Задачи за седмица 6

(от 16.05.20г. до 22.05.20г.)

\* Под всяка задача пишете какво сте направили и ми върнете файла.

**Теоретична задача:**

Прочетете Лекции 6.1 и 6.2. Пишете ми. ако някоя част ви е затруднила с разбирането или имате допълнителни въпроси.

**Решение:**

Прочетена/Непрочетена

Въпроси:

Практически задачи

**Задача 1 (трансферни функции)**

Създайтe скрипт. който създава масива съдържащ елементите:

-5, -4.9, -4.8,…4.9, 5. Пресметнете всички функционални стойности на следните най-често използвани трансферни функции:

1) Линейна функция



2) Сигмоидна функция



3) Хиперболичен тангенс



Начертайте и разгледайте графиките на функциите.

Упътване:

За да начертаете графиките на функциите използвайте най-лесния вариант:

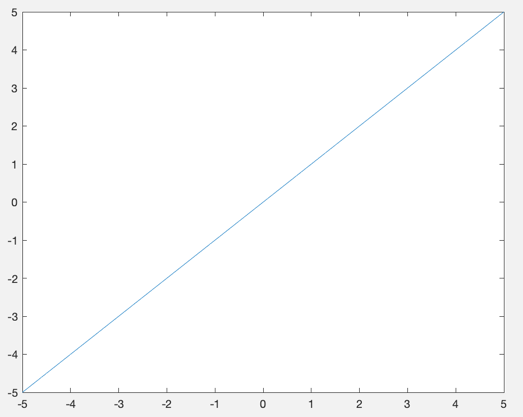
plot(x,y) – начертай графиката на функция представена с точките с координати (х,у)

Решение:

*>> x=-5:0.1:5 % Още един елементарен начин да създадем вектора от условието*

*>> y\_lin=x; % Линейната функция*

*>> plot(x,y\_lin);*



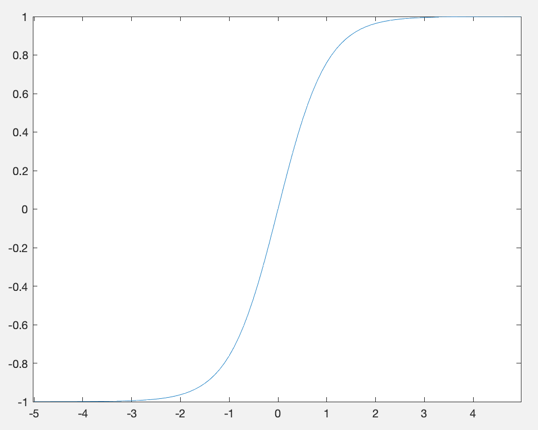
>> y\_sig=1./(1+exp(-x));

>> plot(x,y\_sig);

A close up of a mans face

Description automatically generated

>> y\_tanh=2./(1+exp(-2\*x))-1;



**Задача 2 (трансферни функции в един неврон)**

Повечето трансферни функции имат вградени аналози я Матлаб. Например в горната задача вместо да записваме сложен математически израз за тангенс хиперболичен можем да използваме y\_tan=tansig(x), за сигмоидалната: y\_sig=logsig(x), линейната y\_lin=purelin(x); Използвайте тези функции и създайте скрипт, в който:

1. Потребителя въвежда стойностите три входни променливи за един неврон;
2. Въвежда и вида на трансферната функция на неврона;
3. Данните от въведеното се изпращат към функцията neuron, която връща като резултат изходния сигнал по аксона на този неврон.
4. Оформете подходящо отговора.

Теглата и прага задайте вие

Упътване:

Въвеждането на стойностите на трите входни променливи и типа на трансферната функция не би трябвало да е проблем – подобни задачи сме решавали. Функцията neuron приема като формални параметри стойностите на входовете на неврона и вида на трансферната функция. В самата функция neuron с помощта на някаква условна конструкция, в зависимост от трансферната функция изчислете изходния сигнал на неврона,.

Решение:

prom\_x1="Задайте стойност на променлива x1: ";

x1=input(prom\_x1);

prom\_x2="Задайте стойност на променлива x2: ";

x2=input(prom\_x2);

prom\_x3="Задайте стойност на променлива x3: ";

x3=input(prom\_x3);

prom\_func ="Задайте функция (1-линейна, 2-хиперболичен тангенс, 3-сигмоид";

tr\_func=input(prom\_func);

disp(["Исходния сигнал на неврон е:", neuron(x1, x2, x3, tr\_func)]);

function res = neuron(x1,x2,x3,f)

w1=2

w2=1

w3=-1.5

b=-10

S=w1\*x1+w2\*x2+w3\*x3+b;

switch f

case 1, res=purelin(S);

case 2, res=tansig(S);

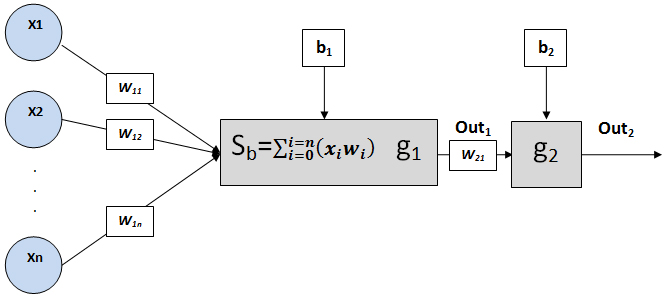
case 3, res=logsig(S);

end

end

**Задача 3 (невронна мрежа от два последователни неврона)**

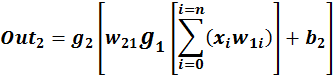
Разгледайте следната система от два последователни неврона:



За n=3, напишете скрипт, който генерира случайни тегла между 0 и 1 по дендритите на двата неврона (wij) и двата прага b1 и b2. При въведени от потребителя три стимула X1, X2, X3 скриптът да изчислява стойността на изходния сигнал по аксона на втория неврон. За трансферна функция g1 на първия неврон използвайте хиперболичен тангенс, а на втория (g2) - линейната f(x)=x.

Упътване:

За ваше улеснение ви напомням, че



Случайно число между нула и едно може да генерирате с функция rand. Например:

*>> w12=rand;*

Решение:

prom\_x1="Задайте стойност на променлива x1: ";

x1=input(prom\_x1);

prom\_x2="Задайте стойност на променлива x2: ";

x2=input(prom\_x2);

prom\_x3="Задайте стойност на променлива x3: ";

x3=input(prom\_x3);

disp(["Исходния сигнал на неврон е:", neuron(x1, x2, x3, tr\_func)]);

function res = neuron(x1,x2,x3)

w11=rand;

w12=rand;

w13=rand;

w21=rand;

b1=rand;

b2=rand;

res = purelin(w21\*tansig(x1\*w11+x2\*w12+x3\*w13+b1)+b2);

end

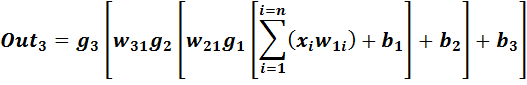
**Задача 4. (невронна мрежа от три последователни неврона)**

Разгледайте следната система от три последователни неврона:

За n=3, напишете скрипт, който генерира случайни тегла между 0 и 1 по дендритите на трите неврона (wij) и трите прага b1, b2 и b3. При въведени от потребителя три стимула X1, X2, X3 скриптът да изчислява стойността на изходния сигнал по аксона на втория неврон. За трансферни функция g1 и g2 на първия и втория неврон използвайте хиперболичен тангенс, а на третия (g3) - линейната f(x)=x.

Упътване:

За ваше улеснение ви напомням, че



Случайно число между нула и едно може да генерирате с функция rand. Например:

*>> w12=rand;*

Решение:

prom\_x1="Задайте стойност на променлива x1: ";

x1=input(prom\_x1);

prom\_x2="Задайте стойност на променлива x2: ";

x2=input(prom\_x2);

prom\_x3="Задайте стойност на променлива x3: ";

x3=input(prom\_x3);

disp(["Исходния сигнал на неврон е:", neuron(x1, x2, x3, tr\_func)]);

function res = neuron(x1,x2,x3)

w11=rand;

w12=rand;

w13=rand;

w21=rand;

w31=rand;

b1=rand;

b2=rand;

b3=rand;

res = purelin(w31\*tansig(w21\*tansig(x1\*w11+x2\*w12+x3\*w13+b1)+b2)+b3);

end

**Задача 5 (средноквадратична грешка на невронна мрежа)**

След обучението си невронна мрежа връща резултатите записани в променливата net при реални стойности real. Изчислете средноквадратичната грешка на мрежата, ако

|  |  |
| --- | --- |
| ***net – резултати на мрежата*** | ***Real – реални стойности*** |
| 10.8085863828891 | 10.7553427138778 |
| 5.70966478549097 | 5.63845506784783 |
| 3.90657177341168 | 3.81257942727674 |
| 2.98069403884092 | 2.90487438815756 |
| 2.41730705563571 | 2.35961303376578 |
| 2.03891224836208 | 1.99313567162069 |
| 1.76770945092586 | 1.72874032840061 |
| 1.56416461545517 | 1.52853248450260 |
| 1.40604551545233 | 1.37149004376560 |
| 1.27988661859547 | 1.24494652130155 |
| 1.17705517505335 | 1.14078598900045 |
| 1.09176196339855 | 1.05355506249551 |
| 1.01998011138416 | 0.979447775597791 |
| 0.958822415687762 | 0.915724203225467 |
| 0.906165235920286 | 0.860360104106184 |
| 0.860412082193075 | 0.811826734597106 |
| 0.820339887080858 | 0.768947508655587 |
| 0.784996081910992 | 0.730801839625468 |
| 0.753627913457085 | 0.696658938618352 |
| 0.725632805379808 | 0.665931192942462 |
| 0.700522801981962 | 0.638140671338345 |
| 0.677898645447379 | 0.612894629100808 |
| 0.657430574454017 | 0.589867307929378 |
| 0.638843896307448 | 0.568786217872667 |
| 0.621908004085185 | 0.549421662574795 |
| 0.606427916562809 | 0.531578645988862 |
| 0.592237690354491 | 0.515090551197376 |
| 0.579195238545752 | 0.499814154088149 |
| 0.567178217892521 | 0.485625653862106 |
| 0.556080736322397 | 0.472417486172490 |
| 0.545810696238810 | 0.460095744433260 |
| 0.536287635046565 | 0.448578077945625 |
| 0.527440957771042 | 0.437791966967931 |
| 0.519208481282387 | 0.427673298085654 |
| 0.511535227964379 | 0.418165180557816 |
| 0.504372420434465 | 0.409216957351155 |
| 0.497676639351694 | 0.400783374470368 |
| 0.491409114317442 | 0.392823879769562 |
| 0.485535124011488 | 0.385302028276605 |
| 0.480023486464647 | 0.378184975605915 |
| 0.474846124089017 | 0.371443044592176 |
| 0.469977691009395 | 0.365049353079229 |
| 0.465395252554111 | 0.358979493020003 |
| 0.461078008604023 | 0.353211252814504 |
| 0.457007053973151 | 0.347724376233605 |
| 0.453165170179859 | 0.342500352421422 |
| 0.449536643927896 | 0.337522232397328 |
| 0.446107108395229 | 0.332774468233848 |
| 0.442863404066867 | 0.328242771705345 |
| 0.439793456369052 | 0.323913989710037 |
| 0.436886167792822 | 0.319775994186631 |
| 0.434131322551378 | 0.315817584594569 |
| 0.431519502110045 | 0.312028401315386 |
| 0.429042010175006 | 0.308398848573647 |
| 0.426690805932047 | 0.304920025678034 |
| 0.424458444500411 | 0.301583665552921 |
| 0.422338023711138 | 0.298382079673774 |
| 0.420323136443542 | 0.295308108641221 |
| 0.418407827856681 | 0.292355077731020 |
| 0.416586556942310 | 0.289516756845336 |
| 0.414854161899871 | 0.286787324364678 |
| 0.413205828900025 | 0.284161334464473 |
| 0.411637063857690 | 0.281633687515000 |
| 0.410143666883062 | 0.279199603230303 |
| 0.408721709120891 | 0.276854596273061 |
| 0.407367511722068 | 0.274594454057125 |
| 0.406077626723712 | 0.272415216520269 |
| 0.404848819639523 | 0.270313157666145 |
| 0.403678053585597 | 0.268284768697581 |
| 0.402562474786602 | 0.266326742583404 |
| 0.401499399325704 | 0.264435959918866 |
| 0.400486301015381 | 0.262609475954844 |
| 0.399520800281286 | 0.260844508684643 |
| 0.398600653962154 | 0.259138427889444 |
| 0.397723745939469 | 0.257488745053460 |
| 0.396888078519711 | 0.255893104069517 |
| 0.396091764499812 | 0.254349272663525 |
| 0.395333019854478 | 0.252855134474273 |
| 0.394610156988806 | 0.251408681730553 |
| 0.393921578506827 | 0.250008008473956 |
| 0.393265771450584 | 0.248651304280503 |
| 0.392641301969198 | 0.247336848438861 |
| 0.392046810380969 | 0.246063004546962 |
| 0.391481006595929 | 0.244828215492302 |
| 0.390942665868008 | 0.243630998784800 |
| 0.390430624850199 | 0.242469942213319 |
| 0.389943777927787 | 0.241343699800553 |
| 0.389481073807258 | 0.240250988032066 |
| 0.389041512340522 | 0.239190582338740 |
| 0.388624141565674 | 0.238161313812448 |
| 0.388228054947911 | 0.237162066137508 |
| 0.387852388804465 | 0.236191772721616 |
| 0.387496319899975 | 0.235249414010951 |
| 0.387159063199268 | 0.234334014976414 |
| 0.386839869765595 | 0.233444642757898 |
| 0.386538024793975 | 0.232580404455469 |
| 0.386252845769188 | 0.231740445056952 |
| 0.385983680739774 | 0.230923945491876 |
| 0.385729906699443 | 0.230130120803381 |
| 0.385490928068516 | 0.229358218429438 |

Упътване:

1. Средноквадратичната грешка в Матлаб между съответните елементи на многофакторни структури (вектори. матрици). който са с една и съща размерност се изчислява с функцията immse . Синтаксисът на функцията е:

immse(X.Y) – връща средноквадратична грешка получена при отклонение на елементите на Х от съответстващите им елементи на У.

Например:

***>> error=immse(X.Y);***

1. Не въвеждайте ръчно вашите променливи net и real. Използвайте създаване на променлива чрез опцията New variable от основната лента и след това copy/past от таблицата в съответната променлива.

Решение:

Ans = 0.0130