقت بالا شناسایی نمایند و این روش‌ها نسبت به روش‌های استفاده شده در بخش (۶-۵) دارای دقت بالاتری می‌باشند. از بین روش‌های ارائه شده در جدول (۶-۲) عملکرد دو روش NNARX و NNARMAX مشابه هم می‌باشند، ولی روش NNOE نسبت به این دو روش خطای بیشتری دارد.

۶-۷- شبیه‌سازی ربات با ورودی chirp

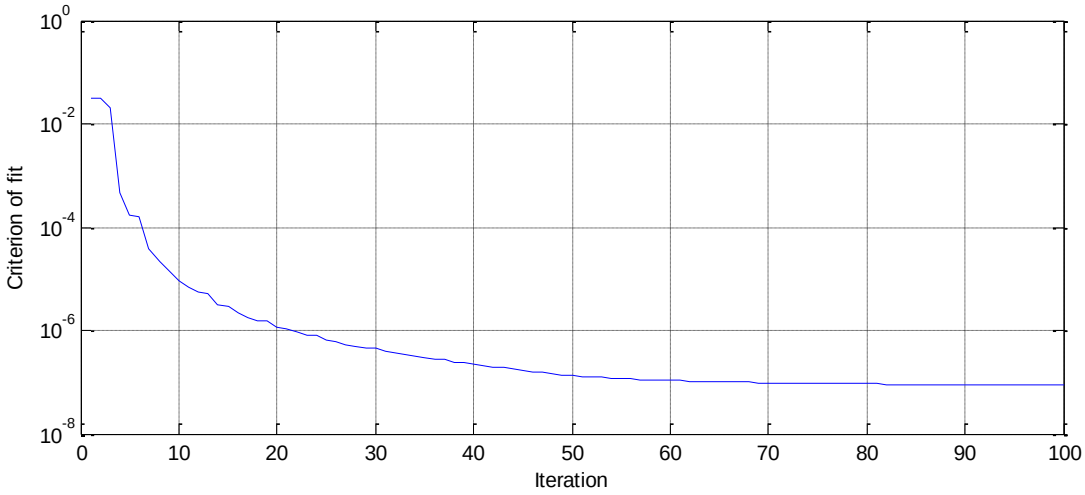
۶-۷-۱-نتایج شبیه‌سازی توسط شبکه عصبی NNARX

با توجه به توضیحات ارائه شده در بخش قبل، در این بخش به ارائه نتایج شناسایی ربات با ورودی جاروب فرکانسی سینوسی چیرپ می‌پردازیم.

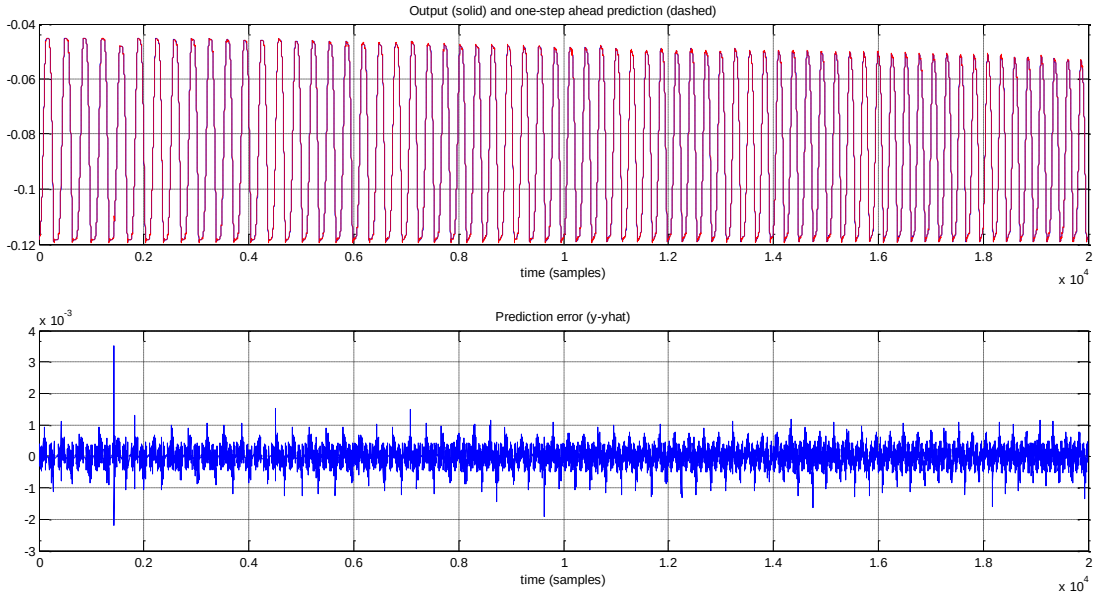


شکل (۶-۳۴): الف: خروجی واقعی و پیش‌بینی یک گام به جلو

ب: خطای پیش‌بینی برای رابط اول شبکه NNARX

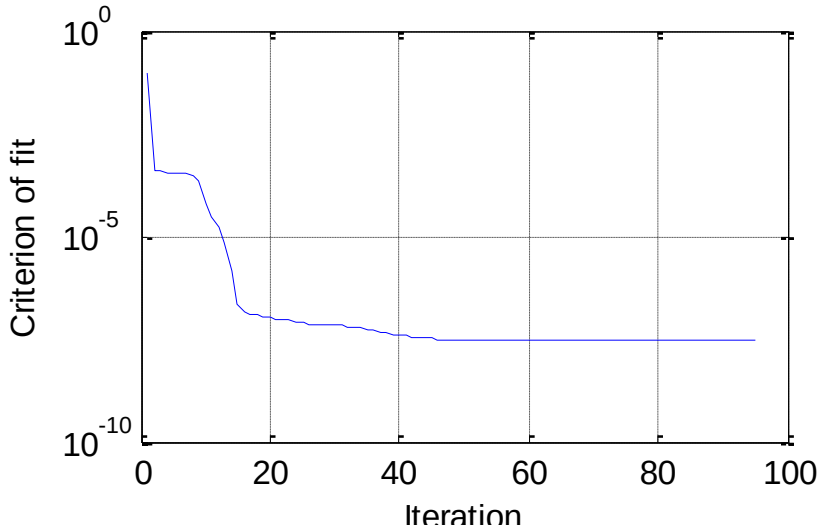


شکل (۶-۳۵): معیار SSE برای رابط اول در شبکه NNARX

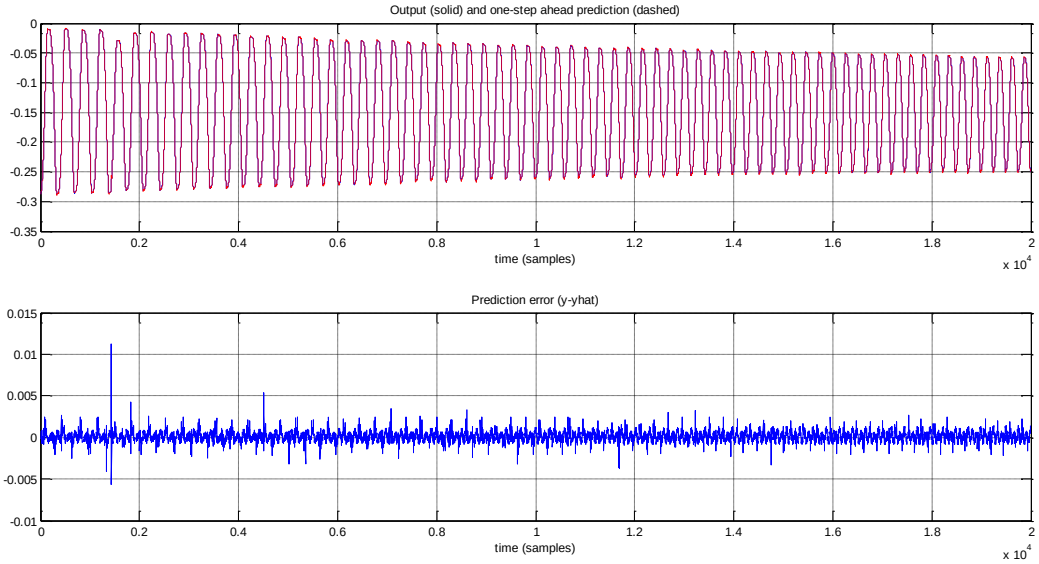


شکل (۶-۳۶): الف: خروجی واقعی و پیش‌بینی یک گام به جلو

ب: خطای پیش‌بینی برای رابط دوم شبکه NNARX

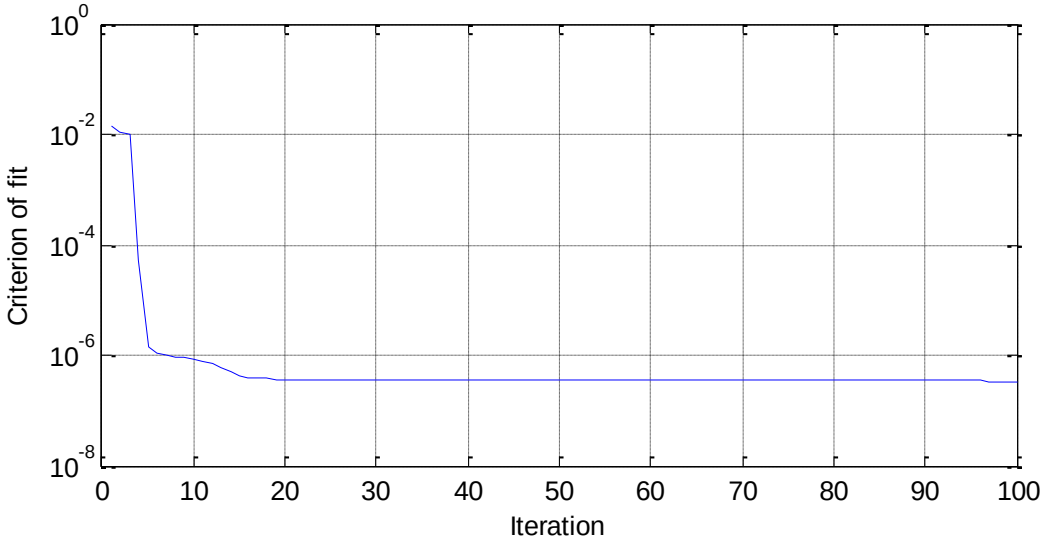


شکل (۶-۳۷): معیار SSE برای رابط دوم در شبکه NNARX



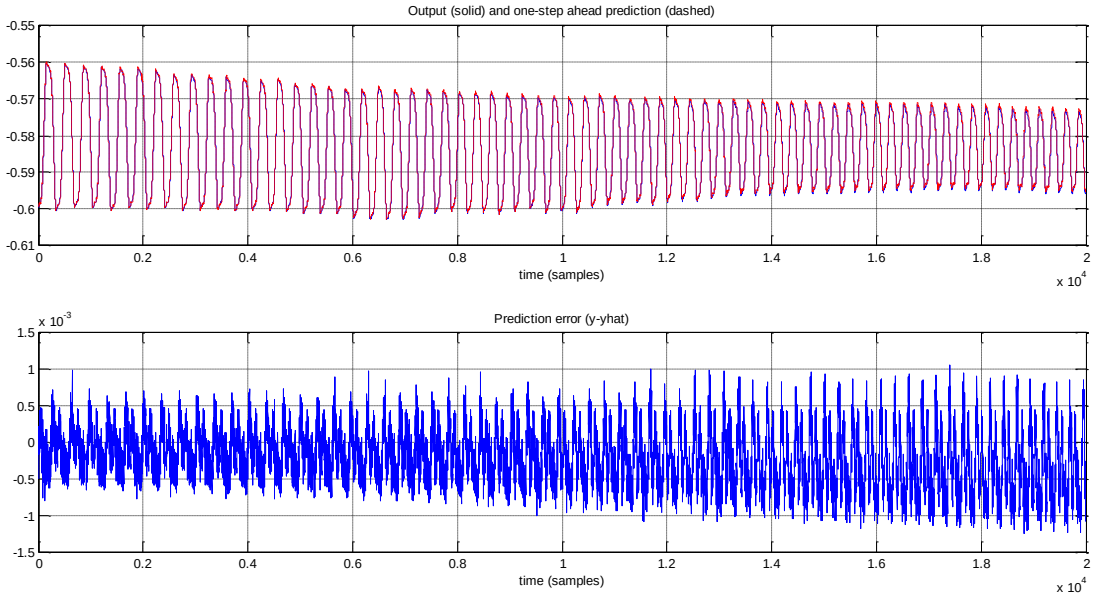
شکل (۶-۳۸): الف: خروجی واقعی و پیش‌بینی یک گام به جلو

ب: خطای پیش‌بینی برای رابط سوم شبکه NNARX



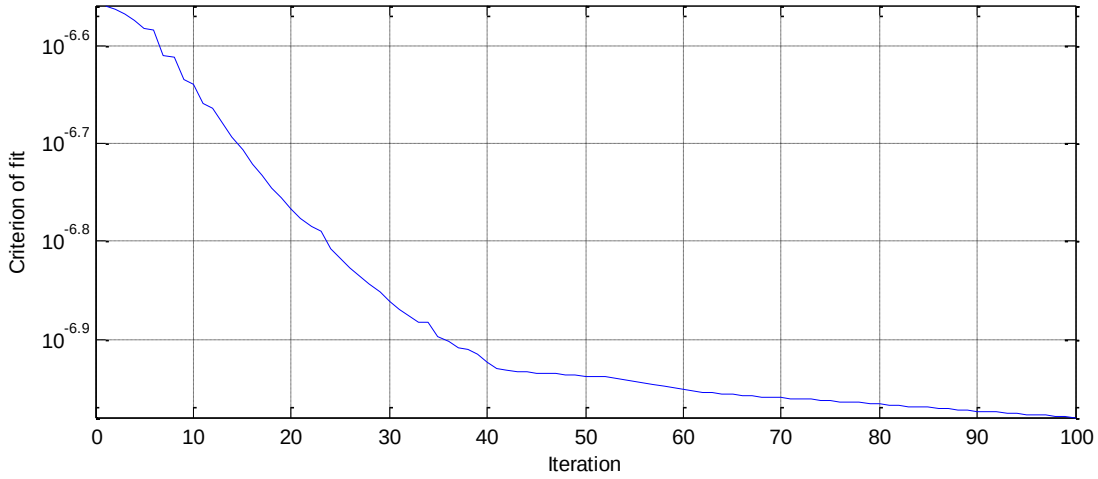
شکل (۶-۳۹): معیار SSE برای رابط سوم در شبکه NNARX

۶-۷-۲-نتایج شبیه‌سازی توسط شبکه عصبی NNARMAX

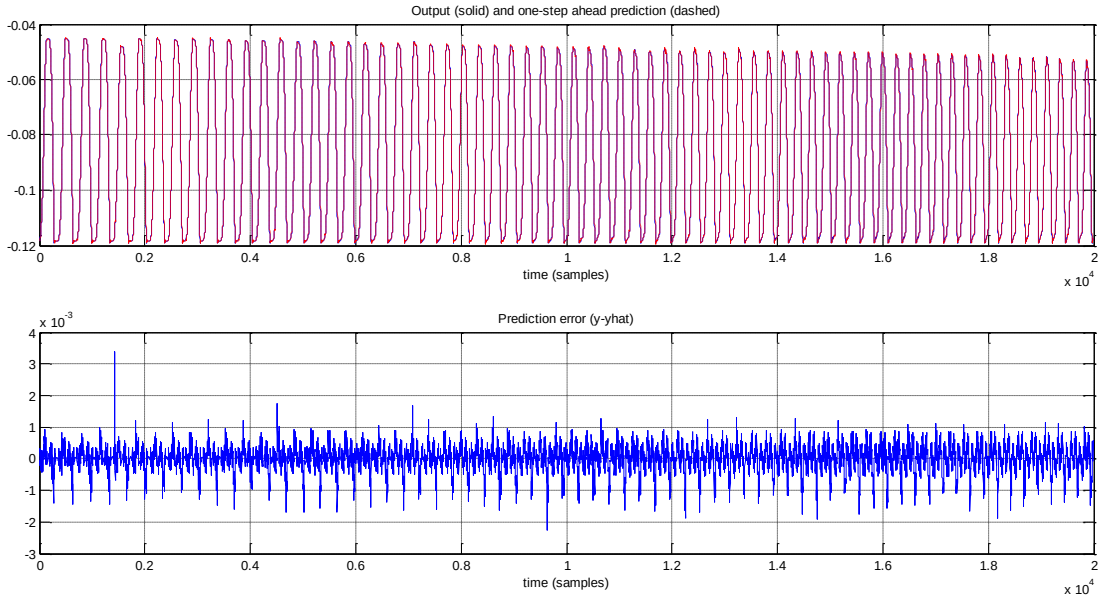


شکل (۶-۴۰): الف: خروجی واقعی و پیش‌بینی یک گام به جلو

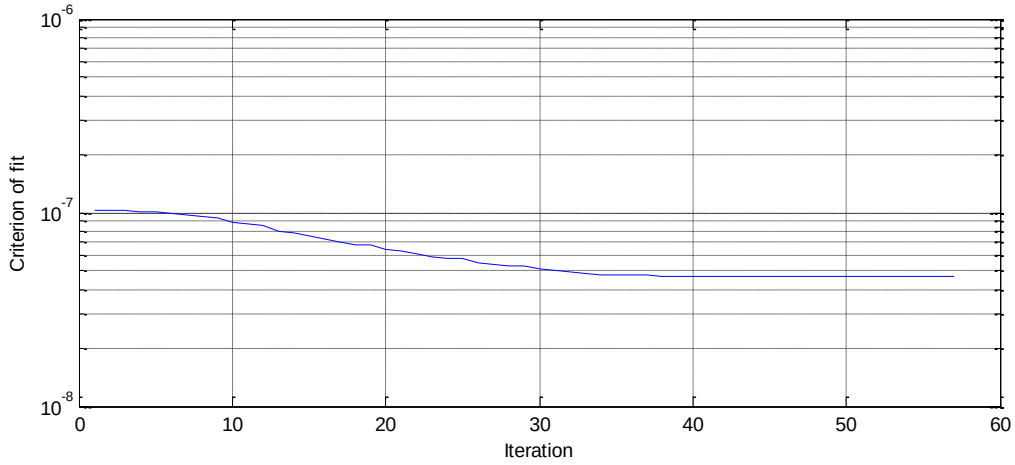
ب: خطای پیش‌بینی برای رابط اول شبکه NNARMAX



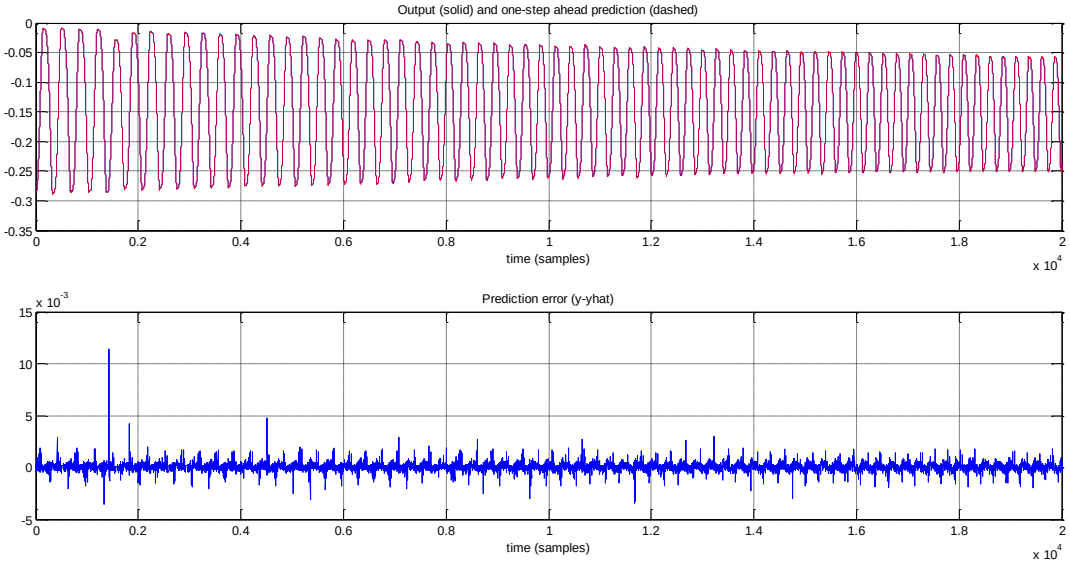
شکل (۶-۴۱): معیار SSE برای رابط اول در شبکه NNARMAX



شکل (۶-۴۲): الف: خروجی واقعی و پیش‌بینی یک گام به جلو
ب: خطای پیش‌بینی برای رابط دوم شبکه NNARMAX

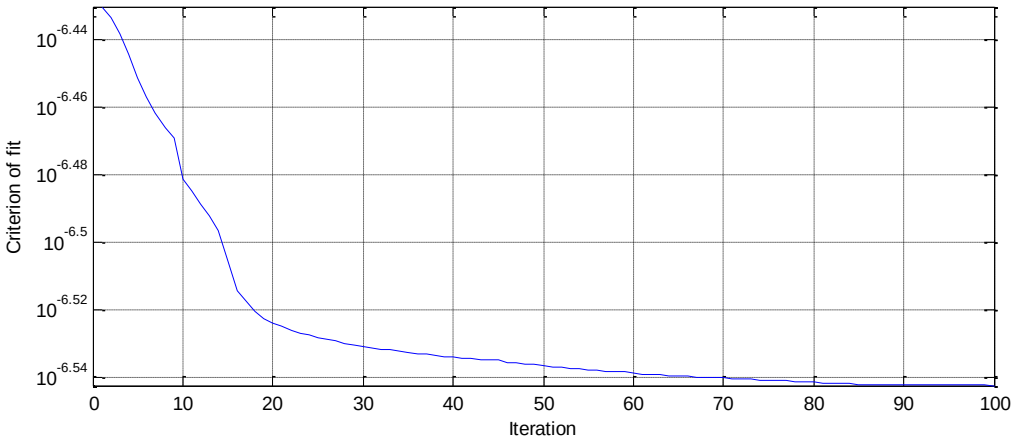


شکل (۶-۴۳): معیار SSE برای رابط دوم در شبکه NNARMAX



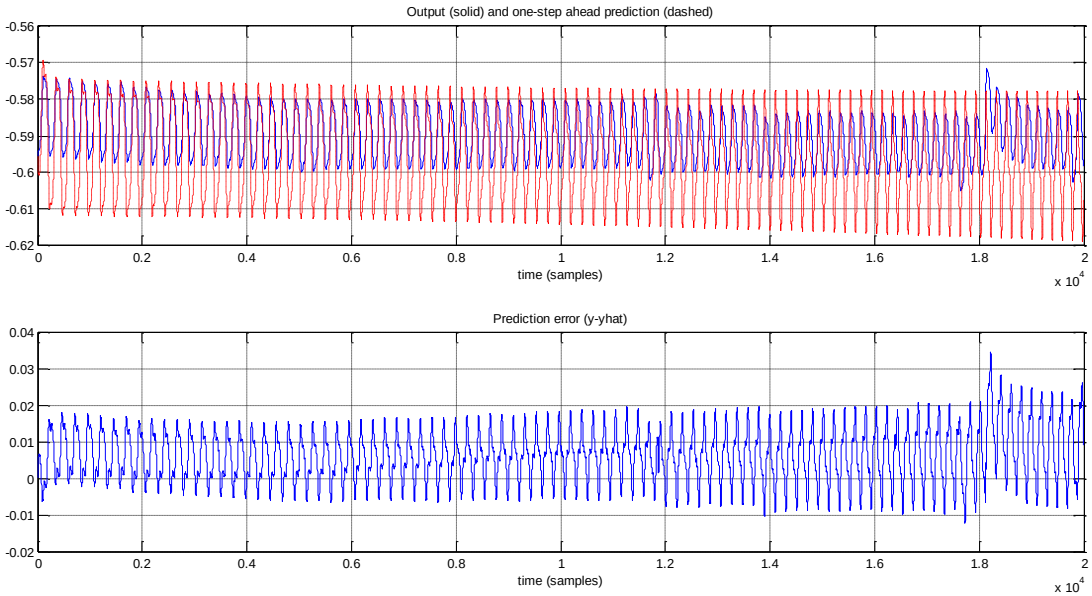
شکل (۶-۴۴): الف: خروجی واقعی و پیش‌بینی یک گام به جلو

ب: خطای پیش‌بینی برای رابط سوم شبکه NNARMAX



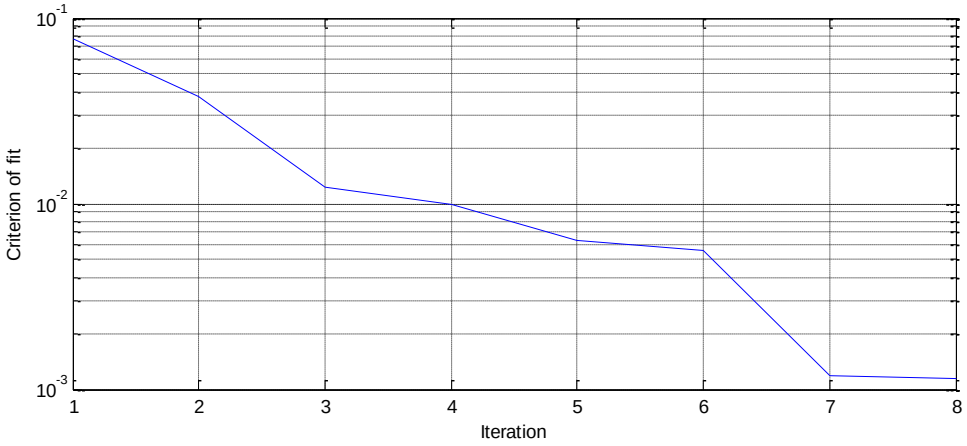
شکل (۶-۴۵): معیار SSE برای رابط سوم در شبکه NNARMAX

۶-۷-۳-نتایج شبیه‌سازی توسط شبکه عصبی NNOE

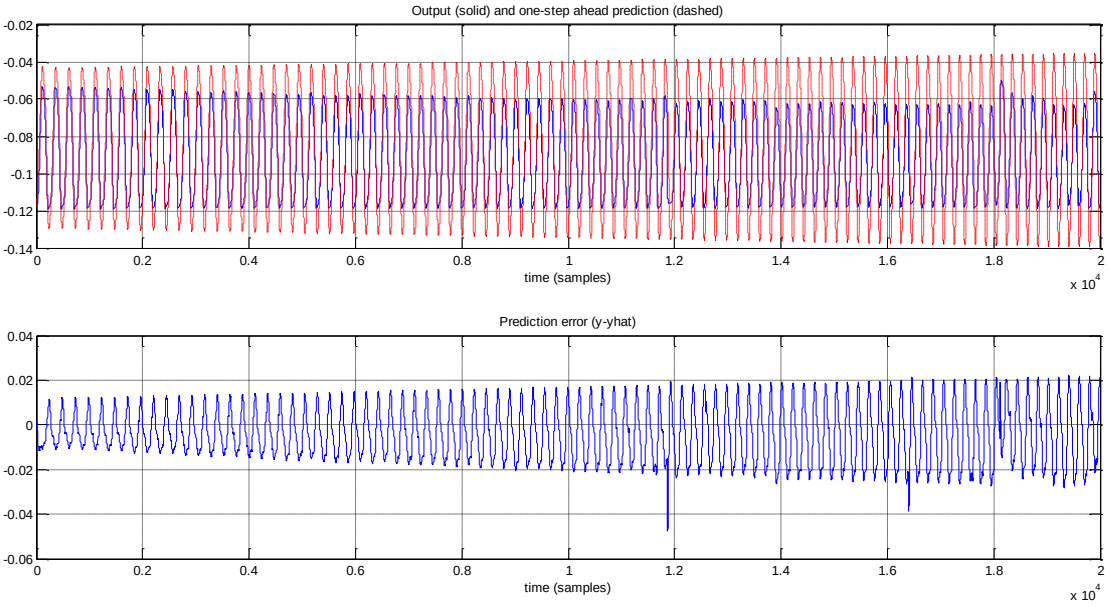


شکل (۶-۴۶): الف: خروجی واقعی و پیش‌بینی یک گام به جلو

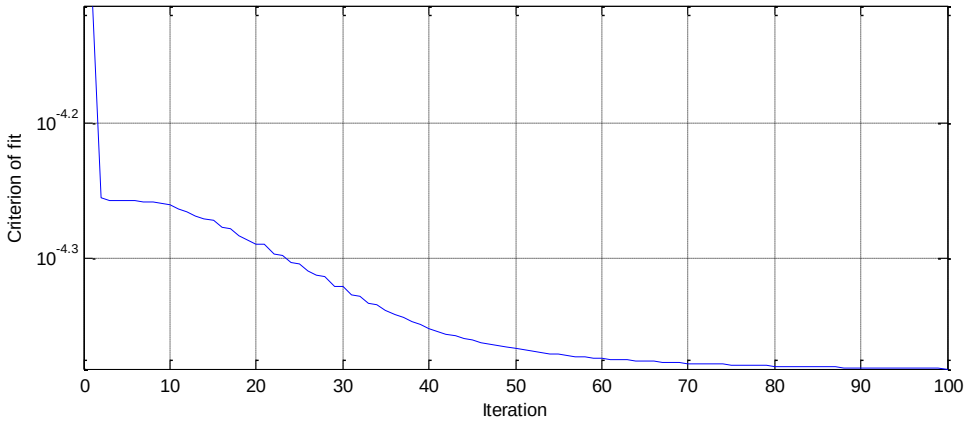
ب: خطای پیش‌بینی برای رابط اول شبکه NNOE



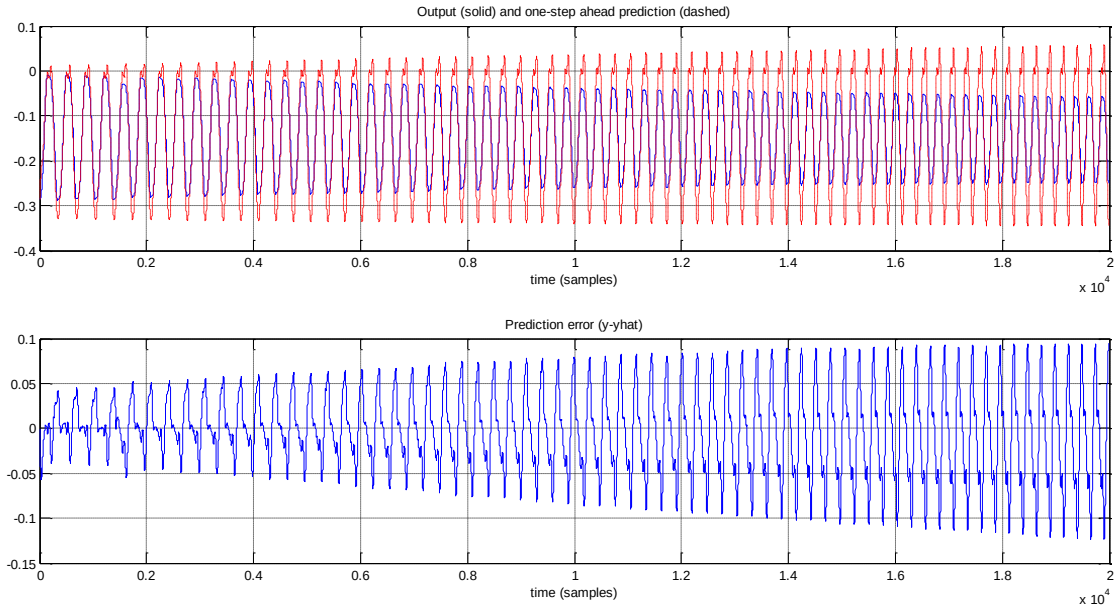
شکل (۶-۴۷): معیار SSE برای رابط اول در شبکه NNOE



شکل (۶-۴۸): الف: خروجی واقعی و پیش‌بینی یک گام به جلو
ب: خطای پیش‌بینی برای رابط دوم شبکه NNOE

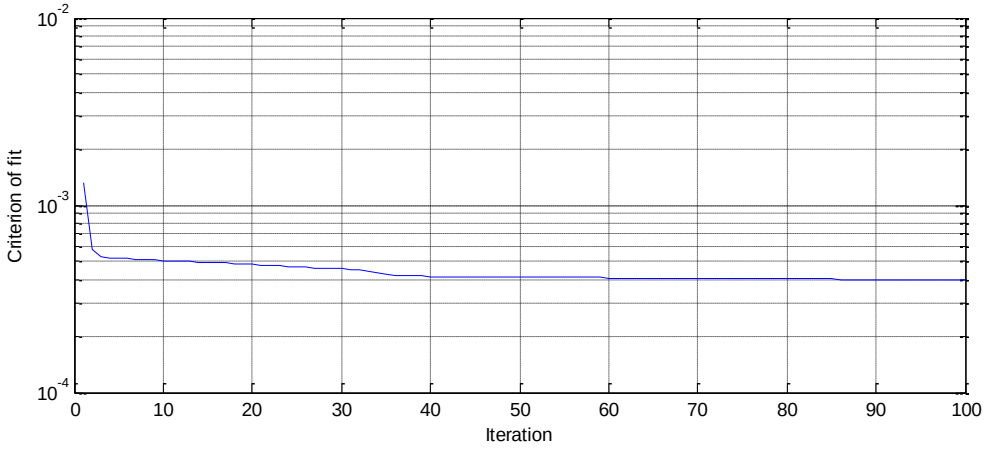


شکل (۶-۴۹): معیار SSE برای رابط دوم در شبکه NNOE



شکل (۶-۵۰): الف: خروجی واقعی و پیش‌بینی یک گام به جلو

ب: خطای پیش‌بینی برای رابط سوم شبکه NNOE



شکل (۶-۵۱): معیار SSE برای رابط سوم در شبکه NNOE

جدول(۶-۳): نتایج شناسایی توسط شبکه عصبی با ورودی چیرپ

	SSE1	RMSE	CC1	SSE 2	RMSE2	CC2	SSE 3	RMSE3	CC3
NNARX	8.859e-008	1.4575e-004	99.9999	2.764e-008	0.0174	99.9913	3.356e-007	0.0482	99.9759
NNARMAX	1.048e-007	1.9878e-004	99.9999	4.685e-008	0.0162	99.9919	2.868e-007	0.0509	99.9746
NNOE	1.145e-003	0.0277	99.9918	4.148e-005	2.0222	98.9903	3.991e-004	7.1420	97.1739

در این فصل توسط روش‌های مختلف شناسایی که در فصل‌های قبل ارائه گردیده بود، به شناسایی عملی ربات فانتوم پرداختیم. نتایج ثبت شده نشان می‌دهد شبکه‌های عصبی بازگشتی NNARX و NNARMAX بیشترین دقت شناسایی و بیشترین درصد همبستگی متقابل بین خروجی واقعی و خروجی مدل را دارند. شبکه NNOE نیز چون از خروجی مدل برای اصلاح وزن‌های شبکه عصبی استفاده می‌نماید، دارای دقت کمتری نسبت به دو روش ذکر شده می‌باشد. به هر حال شبکه‌های عصبی قادرند رفتار دینامیکی سیستم‌های غیرخطی را با دقت بالا تقریب بزنند و سیستم مورد نظر را شناسایی نمایند.

فصل هفتم:

جمع بندی، نتیجه گیری و پیشنهادات برای ادامه کار

جمع بندی و نتیجه گیری:

در این پروژه به معرفی و چگونگی استفاده از شبکه‌های عصبی دینامیک در شناسایی سیستم‌ها پرداختیم و در ادامه برای بررسی کارآیی آنها، از شبکه‌های عصبی برای شناسایی عملی ربات فانتوم استفاده نمودیم. همچنین الگوریتم‌های مختلف بهینه‌سازی که در شناسایی سیستم‌ها کاربرد دارند معرفی گردید. الگوریتم‌های بهینه‌سازی هوشمند از جمله الگوریتم بهینه‌سازی ذرات و الگوریتم ژنتیک که از الگوریتم‌های بهینه‌سازی فراگیر می‌باشند و کاربرد آنها روز به روز در حال افزایش است معرفی و کاربرد آنها در آموزش شبکه عصبی ارائه گردید. این الگوریتم‌ها در مواردی که شبکه عصبی پیچیده و دارای مولفه‌های زیادی می‌باشد و الگوریتم‌های بهینه‌سازی محلی برای یافتن جواب کمینه سراسری پاسخگو نیستند، بکار گرفته می‌شوند.

در پایان سه ساختار شبکه عصبی دینامیکی NNARMAX، NNARX و NNOE برای شناسایی ربات مورد نظر پیشنهاد گردید. نتایج حاصل از آموزش و تست شبکه عصبی نشان می‌دهد که این شبکه‌ها قادرند رفتار دینامیکی ربات مورد نظر را با دقت بالا شناسایی نمایند.

بنابراین با توجه به نتایج بدست آمده و توانایی بالای شبکه‌های عصبی در مدلسازی سیستم‌های غیرخطی، از شبکه‌های عصبی دینامیکی می‌توان به صورت موثر برای شناسایی مدل غیرخطی ربات های صنعتی، لامسه‌ای و عملیات از راه دور استفاده نمود.

پیشنهادات برای ادامه کار:

یکی از مسائلی که پیش روی ما برای انتخاب شبکه عصبی بود، تعیین ساختار، تعداد لایه‌ها، تعداد نرون ها و توابع فعالساز شبکه عصبی بود و با توجه به بررسی‌های به عمل آمده به صورت سعی و خطا این کار صورت می‌پذیرد و شبکه ای که کمترین خطا را به هنگام آزمایش دارا باشد را به عنوان ساختار بهینه انتخاب می‌کنیم. در زمینه بهینه‌سازی پارامترهای شبکه تحقیقات زیادی صورت پذیرفته است ولی در بحث بهینه‌سازی ساختار شبکه، زمینه تحقیقات فراهم است. پیشنهادات زیر برای ادامه کار ارائه می‌گردد:

- استفاده از الگوریتم‌های بهینه‌سازی هوشمند برای یافتن ساختار بهینه شبکه عصبی
- استفاده از الگوریتم‌های هوشمند برای آموزش شبکه عصبی
- تعمیم روش‌های استفاده شده در این پروژه به فرآیندهای مشابه
- شناسایی غیرخطی ربات فانتوم توسط شبکه‌های فازی، نروفازی، توابع موجک

مراجع

[1] M. C. Cavusoglu and D. Feygin, "Kinematics and dynamics of phantom(tm) model 1.5 haptic interface,"*UC Berkeley ERL Memo M01/15*, 2001.

[2] B. Taati, A. Tahmasebi and K. Hashtrudi-Zaad, "Experimental identification and analysis of the dynamics of a PHANTOMTM Premium 1.5 haptic device," *PRESENCE*, 17(4): 327-343, 2008.

[3] A. M. Tahmasebi, B. Taati, F. Mobasser, and K. Hashtrudi-Zaad, "Dynamic parameter identification and analysis of a PHANToMTM haptic device," in Proc. IEEE Conference on Control Applications (CCA'05), Toronto, Canada, Aug. 2005, pp. 1251–1256.

[4] C. ping, "System Identification in Hydraulic Servo System with Diagonal Recurrent Neural Networks" , College of Automation Science and Electric Engineering.

[5] L. Behera & et al, "Identification of Nonlinear Dynamical Systems Using Recurrent Neural Networks", Department of Electrical Engineering Indian Institute of Technology, Kanpur, 208016 INDIA 0512-259-72-198, 2003.

[6] C. Alippi , V. Piuri, "Experimental neural networks for prediction and identification", IEEE Transactions On Instrumentation And Measurement, VOL. 45, NO. 2, APRIL 1996.

[7] Martin T. Hagan , Howard B. Demuth , Orlando , De Jesús, "An Introduction To The Use of Neural Networks In Control Systems", International Journal of Robust and Nonlinear Control, 2002.

[8] K. Hornik and M. Stinchcombe and H.White. "Multilayer feed-forward networks are universal approximators". Department of Encomics, University of Californa, San Diego, La Lolla, CA, 1988.

[9] Ljung, L., System Identification: Theory for the User, 2/e, Prentice-Hall, Englewood Cliffs, NJ, 1999.

[10] O. Nelles, "Nonlinear System Identification- from classical approaches to neural networks and fuzzy models", springers publication, 2000.

[11] S. Mostapha Kalami and Naser Pariz, "Effect of Learning Coefficients and Population Size on Particle Swarm Optimization," in *Proceedings of 3rd International Conference on Information and Knowledge Technology*,October 2007.

[12] دکتر مهدی کراری ، ۱۳۸۸، شناسایی سیستم، چاپ اول، انتشارات دانشگاه صنعتی امیرکبیر.

[13] محمد باقر منهاج، مبانی شبکه‌های عصبی هوش محاسباتی، انتشارات دانشگاه صنعتی امیرکبیر.

[14] F. Mobasser, and K. Hashtrudi-Zaad, "A model-independent force observer for teleoperation systems," *Proc. of IEEE Int. Conf. on Mechatronics and Automation (ICMA)*, 2005.

[15] M. Norgaard, "Neural Network Based System Identification Toolbox", Version 2, Technical Report 2000-E-891, Department of Automation, Technical University of Denmark.

[16] M. Norgaard, O. Ravan, N.K. Poulsen, L.K. Hansen, " Neural network for modelling and control of dynamical systems: a practitioner's handbook", Springer, 2nd printing, 2001.

[17] sexton R.S., Dorsey R.E., Sikander N.A., " Simultaneous Optimization of Neural Network Function and Architecture Algorithm", Decision Support Systems 36,2004 , PP. 283 – 296.

[18] Pham D.T., Karaboga D., " Training Elman and Jordan Networks for System Identification Using Genetic Algorithm", Artificial Intelligence in Engineering 13, 1999 , PP.107 – 117.

[19] Son J.S., Lee D.M., Kim I.S., Choi S.K., " A Study on Genetic Algorithm to Select Architecture of an Optimal Neural Network in the Hot Rolling Process". Journal of Material Processing Technology 153-154 , 2004 , PP. 643 – 648.

[20] Castillo P.A., Merelo J.J., PrietoA., Rivas V., Romero G., " G-Prop: Global Optimization of Multilayer Perceptrons Using Gas", Neurocomputing 35 , 2000 , PP. 149 -163.

[21] sexton R.S., Dorsey R.E., Johnson J.D., " Optimization of Neural Network : A Comparative Analysis of the Genetic Algorithm and Simulated Annealing", European Journal of Operational Research 114, 1999 , PP.589-601.

[22] I. Kaastra, M. Boyd, " Designing a neural network for forecasting financial and economic time series", Neurocomputing Journal, 1996 , PP.215-236.

[23] بهزاد ح، (۱۳۸۹)، پایان نامه ارشد :" شناسایی سیستم‌های مرتبه کسری در فضای حالت"، دانشکده برق و رباتیک، دانشگاه صنعتی شاهرود.

[24] نیکو سخن م، (۱۳۸۳)، پایان نامه ارشد :" شناسایی غیرخطی سیستم دینامیکی هواپیما"، دانشکده مهندسی هوافضا، دانشگاه صنعتی شریف.

Abstract:

There are several ways for modeling a system. Modeling is usually applied using physical relationships governing the system or system identification methods or a combination of both. In general, these models may be static or dynamic, linear or nonlinear, time variant or time invariant, continuous-time or discrete-time, deterministic or stochastic. In one of these methods which is called black box modeling, at the beginning a test is performed for modeling the system and input and output data are recorded and then this data is used for identification and modeling of the intended system.

In this thesis a Phantom robot has been identified using neural network. To evaluate the performance of the proposed methods, a Phantom robot has been chosen and experimentally identified using data extracted from the robot. The Phantom robot has three manipulators and is widely used in haptic and telerobotic applications. Modeling the robot, is important for predictive applications, simulation, control, and fault detection. The results show that neural networks are able to identify a Phantom robot with high precision.

***Key words*— Phantom Robot, Neural Networks, Nonlinear System Identification, Intelligent optimization.**