باسمه تعالی

پروژه درس شبکه عصبی

شبیه سازی شبکه عصبی استفاده شده در مقاله

"کاربرد شبکه عصبی در محاسبه افت فشار در لوله ها"

با استفاده از MATLAB

استاد محترم : دکتر موسوی

هادی بهین

ارشد مکاترونیک

دانشگاه آزاد تهران جنوب

Hadi.behin@gmail.com

تیر۹۴

فهرست مطالب:

چکیده:

در این تحقیق از نخستین کاربرد شبکه های عصبی[۱] یعنی تقریب توابع[۲] برای تقریب دیاگرام مودی[۳] برای استخراج ضریب اصطکاک دارسی[۴] که در محاسبات مربوط به افت فشار سیال لوله ها ناشی از اصطکاک ،استفاده می شود.در مکانیک سیالات ضریب اصطکاک دارسی یا از نمودار مودی استخراج می شود و یا از فرمول های نیمه تجربی مانند معادله کلمبرک[۵] محاسبه می شود.

در این مقاله تعداد ۱۲۰۰ نمونه[۶] از دو داده ورودی به نام عدد بی بعد رینولدز[۷] و زبری نسبی لوله[۸] تولید کرده ایم که داده های معلوم مسئله هستند و بصورت تصادفی و در محدوده مشخص انتخاب شده اند و۱۲۰۰ نمونه داده خروجی ضریب اصطکاک دارسی است که از روی نمودار مودی و با استفاده از داده های ورودی استخراج شده و به عنوان داده ای آموزش[۹] برای شبکه عصبی طراحی شده در نظر گرفته شده اند. آموزش شبکه را با دو سری داده انجام شده است یک سری داده های معمولی و خام و سری دوم داده هایی که ورودی های آنها بصورت غیر خطی نورمالایز لگاریمی شده اند و تفاوت خطای نهایی آنها مورد بررسی قرار گرفته شده است.

بعد از آموزش شبکه و رسیدن به سطح خطای مطلوب ،ضریب اصطکاک را از برنامه گرفته و وارد قسمت دوم برنامه برای محاسبه افت فشار لوله ناشی از اصطکاک سیال با جدار لوله برای جریان کاملا فراگیر و لوله با قطر ثابت کرده و افت هد[۱۰] اصلی را بدست آورده ایم.

---

۱ Neural Network
۲ Fittting app
۳ Moody diagram
۴ Darcy Friction Factor
۵ Colebrook-White equation
۶ Sample
۷ Reynolds
۸ Relative Roughness
۹ Train
۱۰ hed

الف – توضیحاتی در مورد جریان در لوله ها و دیاگرام مودی:

ضریب اصطکاک بصورت آزمایشی تعیین می شود. نتایجی که توسط ال.اف.مودی[۱۱] انتشار یافت به دیاگرام مودی مشهور شده است.این نمودار شامل قسمت های مختلفی چون ناحیه آرام[۱۲]،بحرانی[۱۳]،مغشوش[۱۴]با لوله کاملا زبر می باشد که این تقسیم بندی بستگی به عدد رینولدز بدست آمده برای سیال می باشد.

ضریب اصطکاک را می توان برای هر ناحیه از معادله های تجربی متفاوت بدست آورد:

مثلا در ناحیه آرام یعنی جایی که 2000<Re<4000:

f = 64/Re;

و برای ناحیه مغشوش Re>4000 معادله تجربی کولبروک:

$$\frac{1}{\sqrt{f}} = -2.0 \log_{10}\left(\frac{\epsilon/d}{3.7} + \frac{2.51}{Re\sqrt{f}}\right)$$

عدد بی بعد رینولدز در واقع نسبت نیروی اینرسی به چسبندگی است.

$$Re_D = \frac{\rho V D}{\mu}$$

و زبری نسبی(e) هم از با توجه به جنس لوله از جدول زیر قابل محاسبه است:

Surface	Absolute Roughness Coefficient - k -	
Surface	(m) 10-3	(feet)
Stainless steel	0.015	5 10-5
Steel commercial pipe	0.045 - 0.09	1.5 - 3 10-4
Stretched steel	0.015	5 10-5

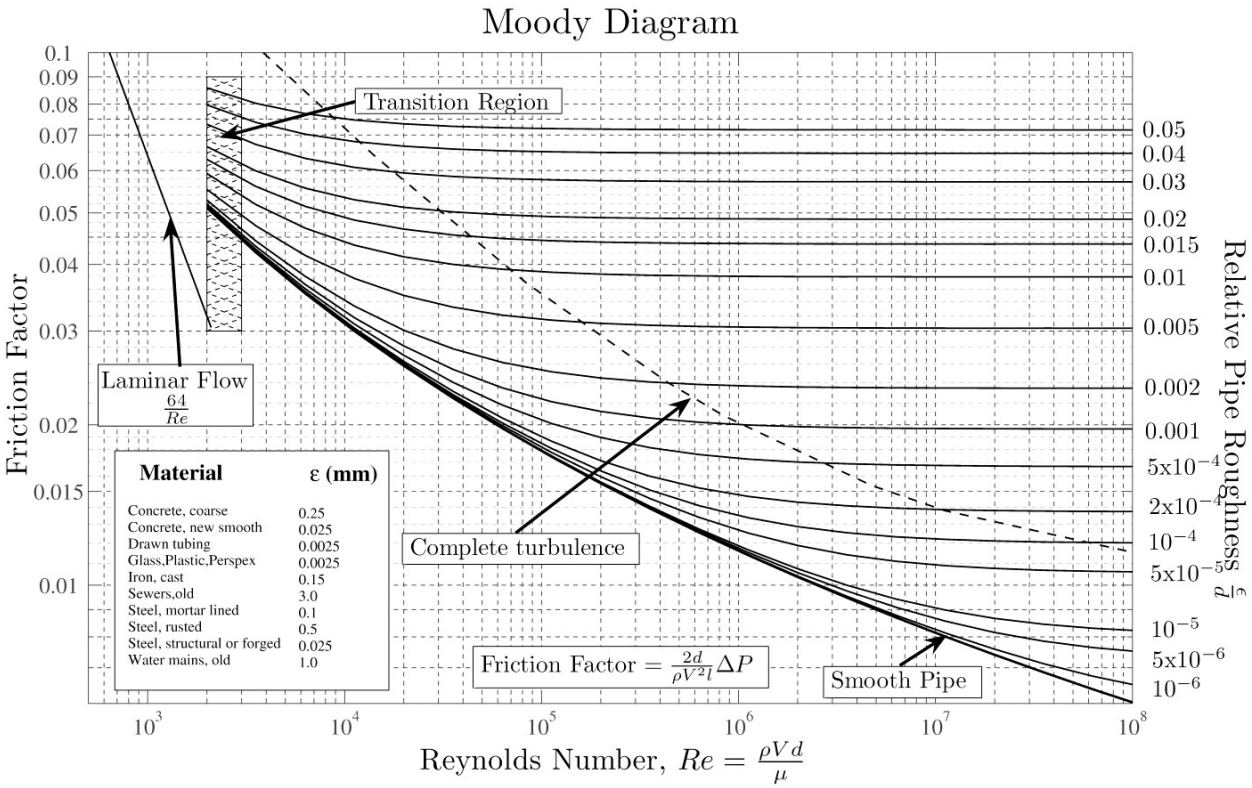
14 more rows

D هم قطر لوله است.

---

[۱۱] Moody
[۱۲] laminar
[۱۳] critical
[۱۴] turbulent

چون استفاده از فرمول های تجربی زمانبر می باشد بنابراین برای محاسبات سریعتر معمولا از دیاگرام مودی استفاده می شود:



همانطوری که مشاهده می شود محورهای دیاگرام مودی بصورت لگاریتمی تغییر میکنند.بنابراین همواره در استفاده از نمودار باید توجه شود که با استفاده از رینولدز و زبری نسبی،مقدار f را بدست آورد یعنی:

$Re \longrightarrow f$

استفاده از این روش بصور برعکس($f \longrightarrow Re$) به دلیل اختلاف زیاد درمقدار رینولدز بدست آمده ،غیر قابل قبول است.

افت فشار:

برای محاسبات مربوط به جریان در لوله ها،معمولا با محاسبه افت فشار سر و کار داریم،افت هد کل جریان در یک لوله مساوی است با :

$h_T = h_l + h_{lm}$

h_l : اتلافات اصلی که بر اثر اصطکاک در جریان کاملا فراگیر در لوله های با مساحت ثابت است که در این مقاله وقتی از افت فشار یا افت هد صحبت میکنیم منظور این افت فشار اصلی است.

h_{lm} : اتلافات فرعی که مربوط به ورودی ها ،اتصالات ،تغییرات مساحت لوله و غیره است که در این مقاله با این افت فشار کاری نداریم ولی با توجه به مفروضات مسئله به سادگی قابل محاسبه است.

یک نکته[۱۵] باید مدنظر داشته باشیم که در حالت کلی افت فشار در لوله با افت هد یکی نیست ولی اگر فرض کنیم که لوله افقی باشد با استفاده از قانون اول ترمودینامیک و با فرض $h_{lm=0}$ خواهیم داشت:

$$\frac{(P_1 - P_2)}{\rho} = \frac{\Delta P}{\rho} = h_1$$

لذا برای جریان کاملا فراگیر در یک لوله افقی با مساحت ثابت ،افت هد اصلی را بر حسب افت فشار می توان بیان کرد.پس در این مقاله هر جا بیان شده است افت هد منظور همان افت فشار است.

و در نهایت مقدار h_1 یا همان افت هد اصلی را می توان از دارسی-ویسباخ[۱۶] زیر محاسبه می شود:

$$h_f = f \cdot \frac{L}{D} \cdot \frac{V^2}{2g}$$

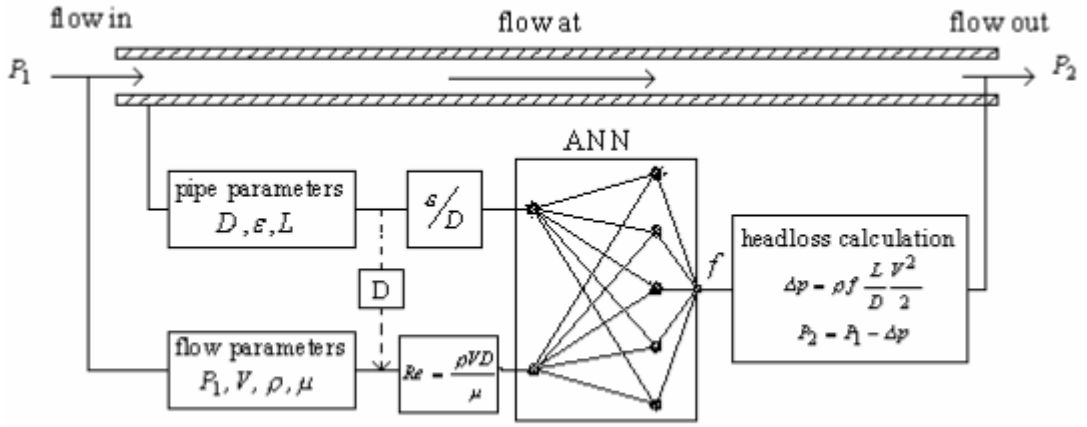
که در آن :

- h_f هد تلف شده به علت وجود اصطکاک
- L طول لوله
- D قطر هیدرولیکی لوله است .(که در مورد لوله با سطح مقطع دایروی همان قطر لوله است.)
- V سرعت میانگین سیال جاری در لوله است.
- g شتاب گرانش محلی زمین(که برای لوله های که ارتفاع ندارند در نظر نمیگیریم)
- f ضریب اصطکاک دارسی است که عدد بی بعدی می‌باشد و مقدار آن با توجه به نوع زبری لوله‌ها و عدد رینولدز و به کمک نمودار مودی چارت یا رابطه کلبروک تعیین می‌شود.

[۱۵] طبق کتاب مکانیک سیالات رابرت دبلیو.فاکس ویراست ششم

[۱۶] Darcy–Weisbach equation

همانطور که ملاحظه می شود برای بدست آوردن افت فشار نیاز به بدست آوردن ضریب اصطکاک دارسی وجود دارد که چون استفاده از فرمول های تجربی زمان بر هستند معمولا از چارت مودی استفاده می شود که در این مقاله چارت مودی را توسط یک شبکه عصبی تقریب زده ایم و بعد از رسیدن به سطح خطای مطلوب ضریب اصطکاک بدست آمده از شبکه را همانطور که در شکل زیر مشاهده می شود در محاسبات مربوط به افت فشار آورده ایم .



یک سیستم عصبی مصنوعی در محاسبات افت فشار در یک لوله

ب – شبیه سازی شبکه عصبی:

داده های ورودی و خروجی:

برای استخراج ضریب اصطکاک از نمودار مودی ما دو ورودی داریم عدد بی بعد رینولدز و دیگری زبری نسبی لوله بنابراین شبکه عصبی دو ورودی دارد و یک خروجی که محدوده آنها را در جدول زیر مشخص کرده ایم.

ورودی ها و خروجی شبکه عصبی و دامنه تغییرات آنها	
دامنه تغییرات	ورودی ها و خروجی
Re<۴۰۰۰ ناحیه آرام ۴۰۰۰<Re<10^8 ناحیه گذرا و مغشوش	$\mathrm{Re} = \frac{\rho VD}{\mu}$ (اینرسی/چسبندگی)
۰,۰۰۰۰۰۱< $\frac{e}{D}$ <۰,۰۵	$\frac{e}{D}$ (زبری نسبی/قطر هیدرولیکی لوله)
۰,۰۰۸< f <0.1	f (ضریب اصطکاک دارسی)

برای اینکه شبکه عصبی خوب آموزش ببیند باید داده های ورودی بصورت تصادفی انتخاب شوند بنابراین برای این منظور با استفاده از متلب خواهیم داشت:

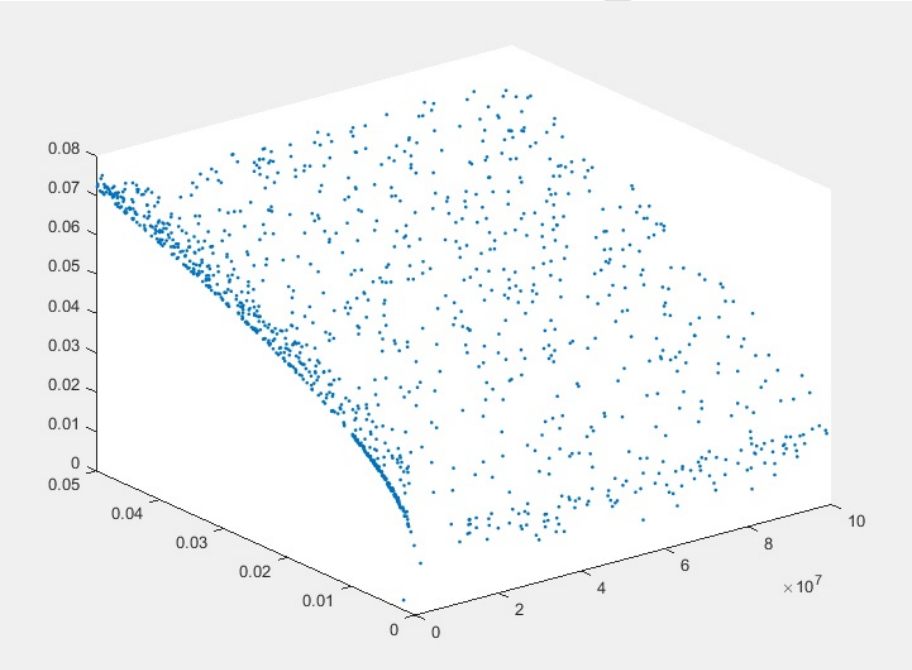
```
Re1 = 4000;
Re2 = 100000000;
x = Re1 + (Re2-Re1)* rand(1200,1);

(e/D)1 = 0.00001;
(e/D)2 = 0.05;
(e/D) = (e/D)1 + ((e/D)2-(e/D)1)* rand(1200,1);

my_dataset = dataset(Re,(e/D));
export(my_dataset,'File','d:\end\3\inputs.txt');
```

بعد از تولید اعداد تصادفی آنها را به عنوان دیتا ست ذخیر میکنیم و با استفاده از دیاگرام مودی خروجی یعنی ضریب اصطکاک را استخراج میکنیم و ذخیره میکنیم.و در شبکه هر جا احتیاج بود این دیتا ست ها را در محیط متلب فراخونی خواهند شد.

بعد از استخراج ضریب اصطکاک از روی دیاگرام مودی با استفاده از اعداد تصادفی تولید شده ، اگر نمودار سه بعدی ورودی ها و خروجی ها را رسم کنیم خواهیم داشت:



همانطوری که مشخص است نوع قرار گیری دیتاها بیان کننده رابطه معنا دار ورودیها و خروجی می باشد و داده ها تصادفی تولید شده و خروجی استخراج شده از نمودار مودی دارای توزیع تقریبا مناسبی می باشد

به علت اینکه دیاگرام مودی رفتار غیر خطی دارد یعنی محورهای عدد رینولدز و زبری نسبی بصورت لگاریتمی تغییر میکنند بنابراین برای بدست آوردن سطح خطای مطلوب باید داده های ورودی بصورت غیر خطی از نوع لگاریتمی و با استفاده از فرمول های زیر ،نرمالایز شوند.

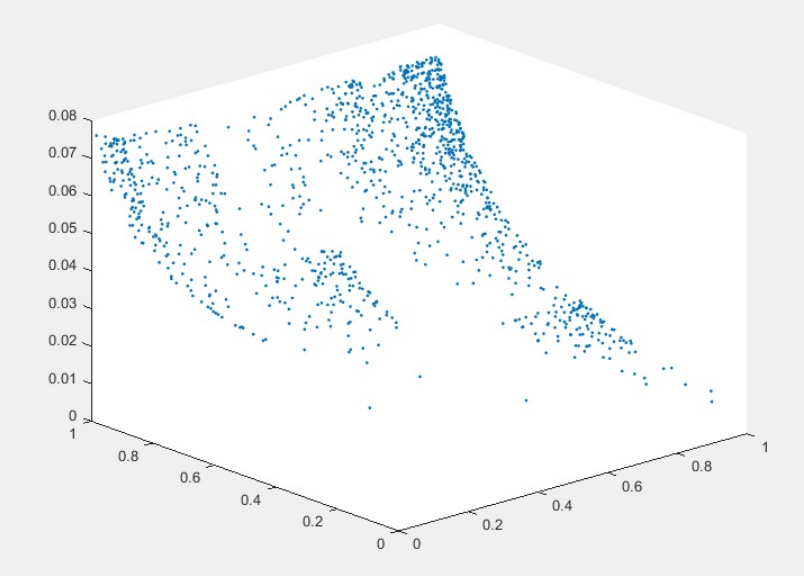
$$normalized = \frac{(\log(actual\,(\mathrm{Re})) - \log(\min(\mathrm{Re})))}{(\log(\max(\mathrm{Re})) - \log(\min(\mathrm{Re})))}$$

$$normalized = \frac{(\log(actual\,(\frac{e}{D})) - \log(\min(\frac{e}{D})))}{(\log(\max(\frac{e}{D})) - \log(\min(\frac{e}{D})))}$$

این فرمول ها را در محیط متلب وارد کرده و دیتا ست های ورودی و خروجی را فراخوانی کرده و عملیات نرمالایز را انجام داده و دیتا ست های ورودی و خروجی نرملایز شده را به عنوان سری دوم دیتاهای آموزش شبکه ذخیره می شوند

```
1
2 -    Re = dataset('XLSFile','d:\input1.xlsx');
3 -    e/D = dataset('XLSFile','d:\input2.xlsx');
4 -    p1 = (log(Re)-log(4000))/(log(100000000)-log(4000))
5 -    p2 = (log(e/d)-log(0.00001))/(log(0.05)-log(0.00001))
6 -    p=[p1 p2]
```

اگر نمودار سه بعدی اعداد نرملایز شده را بر حسب خروجی رسم کنیم خواهیم داشت:



که همانطور که مشاهده می شود شیب نمودار داده ها تغییر کرد و این تغییر باعث بهتر شدن سطح خطای شبکه خواهد شد.

ج – آموزش شبکه عصبی با استفاده از جعبه ابزار شبکه عصبی:

اولین گام در آموزش شبکه عصبی ارائه الگوهایی است که شبکه بتواند با استفاده از آنها آموزش ببیند برای این منظور از ۱۲۰۰ نمونه تولید شده در قسمت قبل استفاده می شود،این داده ها در سه قسمت وارد نرم افزار می شوند ۷۰٪ برای آموزش[۱۷] ، ۱۵٪ برای ارزیابی[۱۸]، ۱۵٪ برای تست[۱۹] استفاده از داده های ارزیابی باعث می شود شبکه تعمیم پذیری بیشتر داشته باشه و از فیت شدن بیش از حد شبکه بر روی نقاط آموزش جلوگیری می کند به این روش توقف پیش از موعد[۲۰] می گویند .

داده در دو سری خام و نرمالایز شده به صورت جداگانه به شبکه داده شده اند و نتایج مورد بررسی قرار گرفته است.

پس از ایجاد پایگاه داده ویژه آموزش شبکه،بایستی شبکه مناسب انتخاب شود.هدف از این تحقیق ارائه یک شبکه عصبی است که بتواند رابطه مطلوب بین ورودی و خروجی را پیدا کند یا همتن تقریب تابع بنابراین یک شبکه پرسپترون چند لایه[۲۱] با اتصالات پیش رو[۲۲] و الگوریتم آموزش انتشار برگشتی[۲۳] از نوع بهبود یافته لورنبرگ مارکورت[۲۴](به علت سرعت بالای همگرایی) که دارای یک لایه پنهان و یک لایه بیرونی تابع انتقال لایه پنهان از نوع tansig به علت پیوسته بودن و مشتق پذیر بودن و از نوع purelin برای لایه بیرونی،تعداد نرون های لایه پنهان با سعی و خطا بدست میآیند در جهت هر چه کمتر کردن خطای شبکه.

ابتدا شبکه را با داده های خام آموزش داده ایم و سپس شبکه را با داده های نرملایز شده آموزش داده ایم و خطای آنها را با هم مقایسه کرده ایم و شبکه ای که داراس سطح خطای کنتری است را انتخاب کرده و با سعی و خطا تعداد نرون های لبه پنهان آن را تعیین کرده و خر.جی شبکه را با داده های واقعی مقایسه کرده ایم

در جدول زیر ساختار شبکه و اطلاعات آن اورده شده اند.

---

[۱۷] Train
[۱۸] validation
[۱۹] Test
[۲۰] Early Stopping
[۲۱] MLP(Multi Layout Perseptron)
[۲۲] FeedForward
[۲۳] Back Propaction(BP)
[۲۴] Levenberg-Marquardt

خلاصه مشخصات و پارامترهای شبکه عصبی بکار رفته در این تحقیق	
پارامتر	**توضیحات**
ساختار شبکه	پرسپترون چند لایه (MLP)
نوع شبکه عصبی	پیش رو(FeedForward)
الگوریتم آموزش	انتشار برگشتی(Back Propaction)
تابع عملکرد خطا	مجمع مربعات خطا(MSE:Mean Square Error)
روش بهینه سازی	Levenberg-Marquardt(LM)
تعداد لایه های پنهان	یک
تعداد نورون های لایه پنهان	۵-۱۷ با سعی و خطا
نرخ یادگیری	۰,۰۰۱ پیش فرضMatlab
تابع تحریک لایه پنهان	تانژانت سیگموئید(Tansig)
تابع تحریک لایه خروجی	خطی(purelin)
تعداد داده های train	840 sample(70%)
تعداد داده های validation	180 sample(15%)
تعداد داده های Test	180 sample(15%)
تعداد نسل آموزش	حداکثر ۱۰۰۰

توضیح جز به جز برنامه نوشته شد در متلب :

- تعریف ورودی ها و خروجی شبکه و فراخوانی دیتا ست های شبکه در محیط متلب:

```
%   Reynolds_RelativePipeRoughness - input data.
%   Friction_Factor - target data.
clc
close all
clear
Reynolds_RelativePipeRoughness = dataset('XLSFile','d:\inputs.xlsx');
Friction_Factor = dataset('XLSFile','d:\target.xlsx');
x = Reynolds_RelativePipeRoughness;
t = Friction_Factor;
```

- انتخاب تابع آموزش که به علت سریع بودن الگوریتم لونبرگ مارکورت را انتخاب میکنیم:

```
% Choose a Training Function
% 'trainlm' is usually fastest.

trainFcn = 'trainlm';  % Levenberg-Marquardt
```

- ساخت ساختار شبکه با تابع و تعیین تعداد نورون های لایه پنهان. این تابع مخصوص تقریب توابع است و این تابع بجایی تابع newff بکار برده می شود برای شبکه های BP تابع newff به پیشنهاد خود Matlab جدیدا استفاده نمی شود.

```
% Create a Fitting Network
hiddenLayerSize = 10;
net = fitnet(hiddenLayerSize,trainFcn);
```

- در این قسمت داده های ورودی را به سه قسمت آموزش ،ارزیابی و تست تقسیم کرده است و با توابع divideFcn و divideMode این تقسیم بندی را بصورت تصادفی و از سراسر دیتاها انجام داده است.

```
% Setup Division of Data for Training, Validation, Testing

net.divideFcn = 'dividerand';  % Divide data randomly
net.divideMode = 'sample';  % Divide up every sample
net.divideParam.trainRatio = 70/100;
net.divideParam.valRatio = 15/100;
net.divideParam.testRatio = 15/100;
```

- انتخاب تابع عملکرد خطا که میانگین مربعات خطا می باشد.

```
% Choose a Performance Function

net.performFcn = 'mse';  % Mean squared error
```

- انتخاب نمودارهایی که باید بعد از آموزش شبکه استخراج شوند برای بررسی عملکرد شبکه.از جمله نمودار خطای آموزش و ارزیاب و تست،نمودار رگرسیون و هیستوگرام تابع خطا و .... :

```
% Choose Plot Functions

net.plotFcns = {'plotperform','plottrainstate','ploterrhist', ...
  'plotregression', 'plotfit' ,'plotwb','plotconfusion'};
```

- شروع کردن آموزش شبکه:

```
% Train the Network
[net,tr] = train(net,x,t);
```

- تست شبکه با داده های تست، همانطور که میدونیم داده های validation در حین آموزش شبکه اعمال می شوند و داده های Test بعد از اتمام آموزش شبکه.

تابع gsubtract اختلاف بین خروجی واقعی یا همان t را با خروجی های شبکه محاسبه کرده و در e قرار میدهد و تابع perform میزان performance خطا را نشان میدهد

```
% Test the Network
y = net(x);
e = gsubtract(t,y);
performance = perform(net,t,y)
```

- در این قسمت مقدار performance شبکه برای داده های train ، validation و Test بصورت جداگانه محاسبه می شود و در خروجی برنامه چاپ می شوند.

```
46      % Recalculate Training, Validation and Test Performance
47 -    trainTargets = t .* tr.trainMask{1};
48 -    valTargets = t  .* tr.valMask{1};
49 -    testTargets = t  .* tr.testMask{1};
50 -    trainPerformance = perform(net,trainTargets,y)
51 -    valPerformance = perform(net,valTargets,y)
52 -    testPerformance = perform(net,testTargets,y)
```

نکته در مورد لاین ۴۷ تا ۴۹:

تابع Mask ا داده های آموزش و ارزیابی وتست را مشخص می کند و وقتی در t ضرب می شود مقدار داده هایی را که برای آموزش یا ارزیابی یا تست بکار رفته اند مشخص می کند.

اگر بخواهییم اندیس داده هایی که برای آموزش و ارزیابی و تست بکار رفته اند را پیدا کنیم از نقیض تابع isnan و با کمک تابع Mask استفاده می شود .مثلا اگر اندیس داده های Test را بخواهیم از دستور زیر داریم:

```
testIndices = find(~isnan(tr.testMask{1}))
```

خروجی برنامه اندیس ۱۸۰ نمونه داده های تست را از ۱۲۰۰ نمونه به ما میدهد که اینجا بعلت کمبود فضا دو خط اول و آخر از خروجی را قرار میدهیم:

```
testIndices =

  Columns 1 through 8

           2           4           6          20          26          32          46          50

 Columns 177 through 180

        1186        1191        1195        1198
```

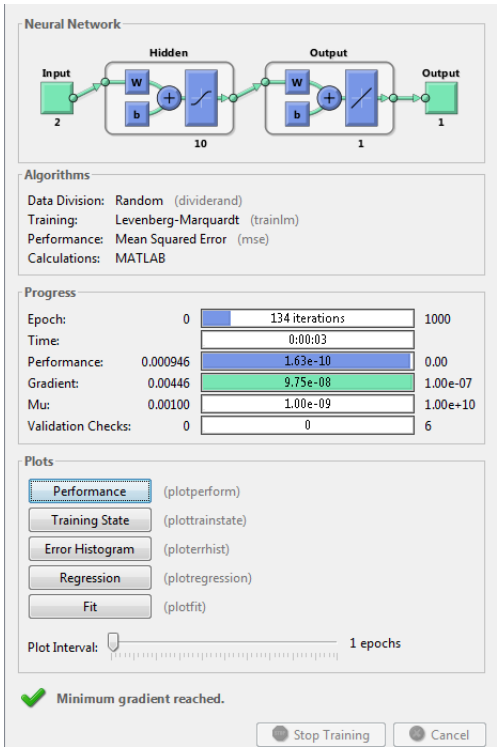
همانطور که ملاحضه می شود اندیس های داده هایی که برای Test شبکه بکار رفته شده است مشخص شده اند که همانطور که قبلا با تابع dividerand تعیین کرده ایم کاملا تصادفی انتخاب شده اند.

- در این قسمت صفحه ای که حاوی اطلاعات آموزش شبکه هست نشان داده می شود:

```
54      % View the Network
55 -    view(net)
```

که همانطور که در شکل مشاهده می شود حاوی اطلاعاتی از شبکه که قبلا توضیح داده شده اند در قسمت Progress داری شش قسمت می باشد که هر کدام از این شش گزینه که زودتر به حد مشخص شده برسد آموزش شبکه متوقف می شود که در

این شکل شبکه بعد از ۱۳۴ نسل آموزش به علت رسیدن مقدار گرادیان تابع خطا به حد مطلوب آموزش شبکه متوقف شده است.



- در قسمت آخر با دستور input ورودی های شبکه یعنی عدد رینولدز و زبری نسبی لوله را از کاربر گرفته و انها را بصورت لگاریتمی نرمالایز کرده و خروجی شبکه را میگیریم:

```
% input data from operator

Re = input('enter the Reynolds :    ');
if Re<4000
    Friction_Factor = 64/Re;
    string=num2str(Friction_Factor);
    final_string=strcat(' the Friction_Factor in Transition flow area is :   ',string,'');
    disp(final_string)

else
 Relative_Pipe_Roughness = input('enter the Relative_Pipe_Roughness(e/D) :  ');
 if Relative_Pipe_Roughness <= 0.05 & Relative_Pipe_Roughness >= 0.00001

p1 = (log(Re)-log(4000))/(log(100000000)-log(4000))
p2 = (log(Relative_Pipe_Roughness)-log(0.00001))/(log(0.05)-log(0.00001))
Friction_Factor = net([p1;p2])

else

string=num2str(Relative_Pipe_Roughness);
final_string=strcat(' the Relative_Pipe_Roughness Not in admissible Rang please Change (e/D)  ');
disp(final_string)
end
end
```

نکته : شرط هایی را تعریف کرده ایم که طبق محدودیت های نمودار مودی، اگر این شرط $0.00001 < \frac{e}{D} < 0.05$ برقرار نباشد برنامه درخواست تغییر $\frac{e}{D}$ را بدهد و همچنین اگر $\mathrm{Re} < 4000$ باشد، همانطور که قبلا بیان شد ضریب اصطکاک از فرمول $f = \frac{64}{\mathrm{Re}}$ بدست آید و نیازی به وارد کردن اطلاعات به شبکه شبیه سازی شده نیست.

- **خروجی شبکه:**

Performance را بطور جداگانه برای داده های آموزش و ارزیابی و تست و داده های کلی محاسبه کرده است و همچنین با ورودی عدد رینولدز و زبری نسبی لوله را از کاربر گرفته و خروجی شبکه که همان ضریب اصطکاک دارسی می باشد را محاسبه کرده است:

```
performance =

   4.0288e-07


trainPerformance =

   5.4938e-07


valPerformance =

   6.1627e-08


testPerformance =

   6.0456e-08

enter the Reynolds :    169600
enter the Relative_Pipe_Roughness(e/D) :  0.002

Friction_Factor =

    0.0234
```

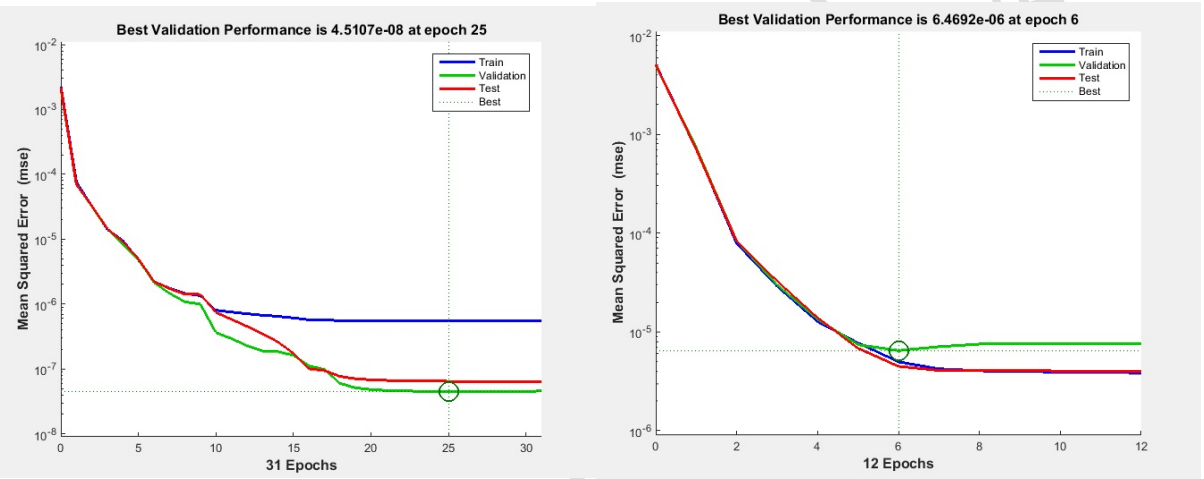
بررسی نمودارهای استخراج شده بعد از آموزش شبکه برای داده های ورودی خام و نرمالایز شده:

یک شبکه طبق دستوراتی که بالا توضیح داده شد، با ۱۰ نرون در لایه پنهان شبیه سازی کرده و این شبکه را یکبار Train کرده و تفاوت نمودار های استخراج شده را برای دو سری از داده ها مقایسه شده اند:

- **نمودار performance :**

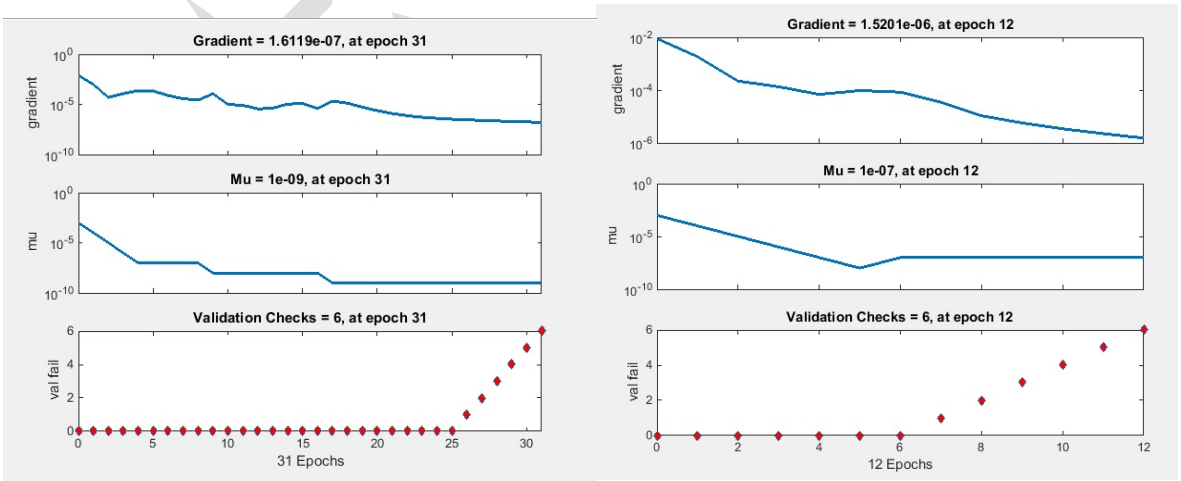
همانطوری که ملاحضه می شود داده های نرمالایز برای داده های Test، performance تابع خطای بهتری را فراهم کرده است .

داده های خام **داده های نرمالایز**



- **نمودار :Training state**

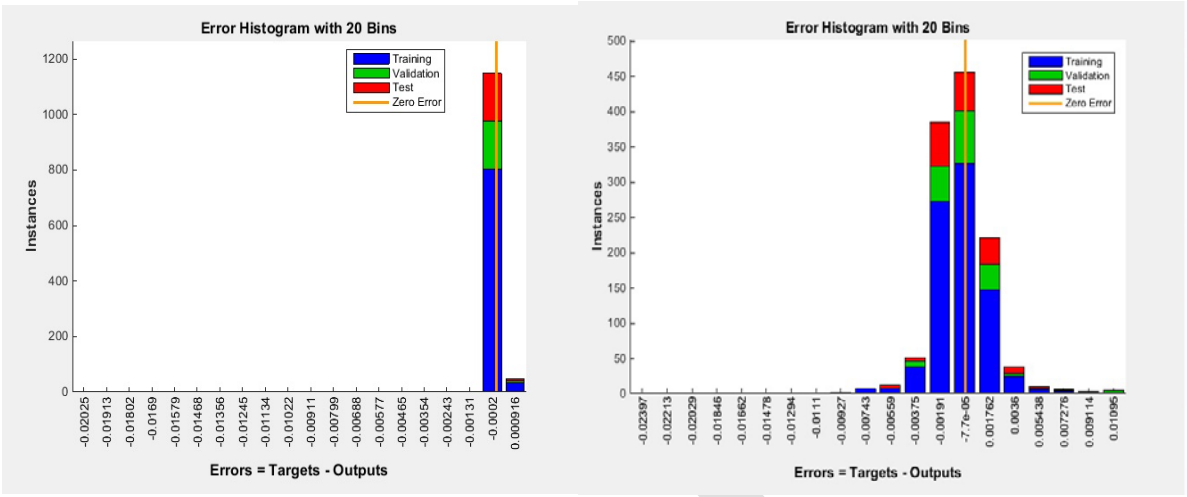
داده های خام داده های نرمالایز



- نمودار **Error Histogram** :

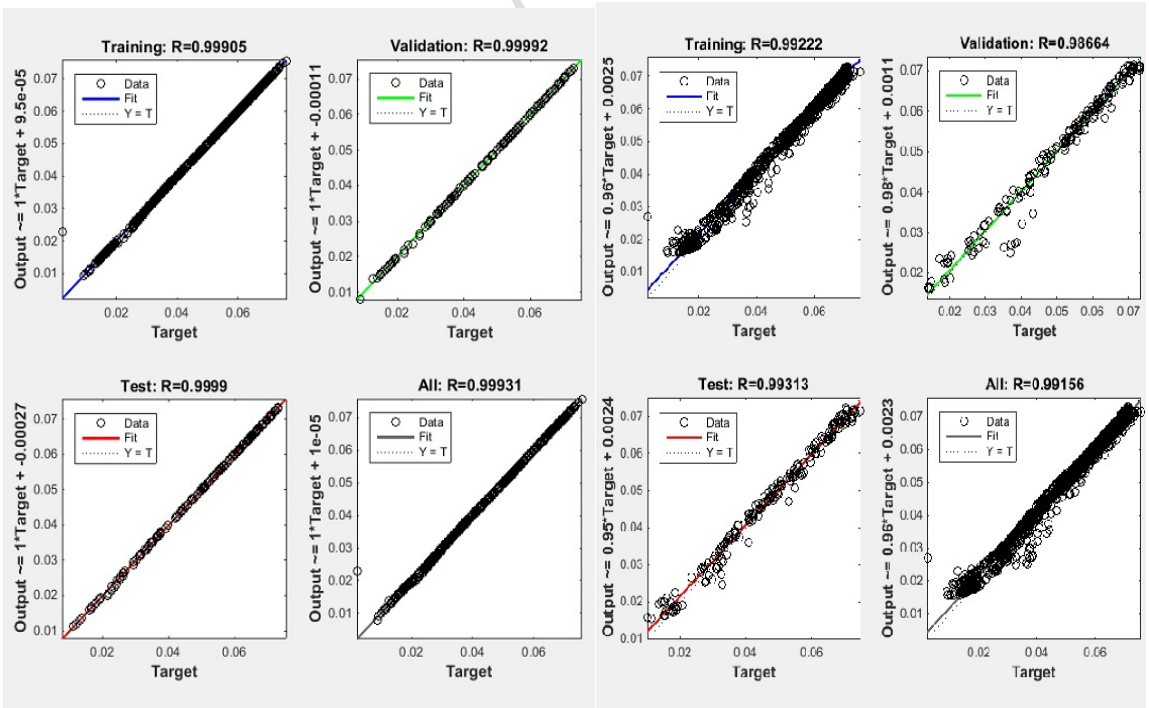
همانطور که مشاهده می شود ستون های داده هاس نرملایز به محدوده خط صفر (خط زرد رنگ) نزدیکتر هستن و نتایج بهتری را فراهم کرده اند.

داده نرمالایز | داده خام



- نمودار **Regression** :

داده نرمالایز | **داده خام**



این نمودار رگرسیون خروجی واقعی به خروجی شبکه عصبی را برای داده های Train,validaition,Test و داده هایی کلی نشان می دهد ضریب رگرسیون کلی داده های خام 0.99156 و ضریب رگرسیون کلی داده های نرمالایز

شده 0.99931 می باشد که با افزایش نرون حتما به یک هم خواهد رسید و این نشان دهنده دقت بالای این شبکه با داده های نرملایز شده است.

همچنین معادله خط رگرسیون کلی داده های خام :Output = 0.96 * Target + 0.0023

و معادله خط رگرسیون کلی داده های نرمالایز شده: Output = 1 * Target + 0.00001

که دارای شیب ۱ و یک عرض از مبدا خیلی کوچک ۰,۰۰۰۰۱ بنابراین میشه گفت با خط ۴۵ درجه یکی است که نشان دهنده دقت بالای شبکه با داده های نرمالایز شده می باشد.

نتیجه:

با توجه به نمودارهای بالا و مقدار performance و ضریب Regression حاصل شده آموزش شبکه با داده های نرمالایز شده خطای کمتری را برای فراهم می کند بنابراین شبکه نهایی را با داده های نرمالایز شده شبیه سازی شده و افت فشار لوله سیال را محاسبه خواهد شد.

- **نمودار خروجی شبکه و خروجی واقعی بر حسب اندیس داده های ورودی train:**

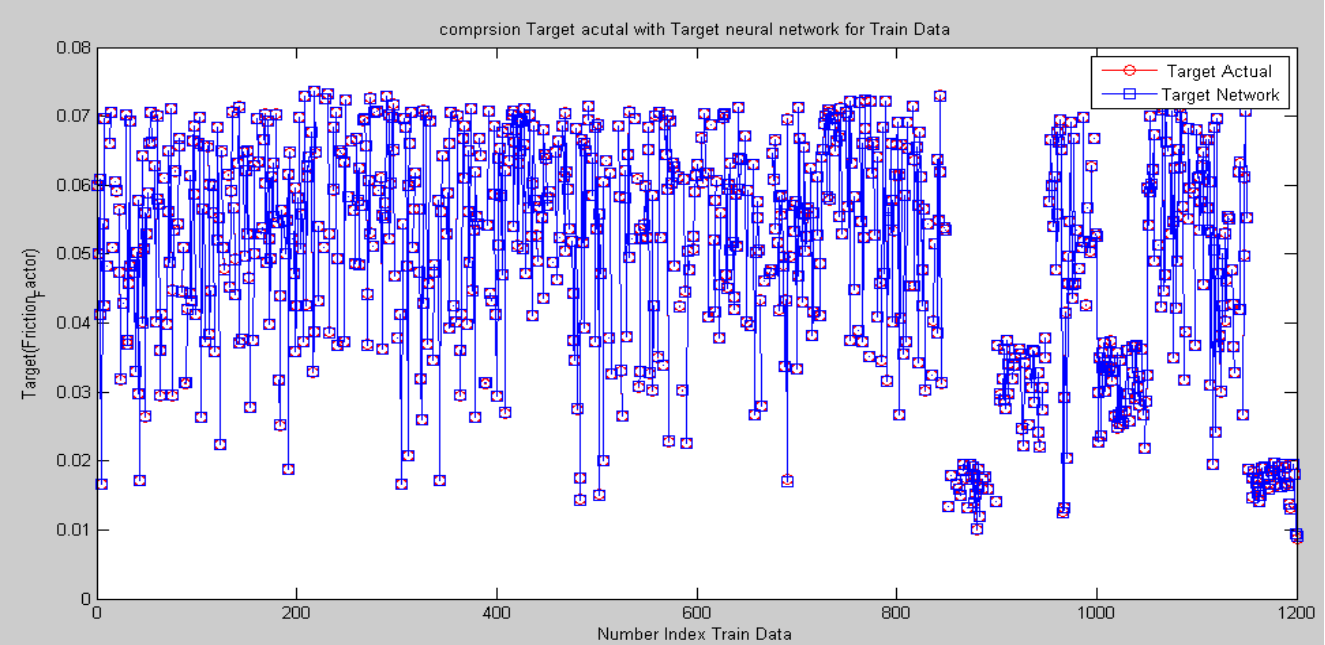
برای اینکه مشاهد شهودی تری از جواب خوب شبکه به داده های ورودی train داشته باشیم

خروجی شبکه و خروجی واقعی را در یک نمودار و روی هم رسم میکنیم.

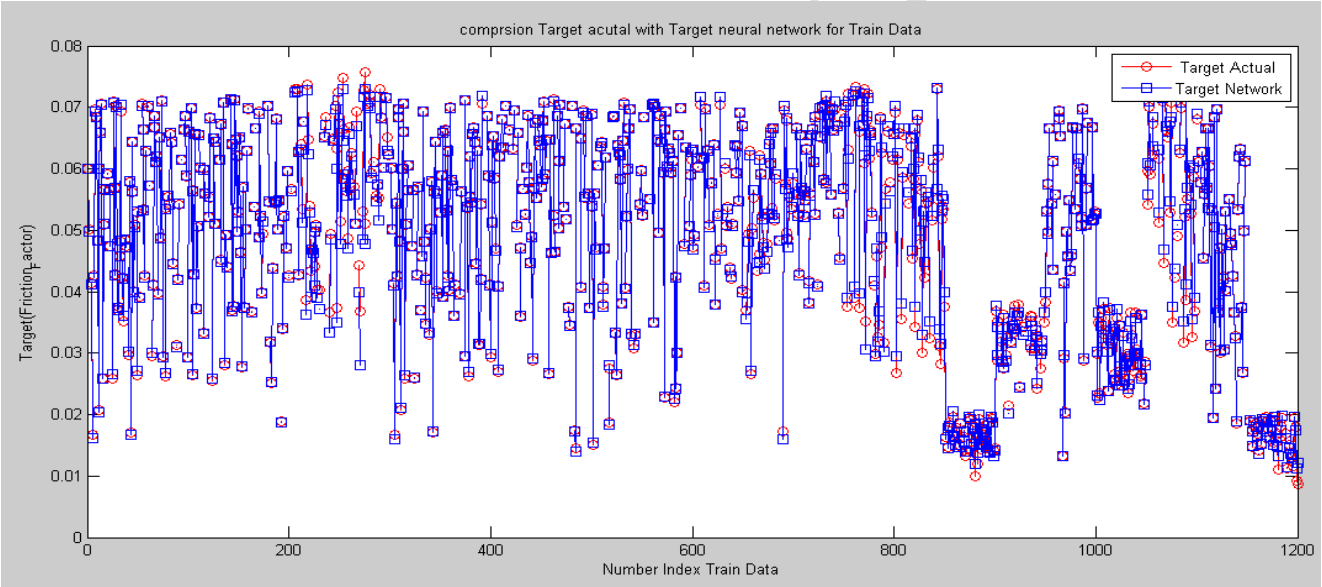
```
% comporsion with Target actual with Target Neural Network for Train Data
traininput1 = x(1,:) .*tr.trainMask{1};
traininput2 = x(2,:) .*tr.trainMask{1};
TrainInput = [traininput1;traininput2];
TargettrNet = sim(net,TrainInput);
figure(1);
plot(trainTargets,'-or'),hold on,plot(TargettrNet,'-sb'),hold off
legend(' Target Actual ','Target Network');
xlabel('Number Index Train Data');
ylabel('Target(Friction_Factor)');
title('comprsion Target acutal with Target neural network for Train Data');
```

در کد بالا ابتدا داده های ورودی که در train شبکه استفاده شده اند را به استفاده از تابع Mask پیدا کرده و شبکه را برای این داده های با تابع sim شبیه سازی کرده و خروجی های مربوط به این داده ها را از شبکه استخراج می شوند و این خروجی ها و خروجی های واقعی مربوط به داده های train را روی یک نمودار رسم میکنیم و همپوشانی آنها را با هم مقایسه میکنیم.

داده نرمالایز:

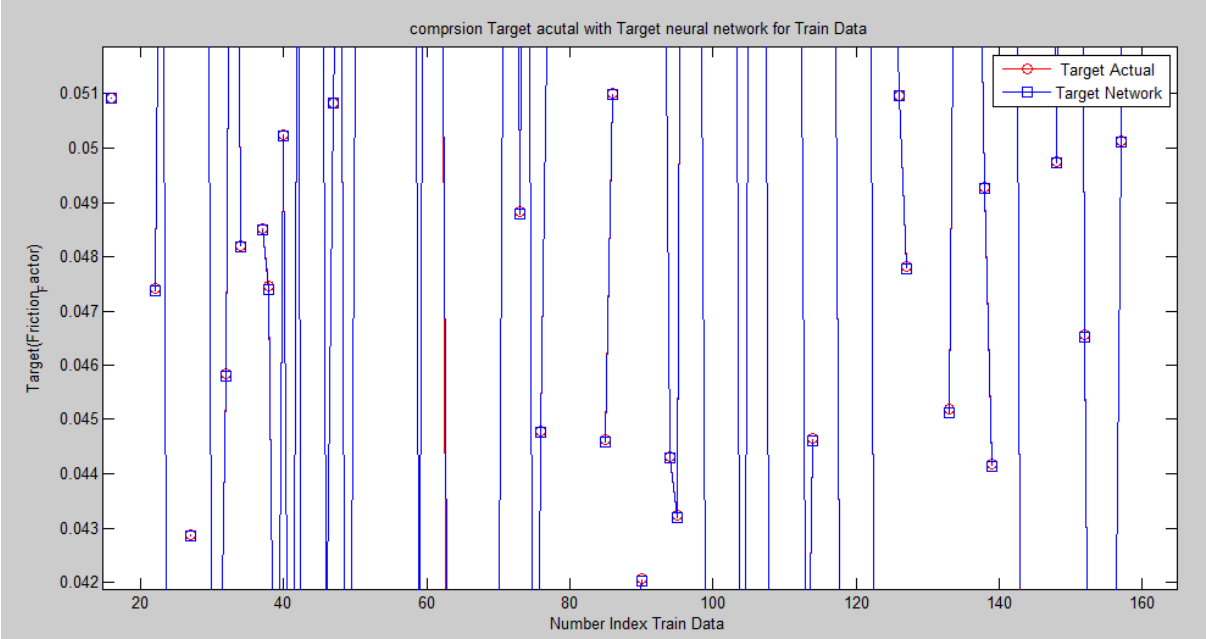


داده های خام:

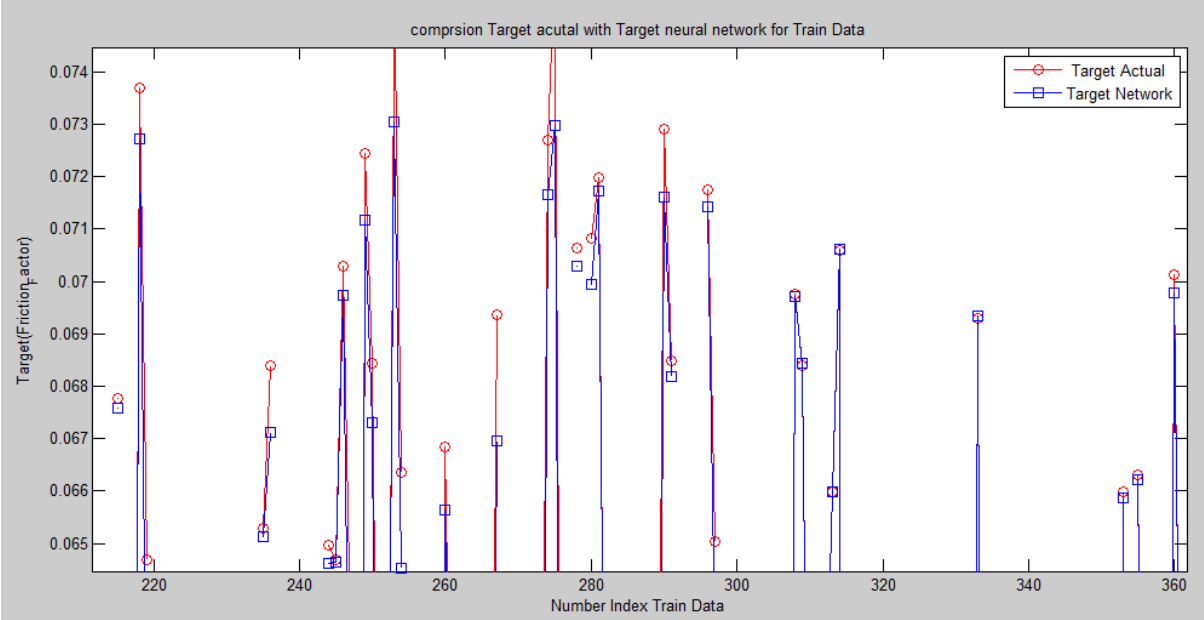


اگر قسمتی از نمودار را زوم کنیم خواهیم داشت:

داده نرملایز:



داده خام:

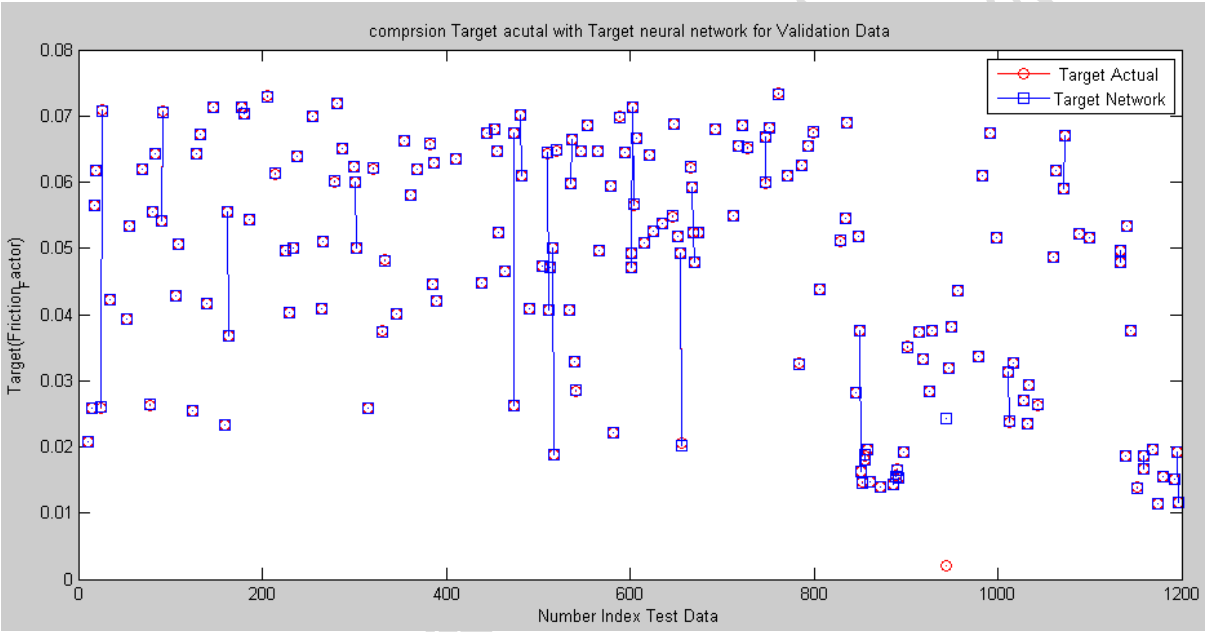


همانطور که مشاهده می شود خروجی های واقعی با خروجی های شبکه در بیشتر جاها دقیقا در داده های نرملایز شده رو هم افتاده اند و نمودار همپوشانی کاملی دارد ولی در نمودار مربوط به داده های خام این همپوشانی کمتر است. این نشان دهنده آموزش خوب شبکه می باشد برای داده های نرمالایز train است.

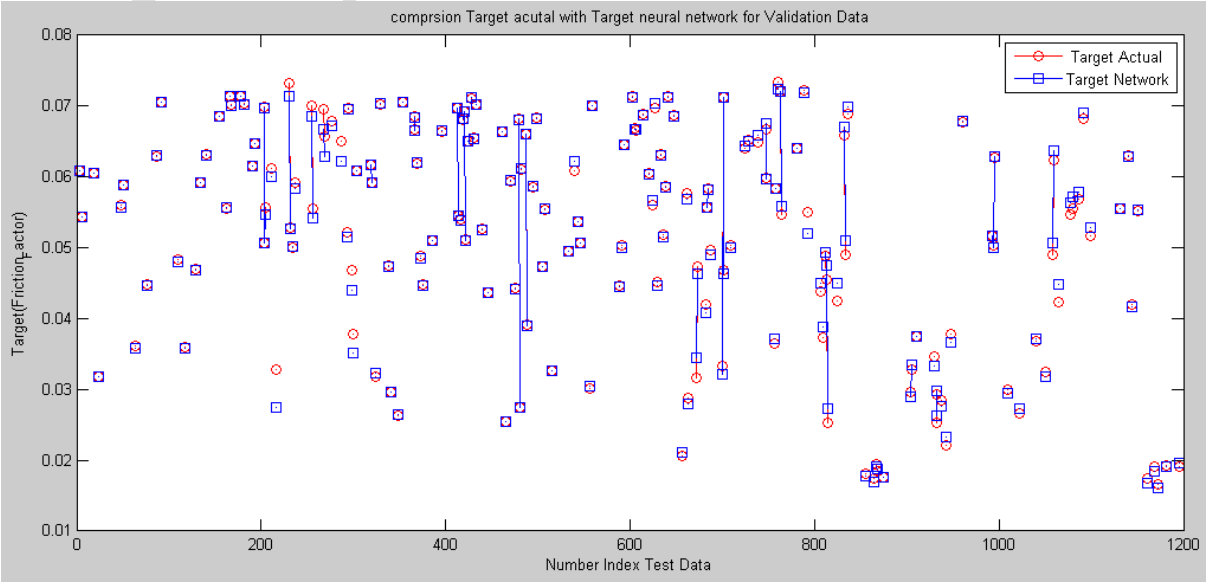
نمودار خروجی شبکه و خروجی واقعی بر حسب اندیس داده های ورودی **Validation**:

```
74      % comporsion with Target actual with Target Neural Network for Validation Data
75 -    valinput1 = x(1,:) .*tr.valMask{1};
76 -    valinput2 = x(2,:) .*tr.valMask{1};
77 -    ValidationInput = [valinput1;valinput2];
78 -    TargetvalNet = sim(net,ValidationInput);
79 -    figure(2);
80 -    plot(valTargets,'-or'),hold on,plot(TargetvalNet,'-sb'),hold off
81 -    legend(' Target Actual ','Target Network');
82 -    xlabel('Number Index Test Data');
83 -    ylabel('Target(Friction_Factor)');
84 -    title('comprsion Target acutal with Target neural network for Validation Data');
```

داده های نرملایز:



داده های خام:



نمودار خروجی شبکه و خروجی واقعی بر حسب اندیس داده های ورودی Test:

```
86      % comporsion with Target actual with Target Neural Network for Test Data
87 -    testinput1 = x(1,:) .*tr.testMask{1};
88 -    testinput2 = x(2,:) .*tr.testMask{1};
89 -    TestInput = [testinput1;testinput2];
90 -    TargettestNet = sim(net,TestInput);
91 -    figure(3);
92 -    plot(testTargets,'-or'),hold on,plot(TargettestNet,'-sb'),hold off
93 -    legend(' Target Actual ','Target Network');
94 -    xlabel('Number Index Test Data');
95 -    ylabel('Target(Friction_Factor)');
96 -    title('comprsion Target acutal with Target neural network for Test Data');
```

داده های نرمالایز:

داده های خام :

تعیین تعداد نورون لایه پنهان :

پس از نوع داده های آموزش باید تعداد نورون های لایه پنهان را برای هر چه کمتر شده خطای بین ضریب اصطکاک دخروجی از شبکه و ضریب اصطکاک واقعی بدست آورده شود.

برای این منظور شبکه را با تعداد متفاوت نورون بین ۵ تا ۱۷ نورون آموزش داده و چند نمونه تصادفی را به شبکه اعمال کرده و نتایج را در جدول زیر آورده خواهد شد:

تست شبکه با نورون های متفاوت

عدد رینولدزRe	زبری نسبیe/D	ضریب اصطکاک خروجی شبکه			ضریب اصطکاک استخراج شده از دیاگرام مودی
		۷ نورون	۱۰ نورون	۱۷نورون	
5000	0.004	0.0413	0.0414	0.0419	**0.0416**
100000	0.02	0.0491	0.0488	0.0490	**0.0490**
4000000	0.006	0.0322	0.0325	0.0321	**0.0321**
900000	0.03	0.0571	0.0571	0.0574	**0.0572**
60500000	0.0009	0.0366	0.0367	0.0366	**0.0365**

معیار رد یا پذیرش جواب هایی ANN این است که خطای محاسبه شده e_k که از رابطه زیر به دست می آید کمتر از یک درصد باشد:

$$\frac{f_{chart}-f_{ANN}}{f_{chart}}\times 100$$

که برای شبکه عصبی با ۷ نورون خطای همه ی نمونه های تست زیر یک درصد و قابل قبول است.

❖ محاسبه افت فشار یک لوله با استفاده از شبکه عصبی طراحی شده:

در پایان یک مثال کامل از محاسبه افت فشار در یک لوله در اثر اصطکاک با ترکیب شبکه عصبی طراحی شده و برنامه نوشته شده برای محاسبه افت فشار.

- **مثال:**

جریان آب با دبی حجمی $Q = 0.01\ {}^{m^3}/_{s}$ در لوله ای افقی صاف و بدون اتصالات و از جنس آهن گالوانیزه دایروی به قطر $D = 75^{mm}$ و با طول $l = 100^{m}$ جریان دارد، افت فشار اصلی(ناشی از اصطکاک) را بدست آورید.

چگالی آب: $\rho_{H_2O} = 999\ {}^{kg}/_{m^3}$

لزجت : $\mu = 1.0\times10^{-3}\ {}^{kg}/_{(m.s)}$

زبری نسبی برای لوله با جنس آهن گالوانیزه: $e = 0.00015m$

سرعت متوسط سیال داخل لوله: $\overline{V} = \frac{Q}{A} = \frac{Q}{\pi D^2} = \frac{4Q}{\pi D^2}$

عدد رینولدز : $\mathrm{Re} = \frac{\rho \overline{v} D}{\mu}$

معادله محاسبه افت هد دارسی – ویسباخ: $h_l = f\ \frac{L}{D}\frac{\overline{v}^2}{2}$

در ادامه برنامه نوشته شده را در متلب اجرا کرده و افت فشار با مفروضات مسئله بدست خواهد آمد.

```
%   Reynolds_RelativePipeRoughness - input data.
%   Friction_Factor - target data.
clc
close all
clear
Reynolds_RelativePipeRoughness = dataset('XLSFile','d:\inputs.xlsx');
Friction_Factor = dataset('XLSFile','d:\target.xlsx');
x = Reynolds_RelativePipeRoughness;
t = Friction_Factor;
  % Choose a Training Function
  % 'trainlm' is usually fastest.

  trainFcn = 'trainlm';  % Levenberg-Marquardt
  % Create a Fitting Network
  hiddenLayerSize = 7;
  net = fitnet(hiddenLayerSize,trainFcn);
 % Setup Division of Data for Training, Validation, Testing

 net.divideFcn = 'dividerand';  % Divide data randomly
 net.divideMode = 'sample';  % Divide up every sample
 net.divideParam.trainRatio = 70/100;
 net.divideParam.valRatio = 15/100;
 net.divideParam.testRatio = 15/100;
  % Choose a Performance Function

  net.performFcn = 'mse';  % Mean squared error
% Choose Plot Functions

net.plotFcns = {'plotperform','plottrainstate','ploterrhist', ...
  'plotregression', 'plotfit' ,'plotwb','plotconfusion'};
  % Train the Network
  [net,tr] = train(net,x,t);
  % Test the Network
  y = net(x);
  e = gsubtract(t,y);
  performance = perform(net,t,y)
 % Recalculate Training, Validation and Test Performance
 trainTargets = t .* tr.trainMask{1};
 valTargets = t  .* tr.valMask{1};
 testTargets = t  .* tr.testMask{1};
 trainPerformance = perform(net,trainTargets,y)
 valPerformance = perform(net,valTargets,y)
 testPerformance = perform(net,testTargets,y)
  % View the Network
  view(net)
```

تا این قسمت برنامه مربوط به شبیه سازی شبکه عصبی است که توضیح داده شد قسمت زیر مربوط به برنامه محاسبه افت فشار توسط معادله دارسی –ویسباخ و با ضریب اصطکاک بدست آمده از خروجی شبکه عصبی.

وارد کردن معلومات مسئله توسط کاربر

```
Q = input('enter the Fluid debit (m^3/s) :     ');
D = input('enter the Pipe  diameter (meter):    ');
l = input('enter the  Pipe length  (meter):       ');
ro = input('enter the Fluid density (kg/m^3):     ');
muu = input('enter the absolute viscosity beetwen Fluid and Pipe  (kg/(m.s)):     ');
e = input('enter the Pipe roughness (m):       ');
```

محاسبه پارامترهای مورد نیاز معادله دارسی ویسباخ(سرعت متوسط سیال،عدد رینولدز،زبری نسبی لوله)

```
70 -    V = (4*Q)/(pi*D^2); Re = (ro*V*D)/(muu)
71 -    Relative_Pipe_Roughness = (e/D)
```

نرمالایز کردن لگاریتمی ورودیهای شبکه مخصوص این مسئله:

```
70 -    p1 = (log(Re)-log(4000))/(log(100000000)-log(4000))
71 -    p2 = (log(Relative_Pipe_Roughness)-log(0.00001))/(log(0.05)-log(0.00001))
```

محاسبه افت فشار و تعریف شرط ها:

شرط هایی را تعریف کرده ایم که طبق محدودیت های نمودار مودی، اگر این شرط $0.00001 < \frac{e}{D} < 0.05$ برقرار نباشد برنامه درخواست تغییر $\frac{e}{D}$ را بدهد و همچنین اگر $\mathrm{Re} < 4000$ باشد، همانطور که قبلا بیان شد ضریب اصطکاک از فرمول $f = \frac{64}{\mathrm{Re}}$ بدست آید و نیازی به وارد کردن اطلاعات به شبکه شبیه سازی شده نیست. بنابراین به این ترتیب ضریب اصطکاک بدست می اید و سپس وارد فرمول دارسی ویسباخ می شود.

```
113 -   if Re<4000
114 -       Friction_Factor = 64/Re;
115 -       string=num2str(Friction_Factor);
116 -       final_string=strcat(' the Friction_Factor in Transition flow area is :   ',string,'');
117 -       disp(final_string)
118 -       Pressure_drop = Friction_Factor*(l/D)*(V^2/2)
119 -   else
120
121 -   if Relative_Pipe_Roughness <= 0.05 & Relative_Pipe_Roughness >= 0.00001;
122
123 -   p1 = (log(Re)-log(4000))/(log(100000000)-log(4000))
124 -   p2 = (log(Relative_Pipe_Roughness)-log(0.00001))/(log(0.05)-log(0.00001))
125 -   Friction_Factor = net([p1;p2])
126 -   Pressure_drop = Friction_Factor*(l/D)*(V^2/2)
127
128 -   else
129 -      string=num2str(Relative_Pipe_Roughness);
130 -     final_string=strcat(' the Relative_Pipe_Roughness Not in admissible Rang please Change (e/D)  ');
131 -     disp(final_string)
132 -   end
133 -   end
```

خروجی شبکه :

```
performance =

   4.3334e-07


trainPerformance =

   5.9586e-07


valPerformance =

   5.0503e-08


testPerformance =

   5.7791e-08

enter the Fluid debit (m^3/s) :    0.01
enter the Pipe  diameter (meter):   0.075
enter the  Pipe length  (meter):      100
enter the Fluid density (kg/m^3):    999
enter the absolute viscosity beetwen Fluid and Pipe  (kg/(m.s)):    1.0e-3
enter the Pipe roughness (m):      0.00015
Re =

   1.6960e+05


Relative_Pipe_Roughness =

    0.0020


Friction_Factor =

    0.0233


Pressure_drop =

   79.5712
```

افت فشار بر اثر اصطکاک ۷۹,۵۷ متر.

منابع و ماخذ:

۱ – کاربرد شبکه عصبی مصنوعی برای محاسبه افت فشار لوله ها(مقاله رفرنس)

احمد رضا عظیمیان،دانشکده مهندسی مکانیک،دانشگاه صنعتی اصفهان

۲- کتاب شبکه های عصبی

هاگان

۳-کتاب شبکه های عصبی با MATLAB و #C

حمید رضا مقسمی،بهروز علیزاده سواره

۴ – کتاب مکانیک سیالات

رابرت دبیلیو.فاکس

توضیح:

برای بررسی صحت برنامه می توانید از سایتی که از لینک زیر قابل دسترسی است ورودی بدهید وخروجی ضریب اصطکاک را بدست آورید و با همان ورودی ها شبکه را تست کنید و با خروجی سایت مقایسه کنید.

http://www.advdelphisys.com/michael_maley/Moody_chart

هزینه استفاده 5 صلوات برای تعجیل در فرج صاحب الزمان(عج)