دانشکده مهندسی برق و رباتیک

گروه کنترل

آموزش شبکه عصبی کوهنن با استفاده از فیلتر ذره ای

دانشجو : سردار افرا

استاد راهنما :

دکتر محمد حدادظریف

استاد مشاور :

دکتر حیدر طوسیان شاندیز

پایان نامه ارشد جهت اخذ درجه کارشناسی ارشد

تیرماه ۱۳۸۸


بسم الله الرحمن الرحيم

دانشکده : مهندسی برق و رباتیک

گروه : کنترل

آموزش شبکه عصبی کوهنن با استفاده از فیلتر ذره ای

دانشجو : سردار افرا

استاد راهنما :

دکتر محمد حدادظریف

استاد مشاور :

دکتر حیدر طوسیان شاندیز

پایان نامه ارشد جهت اخذ درجه کارشناسی ارشد

تیرماه ۱۳۸۸

شماره : ۰۷۶۸۰/آ.ت.ب
تاریخ : ۱۳۸۹/۰۴/۰۷
ویرایش : - - - -

دانشگاه صنعتی شاهرود
مدیریت تحصیلات تکمیلی
فرم شماره (۶)

بسمه تعالی

فرم صورتجلسه دفاع پایان نامه تحصیلی دوره کارشناسی ارشد

با تأییدات خداوند متعال و با استعانت از حضرت ولی عصر (عج) جلسه دفاع از پایان نامه کارشناسی ارشد خانم / آقای **سردار افردار افرا** رشته **برق** گرایش : **کنترل**

تحت عنوان : **آموزش شبکه عصبی کوهنن با استفاده از فیلتر ذره ای**

که در تاریخ **۱۳۸۹/۰۴/۰۷** با حضور هیأت محترم داوران در دانشگاه صنعتی شاهرود برگزار گردید به شرح زیر است :

قبول (با درجه : عالی امتیاز ۱۹٫۴) ☑ دفاع مجدد ☐ مردود ☐

۱ـ عالی (۲۰ ـ ۱۹)
۲ـ بسیار خوب (۱۸/۹۹ ـ ۱۸)
۳ـ خوب (۱۷/۹۹ ـ ۱۶)
۴ـ قابل قبول (۱۵/۹۹ ـ ۱۴)

عضو هیأت داوران	نام ونام خانوادگی	مرتبه علمی	امضاء
۱ـ استادراهنما	محمد حدادزاده	استادیار	
۲ـ استاد مشاور	حمید رضا کربلائی	استادیار	
۳ـ نماینده شورای تحصیلات تکمیلی	سید رضا سروش	استادیار	
۴ـ استاد ممتحن	علیرضا پورباقر	استادیار	
۵ ـ استاد ممتحن	علیرضا ابوالقاسمی	استادیار	

تأیید رئیس دانشکده :

تقدیم به پدر و مادر مهربانم

که هر چه دارم از آنهاست

با تشکر از زحمات:

جناب آقای دکتر محمد حداد ظریف

تعهد نامه

اینجانب ..... دانشجوی دوره کارشناسی ارشد / دکتری رشته ..... دانشکده ..... دانشگاه صنعتی شاهرود نویسنده پایان نامه/ رساله ..... تحت راهنمائی ..... متعهد می شوم .

- تحقیقات در این پایان نامه/ رساله 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/ رساله 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 / رساله تأثیرگذار بوده اند در مقالات مستخرج از پایان نامه/ رساله رعایت می گردد.
- در کلیه مراحل انجام این پایان نامه/رساله ، 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/رساله ، 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 ، برنامه های رایانه ای ، نرم افزارها 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 / رساله بدون ذکر مرجع مجاز نمی باشد.

* متن این صفحه نیز باید در ابتدای نسخه های تکثیر شده پایان نامه/ رساله وجود داشته باشد .

چکیده

در این پایان‌نامه، به بررسی و اصلاح یکی از پرکاربردترین شبکه های عصبی خودسازمانده یعنی شبکه عصبی کوهنن با استفاده از الگوریتم توانمند فیلتر ذره‌ای پرداخته‌ایم. شبکه عصبی کوهنن کاربرد فراوانی در مسائلی همچون تخمین چگالی احتمال، پردازش تصویر و صوت، و روباتیک دارد. در بسیاری از مسائل مهندسی خصوصاً در مواردی که یا چگالی احتمال سیستم نامعین است و یا سیستم تحت نویز و اغتشاش غیر گوسی قرار می گیرد، الگوریتم‌های مونت کارلو و خصوصاً فیلتر ذره‌ای، از کارایی خیره‌کننده‌ای در مقایسه با الگوریتم های قدرتمندی همچون فیلترکالمن توسعه یافته، برخوردارند. امروزه این الگوریتم ها مورد توجه اغلب محققین قرار گرفته است.

یک مشکل اساسی در الگوریتم شبکه عصبی کوهنن ایجاد سلول های مرده پس از گذشت مدت زمان کوتاهی از عمر اجرای الگوریتم است. این مساله باعث می شود تا بعضاً قسمت بزرگی از سلول های شبکه دیگر در فرایند به هنگام سازی شرکت نکرده که باعث تضعیف عملکرد الگوریتم می شود. در راستای حل این مشکل روش های اصلاحی گوناگونی ارائه شده و در این پایان‌نامه ما نیز یک روش جدید برای اصلاح الگوریتم و بهینه نمودن عملکرد آن، ارائه نموده‌ایم. درواقع ضرایب شبکه عصبی کوهنن با استفاده از الگوریتم SIR تخمین زده شده است که این کار نه تنها مشکل اساسی سلول مرده را برطرف نموده، بلکه باعث بهبود عملکرد شبکه در تخمین تابع چگالی احتمال و بالاتر رفتن دقت اجرای الگوریتم نیز شده است.

مزیت استفاده از الگوریتم فیلتر ذره ای برای آموزش شبکه عصبی کوهنن در دقت فزاینده‌ی روش مونت کارلو در پیش‌بینی و تخمین براساس داده های محدود در دسترس، عدم استفاده از پسخور برای مقایسه خروجی با داده های حاصل از مشاهده و اندازه گیری، و همچنین همخوانی این الگوریتم با الگوریتم یادگیری شبکه عصبی کوهنن و به طور کلی با دسته بسیار بزرگ از اینگونه شبکه ها موسوم به شبکه های خودسازمانده بدون ناظر می باشد.

کلمات کلیدی: شبکه عصبی خودسازمانده، شبکه عصبی کوهنن، فیلتر ذره ای، یادگیری، روش مونت کارلو، نمونه برداری اهمیت، نمونه برداری مجدد اهمیت.

فهرست مطالب

فهرست اشکال

فصل اول

مقدمه

۱-۱- انگیزه ها

امروزه مدلها و مدلسازی در مرکز توجه علوم گوناگون قرار دارند که به تدریج در بسیاری از زمینه های مورد علاقه و پرکاربرد علمی از جمله علوم اقتصادی، علوم مهندسی، پزشکی، علوم سیاسی، جامعه شناسی و مدیریت نفوذ کرده اند. از مدلها می توان در پردازش داده ها برای پیش بینی رخدادهای آتی، تخمین و برآورد نتایج و یا حتی شکل دهی داده ها برای استخراج اطلاعاتی خاص استفاده نمود. مدلها در واقع انتزاعی از واقعیت هستند که به واسطه آنها می توان تجربیات ملموسی خلق نمود که به افزایش فهم ما از پدیده های طبیعی کمک می کند.

در حال حاضر دو روش معمول و شناخته شده برای برای ایجاد مدلها وجود دارد. روش اول استفاده از طبیعت استنتاجی است. این روش بر پایه تقسیم سیستم به چندین زیرسیستم است به گونه ای که هر قسمت با استفاده از روابط فیزیکی و ریاضی معمول قابل بیان باشد. این زیرسیستم ها نوعاً به صورت بلوک های شبیه ساز و مجموعه ای از معادلات دیفرانسیل مرتب می شوند که مدل نهائی از ترکیب همه این زیرمدل ها بدست می آید.

در روش دوم از راهبرد استقرایی مدل های تخمین زده شده ی بدست آمده از داده های اندازه گیری بهره می برند. این فرآیند تخمین، یادگیری از داده ها و یا به اختصار یادگیری نامیده می شود. در این پایان نامه، یادگیری، بر یافتن الگوها میان داده ها یا دستیابی به یک نمایش صرفه جو از داده ها که کاربردهای فراوانی نظیر پیش بینی یا طبقه بندی دارد، دلالت می کند. متد یادگیری به دلیل نبود یک تئوری منسجم و جامع درباره سیستم های غیرخطی، از اهمیت ویژه ای در مدل سازی غیر خطی برخوردار است. در ادامه به ذکر برخی نکات در خصوص یادگیری می پردازیم.

در ریاضیات به مساله ای خوش قلق می گویند که دارای سه شرط زیر باشد:

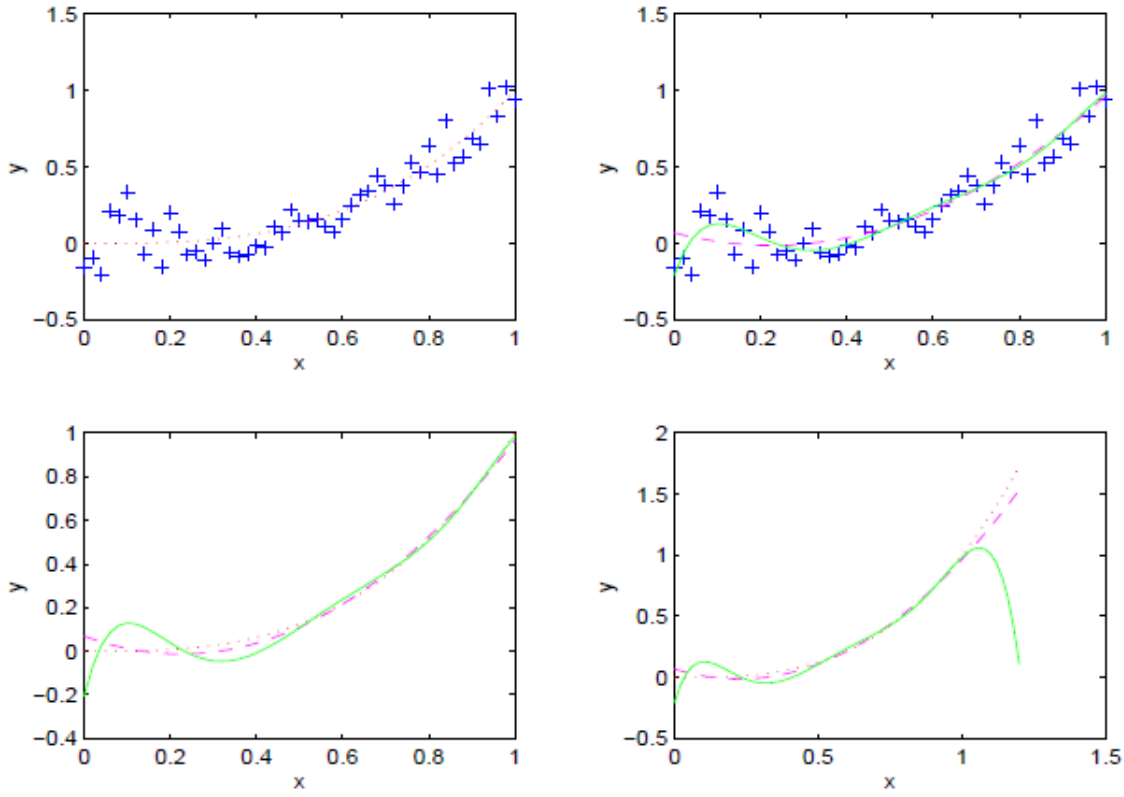
۱) دارای پاسخ باشد؛
۲) این پاسخ یکتا باشد؛
۳) پاسخ به مجموعه ی پیوسته ای از داده ها وابسته باشد.

پرواضح است که در مورد مساله یادگیری چنین شروطی تماماً یا بعضاً برقرار نمی باشد، بنابراین یادگیری از داده ها یک مساله بدقلق است، بدین معنا که برای هر مجموعه خاص از داده ها یک سلسله از پاسخ ها وجود دارد و در نتیجه برای رفع این مشکل باید شروط و محدودیت های خاصی برای رسیدن به پاسخ اعمال شوند. گاهی یک دانش قبلی درباره پدیده ای که باید مدلسازی

شود به صورت یک سری قوانین فیزیکی، ایده های قابل قبول عمومی و یا روابط ریاضی در دسترس هستند. این دانش باید به منظور کاهش مجموعه های پاسخ ممکن به یک مجموعه قابل قبول، در فرآیند مدل سازی اعمال شود. طبیعت بدقلق و دیگر دشواری های ذاتی موجود در مساله یادگیری به راحتی با یک مثال درونیابی نویزی ساده قابل درک خواهد بود [۱۱]. به داده های رسم شده در شکل (۱-۱) که با معادله (۱-۱) تولید شده اند دقت کنید؛

$$y = x^3 + \eta \qquad (۱-۱)$$

که در اینجا x^3 نشانگر رابطه ی صحیح بین ورودی x وخروجی y و η نمایانگر نویز با توزیع یکنواخت و میانگین صفر است. وظیفه الگوریتم یادگیری تخمین رابطه واقعی و صحیح بین ورودی و خروجی با استفاده از داده های نویزی رسم شده در شکل (۱-۱) است.



شکل (۱-۱): نمودارهای مربوط به تخمین

می توان داده ها را با تطبیق چندجمله ای هایی از درجات مختلف با استفاده از روش حداقل مربعات مدل کرد. در اینجا ما این کار را با استفاده از چندجمله ای های از درجه دو و شش انجام داده ایم. همانطور که از شکل (۱-۱) پیداست، اگرچه چندجمله ای درجه شش تخمین بهتری نسبت به بسجمله درجه دو بدست داده است، اگر تابع صحیح را به همراه این دو تخمین رسم نماییم مشاهده می کنید که چندجمله ای درجه دو نه تنها تخمین بهتری بدست داده بلکه برونیابی بهتری نیز برای پیش بینی داده های بعدی ارئه نموده است.

همانطور که در این مثال مشاهده کردید تخمینگرهای پیچیده اگرچه ممکن است تخمین بهتری برای داده های اندازه گیری شده بدست دهند، اما همواره تخمین مشابه تابع اصلی ارائه نمی کنند. به بیان دیگر مقادیر پیش بینی شده توسط آنها به ازای داده هایی غیر از داده های اندازه گیری یک تخمین اشتباه و غیر واقعی خواهد بود. این توانایی در تخمین داده های آتی یا همان برونیابی یکی از خواصی است که در تخمین و پیش بینی داده ها برای مهندسین از اهمیت ویژه ای برخوردار است. نکته ی دیگری که حائز اهمیت است آن است که اگر ما ویژگی های نویز را به عنوان یکی از معلومات مساله فرض کنیم آنگاه مسلماً تخمین با بسجمله درجه شش یک تخمین واقعی تر از داده هاست. بالعکس، اگر ما داده ای در بازه‌ی [1,1.5] داشته باشیم در این صورت دیگر تخمین ارائه شده توسط چندجمله‌ای درجه شش نمی‌تواند قابل قبول باشد. این نکات بیانگر اهمیت مفروضات و معلومات پیشین در یک مساله، بازه‌ی عملکرد توابع تخمینگر و اندازه مجموعه داده‌ها در مسائل یادگیری است.

مثال فوق به راحتی تمامی مشکلات پیش روی در حل یک مساله یادگیری را هنگامی که سعی داریم از روی مقادیر داده های اندازه گیری شده که با نویز آمیخته شده‌اند مدل واقعی سیستم را بیابیم، آشکار می کند. مشکلاتی از قبیل:

- یادگیری یک مساله بدقلق باتعداد نامتناهی پاسخ است؛
- نویز و دیگر محدودیت های سیستم امکان ارائه یک پاسخ بسته عمومی را سلب می کند؛
- در فرآیند حل مساله گاه ناگزیر از انتخاب ساختارهای گوناگون غیرخطی برای ارائه تخمین بهتر هستیم؛
- و در نهایت باید یک الگوریتم روش مند برای پیاده سازی مدل پیشنهادی روی داده ها ارائه نماییم.

در خاتمه باید خاطرنشان ساخت که در بسیاری از مسائل تخمین و یادگیری با مشکلات بزرگتری همچون دینامیک بودن ساختار مساله روبرو هستیم به این معنا که تمامی مقادیر آماری یا همان آماره سیستم متغیر با زمان خواهد بود.

۱-۲- تاریخچه

ترکیب مدل سازی احتمالاتی با تقریب زننده های توابع غیرخطی یک ابزار توانمند در حل بسیاری از مسائل دنیای واقعی است. در علوم مهندسی مثالهای فراوانی از مسائلی با مجموعه داده های گسترده وجود دارند که توسط تقریب زننده های توابع غیرخطی همانند مدل چندلایه ای پرسپترون یا مدل توابع پایه شعاعی، حل شده اند. همچنین مثالهای متعددی از اتفاده فیلتر کالمن برای تخمین وزن سلولها در شبکه های عصبی وجود دارد. از شبکه های عصبی در الگوریتم های تشخیص صحبت (رابینسون۱۹۹۴) و تشخیص ارقام دست نوشته (هووارد، هندرسون، دنکر، لو چان و باسر ۱۹۸۹) استفاده شده که به دلیل دقت در تخمین ها و عملکرد خوب این شبکه ها بوده است. اولین بار ردفورد نیل از تکنیکهای نمونه برداری در شبکه های عصبی استفاده نمود (نیل ۱۹۹۶) و نشان داد تقریب زننده های توابع غیرخطی به فرم شبکه های عصبی پرسپترون قابل پیاده سازی هستند و عملکرد آنها در یک قالب بیزی ارزیابی می شود. پس از این، روشهای گوناگون نمونه برداری دیگری همچون روش نمونه برداری متوالی یا ترتیبی مونت کارلو و روش زنجیره مارکوف-مونت کارلو نیز مورد استفاده قرار گرفتند (مک کی ۱۹۹۲ و بی شاپ ۱۹۹۵). تمامی این روشها کاربردهای فراوانی در علوم کامپیوتر همانند ردیابی در مسائل بینائی ماشین (ایسارد و بلیک ۱۹۹۶)، GPS و الگوریتم های پردازش صدا و تصویر و مسائلی از این دست دارند [۱]، [۱۵] و [۱۹].

۱-۳ اهداف پایان نامه

پیشرفت های اخیر نشان داده است که فیلتر ذره ای یک روش شناسایی قدرتمند و نوظهور با گستره ی وسیعی از کاربردها در علوم و مهندسی است. این روش توجه بسیاری از محققین را در شاخه های مختلف علوم به خود جلب کرده که این علاقه در توانایی آن در کنار آمدن و تطبیق با مسائل غیرخطی و غیرگوسی ریشه دارد. در عمل زمانی که با یک سیستم خطی با نویز گوسی جمع

شونده برخورد می کنیم فیلتر کالمن یک پاسخ بهینه برای آن ارائه می دهد. در مسائلی که نویز غیرگوسی باشد و یا با یک مساله ی غیرخطی روبرو باشیم معمولترین روش استفاده از فیلتر کالمن توسعه یافته است. اما در بسیاری از حالات این فیلتر بایاسی به سیستم تحمیل میکند که نه تنها واگرائی زیادی ایجاد می کند بلکه سبب کاهش پایداری سیستم و ویژگی مقاومت پذیری آن در برابر تغیرات نیز می گردد. این نکات حاکی از عملکرد بسیار ضعیف این فیلتر و عدم اطمینان به پاسخ های ارائه شده آن می باشد حال آنکه تجربیات نشان داده که فیلتر ذره ای از بوته چنین مسائلی به راحتی بیرون می آید. این توانایی فیلتر ذره ای در برخورد با سیستم های غیرخطی با نویز غیرگوسی به دلیل کاربرد مفهوم نمونه برداری با اهمیت ترتیبی و با استفاده از تئوری بیز حاصل شده است. اساسی ترین اصلی که این روش بر آن پایه گذاری شده است تقریب توزیع های وابسته از طریق اندازه گیری های تصادفی است که شامل ذره ها یا نمونه هایی از فضای کاری و وزن های آنها هستند. درواقع روش فیلتر ذره ای یک روش شبیه سازی مونت کارلو است که برای تقریب زدن فیلترهای غیرخطی طراحی می شوند که حالات یک سیستم پویا را تخمین زده و دنبال می کند. همانطور که پیشتر اشاره شد ایده اصلی این روش محاسبه بازگشتی توابع توزیع احتمال وابسته با استفاده از مفهوم نمنونه برداری با اهمیت ترتیبی و تقریب توابع توزیع احتمال توسط اندازه گیری های تصادفی گسسته است.

مکانیسم کاری فیلتر ذره ای به این صورت است که فضای حالت به بخش های (ذره ها) زیادی تقسیم می شود که در آن ذره ها با توجه به برخی اندازه گیری های احتمالاتی رشد می کنند. درواقع ذرات با احتمال بزرگتر چگالتر می شوند. سیستم ذره ای در طول زمان براساس معادلات حالت نمو می یابد و تابع چگالی احتمالی که از معادله فوکر-پلانک-کولموگروف محاسبه می شود نیز نمو می کند.درواقع با استفاده از فیلترهای ذره ای، توزیع های پیوسته توسط اندازه گیری های تصادفی گسسته ای تقریب زده می شوند که خود از ذرات وزندهی شده ای را تشکیل شده اند که این ذرات نمونه هایی از فضای ناشناخته در فضای حالت هستند و وزن های این ذرات جرم های احتمالاتی هستند که با تئوری بیز محاسبه می شوند. مزیت اصلی این روش نسبت به سایر روش ها اینست که تقریب بدست آمده شامل هیچ خطی سازی حول تخمین جاری نیست بلکه بیشتر شامل تقریب هایی در ارائه توزیع های مطلوب با اندازه گیری های تصادفی گسسته می باشد.

بهترین، معمول ترین و شناخته شده ترین مدل از شبکه های عصبی خودسازمانده، نگاشت حافظ توپولوژی است که توسط تئوو کوهنن پیشنهاد شده است. تفاوت اساسی میان شبکه های خودسازمانده و دیگر مدلهای رایج شبکه های عصبی اینست که خروجی صحیح را از طریق قیاس نمی توان تعیین کرد بنابراین یک اندازه گیری عددی از دامنه خطای نگاشت غیرقابل استفاده است. مدل

کوهنن یک مدل بدون ناظر است. در این مدل تعدادی سلول عصبی که معمولاً در یک توپولوژی مسطح کنار یکدیگر چیده می شوند و با رفتار متقابل روی یکدیگر وظیفه شبکه خودسازمانده را که تخمین یک تابع توزیع است ایفا می کنند. بردار $\underline{x} \in \Re^n$ را که هر یک از درایه های آن دارای چگالی احتمال $p_i(x)$ است در نظر بگیرید. در این فضای چگالی نمونه هایی را به تناوب و تصادف انتخاب کرده و به شبکه اعمال می کنیم. براساس موقعیت بردار ورودی در فضای $\Re^n$، وزن سلولها تغییر می کند و در نهایت بردارهای وزن مربوط به سلولها $\underline{w_i} \in \Re^n$ بطور یکنواخت در فضای چگالی احتمال ورودی توزیع می شوند و بدین ترتیب شبکه با پراکندن سلول های خود در فضای ورودی، چگالی احتمال آن را تخمین می زند. حال در این بین با توجه به الگوریتم انتخابی مشکلی برای شبکه ایجاد می شود که به سلول مرده معروف می باشد. درواقع روند الگوریتم شبکه عصبی کوهنن به گونه ای پیش می رود که در نهایت پس از چند گام تنها چند سلول در فرآیند تخمین شرکت می کنند و این یعنی از دست دادن پتانسیل شبکه مبنی بر استفاده از تمامی سلول ها در الگوریتم برای بالا بردن دقت درست مشابه آنچه در مغز انسان رخ می دهد.

در این پایان‌نامه ابتدا توجه خود را به آموزش و پیشگویی متوالی شبکه های عصبی جلب می کنیم و سپس با رویکردی مشابه به پیاده سازی الگوریتم فیلتر ذره ای روی شبکه عصبی کوهنن به منظور ایجاد یک تخمین با دقت بالاتر خواهیم پرداخت. ما از تکنیک های نمونه برداری در مجموعه فضای حالت استفاده می کنیم تا چگونگی آموزش یک شبکه عصبی را به هنگام ورود اولیه داده ها نشان دهیم. همچنین به بررسی عملکرد الگوریتم های نمونه برداری با اهمیت ترتیبی و نمونه برداری های مجدد روی چند مساله پایه ای در تئوری تخمین غیرخطی می پردازیم. و در ادامه قصد داریم تا با استفاده از الگوریتم فیلتر ذره ای در آموزش شبکه کوهنن اولاً با دقت بهتری نسبت به الگوریتم کلاسیک به محاسبه ضرایب شبکه بپردازیم و ثانیاً با اعمال روش نمونه برداری مجدد متوالی (ترتیبی) در گام به هنگام سازی ضرایب شبکه، از بروز مشکل سلول مرده جلوگیری نماییم تا هم بتوان از تمامی قابلیت های شبکه برای تولید خروجی مطلوب بهره برده و هم با دخیل کردن حداکثر تعداد ممکن ازسلولها (نرون ها) در تمامی گام های الگوریتم به بهترین دقت ممکن برای تخمین خروجی دست یابیم.

۱-۴ دورنمای فصول

در فصل دوم ابتدا مساله آموزش شبکه عصبی در فضای حالت را فرمول بندی می کنیم ودر ادامه به معرفی شبکه عصبی خودسازمانده و بالاخص شبکه عصبی کوهنن می پردازیم. متعاقباً در فصل سوم یک جواب تئوریک براساس قانون بیز برای مساله آموزش که در فصل دوم فرموله شد ارائه شده است. از آنجائیکه دشواریهای علمی و عملی به هنگام محاسبه جواب بیزی رخ می نماید، سه روش متداول تخمین انتگرال عددی، تخمین گوسی و روش مونت کارلو برای تخمین پاسخ مساله آموزش شبکه عصبی در فضای حالت، به تفصیل در فصل چهارم آمده است. در فصل پنجم روش عمومی الگوریتم نمونه برداری با اهمیت متوالی را ارائه وبه بیان تفصیلی تمامی شرایط پیاده سازی الگوریتم پرداخته ایم. فصل ششم متشکل از دو بخش است. بخش اول نحوه اعمال الگوریتم فیلتر ذره ای روی شبکه بحث شده است ودر بخش دوم به ارائه چند مثال شبیه سازی شده با استفاده از شبکه بهینه شده پرداخته ایم.

فصل دوم

شبکه های عصبی رقابتی

و

نگاشت های خودسازمانده

۲-۱- مقدمه

قانون یادگیری شبکه های عصبی رقابتی، در گروه یادگیری بدون ناظر قرار می گیرد. در این شبکه ها، چنانکه از نام آنها برمی آید، خروجیهای سلول های عصبی با یکدیگر به رقابت می پردازند تا یکی از آنها که دارای امتیاز بیشتری است در رقابت برنده شده در بین سایر سلولها متمایز شود. مدل همینگ را شاید بتوان یکی از ساده ترین نمونه های شبکه های رقابتی دانست. چنانکه می دانید لایه اول و سوم این شبکه از نوع پیشخور هستند. لایه ی اول در واقع عمل همبستگی ورودی با کلاسهای مرجع را انجام می دهد و لایه سوم هم که در حکم لایه خروجی کل مدل است، الگوی مرجعی که از بیشترین شباهت به ورودی برخوردار است را تعیین می کند.

این فرآیند به کمک رقابت بین سلولهای عصبی در لایه دوم صورت می گیرد. این لایه عملاً یک شبکه رقابتی را نمایندگی می کند. سلولی برنده رقابت است که شماره کلاس مرجع دارای بیشترین شباهت به ورودی را تعیین می کند. با انتخاب مناسب مقادیر بردار وزن لایه دوم، فرآیند رقابت از طریق مکانیسمی موسوم به مهار جانبی ایفا می شود. در شبکه های انجمنی مبتنی بر قانون یادگیری هب، ممکن است چندین سلول در لایه خروجی بطور همزمان فعال شوند. در این بخش خواهیم دید که چگونه ترکیب قوانین یادگیری رقابتی و قوانین یادگیری انجمنی به پیدایش شبکه های عصبی رقابتی بدون ناظر یا همان شبکه های عصبی خودسازمانده، به عنوان یک ابزار توانمند در حل بسیاری از مسائل عملی، منجر می شود.

ایده اولیه یادگیری رقابتی به اوایل دهه شصت برمی گردد. در آن هنگام فرانک روزنبلات[1]، یک شبکه بدون ناظر مبتنی بر پرسپترون معرفی کرد که می توانست بردارهای ورودی را به دو طبقه با تعداد عناصر نسبتاً برابر تقسیم بندی نماید. در سالهای آخر دهه شصت و اویل دهه هفتاد، استفان گراسبرگ[2] تعداد زیادی از شبکه های رقابتی که براساس ایده مهار جانبی، توانایی حذف نویز و نرمالیزه کردن بردارها را داشتند، معرفی نمود. در سال ۱۹۷۳ فون درمالسبورک، یک قانون رقابتی خودسازمانده مبتنی بر ایده یادگیری رقابتی معرفی کرد. بر اساس این قانون یادگیری، شبکه قادر است ورودیها را طوری طبقه بندی کند که سلولهای مجاور به ورودیهای مشابه پاسخ دهند. مشابه این عمل در قشر بینایی و شنوایی مغز انجام می گیرد. قانون یادگیری مالسبورگ برای تضمین نرمالیزه کردن از اطلاعات غیر محلی استفاده می نمود که البته این امر توجیه بیولوژیکی نداشت.

[1] Frank Rosenblatt
[2] Stephan Grassburg

محقق بزرگ دیگری که در زمینه شبکه های عصبی و بطور ویژه شبکه های عصبی رقابتی خودسازمانده نقش بارزی ایفا نموده، تئو کوهنن است. او روی کاربردهای مهندسی و توصیفهای ریاضی گونه از شبکه های رقابتی تاکید داشت. در دهه هفتاد او نوع ساده ای از قانون یادگیری را معرفی و نیز یک راه مؤثر برای پیاده سازی شبکه های رقابتی پیشنهاد کرد.

در ادامه فصل قصد داریم چهارچوب شبکه های رقابتی معرفی شده توسط کوهنن را مورد بررسی قرار دهیم و این کار را با توضیح مختصری روی شبکه عصبی همینگ آغاز می کنیم.

۲-۲- شبکه همینگ [۱۸]، [۱۹] و[۲۹]

به دلیل شباهت زیادی که بین شبکه های رقابتی و مدل همینگ وجود دارد در ادامه به شرح کوتاهی از خصوصیات این مدل می‌پردازیم. شبکه همینگ همانطور که در شکل (۲-۱) رسم شده است از سه لایه تشکیل می شود؛ نخستین لایه در واقع لایه ای از نرونهای اینستار[۳] می باشد. این لایه همبستگی بین بردار ورودی و کلاس مرجع را محاسبه می کند. در لایه دوم عمل رقابت بین سلولها انجام می گیرد و سلول برنده شماره کلاس مرجع مربوط به الگوی ورودی را نشان می دهد. لایه سوم نیز که مجموعه ای از سلولهای اوتستار[۴] است الگوی مرجع انتخاب شده را ارائه می کند.

لایه اول:

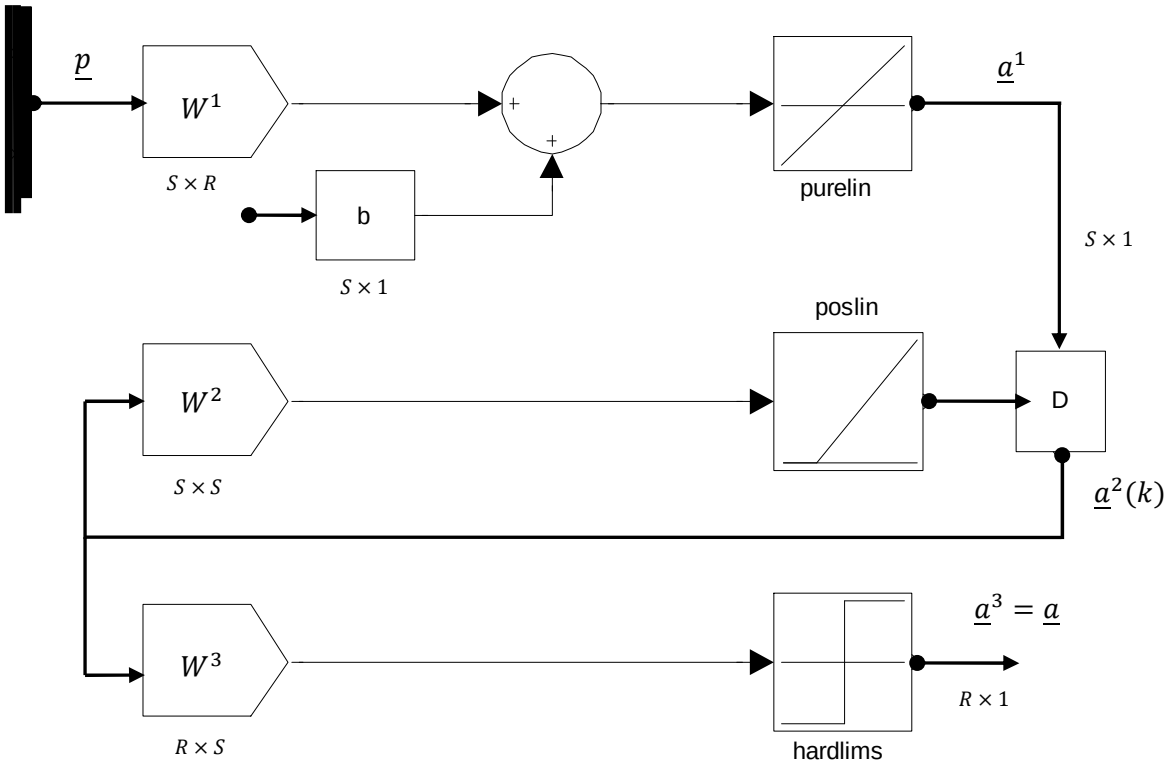
همانطور که می دانید یک سلول اینستار تنها قادر به شناسایی یک الگو است. برای شناسایی و طبقه بندی بیش از یک الگو مسلماً به بیشتر از یک سلول اینستار نیاز خواهیم داشت. این روند در مدل همینگ انجام می شود. فرض کنیم می خواهیم Q الگوی مرجع $\underline{p}^1, \underline{p}^2, \cdots, \underline{p}^Q$ را در شبکه ذخیره کنیم. ماتریس وزن W^1 و بردار $\underline{b}^1$ به ترتیب زیر انتخاب می شوند:

$$W^1 = \left[\underline{p}^1, \underline{p}^2, \cdots, \underline{p}^Q\right]^T, \qquad (۱-۲)$$

$$\underline{b}^1 = [R, R, \cdots, R]^T. \qquad (۲-۲)$$

[۳] شبکه عصبی با ورودی برداری را اینستار(In Star) گویند.

[۴] شبکه عصبی با خروجی برداری را اوتستار (Out Star) گویند.



شکل (۲-۱) : شبکه همینگ

هر سطر ماتریس W^1 یک الگوی مرجع و هر المان بردار $\underline{b}^1$ تعداد عناصر بردار ورودی R را نشان می دهد. تعداد سلولها S، برابر است با تعداد کلاسهای مرجع، Q، که قرار است تشخیص داده شوند. خروجی لایه اول برابر است با :

$$\underline{a}^1 = W^1\underline{p} + \underline{b}^1 = \begin{pmatrix} \left(\underline{p}^1\right)^T \underline{p} + R \\ \vdots \\ \left(\underline{p}^Q\right)^T \underline{p} + R \end{pmatrix}. \qquad (۳-۲)$$

لایه دوم:

این لایه، لایه رقابت بین سلولها و تعیین سلول برنده است. در شبکه های کلاسیک غیر رقابتی به جای سلول اینستار، از تابع تبدیل "sign" استفاده می شود تا در مورد میزان نزدیکی بین بردار ورودی و کلاس مرجع قضاوت شود. در لایه دوم از مدل همینگ تعدادی سلول اینستار وجود دارد و بنابراین مایلیم که بدانیم کدام یک از بردارهای مرجع به بردار ورودی شباهت بیشتری دارد و به جای استفاده از تابع تبدیل "sign" از یک لایه رقابتی جهت انتخاب بردار مرجع استفاده می شود.

سلول های لایه دوم مقدار اولیه خود را از خروجیهای لایه اول دریافت می کنند:

$$\underline{a}^2(0) = \underline{a}^1, \qquad (۴-۲)$$

$\underline{a}^1$ میزان همبستگی بین الگوهای مرجع و الگوی ورودی را تعیین می کند، سپس سلولهای لایه دوم با هم رقابت می کنند. پس از پایان رقابت، تنها یک سلول دارای خروجی غیر صفر و بقیه سلولها تماماً دارای خروجی صفر هستند. سلول برنده، کلاس مرجع استنتاج شده را اعلام می کند. عملکرد لایه دوم به صورت زیر بیان می شود:

$$\underline{a}^2(k+1) = posl\left(w^2\underline{a}^2(k)\right). \qquad (۵-۲)$$

w^2 ماتریس وزنهای لایه دوم بوده و به صورت زیر تعریف می شود:

$$w_{ij}^2 = \begin{cases} 1 & i = j \\ -\varepsilon & i \neq j \end{cases}, 0 < \varepsilon < 1/(S-1). \qquad (۶-۲)$$

این ماتریس موجب می شود که خروجی هر سلول دارای تاثیری بازدارنده روی دیگر سلولها داشته باشد. به عبارت بهتر این ماتریس نقش مهار جانبی را در لایه دوم ایفا می کند.

$$a_i^2(k+1) = posl\left(a_i^2(k) - \varepsilon\sum_{\substack{j\neq i \\ j=i}}^{S} a_j^2(k)\right), i = 1,2,\cdots,S. \qquad (۷-۲)$$

در هر تکرار، خروجی هر سلول متناسب با جمع خروجیهای دیگر سلولها کاهش می یابد. خروجی سلول دارای بیشترین مقدار اولیه، خیلی کندتر از دیگر سلولها که دارای مقادیر اولیه کوچکتر هستند کاهش می یابد. نهایتاً همان سلولی که دارای بزرگترین مقدار اولیه است (و همانطور که می دانید این مقدار اولیه میزان شباهت کلاس مربوطه و بردار ورودی را نشان می دهد)، دارای مقدار خروجی غیرصفر و بقیه سلولها دارای خروجی صفرخواهند بود. این وضعیت حالت ماندگار شبکه است و اندیس سلول در لایه دوم، عملاً اندیس بردار مرجع خواهد بود که بیشترین همبستگی را با بردار

ورودی دارد. مثلاً اگر سلول دوم برنده شده باشد، بردار مرجع دوم، کلاس مرجع مورد نظر است. چون تنها یک سلول دارای مقدار خروجی غیر صفر می باشد، این گونه رقابت موسوم به رقابت "برنده همه را می برد" یا WTA[5] می باشد.

لایه سوم:

لایه سوم در مدل همینگ، عملاً یک شبکه ساده بازیابی است که توسط یک سلول اوتستار مشخص می شود. در این لایه الگوی مرجعی که نزدیکترین شباهت را به بردار ورودی دارا می باشد، ظاهر می شود. ماتریس وزن لایه سوم به ترتیب زیر انتخاب می شود:

$$w_{ij}^{3} = p_i^j, \quad j = 1,2,\cdots,S; i = 1,2,\cdots,R. \qquad (۸-۲)$$

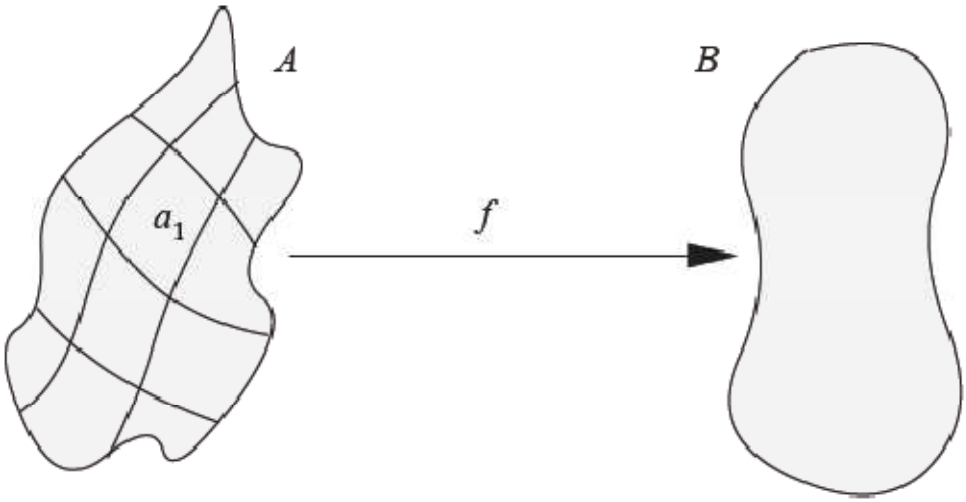
۲-۳- شبکه خود رقابتی سازمانده

در این بخش، شبکه های خودسازمانده را مورد بررسی قرار می دهیم. تفاوت اصلی بین این شبکه ها و مدل های مرسوم در این است که خروجی صحیح را نمی توان به عنوان پیش فرض تعریف نمود و بنابراین اندازه گیری عددی روی بزرگی خطای نگاشت نمی تواند مورد استفاده قرار گیرد. هرچند، فرآیند آموزش مانند قبل، به تعیین پارامترهای شبکه ی خوش- تعریف برای یک کاربرد معین می انجامد.

۲-۳-۱- نمودار فضای ورودی

دراینجا مثالی از یک شبکه خودسازمانده را آورده ایم؛ می دانیم که در یک شبکه خودسازمانده خروجی صحیح از قبل تعریف نمی شود و نگاشت بردارهای وزنی به مراکز خوشه ها یک فرآیند خودکار است.

[5] Winner Take All



شکل (۲-۲): تابع $f: A \to B$

وقتی یک شبکه خودسازمانده مورد استفاده قرار می گیرد، در هر گام یک بردار ورودی ارائه می شود. این بردارها "محیط" شبکه را تشکیل می دهند. هر ورودی جدید، یک انطباق پارامترها را تولید می کند. اگر چنین تغییراتی به درستی کنترل شوند، شبکه می تواند نوعی از بیان درونی محیط را بسازد. از آن جا که در این شبکه ها، فازهای آموزش و "تولید" ممکن است با هم تداخل کنند، این بیان می تواند به طور مستمر به روز شود.

معروف ترین و محبوب ترین مدل شبکه های خودسازمانده، طرح نگهدارنده توپولوژی[۶] است که توسط توو کوهنن[۷] پیشنهاد شده است [۲۴]. شبکه های کوهنن برمبنای ایده هایی هستند که توسط روزنبلات[۸]، فون د. مالزبورگ[۹] و محققین دیگر معرفی شده است. اگرقرار باشد یک فضای ورودی توسط شبکه عصبی پردازش شود، اولین نکته حائز اهمیت، ساختار این فضا می باشد. یک شبکه عصبی با ورودی های واقعی، تابع f را که از یک فضای ورودی A به فضای ورودی B تعریف شده است محاسبه می کند. محدوده ای که تابع f در آن تعریف شده است، به شکلی که برای مثال هنگام انتخاب یک بردار ورودی از محدوده a1 در شکل (۲-۲) فقط یک واحد شبکه آتش شود، می تواند توسط شبکه کوهنن تحت پوشش قرار گیرد. چنین افرازی که در آن فضای ورودی به زیرمحدوده هایی دسته بندی می شود، نمودار یا طرح فضای ورودی نیز نامیده می شود. شبکه های کوهنن به یک روش خود سازمانده، ایجاد طرح هایی از فضای ورودی را می آموزند.

---

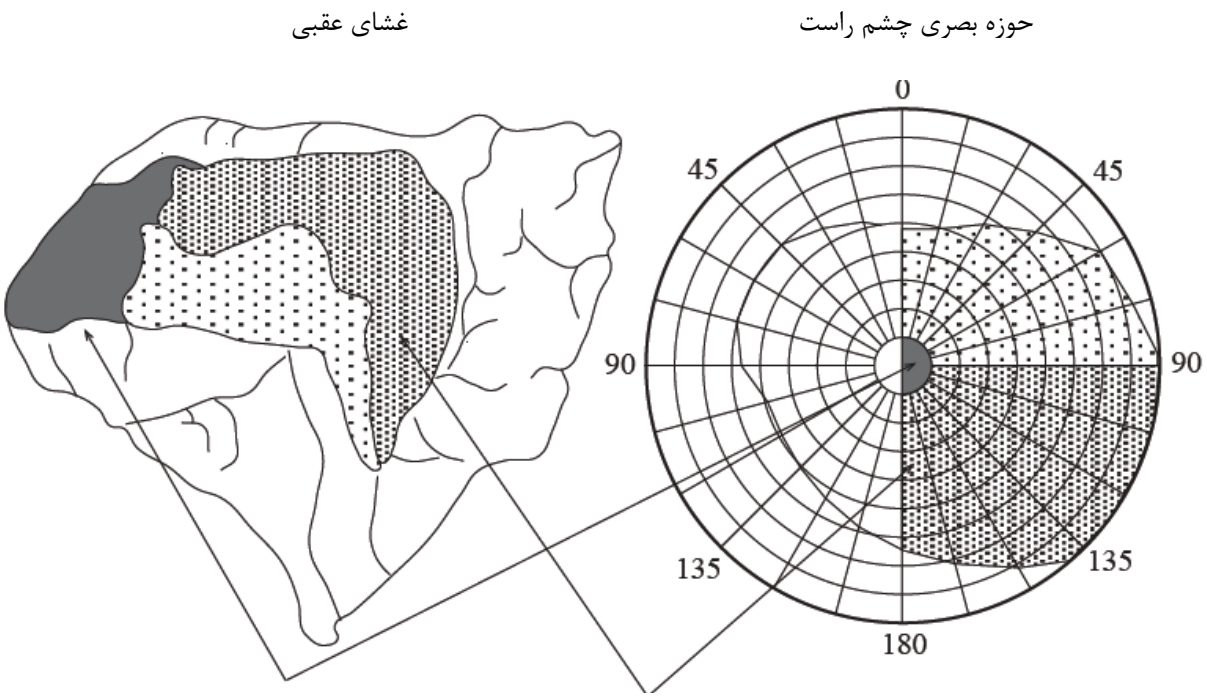
6 topology-preserving
7 Teuvo Kohonen
8 Rosenblatt
9 von der Malsburg

۲-۳-۲- طرح های نگهدارنده توپولوژی

مدل کوهنن یک پیش زمینه زیست شناختی و ریاضیاتی دارد. در زیست شناسی اعصاب معروف است که بعضی ساختارها در مغز، یک توپولوژی خطی یا سطحی دارند؛ یعنی در یک یا دو بعد بسط می یابند. از طرف دیگر، تجربه حسی چندبعدی است. یک رخداد ساده مانند تشخیص رنگ، تعامل سه نوع مختلف گیرنده های نوری را لازم می دارد. چشم ها اطلاعات اضافی درباره ساختار، موقعیت و بافت اشیاء را نیز ثبت می کنند. سؤال این است: چگونه ساختار های سطحی مغز، چنین سیگنال های چندبعدی را پردازش می کنند؟ به عبارت دیگر: چگونه ورودی چندبعدی به ساختارهای عصبی دوبعدی تصویر می شود؟ این سؤال مهم را می توان با دو مثال توضیح داد.

غشای بصری یکی از نواحی شناخته شده در بخش عقبی مغز انسان است. نورون های بسیاری برای رمزگشایی و پردازش ادراک بصری با هم کار می کنند. جالب است که بدانیم اطلاعات بصری علی رغم پیچیدگی مسیر از شبکیه تا ساختارهای غشائی، به عنوان یک تصویر دوبعدی روی این غشاء نگاشت می شود. شکل (۲-۳) طرحی از غشای بصری را نشان می دهد که بسیار شبیه به طرحی است که در ابتدای قرن حاضر توسط گوردن هولمز[10] کشف شد [۲۴]. شکل سمت راست کل حوزه بصری و نواحی مختلف آن را نشان می دهد. دایره داخلی بیانگر مرکز حوزه بصری است که محتویات آن توسط حفره ضبط می شود. شکل سمت چپ با استفاده از سه نوع هاشور تناظر بین نواحی غشای بصری و نواحی حوزه بصری را نشان می دهد. پدیده مهمی را می توان مشاهده نمود: اول این که نواحی همسایه در حوزه بصری توسط نواحی همسایه در غشاء پردازش می شوند؛ دوم این که سطح کنار گذاشته شده از غشای بصری برای پردازش اطلاعات از حفره به طور نامتناسبی بزرگ است. این به آن معنی است که سیگنال ها از مرکز حوزه بصری با جزئیات بیشتر و تفکیک پذیری بالاتری نسبت به سیگنال های محیط حوزه بصری پردازش می شوند. تیزحسی بصری از محیط به مرکز افزایش می یابد.

[10] Gordon Holmes



شکل (۲-۳): نگاشت حوزه بصری روی غشاء

بنابراین غشای بصری یک نوع طرح از حوزه بصری محسوب می شود. از آن جا که این یک تصویر از فضای بصری به گیرنده های نوری کروی در شبکیه است، تناظر کاملی بین شبکیه و غشاء، نیازمند وجود ترکیب بندی کروی در دومی است. هرچند، مرکز حوزه بصری به ناحیه نسبتاً بزرگی از غشاء نگاشت می شود. بنابراین شکل غشای تعمیم یافته می بایست به یک کره تغییرشکل یافته شبیه باشد. آزمایشات جامعه شناختی چنین فرضی را تأیید می کنند.

در غشای انسانی ما نه تنها یک شکل منظم و توپولوژیکی از حوزه بصری می یابیم بلکه تأثیرات حسی از ارگان های دیگر را نیز می بینیم. شکل (۲-۳) برشی از دو ناحیه از مغز را نشان می دهد که در سمت راست، غشای حسی که مسئول پردازش ورودی های مکانیکی است قرار دارد و در سمت چپ غشای محرک نشان داده شده است که حرکت اختیاری قسمت های مختلف بدن را کنترل می کند. هر دوی این نواحی در هر نیم کره مغز وجود دارند و همجوار یکدیگر هستند.

شکل (۲-۳) نشان می دهد که نواحی مغز که مسئول پردازش سیگنال های بدن هستند، به روش نگهدارنده توپولوژی توزیع شده اند. برای مثال ناحیه مسئول سیگنال های بازوها، نزدیک به

ناحیه مسئول سیگنال های دست ها قرار دارد. همان طور که مشاهده می شود، روابط فضایی بین اجزای بدن تا جایی که ممکن است در غشای حسی حفظ شده است. همین پدیده را می توان در غشای محرک نیز مشاهده نمود.

البته هنوز همه جزئیات درباره چگونگی پردازش سیگنال های حسی توسط غشاء روشن نیست. هرچند، به نظر می رسد این فرض صحیح باشد که اولین تعریف از جهان که توسط مغز ساخته می شود، یک تعریف توپولوژیکی است که در آن روابط فضایی خارجی به روابط فضایی مشابه در غشاء نگاشت می شود. ممکن است فکر کنیم که نگاشت جهان حسی به مغز از نظر ژنتیکی تعیین شده است اما آزمایشات روی گربه هایی که از یک چشم نابینا هستند نشان داده است که آن نواحی از غشاء که در یک گربه معمولی اطلاعات را پردازش می کند، از چشم نابینا دوباره تطابق می یابد و می تواند اطلاعات را از چشم دیگر پردازش کند. این تنها در صورتی رخ می دهد که گربه بینایی یک چشم خود را زمانی از دست بدهد که مغز هنوز درحال توسعه است یعنی وقتی که مغز هنوز انعطاف کافی دارد. این به این معنی است که تناظر توپولوژیکی بین شبکیه و غشاء از نظر ژنتیکی به طور کامل معلوم نیست و این که تجربه حسی برای توسعه مداربندی عصبی درست، ضروری است [۲۴].

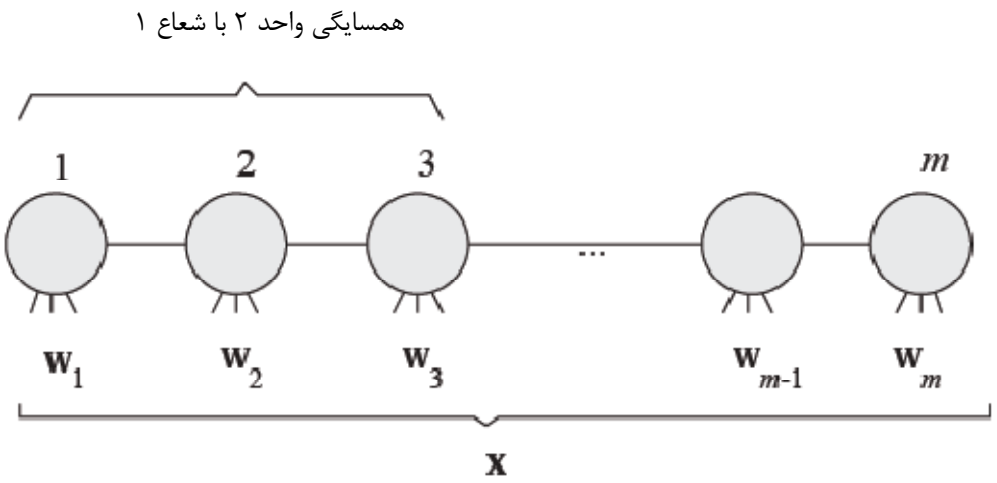
مدل کوهنن از شبکه های خودسازمانده به قلب این پدیده وارد می شود. مدل او با المان هایی کار می کند که از المان های مورد استفاده محققان دیگر چندان متفاوت نیست. مرتبط تر همان تعریف همسایگی یک واحد محاسبه گر است. شبکه های کوهنن شامل تنظیماتی از گره های محاسبه گر در شبکه های توری یک، دو یا چندبعدی است. این واحدها به چندین همسایه اتصالات جنبی دارند. نمونه هایی از این نوع تزویج جانبی، اتصالات بازدارنده مورد استفاده توسط فون د مالزبورگ در مدل های خود سازمانده او می باشد [۲۴]. اتصالاتی از این نوع در شبکیه چشم انسان نیز موجود است.

۲-۴- مدل کوهنن

در این بخش، مثال هایی از ساختارهای منظمی که تحت عنوان شبکه های کوهنن شناخته می شوند مورد بررسی قرار می دهیم. شبکه المان های محاسبه گر به ما اجازه می دهد همسایه های بلاواسطه یک واحد را شناسایی کنیم. این موضوع بسیار مهم است چرا که در حین یادگیری، وزن های واحد های محاسبه گر و همسایگان آنها به روز می شوند. هدف این روش یادگیری این است که واحدهای همسایه یاد بگیرند نسبت به سیگنال های بسیار مرتبط واکنش نشان دهند.

۲-۴-۱- الگوریتم یادگیری

مسأله طرح یک فضای n بعدی با استفاده از یک زنجیره یک بعدی واحدهای کوهنن را درنظر بگیرید. این واحدها همگی به ترتیب منظم شده و از ۱ تا m شماره گذاری شده اند (شکل (۲-۴)). هر واحد، ورودی n بعدی x است و تحریک متناظر با خود را محاسبه می کند. بردارهای وزنی n بعدی w1,w2,…,wm ، برای محاسبه مورد استفاده قرار می گیرند. هدف فرآیند طرح ریزی این است که هر واحد یاد بگیرد روی نواحی مختلف فضای ورودی متخصص شود. وقتی یک ورودی از چنین ناحیه ای به شبکه تزریق می شود، واحد متناظر باید تحریک حداکثر را محاسبه کند. الگوریتم یادگیری کوهنن برای تضمین این که این اثر به دست آید به کار می رود.



شکل (۲-۴): یک شبکه توری یک بعدی از واحدهای محاسبه گر

یک واحد کوهنن فاصله اقلیدسی بین یک ورودی x و بردار وزن w آن را محاسبه می کند. این تعریف جدید از تحریک برای کاربردهای خاصی مناسب تر است و نیز نمایش بصری آن نیز ساده تر است. در شبکه کوهنن یک بعدی، همسایگی شعاع ۱ از یک واحد در موقعیت k ام، شامل واحدهایی در موقعیت های k-1 و k+1 است. واحدهای هر دو انتهای زنجیره همسایگی های نامتقارن دارند. همسایگی شعاع r از واحد k شامل همه واحدهایی است که تا تعداد r موقعیت از k به سمت راست یا چپ زنجیره پیش می رود.

یادگیری کوهنن از یک تابع همسایگی ϕ استفاده می کند که مقدار $\phi(i,k)$ آن بیانگر قدرت تزویج بین واحد i و واحد k در طول فرآیند آموزش می باشد. یک انتخاب ساده، تعریف $\phi(i,k)=1$ برای همه واحدهای i در یک همسایگی به شعاع r از واحد k ام، و $\phi(i,k)=0$

برای همه واحدهای دیگر است. بعدا مسائلی را بررسی خواهیم کرد که با انتخاب انواعی از توابع همسایگی بروز می کند. الگوریتم یادگیری برای شبکه های کوهنن به صورت زیر است:

الگوریتم یادگیری کوهنن

شروع: بردارهای وزنی n بعدی w1,w2,…,wm از m واحد محاسبه گربه صورت تصادفی انتخاب می شوند. یک شعاع اولیه r ، یک ثابت یادگیری η، و یک تابع همسایگی ϕ انتخاب می شوند.

گام ۱: یک بردار ورودی ξ را با استفاده از توزیع احتمال مطلوب روی فضای ورودی، انتخاب کنید.

گام۲: واحد k با تحریک حداکثر انتخاب می شود (یعنی برای آن واحد، فاصله بین wi و ξ حداقل است ($i=1,\ldots,m$)).

گام ۳: بردارهای وزنی با استفاده از تابع همسایگی و قانون به روز رسانی

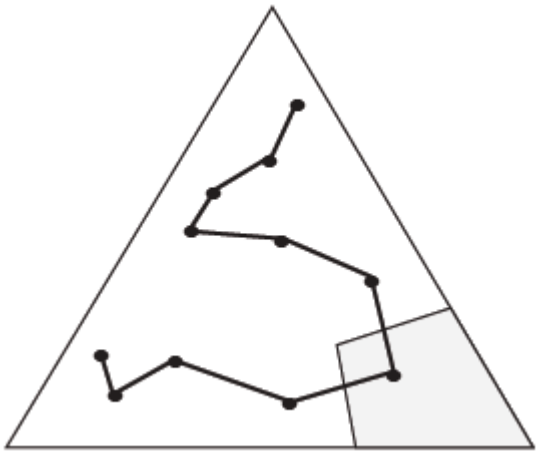
$$w_i \leftarrow w_i + \eta\phi(i,k)(\xi - w_i), \quad i = 1,\cdots,m \text{ برای}$$

به روز می شوند.

گام ۴: اگر به حداکثر تعداد تکرار رسیدید متوقف شوید؛ در غیر این صورت η و ϕ را همان طور که برنامه ریزی شده است اصلاح کنید و با گام ۱ ادامه دهید.

اصلاح بردارهای وزنی (گام ۳) آنها را در جهت ورودی ξ همگرا می کند. با چند بار تکرار این فرآیند ساده، انتظار داریم به یک توزیع یکنواخت بردارهای وزنی در فضای ورودی دست یابیم (اگر ورودی ها نیز به صورت یکنواخت انتخاب شده باشند). شعاع همگرایی بر اساس یک برنامه قبلی که ما آن را برنامه زمانی می نامیم کاهش می یابد. اثر آن این است که هر زمان یک واحد به روز می شود، واحدهای همسایه نیز به روز می شوند. اگر بردار وزنی یک واحد به یک ناحیه در فضای ورودی همگرا شود، همسایه ها نیز همگرا می شوند، اگرچه با یک درجه کمتر. در طول فرآیند یادگیری، هم اندازه همسایگی و هم مقدار ϕ به تدریج کاهش می یابند به گونه ای که اثرگذاری هر واحد روی همسایگانش کاهش می یابد. ثابت یادگیری بزرگی به روز رسانی وزنی را کنترل می کند و آن نیز به تدریج کاهش می یابد. اثر شبکه ای برنامه زمانی انتخاب شده، تولید اصلاحات بزرگ تر در شروع آموزش به جای پایان آن می باشد.

شکل(۲-۵) نتایج آزمایش با یک شبکه کوهنن یک بعدی را نشان می دهد. هر واحد توسط یک نقطه نشان داده شده است. دامنه ورودی یک مثلث است. در پایان فرآیند یادگیری، بردارهای وزنی به توزیعی می رسند که هر واحد را به یک "نماینده" از یک ناحیه کوچک فضای ورودی تبدیل می کند. برای مثال، واحد گوشه پایینی، واحدی است که با بزرگ ترین تحریک برای بردارهای ناحیه سایه دار، پاسخ می دهد. اگر ما استراتژی فعال سازی را به صورت " برنده همه چیز را بردارد " بپذیریم، آنگاه این تنها واحدی است که آتش می شود.



شکل(۲-۵): طرح یک ناحیه مثلثی

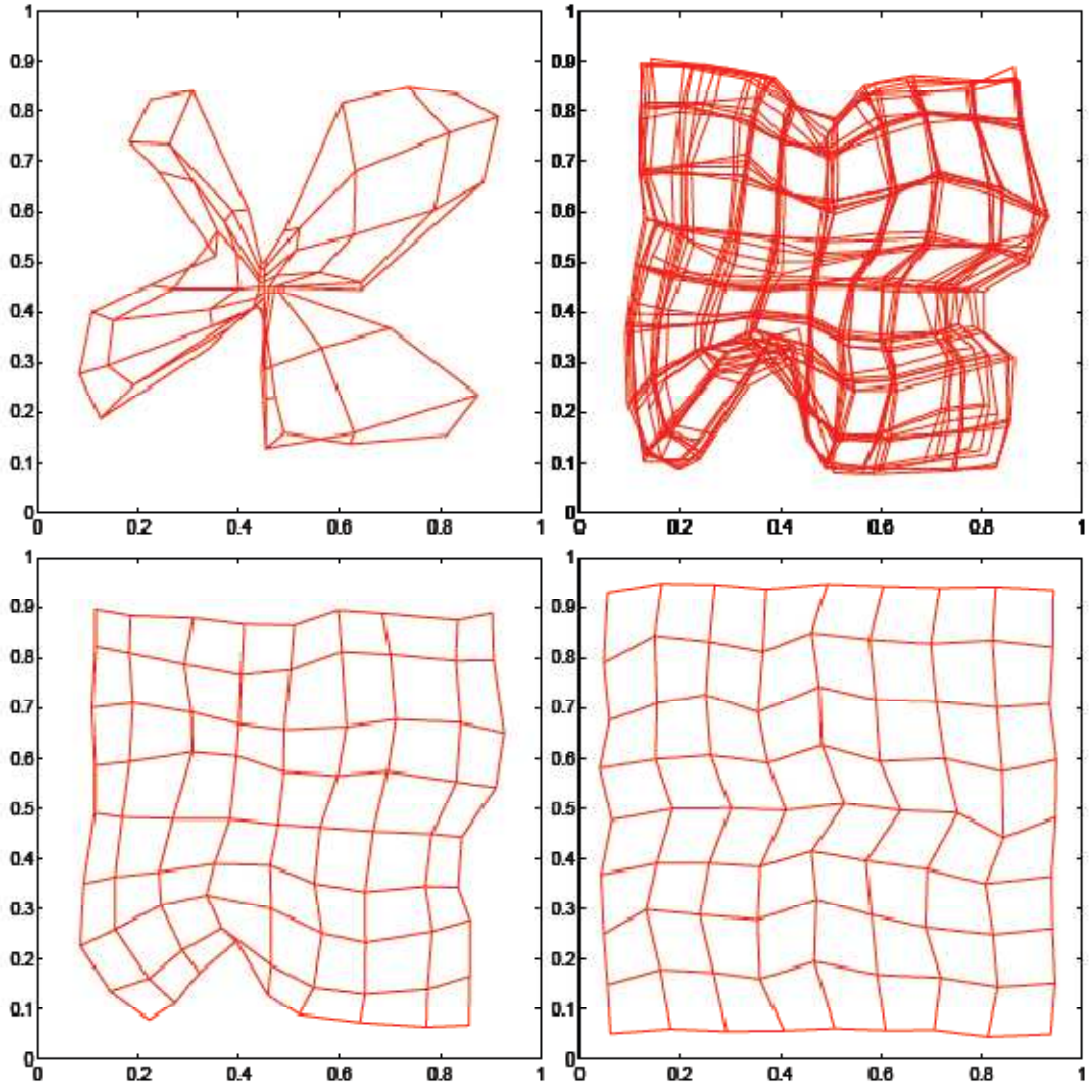
همین آزمایش را می توان برای دامنه های با اَشکال مختلف تکرار نمود. زنجیره واحدهای کوهنن یک شکل تحت عنوان منحنی پینو[11] به خود می گیرد.

شبکه های کوهنن را می توان در شبکه های چندبعدی تنظیم نمود. یک انتخاب جالب همان طور که در شکل (۲-۶) آمده است یک شبکه سطحی است. در این حالت، همسایگی شعاع r از واحد k شامل همه واحدهایی است که تا فاصله r از راست یا چپ، بالا یا پایین شبکه قرار گرفته اند. با این تصور، همسایگی یک واحد، یک جزء درجه دوم از شبکه می باشد. البته ما می توانیم همسایگی های پیچیده تری تعریف کنیم، اما این روش ساده همه آن چیزی است که در اغلب کاربردها مورد نیاز است.

شکل (۲-۶) مسطح کردن یک شبکه کوهنن دوبعدی در یک فضای ورودی درجه دوم را نشان می دهد. چهار نمودار، وضعیت شبکه را بعد از ۱۰۰، ۱۰۰۰، ۵۰۰۰ و ۱۰۰۰۰ تکرار نشان می دهد. در نمودار دوم، چند تکرار روی یکدیگر نشان داده شده است تا تلقی بهتری از فرآیند تکرار به

[11] *Peano curve*

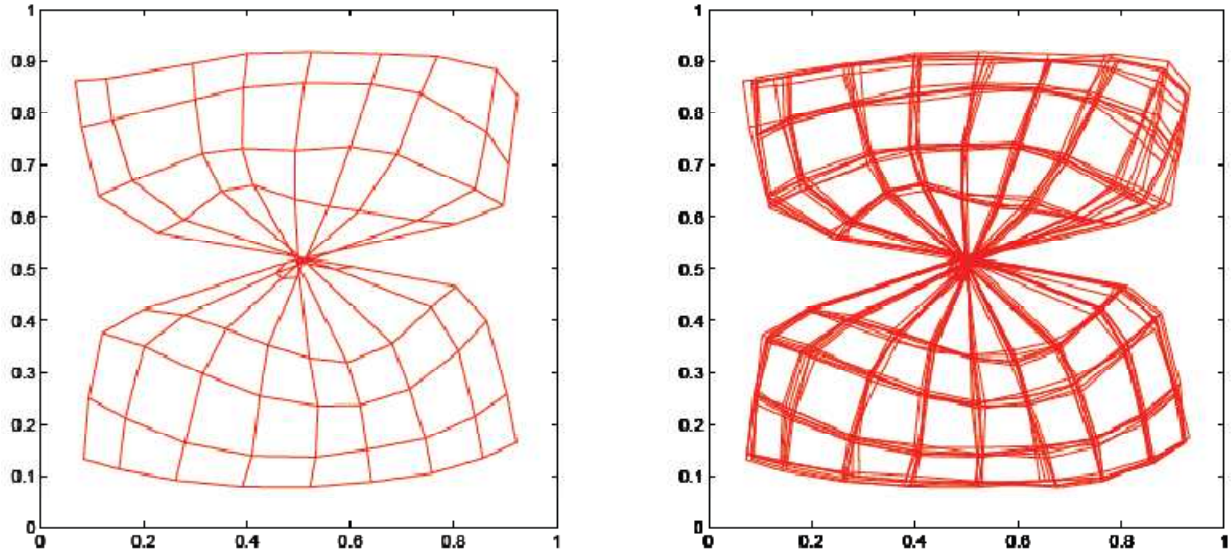
دست دهد. از آن جا که در این آزمایش، بُعد دامنه ورودی و شبکه یکسان است، فرآیند یادگیری به نتیجه بسیار رضایت بخشی می انجامد.



شکل (۲-۶): نگاشت یک مربع با یک شبکه توری دوبعدی. نمودار بالا سمت راست چند تکرار از فرآیند یادگیری را روی هم نشان می دهد. نمودار پایینی حالت نهایی بعد از ۱۰۰۰۰ تکرار را نشان می دهد.

دستیابی به وضعیت پایدار در حالت شبکه های چندبعدی چندان ساده نیست. فاکتورهای زیادی وجود دارند که در فرآیند همگرایی نقش ایفا می کنند؛ مانند اندازه همسایگی منتخب، شکل تابع همسایگی و برنامه ریزی انتخاب شده برای اصلاح هر دو. شکل (۲-۷) مثالی از یک شبکه را نشان می دهد که در رابطه با اصلاح، به وضعیتی بسیار مشکل رسیده است. یک گره کور در حین فرآیند آموزش ظاهر شده است و اگر انعطاف پذیری شبکه به سطح پایینی رسیده باشد، همان طور

که تکرارهای متداخل در نمودار سمت راست در شکل ۱۵-۸ نشان می دهد، گره کور با آموزش بعدی باز نمی شود.

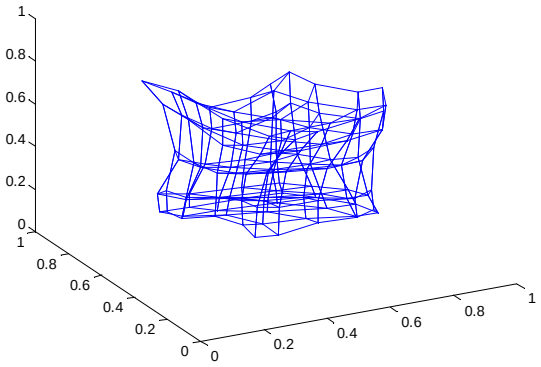


شکل (۲-۷): شبکه سطحی با یک گره کور

چندین اثبات همگرایی برای شبکه های کوهنن یک بعدی در دامنه های یک بعدی داده شده است. هیچ اثبات عمومی برای همگرایی شبکه های چندبعدی وجود ندارد.

۲-۴-۲- نگاشت فضاهای چندبعدی

معمولاً وقتی یک داده تجربی انتخاب می شود، ما ابعاد واقعی آن را نمی دانیم. حتی اگر بردارهای ورودی n بعدی باشند، می توانست چنین باشد که داده به یک نسخه با ابعاد پایین تر متمرکز شود. در حالت کلی، روشن نیست که چه ابعاد شبکه ای برای یک مجموعه داده باید مورد استفاده قرار گیرد. این مشکل کلی، باعث شد کوهنن در نظر بگیرد که هنگام استفاده از یک شبکه با ابعاد پایین برای طرح یک فضای با ابعاد بالاتر، چه اتفاقی می افتد. در این حالت شبکه باید به منظور پر کردن فضای موجود، تا شود. شکل (۲-۸) نتیجه یک آزمایش را نشان می دهد که در آن یک شبکه دو بعدی برای طرح ریزی یک جعبه سه بعدی به کار مورد استفاده قرار گرفت. همان طور که مشاهده می شود، شبکه در ابعاد x و y بسط می یابد و در جهت z ها تا می شود. واحدهای موجود در شبکه تا جایی که ممکن است تلاش می کنند تا فضای موجود را پر کنند، اما همسایگی درجه دوم آنها محدودیت هایی به این فرآیند تحمیل می کند.



شکل (۲-۸): تا شدن شبکه در فرایند تخمین

به خاطر بیاورید که ما پیشتر در رابطه با مشکل تطابق غشای مغزی- سطحی به یک جهان حسی چندبعدی بحث کردیم. شواهدی وجود دارد که نشان می دهد یک فرآیند خود سازمانده از نوع نشان داده شده در شکل (۲-۸) در مغز انسان نیز می تواند رخ دهد؛ اگرچه بسیاری از ساختار مغز در هنگام تولد شکل می گیرد. آزمایشات نشان می دهد که تا شدن های شبکه سطحی، به نوارهایی با رنگ های متناوب می انجامد؛ یعنی نوارهایی که ممکن است به طور متناوب به یک طرف یا طرف دیگر فضای ورودی نگاشته شوند (برای بُعد z). یک مثال معمول برای این نوع ساختار در مغز انسان، غشای بصری است. مغز معمولاً نه یکی بلکه دو تصویر بصری را که یکی به جای دیگری جانشین می شود، پردازش می کند. در این حالت، دامنه ورودی شامل دو ناحیه سطحی است (دو طرف جعبه شکل ۱۵-۹). غشای سطحی باید به همان روش تا شود تا به طور بهینه به ورودی از یک طرف یا طرف دیگر دامنه ورودی پاسخ دهد. نتیجه آن، ظهور نوارهای تسلط بصری است که در سال های اخیر توسط زیست شناس های اعصاب مورد مطالعه قرار گرفته است. شکل (۲-۹) نمایشی از ستون های تسلط بصری در بازسازی لی وی[۱۲] می باشد [۲۴]. مقایسه این نوارها با آنهایی که در آزمایش ساده ما با شبکه کوهنن به دست آمد، جالب است.

[12] Le Vays

شکل (۲-۹): نمودار تسلط بصری در غشای بصری. نوارهای سیاه نماینده یک چشم و نوارهای سفید نماینده چشم دیگر هستند.

این مثال ها، انواعی از نتایج جالبی که می توان از مدل کوهنن به دست آورد را نشان می دهد. در اصل، شبکه های کوهنن به شبکه های غیرنظارتی هم شباهت دارند. در واقع تلاش می کنیم یک فضای ورودی که واحدهای محاسبه گر را توزیع می کند، برای تعریف یک گسسته سازی برداری طرح ریزی کنیم. تفاوت اصلی در این است که شبکه های کوهنن یک توپولوژی از پیش تعریف شده دارند. آنها در یک شبکه سازماندهی شده اند و مسأله یادگیری، یافتن راهی برای توزیع این شبکه در فضای ورودی است. می توان روش دیگری را در پیش گرفت مانند استفاده از یک نوع میانگین k ام الگوریتم به منظور توزیع $n \times n$ واحدها در یک دامنه درجه دوم. سپس می توانیم این واحدها را به عنوان رئوس یک شبکه سطحی که به دلخواه تعریف کرده ایم، به کار بریم. مطمئناً بهتر است اگر ما فقط علاقمند به دستیابی به یک شبکه برای دامنه درجه دوم خودمان باشیم. اگر علاقمند باشیم بدانیم چگونه یک شبکه داده شده (غشاء)، به یک دامنه پیچیده تطابق می یابد، مجبوریم آموزش را با شبکه تا شده شروع کنیم و آموزش را با یک شبکه تا نشده خاتمه دهیم. در این حالت، آنچه ما بررسی می کنیم، تعامل بین "تربیت و طبیعت" است. چه ترکیب بندی های اولیه ای به همگرایی می انجامد؟ چگونه زمان همگرایی، از یک شبکه از پیش تنظیم شده در قیاس با یک شبکه کاملاً تصادفی تأثیر می پذیرد؟ همه این موضوعات از نظر زیست شناسی به هم مرتبط هستند و تنها می توانند با استفاده از مدل کوهنن کامل یا معادل آن پاسخ داده شوند.

۲-۵- کوهنن با ناظر

در ابتدای این بخش قصد داریم تا به انواع شبکه های کوهنن که در کاربردهای عملی مورد توجه محققین می باشد بپردازیم. در واقع می خواهیم اشاره ی کوتاهی به شبکه عصبی کوهنن باناظر داشته باشیم وسپس در فصل پایانی کار را با مقایسه عملکرد این شبکه عصبی با عملکرد شبکه عصبی پرسپترون چندلایه در برخورد با یک مسئله ی واحد، ادامه خواهیم داد. البته همانطور که گفتیم قصد داریم ابتدا شبکه عصبی رقابتی را به یک شکه عصبی رقابتی باناظر تبدیل کنیم سپس رفتار شبکه جدید را که با استفاده از الگوریتم یادگیری پیشنهادی در این پایان نامه آموزش داده شده است را با رفتار یک شبکه عصبی پرسپترون چندلایه مقایسه کنیم.

الف- پیش درآمد

پیش از آنکه به بیان و تشریح نتایج بپردازیم، شرح مختصری راجع به شبکه عصبی کوهنن با ناظر ارائه می شود تا تفاوت های این شبکه با شبکه کلاسیک کوهنن که در واقع یک شبکه بدون ناظر است به خوبی مشخص شود.

شبکه کوهنن همانطور که در پیشتر به طور مفصل آمد، یک نمونه بسیار معروف از شبکه های رقابتی خودسازمانده بدون ناظر است. اما توانایی این گونه شبکه ها محققین را بر آن داشته است تا با ایجاد تغییراتی در ساختار شبکه عصبی کوهنن، از مزایای عملی آن در مسائلی همچون تخمین توابع غیرخطی و به خصوص طبقه بندی الگوهای غیرخطی بهره جویند. البته عملکرد بسیار دقیق شبکه عصبی کوهنن تنها در صورتی در برخورد با مسائلی از این دست کارایی خواهد داشت که یا از چنین شبکه ای در یک ساختار با ناظر مانند شبکه پرسپترون چندلایه استفاده گردد و یا به طریقی با اضافه نمودن یک یا چند لایه به ساختار اصلی شبکه عصبی کوهنن بتوانیم به نحوی در هر مرحله پس از تخمین فضای ورودی، داده های تولیدی (تخمینی) را با یک مجموعه مرجع مقایسه نموده تا به طور همزمان یک از مزایای یک شبکه با ناظر در کنار شبکه کوهنن بهره برده باشیم. نیاز به دقت در تخمین برای چنین مسائلی، باعث شده تا تقریبا پس از سال ۱۹۹۵ میلادی تعداد قابل توجهی الگوریتم در همین راستا پیشنهاد شود که از میان آنها در اینجا به توضیح عملکرد دو الگوریتم پرکاربردتر می پردازیم.

ب- شبکه های خودسازمانده با ناظر

اولین بار در سال ۱۹۹۵ مبتکر شبکه عصبی خودسازمانده، توئو کوهنن، در کتاب مشهور خود با نام نگاشت های خودسازمانده به تشریح عملکرد گونه ی خاصی از شبکه های خودسازمانده پرداخت که یک تفاوت اساسی با دیگر شبکه های خودسازمانده به خصوص شبکه عصبی کوهنن داشت. این فرق اساسی در واقع همان با ناظر بودن این شبکه پیشنهادی جدید بود که باعث می شد رفتار شبکه جدید تا حد زیادی متمایز از رفتار و عملکرد شبکه های خودسازمانده شود. درواقع در الگوریتم پیشنهادی جدید، کوهنن خود اولین اصل اساسی شبکه های خودسازمانده را به بهای گسترده تر کردن دامنه عملکرد و کاربرد اینگونه شبکه ها، که همان رفتار بدون ناظر بود را نقض نمود وبا ارائه یک الگوریتم ترکیبی به واقع یک ماشین عصبی پیشنهاد نمود که در عین اینکه یک شبکه خودسازمانده بدون ناظر بود اما همزمان تمامی خواص شبکه های با ناظری همچون پرسپترون چندلایه را نیز دارا بود. گرچه بعدها کوهنن با ارائه الگوریتم نیرومند LVQ، راهی بهتر برای حفظ و تجمیع خواص متناقض شبکه های با ناظر و بدون ناظر را پیش روی محققین نهاد، اما به هر حال ماشین پیشنهادی وی فتح بابی بود برای دیگران تا از ایده ی ارائه شده توسط کوهنن برای ایجاد شبکه ها و ارائه الگوریتم های جدید از این دست بپردازند. در ادامه قصد داریم پس از تشریح ماشین پیشنهادی کوهنن، به بررسی دو مورد از الگوریتم های ارائه شده برای شبکه عصبی کوهنن با ناظر پرداخته و پس از آن عملکرد چنین شبکه هایی را با عملکرد یک شبکه پرسپترون آموزش یافته با الگوریتم فیلتر ذره ای مقایسه خواهیم نمود.

ج - ماشین پیشنهادی کوهنن برای تولید شبکه عصبی کوهنن با ناظر

همانطور که پیشتر آمد، کوهنن در سال ۱۹۹۵ در کتاب خود، نگاشت های خودسازمانده، الگوریتمی پیشنهاد نمود که در واقع ماشینی برای آموزش شبکه عصبی کوهنن به صورت با ناظر بود. ایده اصلی الگوریتم پیشنهادی کوهنن، تولید بردار وردی توسط الحاق بردار هدف با داده های برچسب دار، و سپس ادامه روند الگوریتم با همان قانون آموزش معمول شبکه های عصبی کوهنن بدون ناظر بود. اما این روش تنها هنگامیکه تعداد کلاسهای یادگیری محدود و کوچک باشد و آن هم فقط برای بعضی مسائل خاص، پاسخ مناسبی ارائه می داد[۲۴]و[۲۵]. البته همانطور که پیشتر هم اشاره شد، کوهنن پس از پیشنهاد این ماشین با ارائه الگوریتم نیرومند LVQ توانست تا حدود زیادی نواقص کار پیشین خود را البته با پیشنهاد نوع جدیدی از شبکه های عصبی رفع نماید.

غیر از ماشین باناظر کوهنن که در بالا شرحش آمد چندین الگوریتم دیگر هم برای تبدیل یک شبکه رقابتی خودسازمانده به یک شبکه باناظر وجود دارد که البته هر کدام از آنها دارای کاربردهای خاص می باشند.در مراجع (ساینس دایرکت و...) می توان برخی از این دست الگوریتم ها را به همراه کاربردهای آنها یافت. در این پایان نامه با توجه به اینکه قصد داریم تا عملکرد الگوریتم تلفیقی را برای آموزش یک شبکه عصبی کوهنن بسنجیم و از طرفی این عملکرد را با شبکه کلاسیک کوهنن مقایسه نماییم تنها به یک مورد خاص از این گونه الگوریتم های ایجاد شبکه ناظر، خواهیم پرداخت. در واقع با توجه به اینکه می خواهیم عملکرد کوهنن را در تخمین حالت بیازماییم، باید به گونه ای ساختار شبکه کوهنن را تغییر دهیم تا همانند یک شبکه پرسپترون چندلایه عمل کند. در ادامه به بررسی این ساختار جدید پرداخته ایم.

د- شبکه عصبی خودسازمانده کوهنن با ناظر

در این قسمت به بررسی شبکه عصبی خودسازمانده با ناظر می پردازیم. این گونه شبکه ها همانند یک شبکه پرسپترون چندلایه و یا یک شبکه تابع پایه شعاعی عمل می کند و در واقع یک شبکه با ناظر می باشد. در اینجا سعی داریم تا با طراحی یک شبکه عصبی دو لایه ای با حفظ خواص خودسازماندهی، یک شبکه با ناظر داشته باشیم. در این پایان نامه از مدل دو لایه ای مرجع [۲۶] استفاده نموده ایم. همانطور که می دانیم، لایه پنهان یک شبکه عصبی در واقع پیچیده ترین بخش آن برای طراحی می باشد. در اینجا برای طراحی این لایه پنهان در یک شبکه عصبی پرسپترون چند لایه از یک نگاشت خودسازمانده استفاده کرده ایم. در مرجع [۲۶] و [۲۷] و [۲۸]، مدل های مشابه مدل این شبکه عصبی پیشنهاد و تشریح شده است.

بسیاری از روشهای کلاسیک طبقه بندی و شناسایی الگو به نحوی، برای طبقه بندی گروه های همسان، از روش جداسازی الگوها استفاده می کنند تا یک رابطه میان داده ها کشف و بر اساس آن تصمیم گیری نمایند. در مقابل شبکه عصبی خودسازمانده در واقع یک الگوریتم تکرارشونده برای طبقه بندی است که می تواند برای یافتن انواع کلاسهای همسان مورد استفاده قرار گیرد. شبکه عصبی که در اینجا معرفی شده از دو لایه تشکیل گردیده است. یک لایه پنهان و یک لایه خروجی. البته دقت کنید که این دو لایه اگرچه از یکدیگر مستقل و جدا هستند، اما در عین حال به هم مربوط و متصل اند. لایه پنهان همانطور که گفته شد، یک شبکه (نگاشت) عصبی خودسازمانده است که به طور کامل به لایه خروجی توسط اتصالات (بردارهای وزنی) پیشخور، متصل شده است. لایه خروجی

هم از یک شبکه از نرون ها که توسط قانون دلتا به روزرسانی می شوند، تشکیل شده است تا بتوان با استفاده از آن خروجی مطلوب را تولید نمود. در ادامه به بررسی و تشریح عملکرد هر دو لایه می پردازیم.

د-۱- لایه پنهان

لایه پنهان شبکه پیشنهادی فوق متشکل از ویرایشی از شبکه عصبی خودسازمانده است به طریقی که در ادامه شرح داده شده است کار می کند. این لایه در اصل یک شبکه $I \times J$ از تعداد ثابتی از نرون ها با توپولوژی یکسان و ثابت است. هر نرون n_{ij} با یک بردار وزن $w_{ij} \in R^n$ متناسب است. تمامی عناصر بردار وزنی با استفاده از اعداد حقیقی تصادفی که در بازه $[0,1]$ توسط یک توزیع یکنواخت تولید شده مقداردهی اولیه می شود و پس از آن نیز تمامی بردارهای وزن نرمالیزه شده و به بردارهای یکه تبدیل می شوند.

در زمان t هر نرون n_{ij} یک بردار ورودی $x(t) \in R^n$ را دریافت می کند. این ورودی شبکه؛ s_{ij} با استفاد از رابطه زیر محاسبه و تولید می گردد.

$$s_{ij} = \frac{x(t).w_{ij}(t)}{\|x(t)\|\|w_{ij}(t)\|} \qquad (۹-۲)$$

همچنین خروجی y_{ij} نرون n_{ij} نیز از معادله زیر که به معادله حداکثرگیری معروف است، محاسبه شده و بدست می آید.

$$y_{ij} = \frac{\left(s_{ij}(t)\right)^m}{max_{uv}\left(s_{uv}(t)\right)^m} \qquad (۱۰-۲)$$

که در رابطه فوق v و u ستون ها و سطر های ماتریس شبکه عصبی و m مرتبه معادله حداکثرگیری است. نرون c را با بردار وزنی $w_c(t)$ که البته شبیه به بردار ورودی اعمالی $x(t)$ به شبکه عصبی است، در نظر بگیرید. از طریق رابطه زیر نرونی که بیشترین تحریک را داراست انتخاب می گردد.

$$c = \arg max_c\{|x(t).w_c(t)|\} \qquad (۱۱-۲)$$

و ضرایب ماتریس وزنی w_{ijk} نیز از رابطه زیر محاسبه می گردد.

$$w_{ijk}(t+1) = w_{ijk}(t) + \alpha(t)G_{ijc}(t)\big[x_k(t) - w_{ijk}(t)\big] \qquad (۱۲-۲)$$

که در آن $0 \le \alpha(t) \le 1$ ضریب یادگیری است و در شرط زیر صدق می کند:

$$\alpha(t) \to 0 \text{ اگر } t \to \infty \qquad (۱۳-۲)$$

همچنین تابع همسایگی نیز با معادله زیر بیان می شود:

$$G_{ijc}(t) = e^{-\frac{\|r_c - r_{ij}\|}{2\sigma^2(t)}} \qquad (۱۴-۲)$$

که در آن $r_c \in R^2$ و $r_{ij} \in R^2$، بردارهای مکان نرون c و n_{ij} هستند. تابع همسایگی G هم یک تابع گوسی کاهنده با زمان است. σ هم شعاع همسایگی در زمان t است که از ضرب مقدار σ در زمان $t-1$ در عدد ۰/۹۹ بدست می آید.

د-۲- لایه خروجی

لایه خروجی هم مانند لایه پنهان در این شبکه عصبی از یک شبکه $I \times J$ از تعداد ثابتی از نرون ها با توپولوژی یکسان و ثابت است. هر نرون n_{ij} با یک بردار وزن $w_{ij} \in R^n$ متناسب است. تمامی عناصر بردار وزنی با استفاده از اعداد حقیقی تصادفی که در بازه $[0,1]$ توسط یک توزیع یکنواخت تولید شده مقداردهی اولیه می شود و پس از آن نیز تمامی بردارهای وزن نرمالیزه شده و به بردارهای یکه تبدیل می شوند.

در زمان t هر نرون n_{ij} یک بردار ورودی $x(t) \in R^n$ را دریافت می کند. همچنین خروجی y_{ij} نرون n_{ij} نیز از معادله زیر محاسبه شده و بدست می آید.

$$y_{ij} = \frac{x(t).w_{ij}(t)}{\|x(t)\|\|w_{ij}(t)\|} \qquad (۱۵-۲)$$

در طول اجرای مرحله آموزش مقادیر ماتریس وزنی w_{ij} با استفاده از رابطه زیر به روزرسانی می شوند. که در آن β ضریب یادگیری و d_{ij} تحریک مطلوب نرون n_{ij} می باشد.

$$w_{ijl}(t+1) = w_{ijl}(t) + \beta x_l(t)\big[d_{ij}(t) - y_{ij}(t)\big] \qquad (۱۶-۲)$$

به این ترتیب با مقدمه ای که گذشت همکنون شبکه عصبی در دست داریم که در عین حالی که یک شبکه یادگیری باناظر است اما در ساختار درونی خود و در لایه پنهان شبکه از یک الگوریتم یادگیری رقابتی خودسازمانده بهره می برد. در ادامه به بررسی یک مثال برای مقایسه عملکرد این شبکه با یک شبکه پرسپترون معمولی که با الگوریتم فیلتر ذره ای آموزش داده شده است می‌پردازیم.

۲-۶- کاربردها[۲۹]

شبکه های کوهنن می توانند به دامنه هایی با عجیب ترین ساختارها تطابق یابند و همان طور که در ادامه نشان داده می شود، در بسیاری از حوزه ها به کار گرفته شده اند.

۲-۶-۱- تخمین توابع

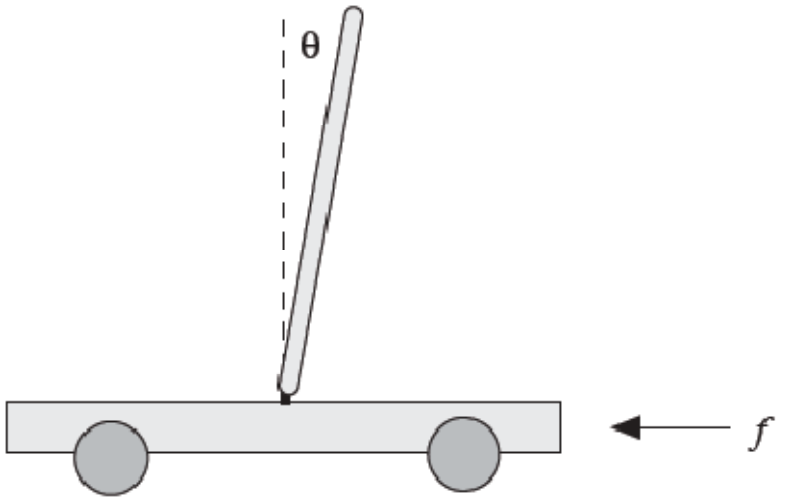
یک شبکه کوهنن را می توان برای تخمین تابع حقیقی پیوسته f در دامنه تعریف $[0,1] \times [0,1]$ به کار برد. مجموعه $P = \{(x, y, f(x,y)), x, y \epsilon [0,1]\}$ یک سطح در فضای سه بعدی محسوب می شود. می خواهیم یک شبکه سطحی به این سطح تطابق دهیم. در این حالت، مجموعه P دامنه ای است که ما تلاش داریم با شبکه کوهنن نگاشته کنیم.

بعد از شروع الگوریتم یادگیری، شبکه سطحی در جهت P حرکت می کند و خود را توزیع می کند تا دامنه را بپوشاند. تابع $z = 5sinx + y$ در نظر بگیرید. این تابع، تابعی است که کنترل یک سیستم تعادل قطب[۱۳] را تضمین می‌کند. چنین سیستمی شامل یک قطب (به جرم ۱) است که به یک اتومبیل در حال حرکت متصل شده است. قطب می تواند در نقطه اتصال بچرخد و اتومبیل تنها

[13] Pole Balancing System

می تواند به چپ یا راست حرکت کند. قطب باید به وسیله حرکت حرکت اتومبیل در یک جهت یا جهت دیگر در تعادل نگه داشته شود. نیروی لازم برای نگه داشتن قطب در تعادل، با رابطه $f(\theta) = \alpha sin\theta + \beta\, d\theta/dt$ داده می شود، که در آن θ زاویه بین قطب و محور عمودی، و α و β ثابت هستند شکل (۲-۱۰). برای مقادیر کوچک θ می توان از یک تخمین خطی استفاده نمود. از آن جا که یک شبکه کوهنن می تواند تابع f را بیاموزد، همچنین می تواند برای تأمین کنترل خودکار در سیستم تعادل قطب به کار رود.

وقتی یک ترکیب x و y داده شده باشد (در این جا θ و $d\theta/dt$)، واحد (i,j) یافت می شود که برای آن فاصله اقلیدسی بین وزن های مرتبط با آن $w_1^{(i,j)}$ و $w_2^{(i,j)}$ و $(\theta, d\theta/dt)$ حداقل است. مقدار تابع در این نقطه، $w_3^{(i,j)}$ آموخته شده است؛ یعنی تخمینی از مقدار تابع f. بنابراین شبکه نوعی از جدول مراجعه ای [۱۴] از مقادیر f می باشد. جدول می تواند بسته به کاربرد مورد نظر پراکنده یا فشرده ایجاد شود. استفاده از یک جدول در حالت کلی کارآمدتر از هر بار محاسبه تابع از اول است. اگر تابع از نظر تحلیلی در دست نباشد، اما توسط برخی مثال های ورودی خروجی آموزش ببیند، شبکه های کوهنن به شبکه های *backpropagation* شبیه می شود. شبکه کوهنن می تواند به تطابق ادامه دهد و در این روش، اگر بعضی از پارامترهای سیستم تغییر کنند (به دلیل این که بعضی قسمت ها شروع به فرسایش کند)، چنین تغییری به صورت خودکار مد نظر قرار می گیرد. شبکه کوهنن مانند یک جدول تطبیقی عمل می کند، که می تواند با استفاده از حداقل مقدار سخت افزار ساخته شود. این موضوع باعث شده که شبکه های کوهنن یک انتخاب جالب در کاربردهای روباتیکی باشند.

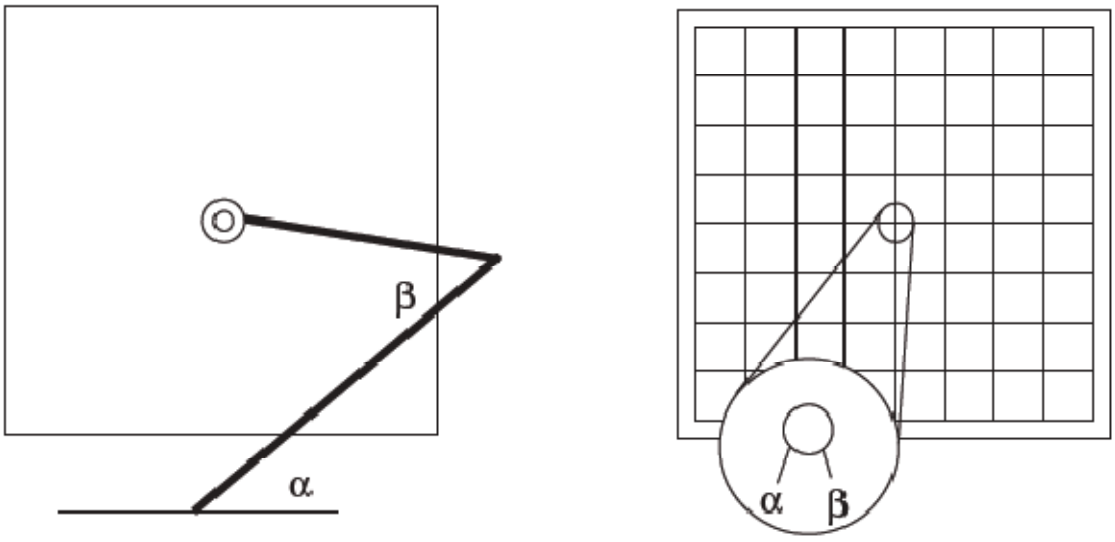


شکل (۲-۱۰): یک سیستم تعادل قطب

[14] look-up table

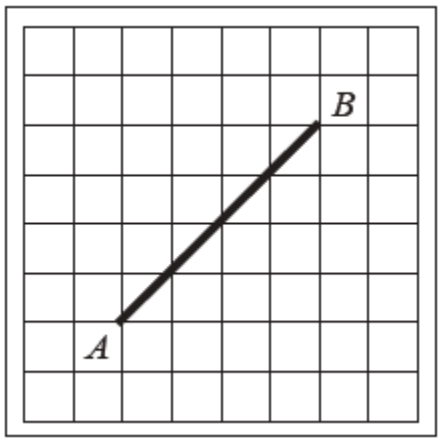
۲-۶-۲- سینماتیک معکوس

کاربرد دوم شبکه های کوهنن نگاشت فضای ترکیب بندی یک بازوی مکانیکی با استفاده از شبکه دو بعدی است. فرض کنید یک بازوی روبات با دو مفصل، همان طور که در شکل(۲-۱۱) نشان داده شده است، برای دستیابی به همه نقاط در جدول به کار رود. بازوی روبات دو درجه آزادی دارد، یعنی زوایای α و β. شبکه با استفاده از یک سیگنال فیدبک از بازو آموزش داده می شود. شبکه کوهنن دو بعدی به صورت همگن در دامنه درجه دوم توزیع می شود. اکنون انتهای بازو به صورت دستی در نقاط تصادفی روی سطح کاری قرار داده می شود. وزن های واحد کوهنن که به این نقطه نزدیک ترین باشد، به روز می شود (و نیز وزن های همسایگان). اما ورودی ها برای به روز رسانی، زوایای مفصل ها هستند. بنابراین شبکه فضای درجات آزادی را طرح می کند. پارامترهای ذخیره شده به عنوان وزن ها در هر گره، نمایانگر ترکیب زوایای مورد نیاز برای دستیابی به یک نقطه مشخص هستند (شکل(۲-۱۱)). در این حالت، دوباره شبکه کوهنن را می توان به صورت یک جدول پارامتر درنظر گرفت. شکل(۲-۱۱) نشان می دهد که اگر بخواهیم بازوی روبات در یک نقطه منتخب قرار گیرد، تنها لازم است پارامترها را از شبکه بخوانیم. توجه کنید که در این حالت، فرض کردیم که فرآیند یادگیری، از دستیابی به یک نقطه با ترکیب متفاوتی از پارامترها اجتناب می کند.



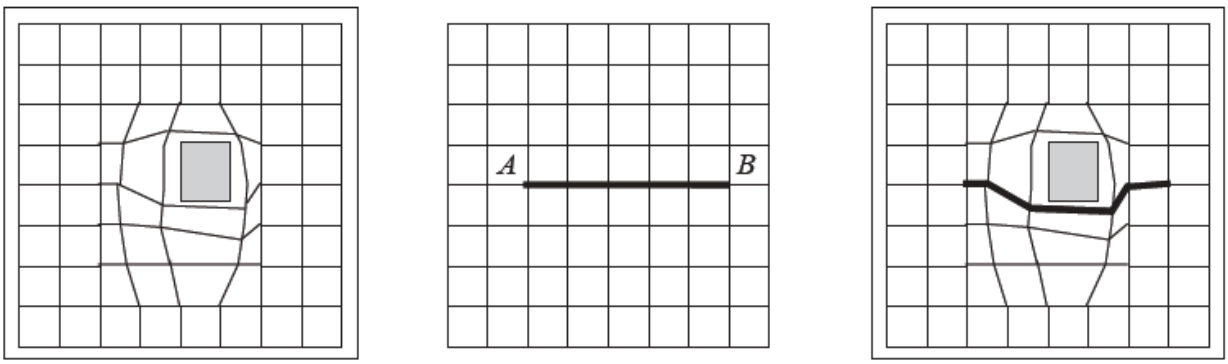
شکل(۲-۱۱): بازوی روبات (سمت چپ) و شبکه آموزش داده شده (سمت راست)

حالت جالب تر وقتی است که بخواهیم انتهای بازو را از نقطه A به نقطه B جابه جا کنیم. تنها لازم است یک مسیر از A به B روی شبکه بیابیم. همه واحدها در این مسیر، پارامترهای لازم برای حرکت را تأمین می کنند. موقعیت های بین آن ها هم می تواند از روی داده ها درون یابی شوند.



شکل(۲-۱۲): مسیر بهینه از A تا B

مشکل می تواند سخت تر شود اگر در سطح کاری با موانعی روبرو باشیم. شکل(۲-۱۳) چنین مانعی را نشان می دهد. اگر یک شبکه کوهنن آموزش داده شود، ترکیب بندی فضای بازو را به نحوی که از نواحی ممنوعه اجتناب کند، طرح ریزی می کند. اگر ما شبکه را در ناحیه کاری بر اساس پارامترهای ذخیره شده توزیع کنیم، نتیجه آن در سمت چپ شکل(۲-۱۳) نشان داده شده است. خود شبکه همچنان یک شبکه دو بعدی است. اگر بخواهیم بازو را از A به B حرکت دهیم، می توانیم این حرکت را روی شبکه به صورتی که قبلاً بحث شد، انجام دهیم. انتخاب مسیر واقعی، همان است که در سمت راست شکل(۲-۱۳) نشان داده شده است. بازو به نحوی حرکت می کند که از موانع اجتناب کند.



شکل(۲-۱۳): موانع در ناحیه کاری

این روش را تنها در صورتی می توان به کار برد که قبلاً مطمئن شده باشیم که لبه های شبکه، درون یابی های معتبری را فراهم می کنند. می تواند این طور باشد که دو گره مجاور، اطلاعات

معتبری را شامل می شوند، اما مسیر بین آن ها به یک برخورد[۱۵] می انجامد.پیش از استفاده از شبکه به روشی که در بالا توضیح داده شد، یک گام اعتبارسنجی قبل از آن ها باید تضمین کند که لبه های شبکه از برخورد آزاد هستند.

شبکه کوهنن تنها موقعی می تواند به کار رود که موانع موقعیت خود را تغییر ندهند. اگر تغییرات خیلی کند باشد هم این امکان وجود دارد که یک الگوریتم یادگیری سریع بتواند شبکه را با وضعیت های در حال تغییر تطابق دهد. شبکه های کوهنن در این نوع از کاربرد خود، به عنوان یک سیستم مختصات تعمیم یافته [۱۶]عمل می کنند. یافتن کوتاه ترین مسیر در شبکه با یافتن کوتاه ترین مسیر در دامنه مسأله متناظر است. حتی وقتی حرکت صورت گرفته توسط بازو، نسبتاً پیچیده و غیرخطی باشد.

۲-۷- لایه های رقابتی در شبکه های عصبی بیولوژیکی

در شبکه های عصبی بیولوژیکی، نرون ها به طور معمول در لایه های دو بعدی آرایش می گیرند. در این آرایش نرون ها به طور فشرده به هم متصل اند، و این اتصال از طریق پسخوردهای جانبی صورت می گیرد. غالباً وزن های ارتباطی بین نرون ها تابعی از فاصله بین نرون ها می باشند.

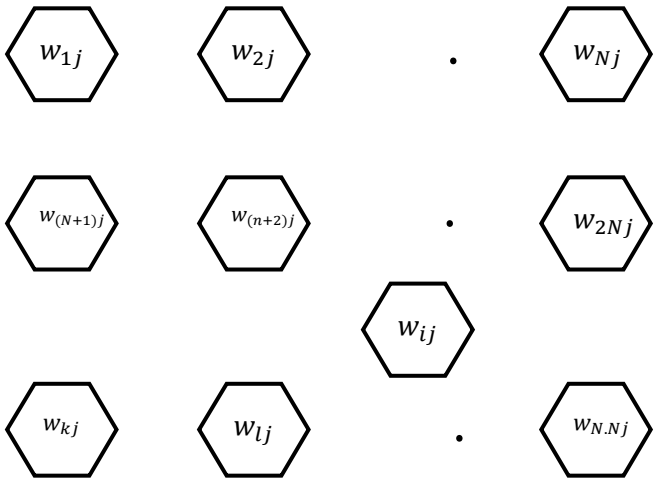
مثلاً برای لایه دوم شبکه همینگ می توانیم بنویسیم:

$$w_{\mathrm{ij}} = \begin{cases} 1 & d_{ij} = 0 \\ -\varepsilon & d_{ij} > 0' \end{cases} \qquad (۱۷-۲)$$

جائیکه d_{ij} فاصله بین نرون های i و j می باشد.

دیاگرام زیر را در نظر بگیرید که نمایش تصویری از ارتباط های وزنی بین (به طور مثال N^2) نرون را ارائه می دهد:

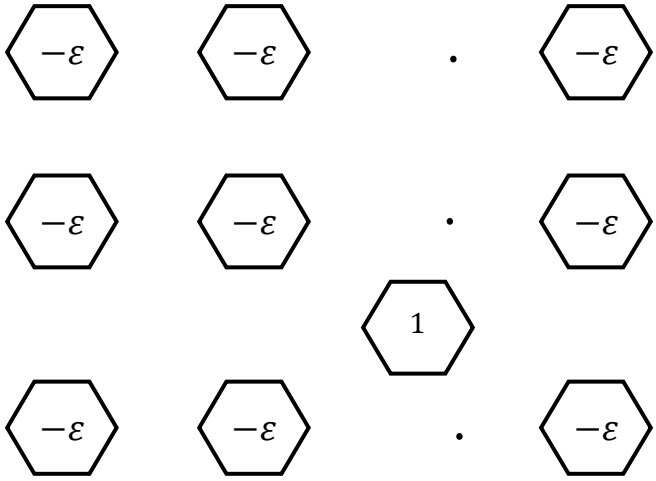
[15] collision
[16] generalized coordinate system



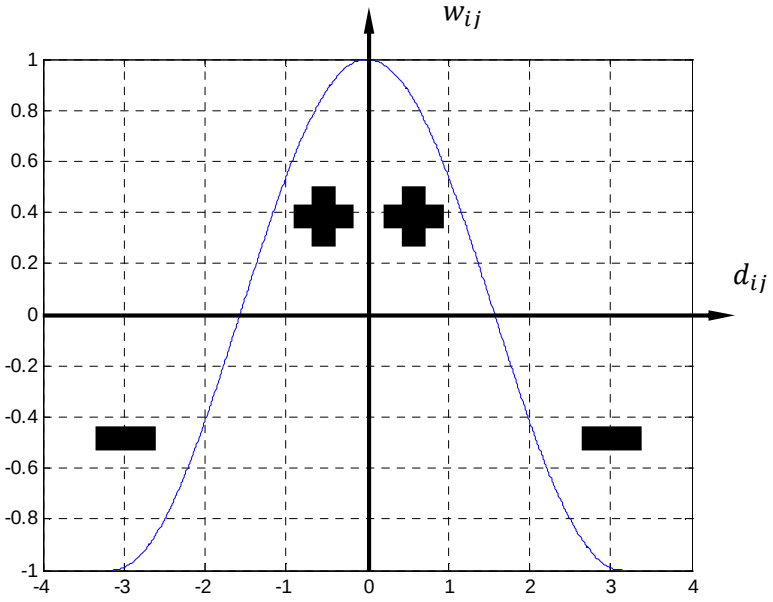
شکل(۲-۱۴): نمایش تصویری از ارتباطات وزنی

همانگونه که از شکل فوق پیداست، قرارداد بر این است که هر نرون با مقدار وزن W_{ij} مشخص می شود، که میزان تاثیر نرون iام به نرون jام را اندازه می گیرد. و مطابق فرمول (۲-۲۳) شکل (۲-۵) را خواهیم داشت. نوع ارتباط بین نرون ها موسوم است به خود مرکز-جانب گریز[۱۷]؛ بدین مفهوم که هر نرون خودش را به عنوام مرکز تقویت می کند و به طور همزمان تمامی نرون های دیگر (نرون های جانبی) را تضعیف می نماید. اینگونه به نظر می رسد که چیزی شبیه به این عمل در سیستم عصبی بیولوژیکی نیز اتفاق می افتد، جائیکه هر نرون نه تنها خودش، بلکه بعضی نرون ها را در همسایگی خود تقویت می کند. به عبارت دیگر، به طور معمول انتقال از تقویت به تضعیف به صورت ملایم، متناسب با افزایش فاصله بین نرون ها اتفاق می افتد. این انتقال را می توان به صورت شکل (۲-۶) تصویر نمود.

[17] On center/off surround



شکل (۲-۱۵): ارتباطات وزنی با توجه به رابطه (۲-۲۳)



شکل (۲-۱۶): ساختار خودمرکز-جانب‌گریز

انتقال از حالتی که w_{ij} مثبت است (حالت تقویتی) به حالتی که w_{ij} منفی است (حالت تضعیف) به صورت ملایم اتفاق می افتد. شکل فوق بیانگر رابطه ایست که میان فاصله بین نرون ها (d_{ij}) و مقدار وزن (w_{ij}) موجود می باشد. نرون هایی که نزدیک هم هستند ارتباط هایی از نوع تقویتی دارند، و میزان تقویت با افزایش فاصله بین نرون ها کاهش می یابد. پس از گذشتن از یک میزان فاصله مشخص، نرون ها از ارتباط های ممانعتی برخوردار می شوند و میزان تضعیف (ممانعت)

با زیاد شدن فاصله افزایش می یابد. هر نرون i براساس نسبت تقویت و تضعیف با علامت "+" و "-" (پررنگ و یا کم رنگ تر) مشخص می شوند که نشان دهنده ی شدت اتصال وزنی w_{ij} آن به نرون j می باشد.

لایه رقابتی در سیستم های عصبی واقعی، از یک حالت تدریجی ملایم تری بین حالت تحریک و حالت تضعیف برخوردارند. به عبارتی جهت انتقال از ناحیه تقویت به ناحیه تضعیف، انتقال به صورت ملایم انجام می گیرد، نه آنگونه که در لایه رقابتی شبکه همینگ انفاق می افتد. به جای اینکه یک نرون در لایه رقابتی برنده شود و مابقی بازنده، در شبکه های رقابتی بیولوژیکی، توده ای از نرون ها که در اطراف نرون برنده قرار می گیرند فعال می شوند.

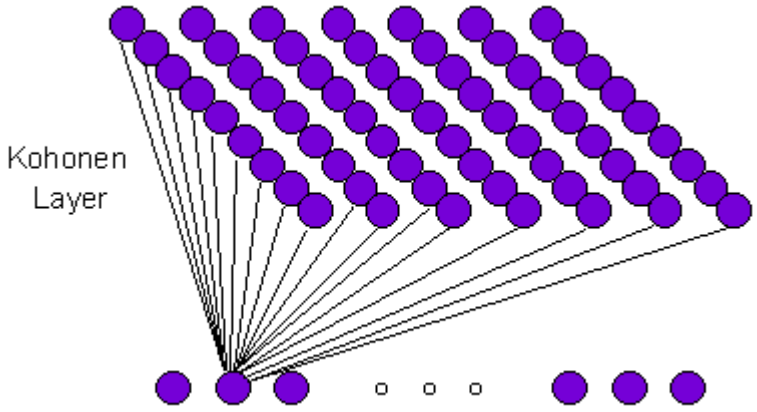
۲-۸- مشکل سلول مرده

گاهی در شبکه وضعیتی پیش می آید که یک یا چند سلول عصبی دارای حالت هایی می شوند که در طی بارها اعمال ورودی، امکان برنده شدن را از دیگر سلول ها گرفته و خود برنده می شوند. در این صورت بعضی از سلول ها از رقابت فاصله گرفته و تاثیر چندانی در شبکه نخواهند داشت. این مشکل باعث می شود تا در بعضی از قسمت های شکل، حالت برخی از سلول ها جا مانده و تغییر چندانی نداشته باشد.

همانطور که در بخش ۲-۴-۱ برای شبکه های عصبی رقابتی یا خودسازمانده اشاره شد، مشکل سلول مرده (واحد مرده) ممکن است به چند دلیل رخ دهد. اول این که ممکن است سلول مورد نظر از ابتدا در ناحیه ورودی قرار داشته باشد اما به دلیل فشردگی بیش از حد سلول ها در مرکز و تاثیر بیشتر ورودی روی سلول های مرزی، حالت این گونه سلول ها –سلول های مرکزی- زیاد تغییر نکند و عملاً پس از چندین تکرار این سلول ها دیگر شانسی برای پیروزی در رقابت نداشته باشند. حلت دوم می تواند به نوع اعمال ورودی تصادفی مربوط باشد به این معنی که سلول های مرده این بار در ناحیه اعمال ورودی واقع نشده باشند و در رقابت پیروز نباشند. و حتی با گذشت زمان و کاهش بهره هم دیگر شانسی برای تصحیح حالت این سلول ها وجود نداشته باشد.

در فصل آخر این پایان نامه سعی کرده ایم از طریق تلفیق روش فیلترهای ذره‌ای با گام تعین ضرایب در شبکه کوهنن نه تنها مشکل سلول مرده را برطرف نماییم بلکه الگوریتم بیهینه ای برای

پیاده سازی شبکه عصبی کوهنن ارائه نماییم. در فصول آتی به بررسی روند الگوریتم کارآمد فیلتر ذره ای و نحوه اعمال آن در شبکه های عصبی به خصوص شبکه عصبی کوهنن خواهیم پرداخت.



شکل (۲-۱۷): مدل فیزیکی شبکه عصبی کوهنن

فصل سوم

یادگیری و فرم انتگرال بیز

۳-۱- یادگیری و شبکه عصبی [۱]، [۱۵]

همانطور که در فصل اول گفته شد کار را با ارائه مدل فضای حالت شبکه عصبی در حوزه زمان آغاز می کنیم:

$$\theta_{t+1} = h(\theta_t, u_t), \qquad (۱-۳)$$

$$y_t = \widehat{f}_t(x_t, \theta_t, v_t). \qquad (۲-۳)$$

که در اینجا $\widehat{f}_t(x_t, \theta_t, v_t)$ نشان‌دهنده یک شبکه عصبی با ماتریس وزنی θ_t است. در حالت کلی در الگوریتم مونت کارلو متوالی، توابع توزیع احتمالاتی جملات نویزی توسط کاربر انتخاب می شوند. نویز فرآیند هم به صورت یک فرآیند گوسی جمع شونده با توزیع یکنواخت $u_t \sim \mathcal{N}(0, Q)$ مدل می‌شود. بنابراین برای معادلات انتقال ماتریس وزنی خواهیم داشت:

$$\theta_{t+1} = \theta_t + u_t. \qquad (۳-۳)$$

همچنین به منظور رگرسیون خروجی می توان نویز اندازه گیری را به صورت یک فرآیند گوسی جمع شونده با توزیع یکنواخت $v_t \sim \mathcal{N}(0, Q)$ مدل کرد. پس برای معادلات اندازه گیری داریم:

$$y_t = \widehat{f}_t(x_t, \theta_t) + v_t. \qquad (۴-۳)$$

توجه داشته باشید که ساختار الگوریتم مونت کارلو متوالی[۱۸] به راحتی این امکان را فراهم می کند که نویز سیستم را در ورودی هم مدل کنیم. برای این کار اگر سیستم طبقه بندی باینری (دودوئی) باشد، می توان تابع توزیع اندازه گیری را با فرآیند برنولی با تابع توزیع احتمال زیر مدل نمود:

$$p(y_t|x_t, \theta_t) = \left(\widehat{f}_t(x_t, \theta_t)\right)^{y_t} \left(1 - \widehat{f}_t(x_t, \theta_t)\right)^{1-y_t}. \qquad (۵-۳)$$

که در آن $\widehat{f}_t(x_t, \theta_t)$ تابع احتمال عضویت کلاس است که توسط یک سلول منطقی تولید می شود. الگوریتم مونت کارلو متوالی از نظر انتخاب توزیع نویز بسیار انعطاف پذیر است.

[18] Sequential Monte Carlo Algorithm

دقت کنید که جملات نویز در این معادلات، ناهمبسته[19]با ماتریس وزنی و شرایط اولیه سیستم هستند. تغییرات ماتریس وزنی براساس یک فرآیند مارکوف[20] با تابع توزیع احتمال اولیه $p(\theta_0)$ و احتمال شرطی انتقال $p(\theta_t|\theta_{t-1})$ صورت می گیرد. همچنین مشاهدات از حالات داده شده سیستم مستقل مشروط هستند. اینها شرایط استانداردی در کلاس بزرگی از مسائل ردیابی و سریهای زمانی هستند.

هدف ما تخمین تابع توزیع احتمال پیشین[21] $p(\theta_{0:t}|d_{1:t})$ و یکی از حاشیه های آن یعنی تابع چگالی احتمال فیلترسازی[22] $p(\theta_t|d_{1:t})$ است که در آن $d_{1:t} = \{x_{1:t}, y_{1:t}\}$ می باشد. در واقع با محاسبه بازگشتی تابع توزیع چگالی احتمال فیلترسازی دیگر نیازی نیست که تمامی داده های ردیابی شده پیشین را نگهداری کنیم.

۳-۲ مدل سازی شبکه عصبی در فضای حالت [۲۱]

با توجه به نوشتار فوق می توانیم معادلات فضای حالت شبکه عصبی را در حالت کلی به فرم زیر بازنویسی کنیم. این دسته از معادلات، معادله انتقال نحوه تغییرات ماتریس وزنی شبکه را تعیین می کند در حالیکه معادلات اندازه گیری روابط غیرخطی میان ورودی و خروجی یک فرآیند فیزیکی خاص را معلوم می نماید:

$$w_{k+1} = w_k + d_k, \qquad (۶-۳)$$

$$y_k = g(w_k, x_k) + v_k. \qquad (۷-۳)$$

در اینجا $y_k \epsilon \Re^o$ اندازه گیری های خروجی، $x_k \epsilon \Re^d$ اندازه گیری های ورودی و $w_k \epsilon \Re^m$ ماتریس وزنی شبکه را نشان می دهد. نگاشت غیرخطی اندازه گیری $g(.)$ توسط یک شبکه عصبی به عنوان مثال کوهنن یا پرسپترون چندلایه تخمین زده می شود. این مطلب که این مدل توانایی تخمین هر تابع غیرخطی پیوسته را با یک دقت دلخواه بدون محدودیت در اندازه را داراست کاملاً پذیرفته شده است. فرض می کنیم داده های اندازه گیری توسط نویز v_k تخریب شده اند. همانطور

---

[19] Uncorrelated
20 Markov Process
21 Posterior Distribution
22 Filtering Density

که پیشتر آمد در ساختار الگوریتم مونت کارلو متوالی، تابع توزیع احتمالاتی نویز توسط کاربر تعیین می شود. در مثال هائی که در این پایان نامه بررسی شده اند یک تابع توزیع احتمال گوسی با میانگین صفر و کوواریانس R در نظر گرفته شده است. دوباره تاکید می کنیم که جملات نویز در این معادلات ناهمبسته با ماتریس وزنی و شرایط اولیه سیستم هستند.

نحوه تغییر مقدار عناصر ماتریس وزنی با توجه به این موضوع که آنها وابسته به مقادیر قبلی w_k و یک مولفه اتفاقی[۲۳] d_k هستند مدلسازی می کنیم. نویز فرآیند d_k می تواند نشانگر عدم قطعیت در مورد چگونگی نمو پارامترها، مدلسازی خطاها و یا ورودی های ناشناس همانند مانور نظامی در کاربردهای ردیابی باشد. همچنین فرض بر اینست که نویز فرآیند یک فرآیند گوسی با میانگین صفر وکوواریانس Q است و با ماتریس وزنی هم ناهمبسته است.

در بسیاری از کاربردها می توان مدل گام اتفاقی فوق را با جایگذاری عبارت گرادیانی زیر قدرتمندتر ساخت:

$$w_{k+1} = w_k + \alpha\big(y_k - \hat{g}(w_k, x_k)\big)\frac{\partial \hat{g}(w_k, x_k)}{\partial w_k} + d_k. \qquad (۸-۳)$$

که در آن پارامتر α نرخ یادگیری و $\hat{g}(w_k, x_k)$ تخمین مدل در زمان t است. در اینجا نیز جملات نویز نا همبسته با ماتریس وزنی و شرایط اولیه سیستم هستند. با توجه به مطالب و نکات مطرح شده در بخش ۲-۱ و شرایط اساسی هاروی برای سیستم های ردیابی و سریهای زمانی، تغییر حالتهای سیستم یا همان ماتریس وزنی سیستم را فرآیند مارکوف با تابع توزیع احتمال اولیه $p(w_0)$ و تابع توزیع احتمال شرطی انتقال[۲۴] $p(w_t|w_{t-1})$ فرض می کنیم. همانگونه که می دانید هدف، تخمین عناصر ماتریس وزنی و ماتریس های نویزR وQ با در دسترس بودن مقادیر اندازه گیری $Y_k = \{y_1, y_2, \cdots, y_k\}$ است. در بخش بعدی به ارائه یک راه حل تئوریک برای مساله و تبدیل آن به فرم انتگرال بیز می پردازیم و با بیان مساله به فرم انتگرال بیز از مزایای قانون بیز و حل انتگرال بیز بهره می جوئیم.

---

6 Stochastic Component
24 Transition Probability

۳-۳ یادگیری متوالی[۲۵] [۱۴]، [۱۶] و [۲۱]

از فصل پیش به یاد داریم که در یادگیری متوالی برای سیستم های دینامیک غیرخطی، پارامترهای مدل متغیر با زمان هستند. مدلی که بتواند این رفتار و خاصیت چنین سیستم هایی را نعکاس دهد، بسیار کاربردی خواهد بود. همچنین در فصل پیشین معادلات فضای حالت شبکه عصبی را برای یک سیستم دینامیکی اتفاقی گسسته بیان کردیم. این معادلات را به دلیل اهمیت بار دیگر یادآوری می کنیم:

$$\theta_{t+1} = \theta_t + u_t. \qquad (۹-۳)$$

$$y_t = \hat{f}_t(x_t, \theta_t) + v_t. \qquad (۱۰-۳)$$

جملات نویزی اغلب نویز فرآیند u_t و نویز اندازه گیری v_t نامیده می شوند. معادله اول یک فرآیند مارکوف مرتبه اول با تابع توزیع احتمال $p(\theta_{t+1}|\theta_t)$ و معادله دوم احتمال مشاهده[۲۶] $p(y_t|\theta_t)$ را تعریف می نماید؛ و بیان مسأله با داشتن مقدار اولیه تابع توزیع احتمال یعنی $p(\theta_0)$ کامل می شود.

[25] Sequential Learning
[26] Observation Likelihood

۳-۴ بیان مسأله به فرم انتگرال بیز

تابع توزیع احتمال پیشین $p(\theta_{0:t}|x_{1:t},y_{1:t})$ ، که در آن $y_{1:t}=\{y_1,y_2,\cdots,y_t\}$ و $\theta_{0:t}=\{\theta_1,\theta_2,\cdots,\theta_t\}$ است، جواب کامل برای مسأله تخمین متوالی را در بر دارد. در بسیاری از کاربردها، همانند ردیابی، محاسبه یکی از حواشی این تابع به نام تابع توزیع چگالی احتمال فیلترسازی $p(\theta_t|x_{1:t},y_{1:t})$ حائز اهمیت می باشد. همان گونه که در فصل ۲ اشاره شد، محاسبه تابع توزیع چگالی احتمال فیلترسازی بسیار به صرفه تر و مناسب تر از محاسبه و نگهداری تابع توزیع احتمال پیشین می باشد. در واقع اگر ما تابع توزیع چگالی احتمال فیلترسازی ماتریس وزنی را برای یک شبکه عصبی بدانیم، به راحتی می توانیم تخمین های گوناگونی برای ماتریس وزنی شامل میانه ها، میانگین ها، مراکز ثقل و بازه های اطمینان بیابیم. در این فصل قصد داریم نشان دهیم که چگونه می توان تابع توزیع چگالی احتمال فیلترسازی را با استفاده از تکنیک های نمونه برداری با اهمیت متوالی تخمین زد. تابع توزیع چگالی احتمال فیلترسازی که در ادامه آن را تابع فیلترسازی می نامیم، در دو مرحله تخمین زده می شود:

۱) پیشگویی

۲) به هنگام سازی

که در شکل (۳-۱) نمودار آن رسم شده است. در گام پیش بینی، تابع فیلترسازی $p(\theta_{t-1}|x_{1:t-1},y_{1:t-1})$ توسط تابع توزیع احتمال انتقال$p(\theta_t|\theta_{t-1})$ به داده های آینده سیستم مانند فرمول (۳-۱۰) تسری می یابد:

$$p(\theta_t|x_{1:t-1}, y_{1:t-1}) \qquad (۱۱-۳)$$
$$= \int p(\theta_t|\theta_{t-1})p(\theta_{t-1}|x_{1:t-1}, y_{1:t-1})d\theta_{t-1}.$$

تابع توزیع احتمال انتقال به فرم جملات مدل احتمالاتی که بر معادلات تغییر حالت های سیستم و نویز فرآیند آماری حاکم است، تعریف شده است. یعنی به صورت زیر:

$$p(\theta_t|\theta_{t-1}) = \int p(\theta_t|u_{t-1}, \theta_{t-1})p(u_{t-1}|\theta_{t-1})du_{t-1} \qquad (۱۲-۳)$$
$$= \int \delta(\theta_t - u_{t-1} - \theta_{t-1})p(u_{t-1})du_{t-1}.$$

که در آن تابع دلتای دیراک $\delta(\cdot)$ ، نشان می دهد می توان θ_t را توسط یک رابطه کاملاً معین محاسبه نمود اگر مقادیر θ_{t-1} و u_{t-1} معلوم باشند. توجه داشته باشید از آن جا که فرض کرده ایم نویز اندازه گیری و نویز فرآیند از حالت های سیستم در همه زمان ها مستقل اند، رابطه $p(u_{t-1}|\theta_{t-1}) = p(u_{t-1})$ درست است.

گام به هنگام سازی شامل اِعمال قانون بیز در زمانی است که داده جدید وارد می شود:

$$p(\theta_t|x_{1:t}, y_{1:t}) = \frac{p(y_t|\theta_t, x_t)p(\theta_t|x_{1:t-1}, y_{1:t-1})}{p(y_t|x_t, y_{1:t-1})}. \qquad (۱۳-۳)$$

و تابع توزیع شانس[۲۷] را نیز می توان به فرم معادلات مدل اندازه گیری به صورت زیر نوشت:

$$p(y_t|\theta_t, x_t) = \int \delta\big(y_t - \hat{f}(x_t, \theta_t) - v_t\big)p(v_t)dv_t. \qquad (۱۴-۳)$$

نرمالیزه کردن مخرج کسر معادله (۳-۴)، یعنی همان تابع توزیع شاهد[۲۸]، نقش مهمی در الگوریتم یادگیری و به کار انداختن تخمین گوسی بازی می کند (جازوینسکی ۱۹۶۹ ، مک کی ۱۹۹۲ ، و سی بی سی ۱۹۸۹) [۲۱]. این کار به صورت زیر انجام می شود:

[27] Likelihood Function
[28] Evidence Density Function

$$p(y_t|x_t, y_{1:t-1}) = \int p(y_t|\theta_t, x_t)p(\theta_t|x_{1:t-1}, y_{1:t-1}). \quad (۱۵-۳)$$

در الگوریتم یادگیری متوالی ممکن است بخواهیم مسأله تخمین پارامترها را برای محاسبه تخمین $\hat{\theta}_t$ از حالت های سیستم (θ) با استفاده از داده های اندازه گیری $\{x_{1:t}, y_{1:t}\}$ ، دوباره فرمول بندی کنیم. به منظور بهینه سازی اغلب نیازمندیم $\hat{\theta}_t$ یک تخمین ثابت، حداقل واریانس[۲۹] و بدون بایاس[۳۰] باشد (گلب ۱۹۷۴). توجه به نکات زیر در مورد این تعاریف ضروری به نظر می رسد:

۱) یک تخمین بدون بایاس تخمینی است که با مقدار داده ای که می خواهیم تخمین زده شود برابر باشد؛

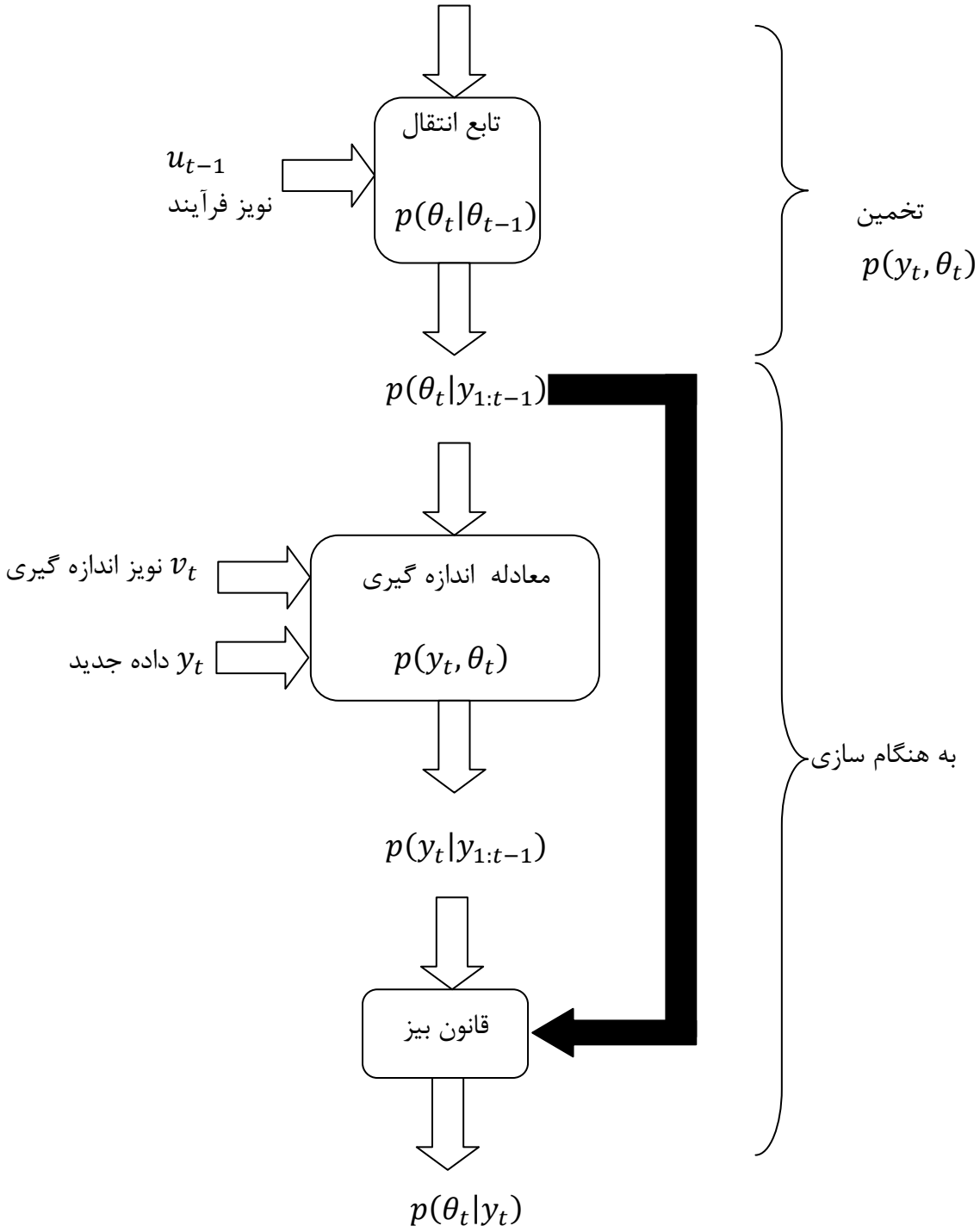
۲) یک تخمین با حداقل واریانس تخمینی است که واریانس آن از هر تخمین بدون بایاس دیگر کمتر باشد؛

۳) یک تخمین ثابت تخمینی است که با افزایش تعداد اندازه گیری ها مقدارش به مقدار داده اصلی همگرا شود؛

در فرمول بندی های فوق یک سوال مطرح است و آن این است که چگونه باید مقدار اولیه تابع توزیع احتمال $p(\theta_0)$ را انتخاب کرد. این مسأله یک بحث مهم در استنتاج بیزی می باشد. اگرچه مسأله بسیار عمومی تر است اما تقریباً با ماهیت بدقلق الگوریتم یادگیری و استنتاج در ارتباط است. یک مجموعه متناهی از داده ها، صورت های ممکن بسیاری را پیشنهاد می کنند در حالی که تنها برخی از آن ها برای محدود کردن فضای جواب به کار می روند. آن چه در این فصل مورد بررسی قرار گرفت تنها بیان کلی معادلات ابتدای فصل به فرم انتگرال بیز بود. در فصل ۴ به بیان راهکارهایی برای حل این انتگرال خواهیم پرداخت. در شکل (۳-۱) چگونگی مراحل اجرای الگوریتم بیزی ترسیم شده است.

---

[29] Minimum Variance
[30] Unbiased



شکل (۳-۱): نمودار تفصیلی چگونگی الگوریتم تخمین بیزی

فصل چهارم

جواب‌های کاربردی

فرم انتگرال بیز

۴-۱- مقدمه

همان گونه که در فصل نخست عنوان شد، به دلیل دشواری های عملی که در هنگام محاسبه انتگرال بیز پیش می آید، نیاز است به راه حلی عملی برای آن دست یابیم. دستیابی به چنین راه حلی از سه طریق ممکن است:

۱) حل عددی انتگرال؛

۲) تخمین گوسی؛

۳) شبیه سازی مونت کارلو؛

در فصل حاضر قصد داریم به شرح این سه روش بپردازیم؛ هرچند روی حل سوم حساسیت بیشتری داریم. تکرار این نکته مفید فایده خواهد بود که در فصل پیش مسأله یادگیری متوالی را به فرم انتگرال بیز بازنویسی نمودیم و به این نتیجه رسیدیم که با حل انتگرال بیز می توان برای مسأله یادگیری متوالی پاسخ و تخمین درستی یافت.

۴-۲- حل عددی انتگرال

روش حل عددی انتگرال، به مسأله تخمین توزیع موردنظر با استفاده از یک توزیع گسسته روی یک شبکه متناهی از نقاط، منتهی می گردد. البته باید توجه داشت که انتخاب این شبکه متناهی کار چندان ساده ای نیست. هنگامی که توزیع در حال محاسبه در نقاط شبکه است، یک الگوریتم برون یابی در حال تخمین توزیع در فضای باقی مانده است. کیتاگاوا در سال ۱۹۸۷ از این روش برای جایگزینی انتگرال فیلترسازی با مجموع های متناهی روی یک مجموعه بزرگ از نقاط شبکه ای متساوی الفاصله استفاده کرد [۱۴]. در واقع او از یک روش برون یابی خطی تکه ای بهره جست.

کرامر و سورنسون هم در سال ۱۹۸۸ روش مشابهی را انتخاب کردند اما از تابع برون یابی ثابت برای آن استفاده کردند. در سال ۱۹۹۰ پل و وست با پیاده سازی یک روش تخصیص شبکه دینامیک، سعی نمودند از پیچیدگی های مسأله انتخاب این شبکه نقاط بکاهند [۸]، [۱۲] و [۱۳].

هنگامی که فاصله نقاط شبکه به قدر کافی نزدیک به هم باشد و شبکه ناحیه به احتمال بالا را دربرگیرد، این روش خوب کار می کند. اگرچه این روش برای پیاده سازی در مسائل چندمتغیره و با ابعاد بالا، نظیر شبکه های عصبی بسیار دشوار می باشد؛ زیرا محاسبه در همه نقاط این شبکه چندبعدی چگال بسیار گران تمام می شود (گلمان ۱۹۹۶) [۱].

۴-۳- تخمین گوسی

تا سال های اخیر بهترین رویکرد به مسأله تخمین متوالی، تخمین گوسی بود. در مسأله تخمین گوسی خطی، فیلتر کالمن یک الگوریتم بازگشتی بهینه را برای تسری و به هنگام سازی میانگین و کوواریانس حالت های پنهان ارائه می کند. در حالت غیرخطی، فیلتر کالمن توسعه یافته، یک بسط ذاتی مناسب محاسباتی برای فیلتر کالمن است. این الگوریتم بر پایه بسط تیلور معادلات غیرخطی اندازه گیری و دینامیکی سیستم حول آخرین حالت پیش بینی شده، بنا نهاده شده است. به عنوان مثال، بسط های مرتبه اول با تخمین گوسی تابع توزیع چگالی احتمال پیشین، برای این منظور به کار می روند. میانگین و کواریانس با مجموعه ساده ای از معادلات مشابه با معادلات فیلتر کالمن، به هنگام می شوند. همان طور که تعداد جملات بسط تیلور افزایش می یابد، پیچیدگی الگوریتم نیز افزایش می یابد و این به دلیل سخت تر شدن محاسبه مشتقات با درجات بالاتر است.

برای مثال، در یک بسط خطی تنها نیازمند محاسبه ژاکوبین[۳۱] هستیم اما در یک بسط مرتبه دوم باید ماتریس هسین[۳۲] را محاسبه کنیم که به مراتب دشوارتر از محاسبه ژاکوبین است.

تخمین گوسی یک راه حل ساده و زیبا ارائه می دهد که با طرح استنتاج و الگوریتم یادگیری مطابقت دارد. در گذشته، نویسندگان بسیاری سعی نموده اند تابع توزیع ماتریس وزنی شبکه عصبی را توسط الگوریتم تابع گوسی تخمین بزنند. یکی از اولین این تلاش ها توسط سیگنال در سال ۱۹۸۸ با آموزش شبکه پرسپترون از طریق الگوریتم فیلتر کالمن بسط یافته پیاده سازی شده است. در روش پیشنهادی آن ها، عناصر ماتریس وزنی شبکه به صورت یک بردار واحد دسته بندی شده و طبق معادلات فیلتر کالمن بسط یافته به هنگام می شود. ورودی های ماتریس ژاکوبین، که همان مشتقات خروجی ها نسبت به وزن ها می باشند، توسط یک الگوریتم بازگشتی با اندازه گیری خروجی ها در سراسر شبکه محاسبه می شوند.

الگوریتم پیشنهادی سیگنال، محاسبات ریاضی قابل توجهی نیاز دارد. پیچیدگی الگوریتم ضریبی از om^2 برای هر گام از تخمین می باشد، که در این جا m تعداد عناصر ماتریس وزنی و o تعداد خروجی های شبکه را نشان می دهد. به طور کلی این روش و الگوریتم تخمین گوسی راه حل بسیار ساده و مناسبی برای طیف بزرگی از مسائل که در آن ها درجه و پیچیدگی سیستم کم است، می باشد؛ اما هنگامی که درجه سیستم افزایش می یابد، این الگوریتم به دلیل افزایش محاسبات ریاضی، راه مناسب و بهینه ای نیست.

[31] Jacobian
[32] Hessian

۴-۴- الگوریتم مونت کارلو

بسیاری از مسائل پردازش تصویر همانند حذف اِکو از سیگنال های مخابراتی، سری های زمانی مالی، تشخیص های پزشکی، ردیابی هدف و تحلیل داده های ژئوفیزیکی، همگی شامل مولفه های غیرگوسی و غیرخطی و غیر ایستا هستند و درنتیجه نمی توان برای آن ها یک فرم بسته از تخمین گر را برپایه ضوابط استانداردی همچون حداکثر شانس، حداکثر احتمال پیشین و خطای میانگین مربعات حداقل، بیان نمود. همچنین باید توجه داشت که تخمین های جبری که برای توزیع های دقیق داده ها به کار می روند، تمامی شاخصه های برجسته آماری فرآیند موردنظر را دربرندارند. روش شبیه سازی مونت کارلو، که یک توصیف کامل از تابع توزیع احتمال داده ها ارائه می دهد، دقت تحلیل ها را افزایش می دهد.

ایده اصلی در شبیه سازی مونت کارلو این است که یک مجموعه وزن دار از نمونه ها که از تابع توزیع چگالی احتمال پیشین ماتریس وزنی شبکه انتخاب شده اند، برای نگاشت نمودن انتگرال ها به مجموع های گسسته در گام استنتاج، مورد استفاده قرار می گیرد. به صورت دقیق تر می توان گفت که از تخمین مونت کارلو زیر استفاده می کنیم:

$$\hat{p}(\theta_{0:t}|y_{0:t}) = \frac{1}{N}\sum_{i=1}^{N} \delta_{\theta_{0:t}^{(i)}}(d\theta_{0:t}). \qquad (۱-۴)$$

که در آن نمونه های تصادفی $\left\{\theta_{0:t}^{(i)}; i = 1, \cdots, N\right\}$ از توزیع چگالی احتمال پیشین انتخاب شده اند و $\delta(\cdot)$ همان تابع دلتای دیراک است. بنابراین هر جمله امید ریاضی به فرم

$$\mathbb{E}\big(g_t(\theta_{0:t})\big) = \int g_t(\theta_{0:t}) p(\theta_{0:t}|y_{1:t}) d\theta_{0:t}. \qquad (۲-۴)$$

با تخمین زیر تقریب زده می شود:

$$\overline{\mathbb{E}\big(g_t(\theta_{0:t})\big)} = \frac{1}{N}\sum_{i=1}^{N} g_t\left(\theta_{0:t}^{(i)}\right). \qquad (۳-۴)$$

که در معادله اخیر فرض کردیم ذرات $\theta_{0:t}^{(i)}$ مستقل و دارای توزیع یکنواخت برای تخمین هستند. درواقع با توجه به قانون اعداد بزرگ داریم:

$$\overline{\mathbb{E}\big(g_t(\theta_{0:t})\big)} \xrightarrow{N\to\infty} \mathbb{E}\big(g_t(\theta_{0:t})\big). \qquad (۴-۴)$$

به علاوه اگر واریانس پیشین $g_t(\theta_{0:t})$ کراندار باشد، یعنی رابطه زیر برقرار باشد:

$$var_{p(\cdot|d_{1:t})}\big(g_t(\theta_{0:t})\big) < \infty, \qquad (۵-۴)$$

آن گاه قضیه حد مرکزی زیر برقرار می باشد [۴]، [۵]، [۸] و[۲۱]:

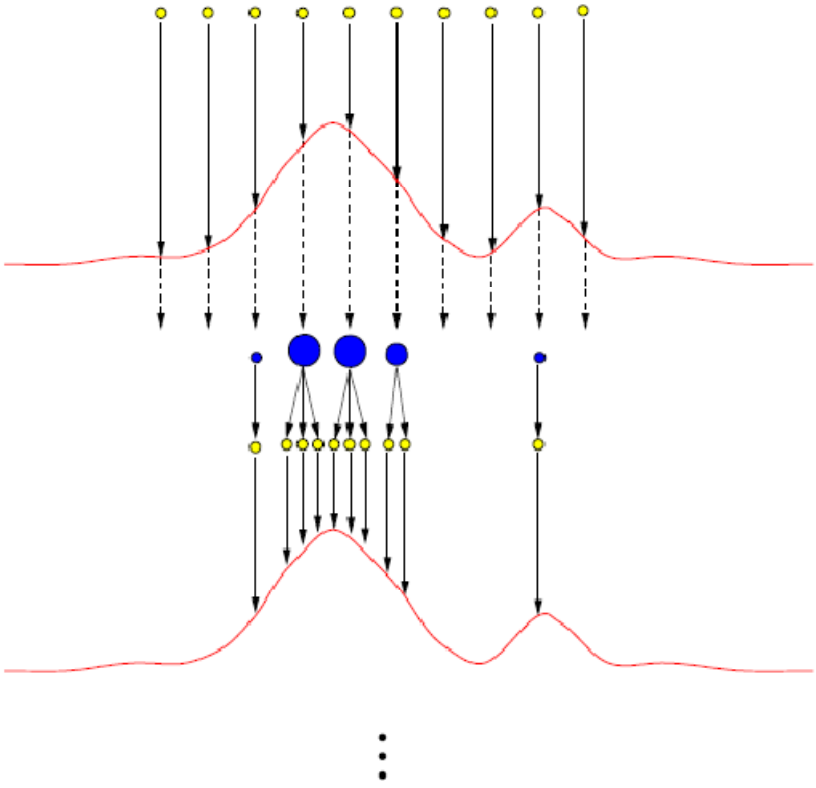
$$\sqrt{N}\left(\overline{\mathbb{E}\big(g_t(\theta_{0:t})\big)} - \mathbb{E}\big(g_t(\theta_{0:t})\big)\right) \qquad (۶-۴)$$
$$\xRightarrow[N\to\infty]{} \mathcal{N}\left(0, var_{p(\cdot|d_{1:t})}\big(g_t(\theta_{0:t})\big)\right).$$

اگرچه در سال های اخیر، بسیاری از محققان در زمینه علوم آماری و پردازش سیگنال ها الگوریتم های زیادی را برای روش مونت کارلو ارائه کرده اند و همان طور که در فصل ۱ آمده است این الگوریتم ها روی مسائل بسیاری پیاده سازی شده اند. اما روش اصلی مونت کارلو در دهه ۶۰ میلادی در زمینه کنترل و اوتوماسیون به کار رفته است. به عنوان مثال، هندشین[۳۳] و ماین[۳۴] در سال ۱۹۶۹ مسأله فیلترینگ غیرخطی را توسط راهبرد نمونه برداری با اهمیت متوالی حل کردند. آن ها روش های تحلیلی و عددی را با هم ترکیب کرده و از تکنیک کنترل متغیرها جهت کاهش واریانس تخمین استفاده کردند. در دهه ۷۰ میلادی افراد بیشتری روی این تکنیک ها کار کردند که از آن جمله می توان به آکاشی[۳۵] و کوماموتو[۳۶] ۱۹۷۷ و زاریتسکی[۳۷] ۱۹۷۵ اشاره نمود. شکل ۴-۱ عملکرد

[33] Hendshin
[34] Maein
[35] Aka Shi
[36] Komamoto
[37] Saritsky

کلی الگوریتم مونت کارلو متوالی را نشان می دهد. دقت کنید تنها ذرات مناسب تر یعنی آن هایی که بالاترین شانس را دارند، در گام به هنگام سازی انتخاب می شوند. سپس بر اساس شانسشان در گام پیش بینی در هم ضرب می شوند.

همان طور که در شکل (۴-۱) مشاهده می کنید در گام به هنگام سازی، شانس (احتمال) هر ذره محاسبه می شود. اندازه هر دایره نشان گر این احتمال است. سپس هر ذره با توجه به احتمال موردنظرش انتخاب می شود. در این فرآیند، ذرات با احتمال بیشتر می توانند بچه های بیشتری داشته باشند. آن گاه الگوریتم مقدار پیش بینی شده هر ذره را با استفاده از معادلات انتقال محاسبه می کند. در نهایت نتیجه این است که ذرات باقیمانده، توصیف وزن دار بهتری نسبت به قبل برای تابع توزیع احتمال ارائه می دهد.



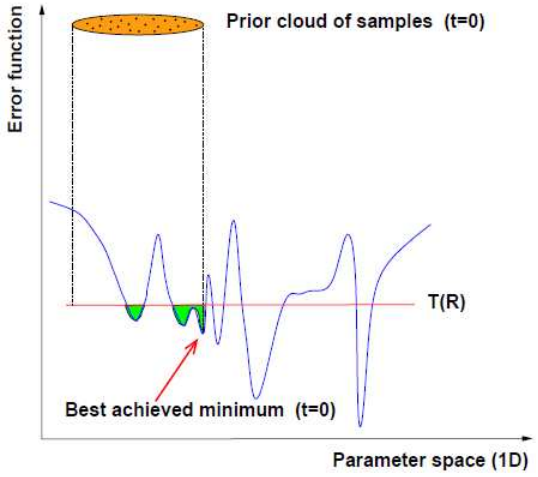
شکل (۴-۱): گام به هنگام سازی [۷]

بررسی مسأله از دیدگاه بهینه سازی می تواند آموزنده باشد. شکل های (۴-۲) و(۴-۳)، چگونگی این امر را نشان می دهد. دیاگرام رسم شده نحوه تأثیرگذاری ماتریس های کوواریانس نویز یعنی R و Q را نشان می دهد. در واقع این ماتریس Q است که تعیین می کند در گام پیش بینی، ابر نمونه ها[۳۸] چقدر باید بسط یابد. با افزایش Q چگالی ابر نمونه ها کاهش می یابد. درنتیجه الگوریتم زمان بیشتری برای همگرایی صرف خواهد کرد. از سوی یک مقدار بسیار کوچک Q، این امکان را از الگوریتم می گیرد تا به نواحی دیگری از فضای نمونه ها نیز دسترسی داشته باشد. پس در واقع به الگوریتم برای تنظیم خودکار Q نیاز داریم. با توجه به شکل (۴-۳)، R دقت گام به هنگام سازی را کنترل می کند [۶]، [۷] و [۲۰]. مقادیر کوچک تر R باعث می شوند فضای احتمال بسیار باریک شود؛ بنابراین تنها تعداد کمی از مسیرهای جواب این امکان را خواهند یافت تا به مرحله بعدی از الگوریتم تخمین را بیابند. اگر مقدار R بسیار کوچک شود، ممکن است الگوریتم بهینه سازی از شناسایی برخی از مدهای مهم تابع توزیع احتمال باز بماند. با افزایش R، ما در واقع دامنه تابع توزیع احتمال و متعاقب آن تعداد مسیرهای مجاز را افزایش داده ایم. توجه داشته باشید اگر مقدار R را بسیار بزرگ انتخاب کنیم، تمامی مسیرها احتمال یکسانی برای ورود به مرحله بعدی تخمین خواهند داشت و الگوریتم هیچ گاه همگرا نمی شود. همان گونه که از شکل ها مشخص است، R به یک مقدار آستانه ای[۳۹] $T(R)$ در تابع خطا می رسد که عرض ابر به هنگام شده نمونه ها را تعیین می کند.

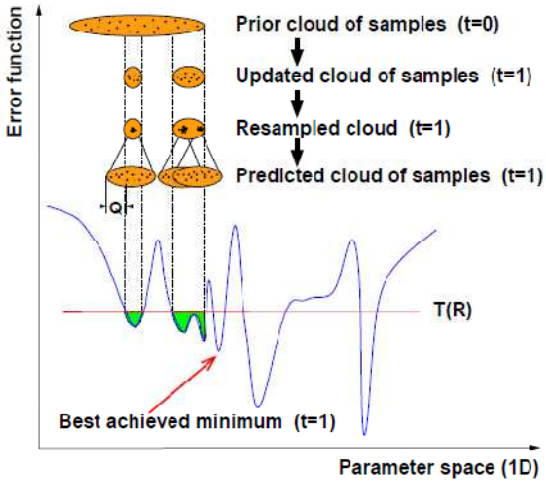
به عبارت دیگر، اندازه ابر نمونه های اولیه، بازه تحقیق را در فضای پارامترها تعیین می کند. هدف یافتن بهترین مینیمم ممکن برای تابع خطاست. پر واضح است که با افزایش نمونه ها، شانس یافتن چنین مینیممی نیز افزایش می یابد. در گام دوم، ابرهای به هنگام شده نمونه ها تولید می شود. عرض این ابرها با تعیین فصل مشترک میان عرض ابرهای پیشین با تابع خطا و مقدار آستانه ای به دست آمده از ماتریس کوواریانس نویز اندازه گیری R، محاسبه می گردد. مشخصاً ابر نمونه های

[38] Cloud of Samples
[39] Threshold

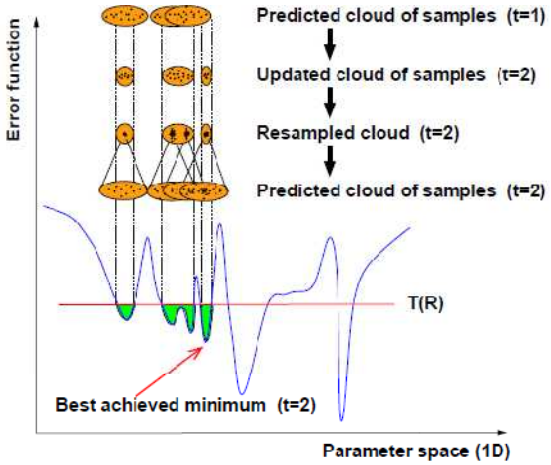
جدید چگال تر از ابر نمونه های پیشین است. سپس در مرحله نمونه برداری مجدد، نمونه ها در نواحی با احتمال بیشتر گروه بندی می شوند. در نهایت، این ابر نمونه ها با استفاده از یک ضریب که از روی ماتریس کوواریانس نویز فرآیند محاسبه می شود، منبسط می گردد. این انبساط به ابرها این امکان را می دهد تا به نواحی از توابع پارامتری دسترسی یابند که در آن ها تابع خطا کمینه است.



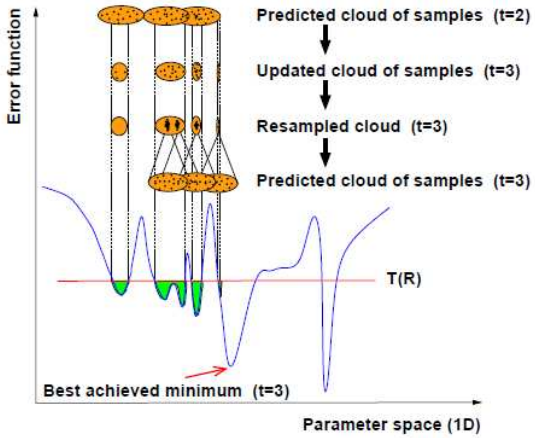
شکل (۴-۲) : گام یکم الگوریتم مونت کارلو [۷]



شکل (۴-۳) : گام دوم الگوریتم مونت کارلو [۷]



شکل (۴-۴) : گام سوم الگوریتم مونت کارلو[۷]



شکل (۴-۵) : گام چهارم الگوریتم مونت کارلو[۷]

شکل های (۴-۴) و (۴-۵) گام های سوم و چهارم مسأله تخمین مونت کارلو را نشان می دهند و همان طور که مشاهده می کنید برای رسیدن به مینیمم سراسری، تعداد نمونه ها افزایش می یابد [۷].

۴-۵- نمونه برداری اهمیت[40]

روش های نمونه برداری مختلفی همچون زنجیره مارکوف و نمونه برداری با اهمیت متوالی را می توان برای نمونه برداری از تابع احتمال پیشین استفاده نمود. ما در این رساله از روش دوم استفاده می کنیم. همان گونه که در قسمت های پیشین نیز اشاره شد، می توان تابع توزیع احتمال پیشین را به احتی با استفاده از یک تابع گسسته متناهی تخمین زد. از قانون اعداد بزرگ استفاده می شود که با افزایش تعداد نمونه ها، امید ریاضی را می توان به مجموع، نگاشت نمود. متأسفانه همواره نمی توان تابع توزیع چگالی احتمال پیشین را نمونه برداری کرد، اما می توان این مشکل را با نمونه برداری از توزیع پیشنهادی و شناخته شده $q(\theta_{0:t}|y_{0:t})$ که نمونه برداری از آن ساده تر است، حل کرد.

با استفاده از جانشینی زیر داریم:

$$\begin{aligned}\mathbb{E}\big(g_t(\theta_{0:t})\big) &= \int g_t(\theta_{0:t})\frac{p(\theta_{0:t}|y_{1:t})}{q(\theta_{0:t}|y_{1:t})}q(\theta_{0:t}|y_{1:t})d\theta_{0:t} \\ &= \int g_t(\theta_{0:t})\frac{p(y_{1:t}|\theta_{0:t})p(\theta_{0:t})}{p(y_{1:t})q(\theta_{0:t}|y_{1:t})}q(\theta_{0:t}|y_{1:t})d\theta_{0:t} \\ &= \int g_t(\theta_{0:t})\frac{w_t(\theta_{0:t})}{p(y_{1:t})}q(\theta_{0:t}|y_{1:t})d\theta_{0:t}.\end{aligned} \quad (۴-۷)$$

که در آن متغیرهای $w_t(\theta_{0:t})$ به عنوان وزن های نرمالیزه نشده با اهمیت شناخته می شوند:

$$w_t(\theta_{0:t}) = \frac{p(y_{1:t}|\theta_{0:t})p(\theta_{0:t})}{q(\theta_{0:t}|y_{1:t})}. \quad (۴-۸)$$

[40] Importance Sampling

همچنین به راحتی می توان چگالی نامعلوم $p(y_{1:t})$ را به جملاتی معلوم تبدیل نمود:

$$\mathbb{E}\big(g_t(\theta_{0:t})\big) = \frac{1}{p(y_{1:t})}\int g_t(\theta_{0:t}) w_t(\theta_{0:t}) q(\theta_{0:t}|y_{1:t}) d\theta_{0:t} \qquad (۹-۴)$$

$$= \frac{\int g_t(\theta_{0:t}) w_t(\theta_{0:t}) q(\theta_{0:t}|y_{1:t}) d\theta_{0:t}}{\int p(y_{1:t}|\theta_{0:t}) p(\theta_{0:t}) \frac{q(\theta_{0:t}|y_{1:t})}{q(\theta_{0:t}|y_{1:t})} d\theta_{0:t}}$$

$$= \frac{\int g_t(\theta_{0:t}) w_t(\theta_{0:t}) q(\theta_{0:t}|y_{1:t}) d\theta_{0:t}}{\int w_t(\theta_{0:t}) q(\theta_{0:t}|y_{1:t}) d\theta_{0:t}}$$

$$= \frac{\mathbb{E}_q\big(w_t(\theta_{0:t}) g_t(\theta_{0:t})\big)}{\mathbb{E}_q\big(w_t(\theta_{0:t})\big)}.$$

دقت کنید که نماد $\mathbb{E}_q$ برای نشان دادن این مطلب امید ریاضی روی توزیع پیشنهادی $q(\cdot\,|y_{1:t})$ گرفته شده است، به کار می رود. پس با انتخاب نمونه ها از تابع پیشنهادی $q(\cdot\,|y_{1:t})$، می توان امید ریاضی را به صورت زیر تخمین زد:

$$\mathbb{E}\big(g_t(\theta_{0:t})\big) = \frac{1/N \sum_{i=1}^{N} g_t\left(\theta_{0:t}^{(i)}\right) w_t\left(\theta_{0:t}^{(i)}\right)}{1/N \sum_{i=1}^{N} w_t\left(\theta_{0:t}^{(i)}\right)} \qquad (۱۰-۴)$$

$$= \sum_{i=1}^{N} g_t\left(\theta_{0:t}^{(i)}\right) \widetilde{w}_t\left(\theta_{0:t}^{(i)}\right).$$

که در آن وزن های نرمالیزه شده $\widetilde{w}_t^{(i)}$ ، توسط رابطه زیر داده می شود:

$$\widetilde{w}_t^{(i)} = \frac{w_t^{(i)}}{\sum_{j=1}^{N} w_t^{j}}. \qquad (۱۱-۴)$$

تخمین حاصله با توجه به قضیه حد مرکزی به شرط برقراری شروط زیر همگرا خواهد بود[۱]و [۲۱]:

۱) $\theta_{0:t}^{(i)}$ از میان یک مجموعه از نمونه ها با توزیع یکنواخت مستقل انتخاب می شود، و پایه های تابع توزیع پیشنهادی شامل پایه های تابع توزیع احتمال پیشین باشد. همچنین امید ریاضی $\mathbb{E}\left(g_t(\theta_{0:t})\right)$ موجود و متناهی باشد؛

۲) امید ریاضی w_t و $w_t g_t^2(\theta_{0:t})$، روی تابع توزیع احتمال پیشین موجود و متناهی باشد؛

با برقراری شرایط فوق و میل کردن مقدار N به بی نهایت، تابع توزیع احتمال پیشین به راحتی با رابطه زیر تخمین زده می شود:

$$p(\theta_{0:t}|y_{1:t}) = \sum_{i=1}^{N} \widetilde{w}_t^{(i)} \delta_{\theta_{0:t}^{(i)}}(d\theta_{0:t}) \,. \qquad (۱۲-۴)$$

در فصل بعدی به بیان الگوریتم نمونه برداری اهمیت متوالی می پردازیم.

فصل پنجم

الگوریتم نمونه برداری

اهمیت متوالی

۵-۱- مقدمه

در این فصل می خواهیم به بیان الگوریتم بسیار کارای نمونه برداری با اهمیت متوالی[41] بپردازیم. با توجه به مطالبی که در فصل گذشته آمد، می توان تابع توزیع احتمال پیشنهادی را به صورت زیر بسط داد:

$$q(\theta_{0:t}|y_{1:t}) = q(\theta_0|y_{1:t}) \prod_{j=1}^{t} q\left(\theta_j|\theta_{1:j-1}, y_{1:t}\right). \qquad (۱-۵)$$

اما به منظور محاسبه یک تخمین متوالی از تابع توزیع چگالی احتمال پیشین در زمان t، بدون اصلاح حالت های شبیه سازی شده قبلی $\theta_{0:t-1}$، تابع توزیع زیر را می توان استفاده نمود:

$$q(\theta_{0:t}|y_{1:t}) = q(\theta_0) \prod_{j=1}^{t} q\left(\theta_j|\theta_{1:j-1}, y_{1:j}\right) \qquad (۲-۵)$$

$$= q(\theta_{0:t-1}|y_{1:t-1}) q(\theta_t|\theta_{0:t-1}, y_{1:t}).$$

در این مرحله نیاز است بار دیگر تأکید کنیم که فرض بر این است که حالت های سیستم، یک فرآیند مارکوف و مقادیر مشاهده ها از حالت های سیستم مستقل مشروط می باشند. یعنی:

$$p(\theta_{0:t}) = p(\theta_0) \prod_{j=1}^{t} p\left(\theta_j|\theta_{j-1}\right). \qquad (۳-۵)$$

و

$$p(y_{1:t}|\theta_{0:t}) = \prod_{j=1}^{t} p\left(y_j|\theta_j\right). \qquad (۴-۵)$$

[41] Sequential Importance Sampling (SIS)

با جاگذاری این فرمول ها در فرمول وزن که در فصل قبل ارائه شد، می توان به یک رابطه بازگشتی برای وزن های با اهمیت دست یافت:

$$w_t = \frac{p(y_{1:t}|\theta_{0:t})p(\theta_{0:t})}{q(\theta_{0:t-1}|y_{1:t-1})q(\theta_t|\theta_{0:t-1}, y_{1:t})} \qquad (۵-۵)$$
$$= w_{t-1}\frac{p(y_{1:t}|\theta_{0:t})p(\theta_{0:t})}{p(y_{1:t-1}|\theta_{0:t-1})p(\theta_{0:t-1})}$$
$$\cdot \frac{1}{q(\theta_t|\theta_{0:t-1}, y_{1:t})} = w_{t-1}\frac{p(y_t|\theta_t)p(\theta_t|\theta_{t-1})}{q(\theta_t|\theta_{0:t-1}, y_{1:t})}.$$

همان طور که ملاحظه می شود، معادله اخیر یک مکانیسم بازگشتی برای به هنگام سازی وزن های با اهمیت ارائه می کند. پس می توان به راحتی از تابع پیشنهادی نمونه برداری کرد و مقادیر احتمال و شانس ذره ها را محاسبه نمود. در واقع همه آن چه باید انجام دهیم، تولید یک مجموعه از نمونه های پیشین و محاسبه تکراری مقادیر وزن های با اهمیت است. این روند همان الگوریتم نمونه برداری با اهمیت متوالی (SIS) می باشد. نکته حائز اهمیت در مورد این الگوریتم این است که واریانس ضریب $\widetilde{w}_t \triangleq \frac{p(\theta_{0:t}|y_{1:t})}{q(\theta_{0:t}|y_{1:t})}$ ، با گذشت زمان و انتخاب تصادفی مشاهده ها افزایش می یابد.

قضیه ۵-۱ : واریانس غیرمشروط (یعنی هنگامی که مشاهدات تصادفی باشند) ضرایب اهمیت، با گذشت زمان افزایش می یابد (دوکت ۱۹۹۹) [۶]، [۸] و [۲۱].

به عبارت دیگر، چون از تابع توزیع احتمال پیشین نمونه برداری کرده ایم، بنابراین مطلوب این است که تابع توزیع احتمال پیشنهادی به تابع توزیع پیشین نزدیک باشد (از لحاظ تخمینی)؛ و اگر چنین شود آن گاه روابط زیر برقرار خواهد بود:

$$\mathbb{E}_q\left[\frac{p(\theta_t|\theta_{0:t-1},y_{1:t})}{q(\theta_t|\theta_{0:t-1},y_{1:t})}\right]=1. \qquad (۵-۶)$$

و

$$var_q\left[\frac{p(\theta_t|\theta_{0:t-1},y_{1:t})}{q(\theta_t|\theta_{0:t-1},y_{1:t})}\right]=\mathbb{E}_q\left[\left(\frac{p(\theta_t|\theta_{0:t-1},y_{1:t})}{q(\theta_t|\theta_{0:t-1},y_{1:t})}-1\right)^2\right]=0. \qquad (۵-۷)$$

به عبارت دیگر، باید مقدار واریانس نزدیک به صفر باشد تا تخمین دقیقی داشته باشیم. پس افزایش واریانس می تواند مخرب باشد و روی مشاهدات تأثیر بگذارد.

۵-۲- نمونه برداری مجدد[۴۲]

برای جلوگیری از اثر تخریبی افزایش واریانس در الگوریتم SIS ، اضافه کردن یک گام نمونه برداری دیگر در الگوریتم برای حذف نمونه های با ضریب اهمیت کم و عبور نمونه های با ضریب بالا، می تواند مفید باشد. این نمونه برداری مجدد شامل نگاشت مجموعه اندازه‌گیری‌های تصادفی دیراک $\left\{\theta_{0:t}^{(i)},\widetilde{w}_t^{(i)}\right\}$ به داخل یک مجموعه اندازه‌گیری‌های تصادفی با وزن‌های یکسان $\left\{\theta_{0:t}^{(i)},N^{-1}\right\}$ صورت می‌گیرد. این کار با نمونه برداری یکنواخت از مجموعه گسسته

$$\left\{\theta_{0:t}^{(i)},i=1,2,\cdots,N\right\}. \qquad (۵-۸)$$

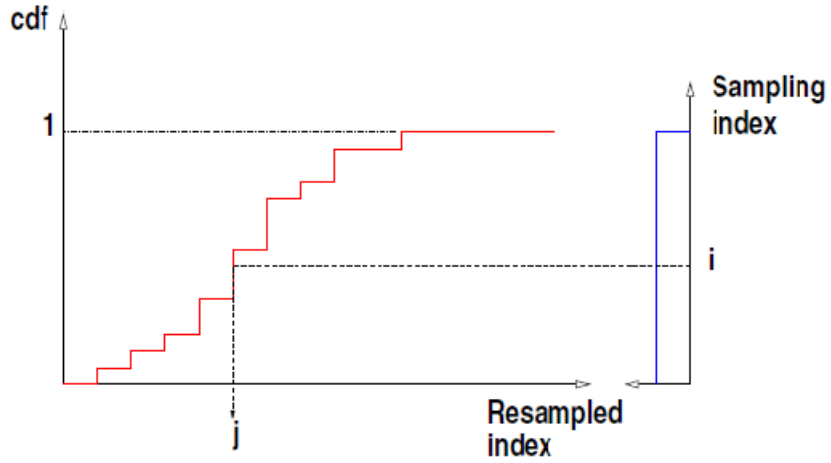
و احتمال

$$\left\{\widetilde{w}_t^{(i)},i=1,2,\cdots,N\right\}. \qquad (۵-۹)$$

انجام می شود.

[42] Resampling

شکل (۵-۱) یک مدل از این نحوه نمونه برداری را نشان می دهد. پس از تولید این توزیع تجمعی از مجموعه گسسته، شاخص نمونه برداری یکنواخت یعنی i را ابتدا روی برد این تابع و سپس روی دامنه آن تصویر می کنیم. فصل مشترک حاصل با دامنه شاخص نمونه برداری جدید j را به دست می دهد. در واقع بردار $\theta_{0:t}^{(j)}$ به عنوان نمونه جدید انتخاب می شود. توجه دارید که بردارهای با ضریب نمونه برداری بزرگ تر با کپی های بیشتری مرحله نمونه برداری مجدد را پشت سر می گذارند. در گام بعدی پیش بینی، اغتشاش های تصادفی به نمونه ها افزوده می شود که در نتیجه یک نوع گوناگونی به نمونه های غالب می بخشد.



شکل (۵-۱): نمونه برداری یکنواخت [۸]

لیو و چن (۱۹۹۵ و ۱۹۹۸) استدلال کردند هنگامی که ضرایب مهم تقریباً مساوی هستند، نمونه برداری مجدد تنها تعداد این شاخص ها را کاهش می دهد و یک تناوب در شبیه سازی ایجاد می کند [۸] و [۲۱]. با استفاده از نتایج تحقیقات کونگ ۱۹۹۴، آن ها پیشنهاد کردند که نمونه برداری مجدد فقط هنگامی انجام شود که اندازه (تعداد) موثر نمونه ها N_{eff} از یک مقدار آستانه ای ثابت کمتر باشد؛ که مقدار موثر با رابطه زیر بیان می شود:

$$N_{eff} = \frac{1}{\sum_{i=1}^{N}\left(\widetilde{w}_t^{(i)}\right)^2}. \qquad (۱۰-۵)$$

تنها مسأله ای که باقی می ماند، نحوه انتخاب توابع چگالی احتمال شانس و پیشنهادی است. در ادامه به شرح روش انتخاب آن ها پرداخته و سپس در پایان این فصل الگوریتم نمونه برداری مجدد را ارائه می کنیم.

۵-۳- انتخاب توابع شانس و پیشنهاد

تابع چگالی احتمال شانس را می توان بر اساس جملاتی از معادلات اندازه گیری به راحتی بیان نمود. در رساله حاضر فرم گوسی زیر را پیشنهاد می کنیم:

$$p(y_t|\theta_t) \propto exp\left(-\frac{1}{2}\left(y_t - \hat{f}_t(x_t,\theta_t)\right)' R^{-1}\left(y_t - \hat{f}_t(x_t,\theta_t)\right)\right). \qquad (۱۱-۵)$$

و برای الگوریتم های طبقه بندی دودوئی با توابع خروجی منطقی نیز می توان فرم توزیع برنولی زیر را پیشنهاد نمود:

$$p(y_t|\theta_t) = \hat{f}_t(x_t,\theta_t)^{y_t}\left(1 - \hat{f}_t(x_t,\theta_t)\right)^{1-y_t}. \qquad (۱۲-۵)$$

انتخاب تابع توزیع چگالی احتمال پیشنهادی در قسمت پیشین، در واقع یکی از مسائل باز و بحث برانگیز در زمینه الگوریتم نمونه برداری با اهمیت متوالی می باشد. اولین بار دوکت در سال ۱۹۹۹ نشان داد که تابع توزیع پیشنهادی، واریانس ضرایب با اهمیت را کاهش می دهد.

قضیه ۵-۲ : توزیع اهمیت $q(\theta_t|\theta_{0:t-1}, y_{1:t}) = p(\theta_t|\theta_{0:t-1}, y_{1:t})$ ، واریانس ضرایب اهمیت مشروط بر $\theta_{0:t-1}$ و $y_{1:t}$ را کاهش می دهد (دوکت ۱۹۹۹) [۶]، [۸] و [۲۱].

مسأله انتخاب تابع توزیع چگالی احتمال پیشنهادی، همواره یکی از موارد مورد علاقه محققان بوده است. از جمله افرادی که در این زمینه کار کرده اند می توان به کونگ [۴۳]۱۹۹۴، لیو و چن [۴۴]۱۹۹۵ و زاریتسکی[۴۵] ۱۹۷۴ اشاره نمود. اما این دوکت بود که در واقع قضیه را اثبات کرد و توزیع بسیار معروف و پرکاربرد زیر را ارائه نمود:

$$q(\theta_t|\theta_{0:t-1}, y_{1:t}) = p(\theta_t|\theta_{t-1}). \qquad (۱۳-۵)$$

۵-۴- الگوریتم SIR

در بخش های گذشته، نحوه تقویت الگوریتم نمونه برداری توسط نمونه برداری مجدد و طریقه محاسبه متوالی ضرایب اهمیت را بیان کردیم. حال قصد داریم الگوریتم نمونه برداری اهمیت متوالی را ارائه کنیم. در ادامه الگوریتم کامل نمونه برداری اهمیت متوالی را برای پیاده سازی روی شبکه عصبی پرسپترون، با جزئیات آن آورده ایم:

الگوریتم SIR

---

۱) در زمان $t = 0$؛

برای $i = 1,2, \cdots, N$، وزن های $\theta_0^{(i)}$ را از لایه های اول $p_1(\theta_0)$ و دوم $p_2(\theta_0)$ پیشین بسازید.

۲) برای $t = 1,2, \cdots$؛

۲-۱) گام نمونه برداری اهمیت بیزی

---

[43] Kong
[44] Leu & Chen
[45] Saritsky

- برای $i = 1,2,\cdots,N$،

$$\hat{\theta}_t^{(i)} \sim q\left(\theta_t \middle| \theta_{0:t-1}^{(i)}, y_{1:t}\right)$$

و

$$\hat{\theta}_{0:t}^{(i)} \triangleq \left(\hat{\theta}_{0:t-1}^{(i)}, \hat{\theta}_t^{(i)}\right)$$

- برای $i = 1,2,\cdots,N$، تمامی ضرایب اهمیت را تا رسیدن به مقدار آستانه ای زیر محاسبه کنید:

$$w_t^{(i)} = w_{t-1}^{(i)} \frac{p\left(y_t \middle| \hat{\theta}_t^{(i)}\right) p\left(\hat{\theta}_t^{(i)} \middle| \hat{\theta}_{t-1}^{(i)}\right)}{q\left(\hat{\theta}_t^{(i)} \middle| \hat{\theta}_{0:t-1}^{(i)}, y_{1:t}\right)}$$

- برای $i = 1,2,\cdots,N$، وزن های اهمیت را نرمالیزه کنید:

$$\widetilde{w}_t^{(i)} = w_{t-1}^{(i)} \left[\sum_{j=1}^{N} w_t^{(i)}\right]^{-1}$$

۲-۲) اگر (مقدار آستانه ای $\geq N_{\text{eff}}$)؛

- برای $i = 1,2,\cdots,N$، قرار بده: $\theta_{0:\mathrm{t}}^{(\mathrm{i})} = \hat{\theta}_{0:\mathrm{t}}^{(\mathrm{i})}$
- در غیر این صورت نمونه های $\hat{\theta}_{0:\mathrm{t}}^{(\mathrm{i})}$ را با وزن های به اهمیت بالای $\widetilde{w}_t^{(i)}$ ضرب کنید تا به N نمونه تصادفی $\theta_{0:\mathrm{t}}^{(\mathrm{i})}$ برسید که نسبت به $p\left(\theta_{0:\mathrm{t}}^{(\mathrm{i})} \middle| y_{1:t}\right)$ توزیع شده اند.
- برای $i = 1,2,\cdots,N$، قرار بده: $w_t^{(i)} = \widetilde{w}_t^{(i)} = \frac{1}{N}$.

الگوریتم عمومی SIR برای پیاده سازی بسیار سرراست است. اگر بخواهیم از تابع توزیع احتمال پیشین $p(\theta_t|\theta_{0:t-1})$ نمونه برداری کنیم، آن گاه داریم:

$$\hat{\theta}_{t+1}^{(i)} = \theta_t^{(i)} + u_t^{(i)}. \qquad (۱۴-۵)$$

که در آن $u_t^{(i)} \sim \mathcal{N}(0, Q)$. برای این انتخاب، وزن های اهمیت به صورت زیر در می آیند:

$$w_t^{(i)} = w_{t-1}^{(i)} p\left(y_t \middle| \hat{\theta}_t^{(i)}\right). \qquad (۱۵-۵)$$

در فصل پایانی ابتدا به بررسی الگوریتم اصلاحی فیلتر ذره ای برای شبکه عصبی کوهنن پرداخته و در ادامه الگوریتم فصل حاضر و الگوریتم کوهنن اصلاحی را در چند مثال کاربردی خواهیم آزمود.

فصل ششم

پیاده سازی الگوریتم کوهنن اصلاحی

و

شبیه سازی کامپیوتری

۶-۱- مقدمه[۴۶]

در این فصل ابتدا به بررسی عملکرد الگوریتم پیشنهادی کوهنن تلفیقی می‌پردازیم، سپس در ادامه چند مساله شبیه سازی شده در محیط نرم افزار MATLAB با استفاده از الگوریتم کوهنن تلفیقی ارائه می‌نماییم. الگوریتم پیشنهادی بر پایه مطالب فصول گذشته تنظیم شده و همانطور که پیشتر در فصل دوم گفته شد، توانستیم با این الگوریتم هم مشکل واحد مرده را رفع نماییم و هم گام تعیین ضرایب شبکه را اصلاح کنیم. در پایان فصل نیز به تحلیل نتایج حاصله از پایان نامه، بررسی مزایا و معایب استفاده از این الگوریتم، و در نهایت بیان چند زمینه پرکاربرد شبکه های عصبی رقابتی از جمله شبکه عصبی کوهنن پرداخته‌ایم.

۶-۲- آموزش شبکه عصبی کوهنن با استفاده از الگوریتم فیلتر ذره ای

همانطور که در فصل دوم آمد، تفاوت اساسی شبکه‌های عصبی خودسازمانده با دیگر شبکه‌های عصبی کلاسیک در این است که خروجی دقیق و صحیح را نمی‌توان به عنوان یک دانسته مساله فرض نمود و بنابراین یک مقیاس عددی از میزان دامنه خطای نگاشت شبکه عصبی در دسترس نیست. یعنی به دلیل عدم دسترسی به خروجی واقعی و صحیح، امکان استفاده از پارامتر خطای نگاشت خروجی به عنوان یک پارامتر مقایسه‌ای و یک مقیاس مناسب برای بهتر کردن پاسخ شبکه و بهینه نمودن آموزش شبکه وجود ندارد. البته همچنان مانند دیگر شبکه‌های کلاسیک این قانون یادگیری است که باعث بهبود ضرایب شبکه شده و سلول‌ها را به سمت ورودی شبکه حرکت می‌دهد.

از طرفی به دلیل ماهیت شبکه دیدیم که ممکن است برخی از سلول‌های شبکه خودبه‌خود به دو صورت از دور رقابت خارج شوند. اول این که یا از همان ابتدا به دلیل واقع نشدن سلول در ناحیه اعمال ورودی امکان تغییر ضرایب ماتریس وزن این سلول‌ها وجود ندارد و در نهایت آنقدر فاصله‌ی ضرایب آنها با دیگر سلول‌ها زیاد می شود که دیگر شانسی برای پیروزی در رقابت نخواهند داشت. دومین دلیل نیز آنگونه که پیش از این اشاره نمودیم می تواند به دلیل تمرکز بیش از حد سلول‌ها در مرکز صفحه ورودی و اعمال ماتریس ورودی به سلول‌های مرزی و تاثیر بیشتر بر این گونه سلول‌ها باشد. که در این صورت نیز حتی با کاهش سرعت تغییرات نبز دیگر این سلول‌ها امکان برنده شدن در رقابت را ندارند. در این پایان نامه سعی نموده‌ایم تا با استفاده از الگوریتم فیلتر ذره ای این دو مشکل شبکه عصبی کوهنن را رفع کرده، الگوریتم بهتری برای اجرای شبکه ارائه نماییم.

[۴۶] تمامی مطالب این فصل و شکل ها و نتایج شبیه سازی از نگارنده پایان نامه می باشد.

در اینجا در واقع باز هم از همان معیار فاصله اقلیدسی برای تعیین سلول برنده استفاده می-کنیم اما در نحوه اجرای آن تغییرات مختصری اعمال شده است تا بتوان در مدل شبکه عصبی رقابتی خودسازمانده کوهنن نیز همانند مدل فضای حالت شبکه‌های عصبی کلاسیک با استفاده از الگوریتم فیلتر ذره‌ای ماتریس ضرایب شبکه عصبی را تخمین زد تا در نهایت با حرکت سلول‌ها به سمت ورودی تخمین بهتری از فضای ورودی اعمالی به شبکه دست یابیم. همانطور که اشاره شد در اینجا هم قانون یادگیری پایه برای اجرای شبکه همان الگوریتم کوهنن است که فقط در هر مرحله تکرار یک گام تصحیح برای اصلاح ضرایب سلول‌های شبکه که شانس برنده شدنشان کمتر از دیگر سلول هاست اضافه نموده‌ایم. یک نکته دیگر هم در مورد الگوریتم تلفیقی حائز اهمیت است و آن این که مقداردهی اولیه ماتریس وزن شبکه عصبی کوهنن را نیز توسط روش مونت کارلو برای تولید رشته تصادفی اعداد انجام داده‌ایم. این کار در واقع برای جلوگیری از تمرکز بیش از اندازه سلول ها در ابتدای کار و تنظیم حالت اولیه ضرایب ماتریس وزنی شبکه صورت گرفته است. در ادامه به بررسی نحوه عملکرد این گام تصحیحی خواهیم پرداخت و در پایان نیز نتایج شبیه سازی‌های کامپیوتری را با استفاده از این الگوریتم تصحیح شده آورده ایم.

مکانیسم گام تصحیح بسیار ساده عمل می کند در هر گام پس از تعیین سلول برنده ابتدا سلول‌هایی که در رقابت شرکت نداشته اند –همان واحدهای مرده شبکه عصبی کوهنن- را شناسایی می شوند و بعد آن سلول یا سلول‌ها از رقابت حذف شده، سلول جدیدی به جای آنها جایگزین می شود. وزنی که به این سلول‌های جدید اختصاص می یابد با توجه به مقدار آستانه -که از روی وزن سلول‌های برنده با استفاده از معیار فاصله اقلیدسی شبکه کوهنن تعیین شده- معلوم می شود. اصولاً دو گروه از سلول‌ها در این گام تصحیحی حذف می شوند. یکی آن دسته از سلول ها که مقدار وزن آنها در مقایسه با مقدار آستانه ای بسیار بزرگتر است و دیگری آن سلول هایی که دارای یک همسایه و وزن بسیار کوچکی در مقایسه با مقدار آستانه ای می باشند.

در مورد گروه اول باید گفت اینگونه سلول ها بیشتر در مناطق مرزی متمرکزاند که آن مناطق مرزی در ناحیه اعمال ورودی قرار نگرفته اند و همین مساله باعث کاهش شانس سلول های این دسته یا حتی صفر شدن شانس آنها در رقابت شده است. در مورد دسته دوم نیز همانطور که در بخش۲-۵-۲-۵ در مورد توپولوژی‌های معمول شبکه‌های عصبی رقابتی خودسازمانده آمد، هر سلول در توپولوژی مسطح اینگونه شبکه ها چهار همسایه دارد که در برخی از موارد که البته تعداد آن برای شبکه عصبی کوهنن کم هم نیست، برخی سلولها به حدی نزدیک یکدیگر قرار می گیرند که به نظر می رسد روی هم قرار گرفته‌اند. مسلماً حذف ایگونه سلول‌ها نه تنها به بهتر شدن پاسخ شبکه -یا همان نگاشت

شبکه- برای تخمین فضای ورودی کمک خواهد کرد، بلکه اجرای الگوریتم را در گام های بعدی تسهیل می کند. آخرین نکته در مورد الگوریتم که باید در اینجا به آن اشاره کرد اینست که همانطور که در فصل دوم گفته شد، دو روش برای تغییر ضرایب وزنی سلول های شبکه وجود دارد. یکی اینکه فقط ضرایب سلول برنده و همسایه های آن را با توجه به شعاع همسایگی الگوریتم که توسط کاربر تعیین می شود، تغییر دهیم. روش دوم تغییر ضرایب وزنی تمامی سلول های شبکه عصبی بدون توجه به اینکه آیا در همسایگی سلول برنده قرار دارند یا خیر، می باشد. ما نیز در اینجا روش دوم را انتخاب نموده ایم و حین آموزش شبکه عصبی خودسازمانده کوهنن، ضرایب تمامی سلول های شبکه را در فرآیند اجرای الگوریتم تغییر می دهیم و کار تصحیح را به سلول های برنده و همسایه های آنها محدود نکرده ایم.

در این قسمت گام های الگوریتم تصحیح شده برای شبکه عصبی کوهنن را را که پیشتر در بخش ۶-۲ به تفصیل شرح داده شد را آورده ایم. تمامی مثال های شبیه سازی شده این فصل با استفاده از این الگوریتم تصحیح شده اجرا شده اند. همانطور که پیشتر نیز آمد، در این الگوریتم برای پیشگیری از تمرکز اولیه ضرایب وزنی شبکه عصبی کوهنن از الگوریتم مونت کارلو برای تولید یک رشته تصادفی از اعداد بهره جستیم و مقادیر این رشته تصادفی را به عنوان ماتریس شرایط اولیه ضرایب وزنی شبکه عصبی خودسازمانده کوهنن به کار برده‌ایم.

در گام شناسایی سلول مرده و تعیین سلول جدید با ضریب نزدیک به ضریب سلول های برنده نیز از الگوریتم فیلتر ذره ای استفاده شده است. در این مرحله با استفاده از الگوریتم فیلتر ذره ای همانطور که در بخش قبل گفته شد، ضرایب تمامی سلول ها را تغییر می دهیم. هم سلول های برنده و هم سلول های بازنده تا به این روش هم از مسائل مربوط نحوه تعیین همسایگی و فاصله وهم از مشکلات ناشی از روش همسایگی اجتناب کنیم. می دانید که در روش همسایگی به دلیل بروز شدید مشکل سلول مرده، این امکان وجود دارد که فضای تخمین خروجی از شبکه در قسمتهای مرکزی بی هیچ تغییری و بسیار فشرده تا پایان اجرای الگوریتم باقی بماند و باعث تولید تخمین نادرست و یا تخمینی با دقت بسیار پایین از فضای ورودی اعمالی به شبکه عصبی شود. در ادامه گام های الگوریتم بیان می‌شود.

الگوریتم:

گام یکم- تعیین مقادیراولیه تصادفی ماتریس وزنی شبکه عصبی با استفاده از الگوریتم مونت کارلو؛

گام دوم- انتخاب تصادفی یک نمونه از بردارهای ورودی و اعمال آن به شبکه؛

گام سوم- یافتن سلول برنده

گام چهارم- تغییر بردار سلول برنده به طوری که به سوی بردار ورودی حرکت نماید؛

گام پنجم- تغییر بردار سلول های دیگر؛

گام ششم- تکرار گام یکم تا چهارم به تعداد تصادفی؛

گام هفتم- محاسبه مقدار آستانه ای و تعیین سلول تنبل ویا مرده؛

گام هشتم- حذف سلول بازنده و جایگذاری سلولی جدید با وزن تعیین شده بر حسب مقدار آستانه؛

گام نهم- بازگشت به گام یکم؛

در پایان این فصل نتایج شبیه سازی کامپیوتری حاصله از پیاده سازی دو الگوریتمی که در این پایان نامه به آن پرداخته شد، آورده ایم. مثال های ۱و۲ مربوط است به الگوریتم آموزش شبکه عصبی با استفاده از فیلتر ذره ای و پیاده سازی آن روی یک شبکه پرسپترون چند لایه، و دو مثال انتهایی نیز برای نمایش نتایج و عملکرد الگوریتم آموزش شبکه عصبی کوهنن با استفاده از فیلتر ذره ای ارائه شده است.

۶-۳- شبیه سازی کامپیوتری

در آخرین بخش پایان نامه ای که پیش رو دارید سعی داریم تا با ارائه چند مثال شبیه سازی به بررسی عملکرد الگوریتم کوهنن تلفیقی در مواجهه با مسائل گوناگون مهندسی بپردازیم و در کنار آن رفتار شبکه پیشنهادی در این پایان نامه را با رفتار دیگر شبکه های عصبی معمول مقایسه کنیم تا در انتهای فصل نیز بتوانیم یک تحلیل کامل و مشروح از نحوه پاسخ دهی شبکه عصبی پیشنهادی ارائه نماییم.

در مثال اول یک شبکه عصبی پرسپترون چندلایه را با الگوریتم فیلتر ذره ای آموزش داده ایم و از این شبکه برای حل یکی از مسائل اساسی و مشکل مهندسی مخابرات که همان شناسایی سیگنال اصلی در حضور نویز می باشد، استفاده خواهیم نمود. سپس در مثال دوم نحوه عملکرد یک شبکه عصبی کوهنن؛ که با الگوریتم پیشنهادی در پایان نامه آموزش داده شده است؛ را در برخورد با یک مساله تخمین حالت غیرخطی خواهیم دید و در عین حال پاسخ های حاصله را با رفتار یک شبکه پرسپترون عادی مقایسه خواهیم نمود. در مثال پایانی نیز یکی از مثال های رایج در شبکه های عصبی رقابتی و خودسازمانده نظیر شبکه عصبی کوهنن را با استفاده از الگوریتم ارائه شده در پایان نامه تحلیل کرده ایم. در واقع مساله تخمین توزیع احتمال فضای ورودی به یک شبکه عصبی، مساله ایست که مطمئناً دید مناسبی از نحوه عملکرد شبکه عصبی کوهنن با الگوریتم تلفیقی، ارائه خواهد داد. البته برای سادگی و سهولت در مقایسه، تمامی تخمین ها برای هر دو حالت شبکه عصبی کوهنن یعنی الگوریتم یادگیری کلاسیک و الگوریتم یادگیری تلفیقی؛ که در این فصل پیشنهاد شده است؛ آورده شده و شبیه سازی توسط نرم افزار MATLAB صورت گرفته است.

و در نهایت به بررسی نقاط تمایز الگوریتم کوهنن تلفیقی و تمامی مسائل و مشکلات پیش روی آن خواهیم پرداخت. آنچه در ادامه می آید می تواند برای شروع و تعریف پایان نامه های دیگری از این دست برای رفع نقایص الگوریتم پیشنهادی در اینجا مطمئناً مفید فایده خواهد بود.

مثال۱: آشکارسازی سیگنال[۴۷]

مسئله آشکارسازی مولفه های یک سیگنال دریافتی در حضور نویز یکی از معضلات اصلی مهندسی مخابرات است که ذهن بسیاری از محققین این رشته را به خود مشغول نموده است. مساله ای که در اینجا ارائه شده است درواقع یک مثال ساده از آشکارسازی مولفه های گوسی یک سیگنال آمیخته با نویز است. داده ها توسط سیستم تک متغیره آمیخته به نویز با معادله زیر تولید شده اند:

$$y = x + 2e^{-16x^2} + 2e^{-16(x-0.7)^2} + n, \qquad (۱-۶)$$

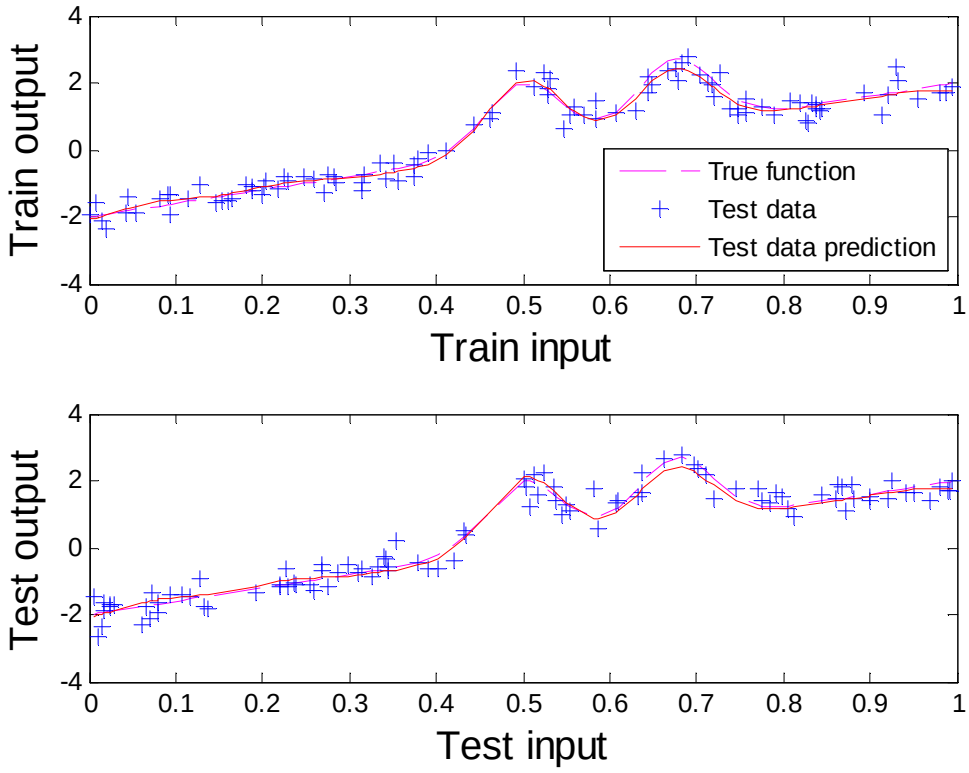
که در آن :

$$n \sim \mathcal{N}(0, \sigma^2). \qquad (۲-۶)$$

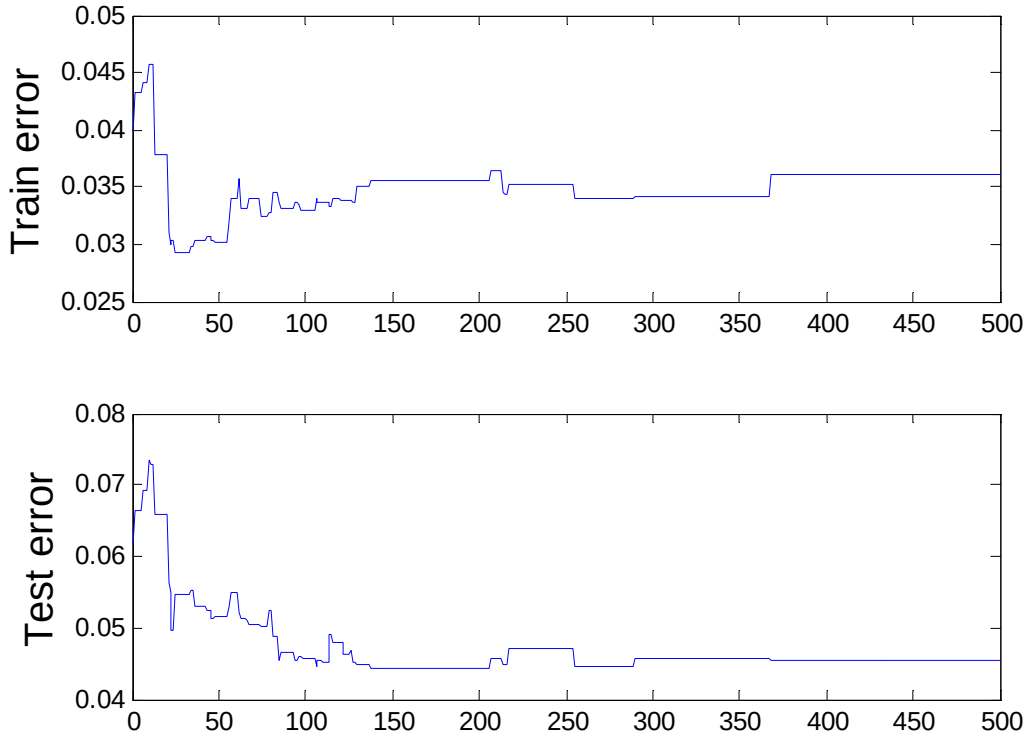
همانطور که پیشتر هم توضیح داده شد، در این مسئله از یک شبکه پرسپترون چندلایه در شبیه سازی استفاده شده است. قانون یادگیری الگوریتم SIR فیلتر ذره ای است که در فصل پنجم به طور مبسوط شرح داده شد. البته در اینجا هدف یافتن مولفه های گوسی در یک سیگنال نویزی مدنظر بوده است. الگوریتم به میزان ۱۰۰ مرتبه تکرار شده و نتایج شبیه سازی در شکل های (۶-۱) و (۶-۲) آمده است. در شکل (۶-۱)، مقادیر داده های اصلی برای آموزش شبکه و همچنین خروجی تخمینی شبکه ترسیم شده است. شکل (۶-۲) نیز خطای تخمین را در مرحله آموزش و تست شبکه نمایش می دهد. همانطور که در شکل های حاصل از نتایج شبیه سازی مشهود است الگوریتم یادگیری فیلتر ذره ای توانسته است تا تخمین بسیار مناسبی از مولفه های گوسی در حضور نویز ارائه دهد.

در مثال دوم توانایی الگوریتم پیشنهادی را برای شبکه کوهنن خواهیم آزمود. نتایج مثال بعد می تواند قدرت الگوریتم پیشنهادی کوهنن تلفیقی را بیشتر نمایان سازد.

[۴۷] برای توضیحات بیشتر در مورد معادله استفاده شده در این مثال رجوع کنید به مرجع ۲۱.



شکل (۶-۱): داده های استفاده شده برای آموزش شبکه عصبی و خروجی شبکه در مثال ۱



شکل (۶ -۲): نمودار خطای تخمین در مثال ۱ برای یافتن مؤلفه های گوسی سیگنال در حضور نویز

مثال ۲: تخمین حالت های یک سیستم غیرخطی متغیر با زمان

در این بخش قصد داریم تا عملکرد شبکه عصبی کوهنن باناظر را با عملکرد شبکه عصبی پرسپترون چندلایه در برخورد با یک مسئله ی واحد، مقایسه نماییم. در واقع در این قسمت از پایان نامه ابتدا ساختار شبکه عصبی کوهنن باناظر را بررسی می کنیم و سپس با استفاده از الگوریتم تلفیقی ارائه شده در بخش (۶-۳) شبکه را آموزش می دهیم. در پایان، رفتار این شبکه در تخمین حالت های یک سیستم غیرخطی متغیر با زمان در حضور نویز را با عملکرد یک شبکه پرسپترون معمولی مقایسه می کنیم. لازم به ذکر است در مقالات و پایان نامه های بسیاری عملکرد الگوریتم های مختلف فیلتر ذره ای در برخورد با مسائل مهندسی تجزیه و تحلیل شده و در اینجا ما فقط کار مقایسه را برای الگوریتم کوهنن تلفیقی انجام داده ایم. البته همانطور که گفتیم قصد داریم ابتدا شبکه عصبی رقابتی را به یک شکه عصبی رقابتی باناظر تبدیل کنیم سپس رفتار شبکه جدید را که با استفاده از الگوریتم یادگیری پیشنهادی در این پایان نامه آموزش داده شده است را با رفتار یک شبکه عصبی پرسپترون چندلایه مقایسه کنیم.

پس از تشریح شبکه کوهنن با ناظر و الگوریتم یادگیری آن، اکنون نوبت بررسی مثال دوم فرارسیده است. در اینجا همانطور که پیشتر گفتیم، قصد داریم عملکرد شبکه عصبی پرسپترون چند لایه را با عملکرد شبکه عصبی کوهنن با ناظری که توسط الگوریتم تلفیقی آموزش داده شده است، مقایسه نماییم. در واقع هدف تخمین حالت های یک سیستم غیرخطی گسسته متغیر با زمان با استفاده از هر دوی این شبکه ها می باشد و سپس رفتار پاسخ ها را بررسی وتجزیه و تحلیل می-نماییم.

روابط سیستم این مثال، اولین بار برای سنجش توانایی الگوریتم های عددی مختلف از جمله تخمین مونت کارلو، در تخمین حالت های یک سیستم غیرخطی، توسط کیتاگاوا[۴۸] در سال ۱۹۹۶ پیشنهاد شده است و همکنون به عنوان یکی از سیستم های مرجع، در جهت اثبات کارایی الگوریتم های جدید و ایده های بکر تئوری تخمین فیلترهای غیرخطی کاملاً شناخته شده می باشد. در ادامه روابط و نتایج شبیه سازی آورده شده است.

[48] Kitagava

معادلات سیستم توسط روابط زیر بیان می شوند:

$$x_{t+1} = \frac{x_t}{2} + \frac{25x_t}{1+x_t^2} + 8\,cos(1.2t) + w_t, \quad (۱۱-۶)$$

$$y_t = \frac{x_t^2}{20} + e_t. \quad (۱۲-۶)$$

که در آن $x_0 \sim \mathcal{N}(0,5)$ و w_t و e_t نیز نویز سفید گوسی مستقل از هم هستند و برای آنها داریم:

$$w_t \sim \mathcal{N}(0,10), \quad (۱۳-۶)$$

$$e_t \sim \mathcal{N}(0,1). \quad (۱۴-۶)$$

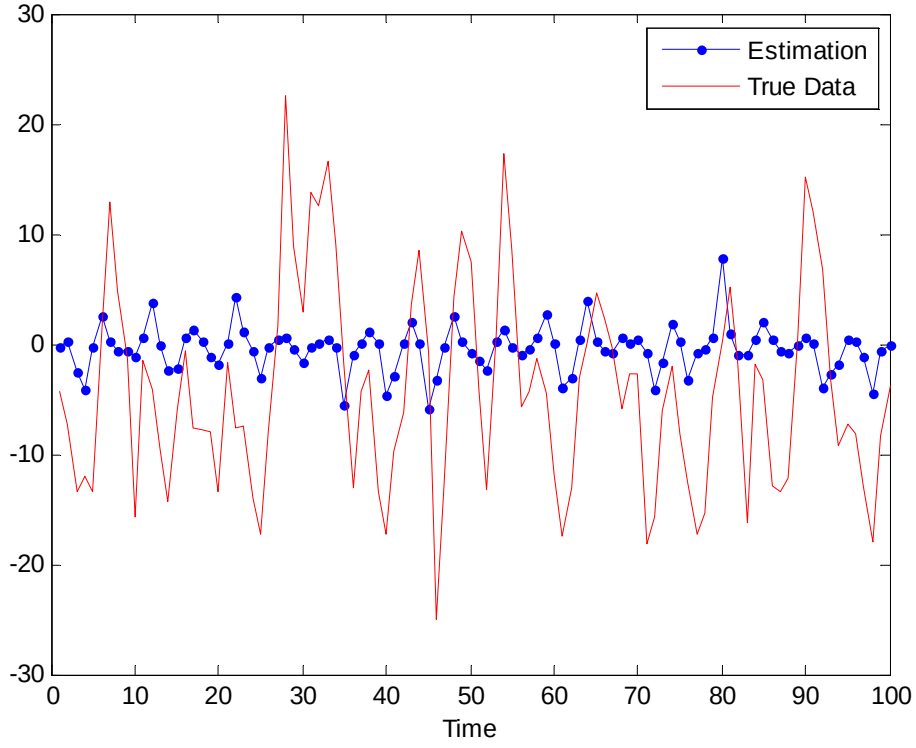
نتایج شبیه سازی در شکل های (۶-۳)، (۶-۴)، (۶-۵) و (۶-۶) رسم شده است. شکل (۶-۳)، داده های اصلی تولید شده توسط سیستم غیرخطی به همراه تخمین آنها را برای شبکه عصبی پرسپترون چندلایه با همان قانون یادگیری معمول اینگونه شبکه ها، نشان می دهد و شکل (۶-۴) هم، پاسخ نهایی تخمین حالات سیستم را با استفاده از شبکه عصبی کوهنن با ناظر را نمایان ساخته است. باز هم تاکید می کنیم که الگوریتم یادگیری شبکه عصبی کوهنن با ناظر همان الگوریتم تلفیقی فیلتر ذره ای است که در این پایان نامه پیشنهاد شده است و از این پس آن را با عنوان الگوریتم کوهنن با ناظر تلفیقی می خوانیم. همانطور که می بینید درصد پیروزی الگوریتم کوهنن با ناظر تلفیقی در تخمین حالات سیستم به مراتب بیشتر از پاسخ الگوریتم شبکه عصبی پرسپترون چندلایه است. لازم به ذکر است که الگوریتم شبکه عصبی پرسپترون با پنج نرون اجرا شده و هر دو الگوریتم را ۱۰۰ مرتبه تکرار نموده ایم و نتایج شبیه سازی بدست آمده است. در شکل های (۶-۵) و (۶-۶) هم نمودار میزان خطای تخمین هر دو الگوریتم ارائه شده است. این نمودار ها نیز نشانگر دقت بالاتر الگوریتم کوهنن با ناظر تلفیقی است. بحثی که ممکن است در اینجا وجود داشته باشد مسئله زمان اجرای الگوریتم و انتخاب تعداد ذرات برای اجرای الگوریتم کوهنن با ناظر تلفیقی است. در مورد این مثال زمان اجرای الگوریتم کوهنن با ناظر تلفیقی ۵۵ ثانیه و زمان اجرای الگوریتم پرسپترون چندلایه ۳۸ ثانیه بوده است. همچنین تعداد ذرات با اهمیت در الگوریتم کوهنن با ناظر تلفیقی، نیز ۱۰۰۰ ذره در نظر گرفته شده است.

باید تاکید کنیم که اصولاً در الگوریتم های ارائه شده برای فیلتر ذره ای تا حدودی با توجه به مسئله ای که با آن درگیر هستیم به بهای بالا بردن دقت تخمین و بهبود پاسخ، زمان را از دست می

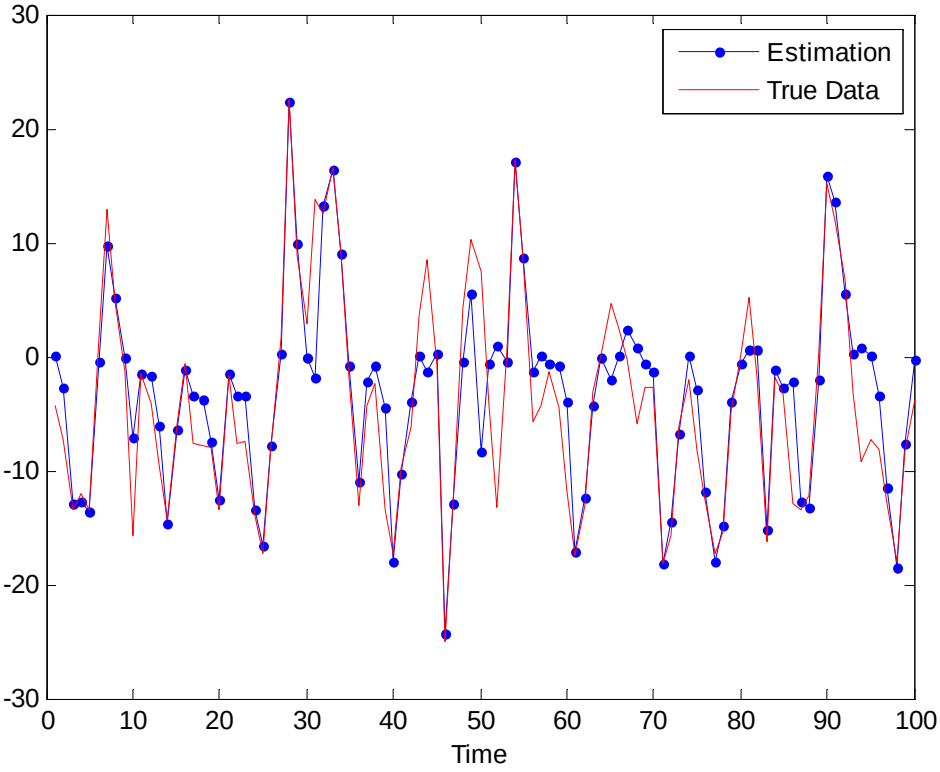
دهیم. که این مسئله رابطه ی مستقیمی با نحوه انتخاب الگوریتم، تعداد دفعات تکرار آن، تعداد ذرات با اهمیت و نهایتاً وجود یا عدم وجود گام نمونه برداری مجدد، دارد. البته در سال های اخیر توجه بسیاری از محققین علوم کامپیوتر برای اصلاح الگوریتم های فیلتر ذره ای، جلب شده است و کارهای قابل تاملی در این زمینه انجام شده و مقالات زیادی به چاپ رسیده و یا در دست چاپ می باشد. و تاحدودی توانسته اند زمان اجرای الگوریتم ها را البته با اعمال شرایطی خاص، کاهش داده و در واقع عملکرد زمانی الگوریتم را بهبود بخشیده و باب جدیدی برای استفاده از چنین الگوریتم های نیرومندی در پردازش های بلا درنگ[۴۹]، مفتوح گردد.

در ادامه سومین مثال برای بررسی عملکرد الگوریتم کوهنن با ناظر تلفیقی ارائه شده است. پس از آن در بخش پایانی به ارائه برخی نتایج و پیشنهادات برای ادامه کارهای بعدی پرداخته شده است.

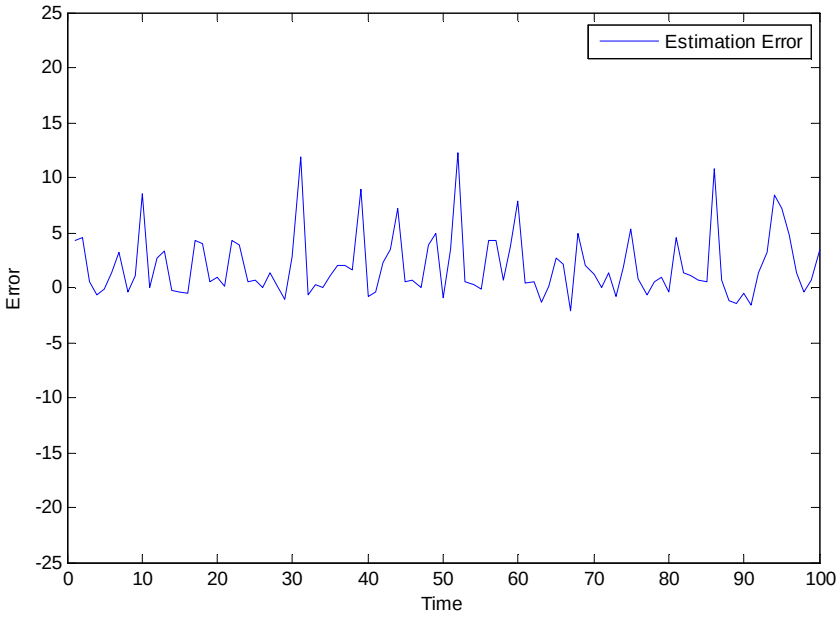
[49] Real Time



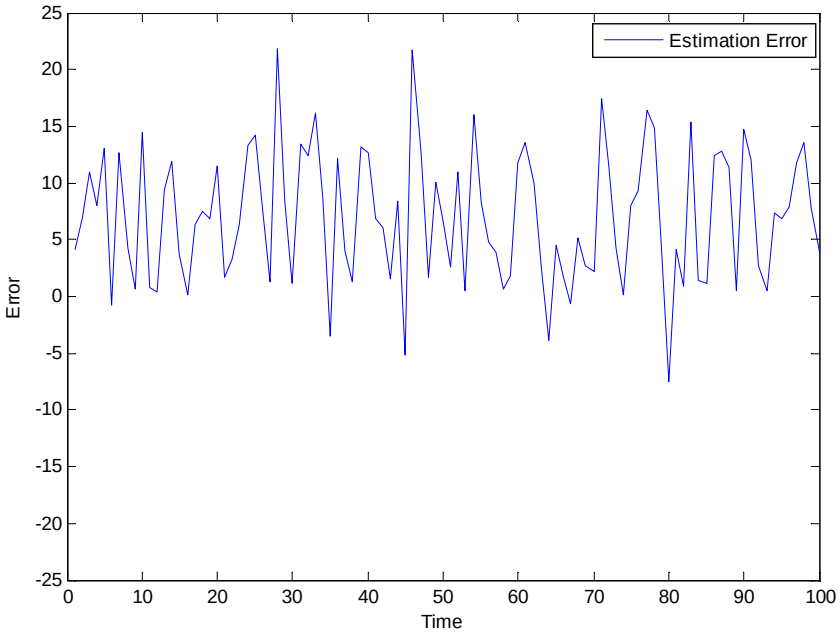
شکل (۶-۳): نتایج شبیه سازی مثال ۲ برای الگوریتم پرسپترون چندلایه



شکل (۶-۴): نتایج شبیه سازی مثال ۲ برای الگوریتم کوهنن با ناظر تلفیقی



شکل (۶-۵): نمودار خطای تخمین برای الگوریتم پرسپترون چندلایه در مثال ۲



شکل (۶-۶): نمودار خطای تخمین برای الگوریتم کوهنن با ناظر تلفیقی در مثال ۲

مثال۳: تخمین توزیع های مختلف

در مثال پایانی نیز کارایی الگوریتم کوهنن با ناظر تلفیقی را در تخمین توزیع های گوناگون مانند حلقه، مکعب و مربع نمایش داده شده است. در اجرای الگوریتم سعی شده است از مراحل اولیه اجرا تا مراحل پایانی تصویری گذاشته شود تا روند پیشروی الگوریتم یادگیری شبکه عصبی کوهنن با ناظر تلفیقی در تخمین فضای ورودی، برای خواننده قابل درک و ملموس باشد.

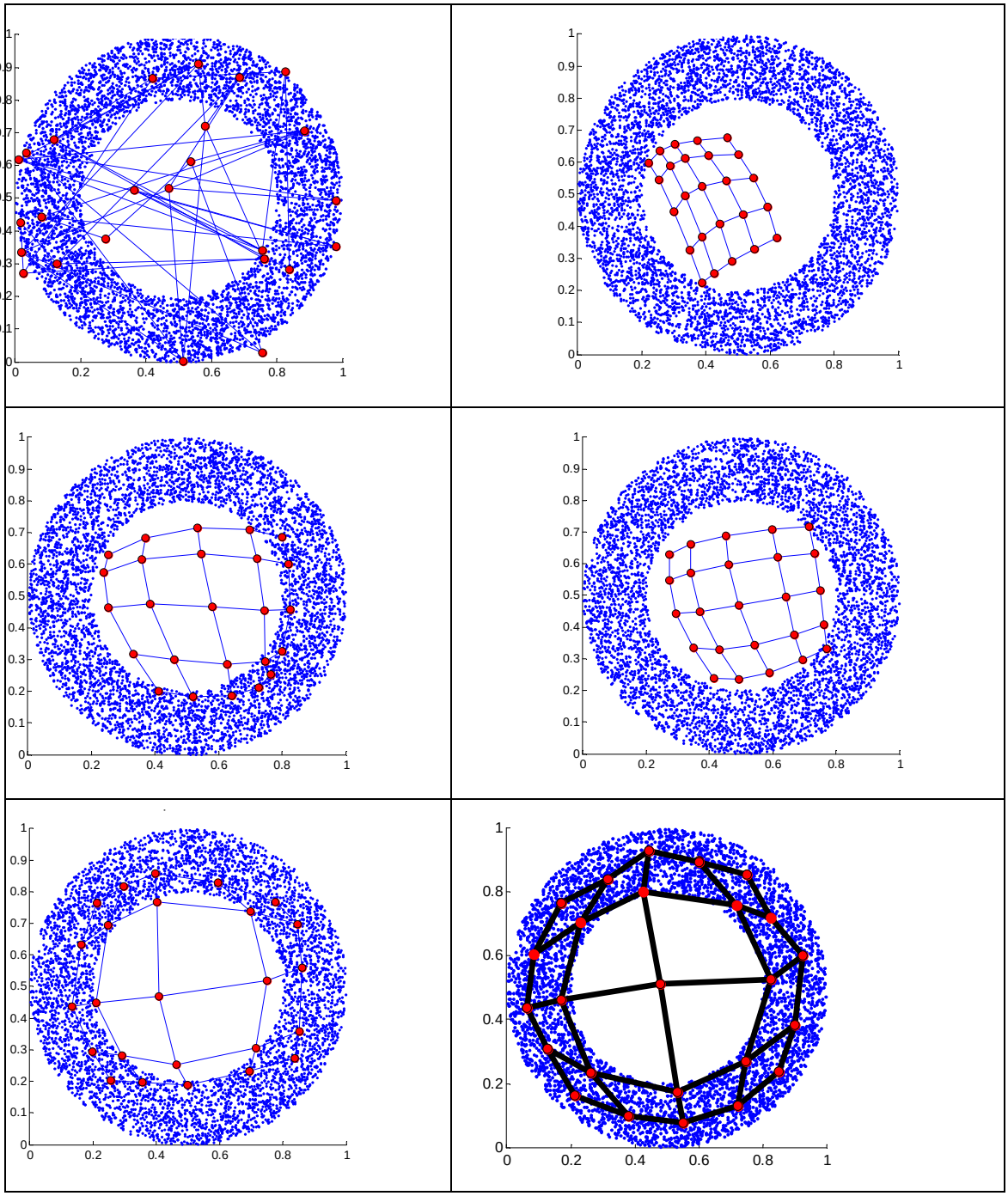
در این مثال تمامی توزیع ها را با استفاده از الگوریتم کوهنن با ناظر تلفیقی تخمین زده ایم و در واقع هدف، اثبات کارایی الگوریتم پیشنهادی پایان نامه در تخمین موفق یک فضای ورودی پیچیده و آن هم با یک زمان قابل قبول، بوده است. پس از آن سعی کرده ایم پاسخ های بدست آمده را با پاسخ تولید شده توسط یک شبکه عصبی کوهنن با قانون یادگیری کلاسیک، مقایسه کنیم تا کارایی الگوریتم کوهنن با ناظر تلفیقی بیشتر مورد بررسی و آزمون قرار گیرد. توجه شود که در هر سه مورد گام های مختلف تخمین ترسیم شده تا نحوه پیشرفت الگوریتم کاملاً نمایان شود. توزیع ابتدایی ضرایب وزنی سلول های شبکه عصبی توسط گام یکم الگوریتم اصلاحی یا همان الگوریتم مونت کارلو تعیین شده و سپس در ادامه روند اجرای الگوریتم در هر گام با تشخیص سلول های مرده و حذف آنها به بهبود پاسخ نهایی الگوریتم کمک می کند. در صفحات بعدی پاسخ های شبیه سازی شده الگوریتم اصلاحی ترسیم شده است. در انتها نیز بررسی و تجزیه و تحلیل اجمالی الگوریتم پرداخته ایم و برخی از معایب و مزایای آن را برشمرده ایم.

در شکل (۶-۷) مراحل تخمین فضای ورودی که به صورت یک حلقه می باشد را برای اجرای الگوریتم با تکرار ۳۰۰۰ نمایش داده شده است و بلافاصله نتایج اجرای الگوریتم کوهنن کلاسیک نیز در شکل (۶-۸) آمده است. شروع اجرای هر دو الگوریتم با ۲۵ نرون بوده است.

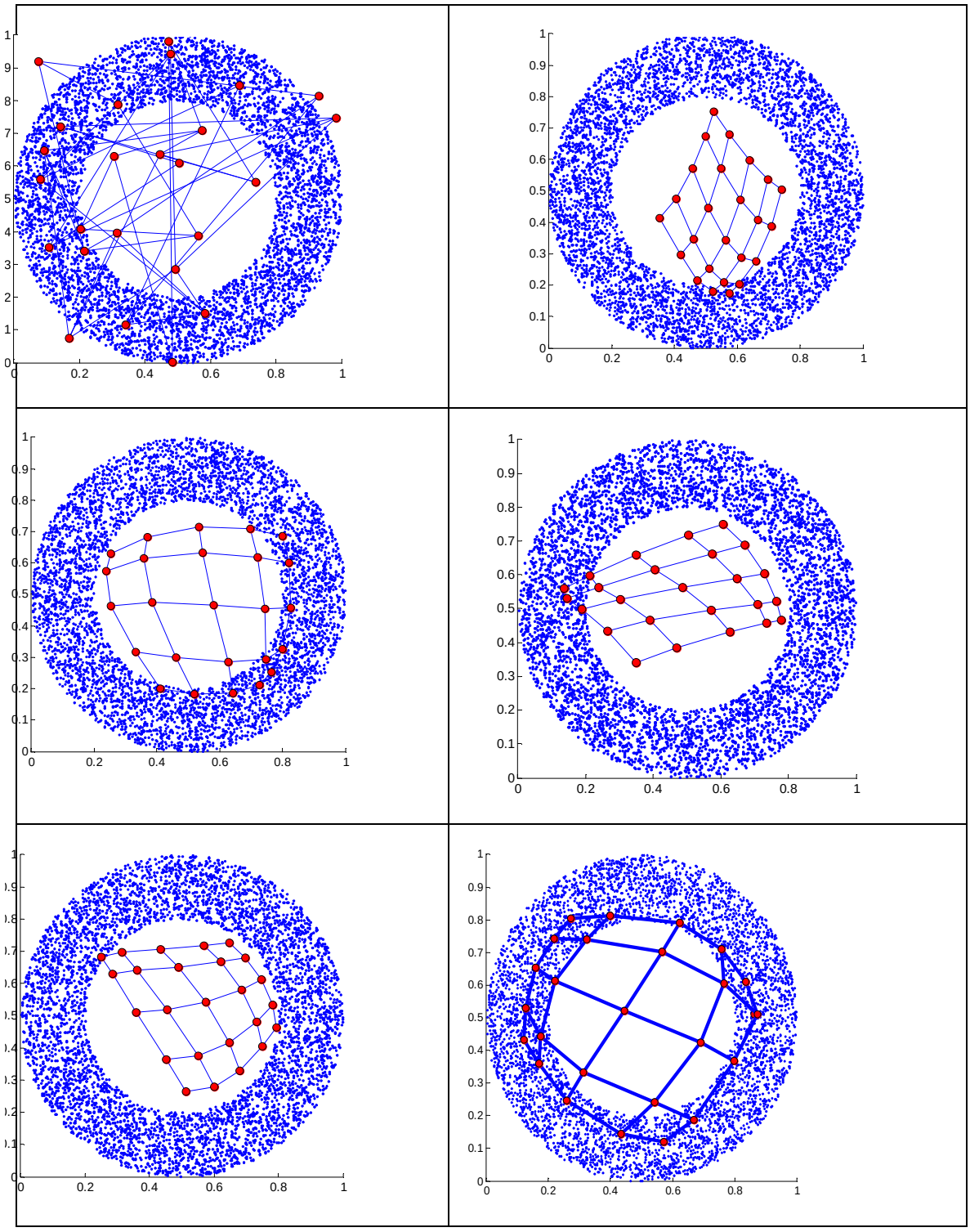
شکل (۶-۹)، مراحل اجرای الگوریتم برای شبیه سازی یک مکعب -که در واقع همان فضای ورودی اعمالی به شبکه کوهنن است- برای تکرار ۴۰۰۰ نشان داده شده است و همانند تخمین حلقه، نتایج شبیه سازی شده از اجرای الگوریتم کوهنن کلاسیک نیز آورده شده است. تعداد نرون های اولیه برای شروع اجرای هر دو الگوریتم با ۵۰ نرون بوده است.

و نهایتاً، نتایج حاصله از شبیه سازی کامپیوتری الگوریتم کوهنن با ناظر تلفیقی برای تخمین فضای ورودی مربع و همینطور نتایج بدست آمده از اجرای الگوریتم کوهنن کلاسیک برای تخمین همین فضای ورودی به ترتیب در شکل های (۶-۱۰) و (۶-۱۱) آمده است. نتایج هر دو الگوریتم

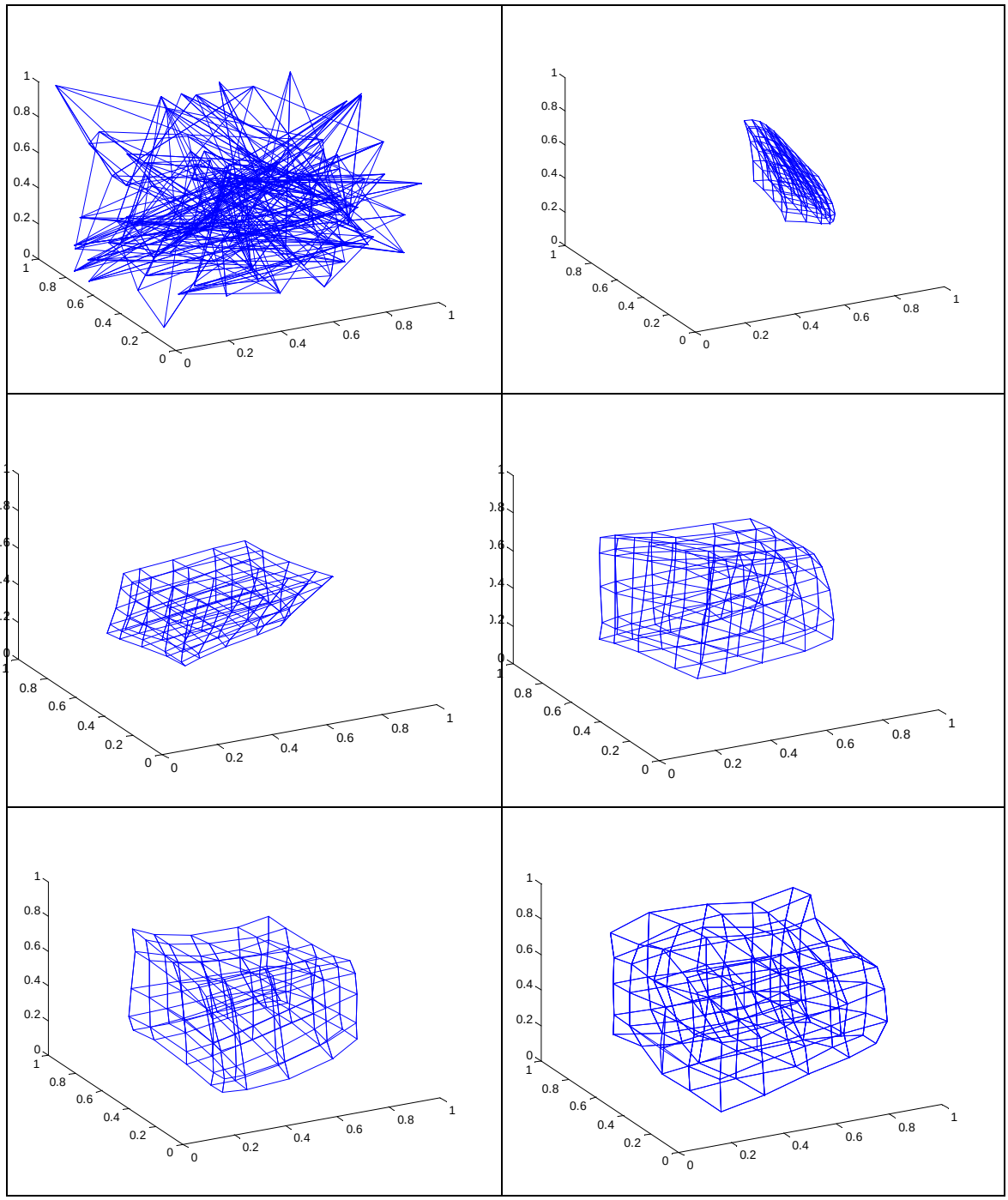
برای ۲۰۰۰ تکرار ترسیم شده است و تعداد نرون های اولیه برای شروع اجرای هر دو الگوریتم با ۲۵ نرون بوده است. بیان این نکته حائز اهمیت می باشد که مقایسه تنها به این دلیل انجام شده است که توانایی الگوریتم کوهنن با ناظر تلفیقی در اجرا و ارائه تخمین و پاسخ مناسب و همچنین کارایی آن راحتر قابل بررسی باشد.



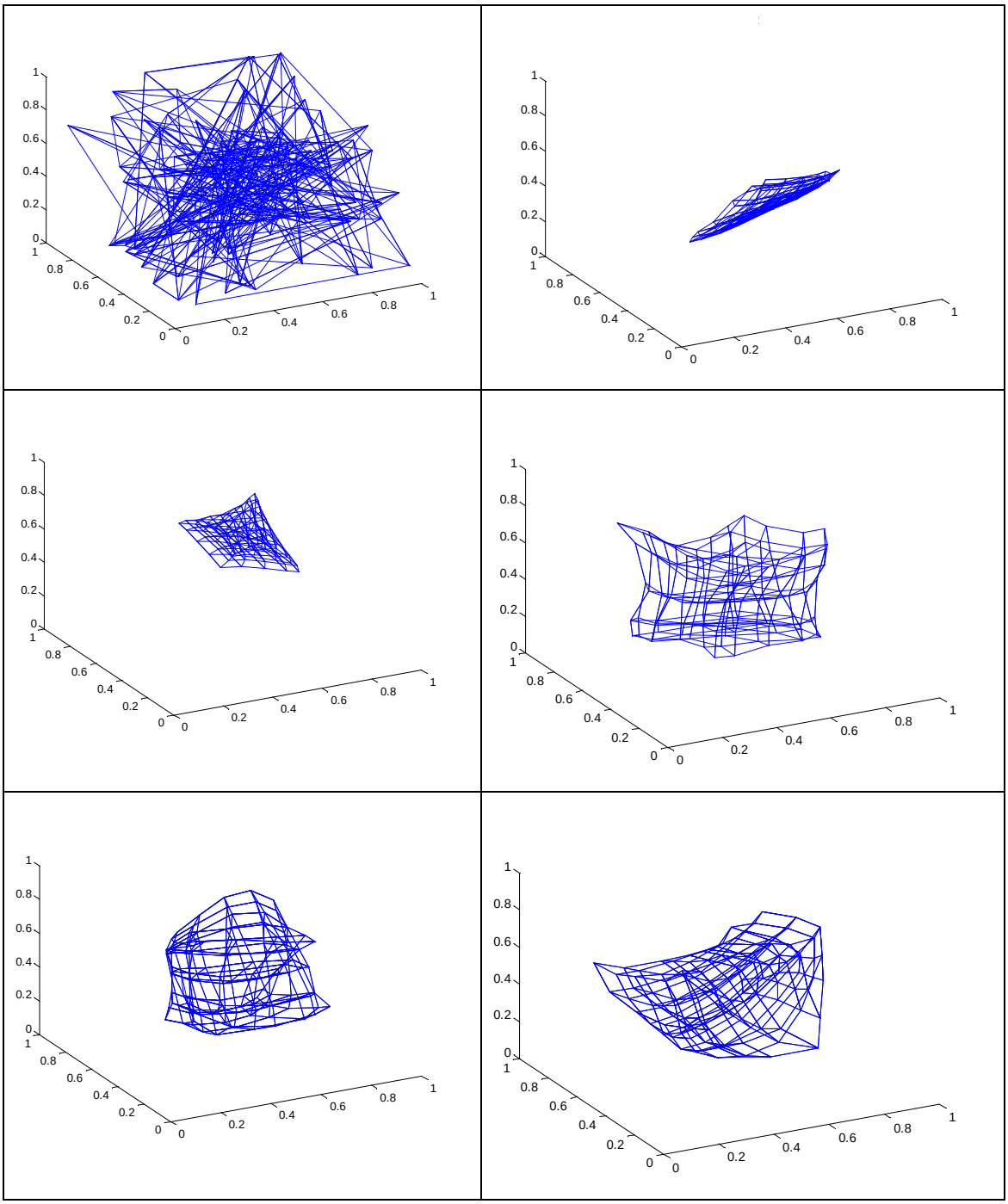
شکل (۶-۷): نتایج شبیه سازی برای تخمین یک حلقه با استفاده از الگوریتم کوهنن با ناظر تلفیقی، مثال ۳



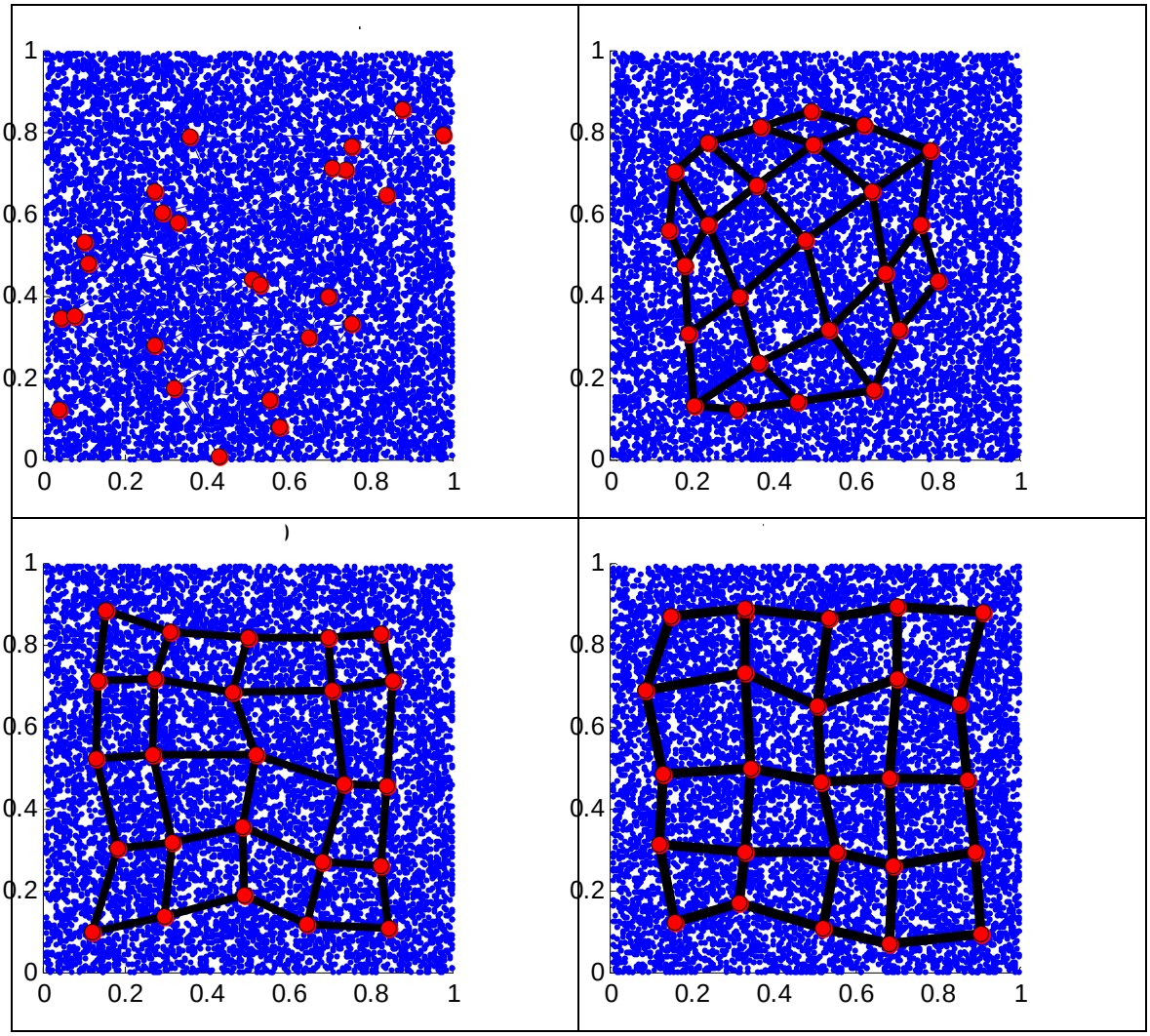
شکل (۶-۸): نتایج شبیه سازی برای تخمین یک حلقه با استفاده از الگوریتم کوهنن کلاسیک، مثال ۳



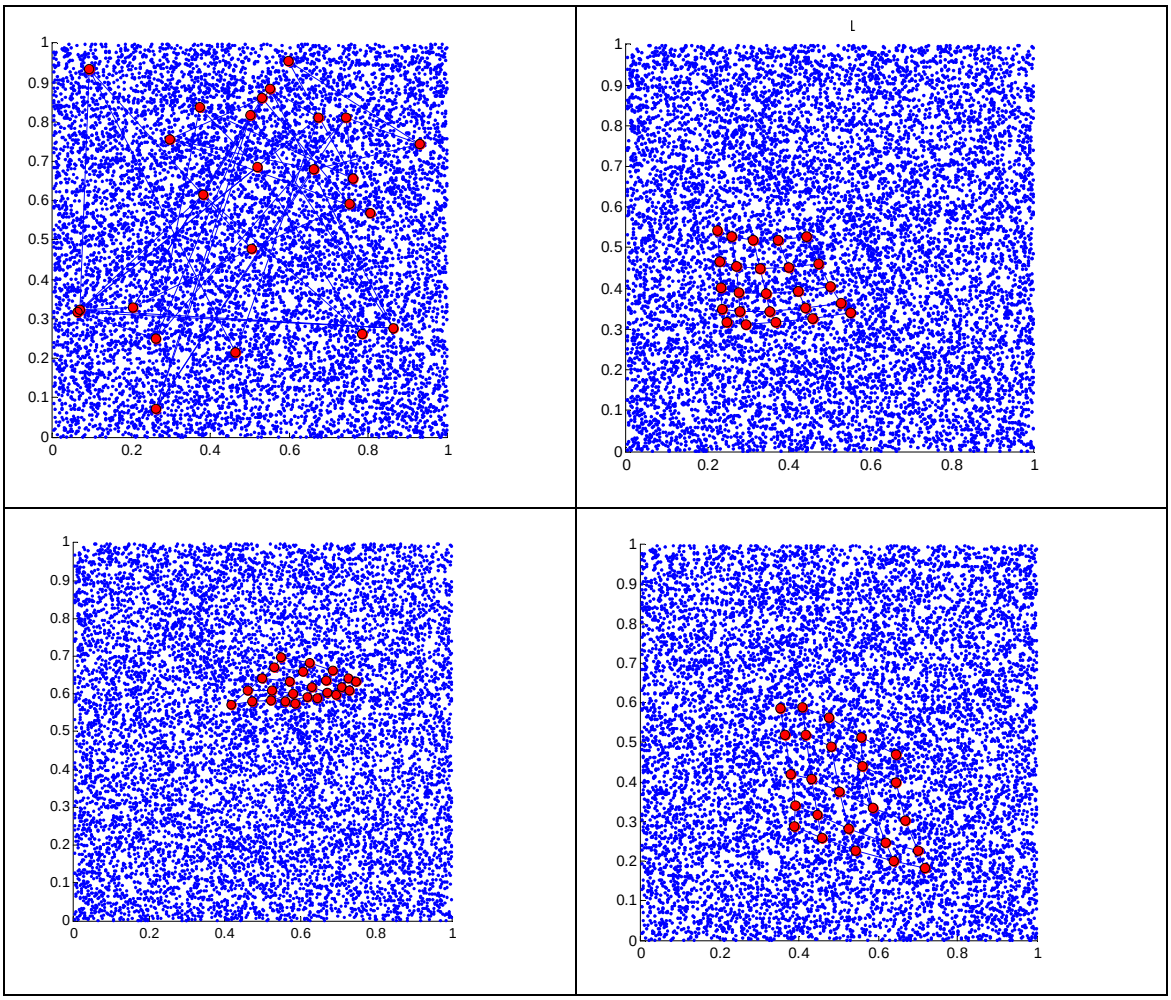
شکل (۶-۹): نتایج شبیه سازی برای تخمین یک مکعب توسط الگوریتم کوهنن با ناظر تلفیقی، مثال ۳



شکل (۶-۱۰): نتایج شبیه سازی برای تخمین یک مکعب توسط الگوریتم کوهنن کلاسیک، مثال ۳



شکل (۶-۱۱): نتایج شبیه سازی برای تخمین یک مربع توسط الگوریتم کوهنن با ناظر تلفیقی، مثال ۳



شکل (۶-۱۲): نتایج شبیه سازی برای تخمین یک مربع توسط الگوریتم کوهنن کلاسیک، مثال ۳

با توجه به نتایج شبیه سازی کارایی الگوریتم توسط الگوریتم کوهنن با ناظر تلفیقی به خوبی روشن است. این نتایج به وضوح کارآمدی الگوریتم پیشنهادی را هم در رفع مشکل سلول مرده و هم در اجرای الگوریتم به صورت درست برای حصول نتایج مورد انتظار در تخمین فضای ورودی اعمالی به شبکه عصبی خودسازمانده کوهنن، نشان می دهد. با توجه به استفاده از الگوریتم فیلتر ذره ای برای تنظیم ضرایب وزنی شبکه و تشخیص درست سلول های مرده، دقت تخمین توسط شبکه نسبت به الگوریتم کلاسیک بالاتر است. همچنین استفاده از الگوریتم مونت کارلو برای ایجاد یک رشته تصادفی که به عنوان شرایط اولیه ماتریس ضرایب شبکه مورد استفاده قرار گرفته، کمک شایانی به تمرکز زدایی در ابتدای کار نموده که خود این مساله باعث تولید تخمین بهتری از فضای ورودی اعمالی به شبکه شده است. در واقع یکی از مشکلات اساسی الگوریتم کوهنن کلاسیک همین مسئله تمرکز است که باعث شده، همانطور که از شکلها و نمودارها معلوم است، که الگوریتم در پایان کار تخمین مناسب و درستی از فضای ورودی اعمالی به شبکه را ارائه نمی نماید. بررسی نتایج هر دو الگوریتم نشان می دهد غیر از مسئله زمان اجرا هیچ مشکلی برای اجرای الگوریتم توسط الگوریتم کوهنن با ناظر تلفیقی متصور نیست. البته در این مثال ها عملکرد زمانی الگوریتم پیشنهادی توسط الگوریتم کوهنن با ناظر تلفیقی قابل قبول بود. اما همانطور که قبلاً هم گفته شد، این مسئله بهبود عملکرد زمانی الگوریتم می تواند موضوع ادامه کار حاضر در یک پایان نامه دیگر باشد.

در تمامی تخمین ها در گام های مشابه نمودار تخمین آن لحظه رسم شده است. اولین نمودار در هر تخمین لحظه شروع الگوریتم را نشان می دهد و نمودار پایانی که در آخرین ردیف شکل ها سمت راست آورده شده است نمودار لحظه پایان اجرای الگوریتم می باشد. در واقع پاسخ نهایی شبکه عصبی برای تخمین فضای ورودی اعمالی به شبکه عصبی می باشد. تمامی نمودارها به ترتیب اعداد ریاضی یعنی از چپ به راست پیشرفت الگوریتم را در گام های متفاوت حین اجرای آن نشان می دهد. برای تخمین اول گام های انتخابی در رسم نتایج، هر پانصد تکرار یکبار بوده است. برای تخمین دوم که تخمین مکعب بوده گامها به ترتیب، گام های صفرم (شروع)، پانصدم، هزارم، یکهزار و پانصدم، دو هزارم، سه هزارم و نهایتاً آخرین مرحله تکرار الگوریتم می باشد. و در مورد آخرین تخمین یعنی تخمین توزیع مربعی، گامها به ترتیب، گام های صفرم (شروع)، دویست و پنجاهم، پانصدم، هزارم، یکهزار و پانصدم و نهایتاً آخرین مرحله تکرار الگوریتم یعنی دو هزارمین گام، می باشد.

نتیجه گیری و پیشنهاد برای کارهای آینده

در این پایان نامه همانطور که ملاحظه نمودید، الگوریتم جدیدی برای تخمین ضرایب شبکه عصبی خودسازمانده کوهنن معرفی شد که در تخمین و بازسازی چگالی احتمال نامشخص فضای ورودی اعمالی به شبکه موفق تر از الگوریتم کلاسیک عمل می نماید و نتایج شبیه سازی فصل ششم نیز پیاده سازی الگوریتم را برای مثال های کاربردی معمول نمایش می دهد. دلیل برتری الگوریتم تصحیح شده بر روش کلاسیک اینست که علاوه بر اینکه مشکل سلول مرده را به کلی برطرف می کند، تخمینی با دقت بالاتر ارائه می نماید که ناشی از به کارگیری روش توانمند مونت کارلو در تعیین ضرایب شبکه می باشد که به تازگی در بسیاری از کاربردهای مهندسی شبکه های عصبی مورد استفاده قرار گرفته است. تنها مشکلی که می توان به این گونه الگوریتم ها نسبت داد زمان اجرای برنامه و الگوریتم است که تا حدی به نسبت الگوریتم کلاسیک بیشتر است. در واقع در زمینه عملکرد شبکه بهبود فوق العاده ایجاد شده اما زمان رسیدن به این پاسخ بهینه و تخمین بهتر، بیشتر از حالت اجرای الگوریتم معمول پیشین می باشد. البته توجه دارید که در هر پروژه‌ی مهندسی چنین مصالحه هایی به وفور یافت می شود و این مساله با توجه به کاربدهای فراوان الگوریتم ارائه شده، مسلماً نشانه ضعف عملکرد الگوریتم پیشنهادی نمی باشد.

به عنوان ادامه کار فعلی می توان با تعریف یک پایان نامه دیگر و استفاده از الگوریتم های دیگر هم خانواده با الگوریتم فیلتر ذرهای سرعت اجرای برنامه را بهبود بخشید. همچنین می توان با تحقیق روی مسائل گوناگونی که به عنوان مثال های پایه ای[50] در مهندسی کنترل و یا پردازش تصویر شناخته شده اند توانایی های الگوریتم جدید آزموده شود. در حال حاضر در تمامی مجتمع علمی به روز دنیا کار بر روی الگوریتم فیلتر ذره ای ادامه دارد و همچنان الگوریتم های قوی تر و سریعتری معرفی می شوند. در مورد شبکه های عصبی خودسازمانده نیز کاربردهای جدیدی در زمینه پردازش تصویر و صوت، طراحی و کاهش نویز در بکه های بی سیم، بهینه سازی عملکرد شبکه های فازی و عصبی، بهینه سازی طراحی شبکه های قدرت، روباتیک، نانو تکنولوژی و بسیاری دیگر از علوم مهندسی برای این الگوریتم در حال ارائه می باشد. در تمامی این زمینه ها می توان با تعریف یک پایان نامه کارشناسی یا کارشناسی ارشد به ادامه کار پرداخت.

[50] Bench Mark

مراجع

[1] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, **(2002)** "**A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking**", IEEE Trans. Signal Processing, vol. 50, pp. 174-188.

[2] J. M. Hammersley and K. W. Morton, **(1954)** "**Poor man's Monte Carlo**", J. Roy. Stat. Soc. , Series B, vol. 16, pp. 22-38.

[3] M. N. Rosenbluth and A. W. Rosenbluth, **(1956)** "**Monte Carlo calculation of the average extension of molecular chains**", J. Chem. Phys. , vol. 23, pp. 356-359.

[4] D. Crisan, **(2001)** "**Particle filters - A theoretical perspective," in Sequential Monte Carlo Methods in Practice**", A. Doucet, J. F. G. de Freitas, N. J. Gordon, Eds. Berlin: Springer Verlag.

[5] S. Challa and Y. Bar-Shalom, **(2000)** "**Nonlinear filter design using Fokker-Planck-Kolmogrov probability density evolutions**", IEEE Trans. Aero. Elect. Syst. , vol. 36, pp. 309-315.

[6] A. Doucet, N. de Freitas, and N. Gordon, **(2001)**, Eds. "**Sequential Monte Carlo Methods in Practice**", Springer.

[7] -----, Ed. , **(2001)**, "**Kalman Filtering and Neural Networks**", New York: Wiley.

[8] A. Doucet, S. J. Godsill, and C. Andrieu, **(2000)**, "**on sequential Monte Carlo sampling methods for Bayesian filtering**", Stat. Comput. , vol. 3, pp. 197-208.

[9] N. J. Gordon, D. J. Salmond, and A. F. M. Smith, **(1993)** "**Novel approach to nonlinear/non-Gaussian Bayesian state estimation**", IEE Proc. –F, vol. 140, pp. 107-113.

[10] de Frietas, J. F. G., Niranjan, M. and Gee, A. H., **(1997)** "**Hierarchichal Bayesian-Kalman models for regularization and ARD in sequential learning**", Technical Report, Cambridge University.

[11] de Frietas, J. F. G., Niranjan, M. and Gee, A. H., **(1998)** "**the EM algorithm and neural networks for nonlinear state space estimation**", Technical Report, Cambridge University.

[12] P. Del Moral, A. Doucet, A. Jasra, **(2006)** "**Sequential Monte Carlo samplers**", J. R. Stat. Soc., Series B, pp. 411-436.

[13] D. Crisan, A. Doucet, **(2002)** "**A survey of convergence results on particle filtering for practitioners**", IEEE Trans. Signal Process., vol. 50, pp. 736-746.

[14] W. S. Chen, B. R. Bakshi, P. K. Goel, S. Ungarala, **(2003)** "**Bayesian estimation by sequential Monte Carlo sampling, Application to large-scale non linear dynamical systems**", Technical Report, Ohio State University.

[15] Georges Oppenhiem, Anne Philippe, Jean de rigal, **(2008)** " **The particle filters and their applications**", Chemometrics and Intelligent Laboratory Systems, vol. 91, pp. 87-93.

[16] P. Fearnhead, **(2005)** "**Sequential Monte Carlo methods in filter theory**", PhD. Thesis, Oxford University.

[17] Z. Pang, **(2003)** "**tremor disorder Classification by neural networks and particle filters**", PhD. Thesis, University of Missouri.

[18] Haykin, S. **(1994)** " **Neural Networks: A Comprehensive Foundation**", Macmillan College Publishing Company.

[19] R. Rojas, **(1996)** " **Neural Networks**", Springer-Verlag, Berlin.

[20] ZHE CHEN, **(2002)** "**Bayesian Filtering: from Kalman filters to Particle Filters, and Beyond**", Wiely.

[21] J. F. G. de Freitas, **(2001)** " **Bayesian Methods for Neural Networks**", PhD. Thesis, Cambridge University.

[22] W. Melssen, R. Wehrens, **(2007) " Supervised Kohonen networks for classification problems",** jurnal of chemometrics and intelligent laboratory systems, 83.

[23] D. Gil, M. Johnsson, **(2010) " Supervised SOM vs. Perceptron Network "**, neural computation, 24.

[24] T. Kohonen, **(2005) " Self-Organizing Maps "**, ser Springer Series in information sciences. Berline, Springer, vol. 30.

[25] M. Hagenbuchner, A. Tosi, and A. Sperduti, **(2001) " A supervised Self-Organizing map for structured data "**, Advances in Self-Organizing Maps, pp 21-28.

[26] Fi-John Chang, Li Chang, H. Shan, and G. Wu, **(2010) " assessing the effort of meteorological variables for evaporation estimation by self organizing map neural network "**, Journal of Hydrology.

[27] P. Piela, **(2002) " Introduction to Self-Organizing Maps modeling for imputation techniques and technology "**, Research in Official Statistics, 2:5-19.

[28] H. Werner and M. Obeach, **(2001) " New neural network types estimation the accuracy of response for ecological modeling "**, Ecological Modelling, 146, pp 289-298.

[29] R. Rojas, **(1996) "Neural Networks"**, Springer-Verlag, Berlin, pp 408-412.

[۳۰] منهاج، محمد باقر، (**۱۳۸۶**) "**هوش محاسباتی، مبانی شبکه های عصبی**" جلد اول، چاپ چهارم، انتشارات دانشگاه صنعتی امیر کبیر، تهران، صص ۳۳۲-۳۳۵.

Abstract

In this thesis, we consider optimizing of one of the most important and practical self-organization neural networks, that is, Kohonen neural network using particle filter algorithm. Kohonen neural network has several applications in several areas such as density estimation, image and voice processing, and robotics. In many engineering problems particularly in places that a system has uncertainty or there is a non-Gaussian noise, Monte Carlo methods and especially particle filter have interesting results in compare to other powerful methods such as extended Kalman filter.

One serious problem with Kohonen neural network is dead units. This problem, in some cases, leads to a kind of inactivity in network's units, that is, those inactive units do not take part in update stage in algorithm and this may alters the performance of network worst. Various methods have been proposed to solve this problem and in this thesis we proposed a novel extension to correct and optimize the performance of network. Indeed, network's coefficients are estimated by SIR algorithm which by this we could not only solve the dead unit problem, but also we could promote both performance of the network in density estimation and accuracy of the estimation.

Preferences of using the particle filter algorithm for learning of Kohonen neural network are high accuracy of Monte Carlo method in estimation and prediction based on restricted data, disuse of feedback to compare the network output to observations, and homogeneous between Kohonen neural network as same as such other unsupervised self-organized neural networks and particle filter algorithm.

Keywords: self-organized neural network, Kohonen neural network, particle filter, learning, Monte Carlo method, importance sampling, sequential importance resampling.

Shahrood University of Technology

Faculty: Electrical and Robotic Engineering

Kohonen Neural Network Training by Particle Filter

Sardar Afra

Supervisor:

Dr. Mohammad Hadad Zarif

JUNE 2010