Proceedings of the Jangjeon Mathematical Society
21 (2018), No. 4. pp. 719 - 731
www.jangjeon.or.kr
http://dx.doi.org/10.17777/pjms2018.21.4.719

Recurrent Neural Network Model For Bidirectional Associative Memory

Neeraj Sahu [1*] and Poonam Sinha[2]

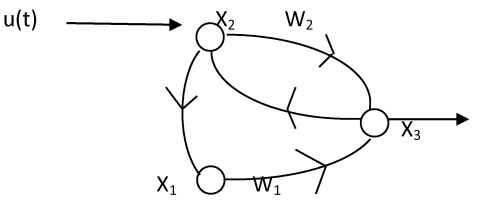
[1] Raffles University Neemrana Rajasthan
neeraj_maths1@yahoo.co.in
[2] S.M.S Govt. Science college Gwalior
poonamsinha_1968@yahoo.co.in

Abstract. Here we describe three neurons recurrent network model generate a limit cycle. This model describe recurrent neural networks of bidirectional associative memory (BAM) model for local asymptotical stability problem and some conditions which appear linear inequality relations.

Keywords: Neural network, Limit cycle, Nonlinear dynamics, Bidirectional associative memory. Asymptotic stability. Transcendental function.

1 Introduction

The recurrent network have been developed for solving a broad diversity of algebra and optimization problem [2],[6]-[8]. These type investigations have prostrated a strong basis for pole assignment using recurrent neural network. A pair of recurrent neural network is defined in [9] for on line pole assignment. This method is useful for solving two pair of matrix equation. This type recurrent neural networks useful to prove asymptotically stability. In stability analysis of neural network the general method is to translate its equilibrium to zero solution, then analyze the stability of the result convert system near the origin. B. Kosko [3] describe Bidirectional associative memory (BAM) and generalize a model for separate layer change two layer and kosko also verified the BAM is a structurally stable and globally stable in dynamical system[4],[5]. A new method for learning algorithm is call bidirectional excessive learning machine (B-ELM). This machine a few secreted nodes are not arbitrarily preferred, and algorithms decrease network output error to zero at an exceptionally before time learning stage [11]. Recently Sangwoo Moon, Hairong Qi [12] define a new method for search projection throughout optimization, is important structural threat and statistics independence known as hybrid dimensionality reduction method. In this we define a new model for recurrent neural networks through the translacendental function. Here we well known that networks demand and set of library pattern for the system of asymptotical stable. We also trained the model by gradient descent algorithm.



In above diagram u (t) indicate the input and y (t) indicates the output of the given network. In this network is explaining differential equation of the system.

$$\dot{X}_1 = -X_1 + \tan h\,(X_2\,(t)) \tag{1}$$

$$\dot{X}_2 = -X_2 + \tan h\,(X_3\,(t)) \tag{2}$$

$$\dot{X}_3 = -X_3 + W_1 \tan h\,(X_1\,(t)) + W_2 \tan h\,(X_2\,(t)) \tag{3}$$

$$Y\,(t) = \tan h(X_3\,(t))$$

The network parameter weights w_1, w_2 and X (t) $\in R^n$ n=1,2,3 is the state . Here X_1, X_2 and X_3 neurons describe (X_{11}, X_{12}, ……X_{1i}),(X_{21}, X_{22},….. X_{2j}), (X_{31}, X_{32},… X_{3k}) layers of neurons respectively.

The system of differential equation generate a limit cycle when X=0 collective with bounded the set of solution. When we determine the location of pole by the linearized system, we find influence amplitude and frequency of the equivalent Limit cycle.

We also linearization equation (1) to(3) at X =0

$$\dot{z}_1 = -z_1 + z_2$$

$$\dot{z}_2 = -z_2 + z_3$$

$$\dot{z}_3 = -z_3 + W_1 z_1 + W_2 z_2 \tag{4}$$

Let

$$A = \begin{bmatrix} -1 & 1 & 0 \\ 0 & -1 & 1 \\ W_1 & W_2 & -1 \end{bmatrix}$$

Here A -λI = 0 define the roots of the characteristic equation

$$(-1- \lambda)[(1+ \lambda)^2 - W_2\,] - 1[0 - W_1\,] = 0$$

Let $f(\lambda) : = -[\lambda^3 + 3\,\lambda^2 + 3\,\lambda + 1] + W_2(1+ \lambda) + W_1$ (5)

When we differentiating (5)

$$f'(\lambda) : = 3\lambda^2 + 6\,\lambda + 3 - W_2 = 0 \tag{6}$$

The roots of characteristic equation (6) is

$$\lambda_1 = -1 + \frac{\sqrt{W_2}}{\sqrt{3}} \quad , \quad \lambda_2 = -1 - \frac{\sqrt{W_2}}{\sqrt{3}} \tag{7}$$

Ruiz, Owens and Townley [1] define a couple of complex conjugate root for f (λ) in necessary and sufficient condition f (λ_1) f (λ_2) > 0. Then we find

$$f(\lambda_1) f(\lambda_2) = 1 - \frac{W_2}{3} > 0 \tag{8}$$

This equation (8) produces limit cycle on the origin. In the sight of equation (1) and (2) entail that $X_1 = X_2$

Let $X_1 = X_2 = X$ and $W_1 = W_2 = W$

2. A BAM'S EQUILIBRIUM FOR LOCAL ASYMPTOTICAL STABILITY RECURRENT NEURAL NETWORK

Here continuous BAM recurrent neural network structure is pair of nonlinear equation in dynamical system and this model is given by modifying (2) and (3). Now the new set of equation is given below

$$\dot{X}(t) = -X(t) + \tanh X_3(t) + I \tag{9}$$

$$\dot{X}_3(t) = -X_3(t) + 2W \tanh X(t) + J \tag{10}$$

Where $X(t)=(x_1(t),\text{-----}x_N(t))^T \in R^N$, $X_3(t)=(x_{31}(t),\text{-----}x_{3p}(t))^T \in R^N$ Differential $I=(I_1,\text{----------}I_N)^T$ and $J=(J_1\text{---------}J_P)^T$ stand for constant Here layers of BAM W are NxP and PxN outer input of the real matrix with opening representing the involving weights of the formal neurons between the two layers.

Theorem 1:- The (X^c,X_3^c) is the equilibrium and stable asymptotically state of the equation (9) to(10) if they satisfy the following conditions.

$$X^c - \tanh X_3^c = I \tag{11}$$

$$X_3^c - 2W\tanh X^c = J \tag{12}$$

$$\sum_{j=1}^{p} \left| \frac{\partial \tanh X_{3j}}{\partial X_{3j}} \right|_{X_s = x_{sj}^c} < 1 \quad i = 1 \ldots\ldots N \tag{13}$$

$$\sum_{j=1}^{N} \left| \frac{2W_{ij}\,\partial \tanh X_j}{\partial X_j} \right|_{X_j = x_j^c} < 1 \quad i = 1 \ldots\ldots P \tag{14}$$

Proof :- If (11) and (12) are satisfied then

$$\frac{dX^c}{dt} = -X^c + \tanh X_3^c + I = 0 \tag{15}$$

$$\frac{dX_3^c}{dt} = -X_3^c + 2W\tanh X^c + J = 0 \tag{16}$$

Which means that (X^c,X_3^c) is an equilibrium state couple. We expand the right Side of (9)-(10) in Taylor series concerning (X^c,X_3^c) with examine the local point of resulting in asymptotical stability.

$$\frac{dX(t)}{dt} = -X^c + \tanh X_3^c + I - (X - X^c)\Psi_1[\tanh X_3 - \tanh X_3^c]$$

$$+0(\|X - X^c\|^2 + \|\tanh X_3 - \tanh X_3^c\|^2)$$

$$\frac{dX_3(t)}{dt} = -X_3^c + 2W\tanh X^c + J - (X_3 - X_3^c) + 2W\Psi_2[\tanh X - \tanh X^c]$$

$$+0(\|X - X^c\|^2 + \|\tanh X - \tanh X^c\|^2)$$

Where

$$\Psi_1 = \operatorname{dig}\left[\frac{\partial f(X_1)}{\partial X_1}\Big|X_1 = X_1^c \ldots\ldots \ldots \frac{\partial X_N}{\partial X_N}\Big|X_N = X_N^c\right]$$

$$\Psi_2 = \operatorname{dig}\left[\frac{\partial f(X_{31})}{\partial X_{31}}\Big|_{X_{31}=X_{31}^c} \ldots\ldots \frac{\partial X_{3p}}{\partial X_{3p}}\Big|_{X_{3p}=X_{3p}^c}\right]$$

$0(\|X-X^c\|^2+\|\tanh X-\tanh X^c\|^2)$ are the expressions higher and second order which are disappear small as $(X,X_3)\rightarrow(X^c,X_3^c)$ Merging with situation (15)-(16) we can write

$$\begin{bmatrix}\frac{dX}{dt}\\ \frac{dX_3}{dt}\end{bmatrix}=\begin{bmatrix}-1 & \Psi_1\\ 2W\Psi_2 & -1\end{bmatrix}\begin{bmatrix}X-X^c\\ X_3-X_3^c\end{bmatrix}=J\begin{bmatrix}X-X^c\\ X_3-X_3^c\end{bmatrix} \tag{17}$$

Where J is square of value (N+P) for f(x)and f(y) Jacobian calculate at (X^c,X_3^c);

$$J = \begin{bmatrix}-1 & \Psi_1\\ 2W\Psi_2 & -1\end{bmatrix} \tag{18}$$

In equation (17) we also know that (X,X_3^c) is nearly asymptotical stable if the eigen value of J are all in the left plane according to well know Gersgorin Theorem for eigen values localization any egien value of λ of J is contained in the union of the N+P disk of complex λ plane

$$|\lambda - J_{ii}| \le \sum_{j=1}^{N+P}|J_{ii}| \qquad i = 1 \ldots\ldots\ldots\ldots\ldots\ldots N+P \tag{19}$$

from equation (19) it clearly describe for condition J in stable matrix $J_{ii} < 0$ and $|J_{ii}| \le \sum_{j=1}^{N+P}|J_{ii}|$ here first situation is satisfy as being positive. The second condition of the definite entries of J can be written as

$$\sum_{j=1}^{p}\left|\frac{\partial \tanh X_{3j}}{\partial X_{3j}}\Big|_{X_3=X_{3j}^c}\right| < 1 \qquad i = 1 \ldots\ldots N$$

$$\sum_{j=1}^{N}\left|\frac{2W_{ij}\,\partial \tanh X_j}{\partial X_j}\Big|_{X_j=X_j^c}\right| < 1 \qquad i = 1 \ldots\ldots P$$

Which are in equalities (13) and (14)

Ostrowski shows the localization theorem for another eigen values then we get

Theorem 2-:- The (X^c,X_3^c) is the equilibrium and asymptotically stable of the (9) and (10). We satisfy (11)-(12) and the following conditions for any α

$$\left[\sum_{j=1}^{p}\left|\frac{\partial \tanh X_{3j}}{\partial X_{3j}}\Big|_{X_3=X_{3j}^c}\right|\right]^{\alpha}\left[\left|\frac{\partial \tanh X_j}{\partial X_i}\Big|_{X_i=X_i^c}\right|\sum_{j=1}^{N}2|W_{ij}|\right]^{1-\alpha} < 1$$

i = 1……N (20)

$$\left[\sum_{j=1}^{p}\left|\frac{\partial \tanh X_j}{\partial X_j}\right|_{X_j=X_j^c}\right]^{\alpha}\left[\left|\frac{\partial \tanh X_{3i}}{\partial X_{3i}}\right|_{X_{3i}=X_{3i}^c}\left|\sum_{j=1}^{p}|1|\right|\right]^{1-\alpha}<1$$

i = 1……P (21)

We can proof of theorem2 is similar proof of theorem1.

When we are transforming of matrix J to a new matrix with equal eigenvalue. The possibility of the eigenvalue is more exact to possible to localization. When we generalization the examination [10], we can find following result.

Theorem-3 The (X^c,X_3^c) is equilibrium for asymptotically stable of (9)-(10) it W, satisfy (11)-(12) and

$$\left[\left|\frac{\partial \tanh X_i}{\partial X_i}\right|_{X_i=X_i^c}\right]^{\frac{1}{2}}\sum_{j=1}^{p}|1|\left[\left|\frac{\partial \tanh X_{3j}}{\partial X_{3j}}\right|_{X_{3j}=X_{3j}^c}\right]^{\frac{1}{2}}<1 \quad (22)$$

i = 1……N

$$\left[\left|\frac{\partial \tanh X_{3i}}{\partial X_{3i}}\right|_{X_{3i}=X_{3i}^c}\right]^{\frac{1}{2}}\sum_{j=1}^{N}|2W_{ij}|\left[\left|\frac{\partial \tanh X_j}{\partial X_j}\right|_{X_j=X_j^c}\right]^{\frac{1}{2}}<1 \quad (23)$$

i = 1…… P

Proof - Let

$$J=\begin{bmatrix}-\Psi_2^{-1} & 1\\ 2W & \Psi_1^{-1}\end{bmatrix}A=\begin{bmatrix}\Psi_1 & \\ & \Psi_2\end{bmatrix}$$

Here we explain that $J = A\Psi$ let λ is a random eigen value of J equivalent to an eigen vector Z,

$$J Z = A \Psi Z = \lambda Z$$

It is not difficulty to demonstrate that λ is also an value of $\Psi_1^{\frac{1}{2}} A \Psi_2^{\frac{1}{2}}$ and

$$\Psi_1^{\frac{1}{2}} A \Psi_2^{\frac{1}{2}}=\begin{bmatrix}-1 & \Psi_2^{\frac{1}{2}}\Psi_1^{\frac{1}{2}}\\ \Psi_1^{\frac{1}{2}} 2W \Psi_2^{\frac{1}{2}} & -1\end{bmatrix}$$

The following inequality satisfy Gersgorin theorem λ of the given condition

$$(\lambda+1)\leq\left[\left|\frac{\partial \tanh X_i}{\partial X_i}\right|_{X_i=X_i^c}\right]^{\frac{1}{2}}\sum_{j=1}^{p}|1|\left[\left|\frac{\partial \tanh X_{3j}}{\partial X_{3j}}\right|_{X_{3j}=X_{3j}^c}\right]^{\frac{1}{2}}$$

$$(\lambda+1)\leq\left[\left|\frac{\partial \tanh X_{3i}}{\partial X_{3i}}\right|_{X_{3i}=X_{3i}^c}\right]^{\frac{1}{2}}\sum_{j=1}^{N}|2W_{ij}|\left[\left|\frac{\partial \tanh X_j}{\partial X_j}\right|_{X_j=X_j^c}\right]^{\frac{1}{2}}$$

A condition for (X^c, X_3^c) shows equilibrium for asymptotically stable i eigen value of λ when situate J of the left side plane where following condition are satisfied

$$\left[\left|\frac{\partial \tanh X_i}{\partial X_i}\right|_{X_i=X_i^c}\right]^{\frac{1}{2}} \sum_{j=1}^{p} |1| \left[\left|\frac{\partial \tanh X_{3j}}{\partial X_{3j}}\right|_{X_{3j}=X_{3j}^c}\right]^{\frac{1}{2}} < 1$$

$$i = 1 \ldots\ldots\ldots\ldots\ldots \quad N$$

$$\left[\left|\frac{\partial \tanh X_{3i}}{\partial X_{3i}}\right|_{X_{3i}=X_{3i}^c}\right]^{\frac{1}{2}} \sum_{j=1}^{N} |2W_{ij}| \left[\left|\frac{\partial \tanh X_j}{\partial X_j}\right|_{X_j=X_j^c}\right]^{\frac{1}{2}} < 1$$

$$i = 1 \ldots\ldots\ldots\ldots\ldots \quad P$$

Hence proved

3. Training part of the structure

The structure of recurrent neural network describe learning capability of the equation (1) to (3), here beginning time at $t = t_0$. Where we fed as input sinh (t), tanh(t) is a periodic signal for the network. Where beginning time $t_0 = 0.0$ second up to a time $t = 1.8$ second, in this network (1) - (3) change its parameters W_1 , W_2. When the network is trained the output tanh(t) is fed back so as to replace the initial period signal sinh (t), tanh(t).

In this recurrent neural network explain by the system

$$\begin{aligned}
\dot{X}_1 &= -X_1 + \tanh(X_2(t)) \\
\dot{X}_2 &= -X_2 + u^*(t) \\
\dot{X}_3 &= -X_3 + W_1 \tan h(X_1(t)) + W_2 \tan h(X_2(t))
\end{aligned} \tag{24}$$

with X (t) $\in R^3$, X(0) $\neq 0$ and u(t) given by

$$U^*(t) = \begin{cases} u(t) = \sinh(t) \text{ or } \sinh(X_3(t)) & 0 \le t < t_0 \\ y(t) = \tanh(X_3(t) & t_0 \le t \end{cases} \tag{25}$$

When $t_0 = 1.8$ second, the weight W_1 , W_2 are update for minimizing energy function

$$E(t) = \tfrac{1}{2} [\, y(t) - u(t) \,]^2 \tag{26}$$

A. Ruiz and several authors [1] use gradient descent algorithm explain the equation

$$(\partial E / \partial W_1) = (y - u)\, [(1 - \tanh^2(X_3(t)) - \cosh(X_3(t)) \text{or } \cosh(t)]\, \Delta_d \tanh^2(X_1(t))$$

$$(\partial E / \partial W_2) = (y - u)\, [(1 - \tanh^2(X_3(t)) - \cosh(X_3(t)) \text{or } \cosh(t)]\, \Delta_d \tanh^2(X_2(t))$$

The differential operator Δ_d was incorrect and the correct one is given by $\Delta_d := (1+d/dt)^{-1}$ with the auxiliary variables V_1, V_2 defined as $V_1 := (\partial E / \partial W_1)$, $V_2 := (\partial E / \partial W_2)$, then we have

$$\dot{V}_1 = -V_1 + \tanh(X_1(t))$$
$$\dot{V}_2 = -V_2 + \tanh(X_2(t)) \qquad (27)$$

and updates the rules for W_1, W_2 by A. Ruiz, and several autors[1].

$$\dot{W}_1 = -\xi\,(\partial E / \partial W_1) = -\xi\,(y-u)[(1-\tanh^2(X_3(t)) - \cosh(X_3(t)) \text{ or } \cosh(t)]\; V_1 \quad \xi > 0$$

$$\dot{W}_2 = -\xi\,(\partial E / \partial W_2) = -\xi\,(y-u)\,[(1-\tanh^2(X_3(t)) - \cosh(X_3(t)) \text{or } \cosh(t)\,]\; V_2 \qquad (28)$$

With $W(0) = (W_1, W_2)'(0) =: W_0$, $V(0) = (V_1, V_2)'(0) =: V_0$ arbitrary.

4. CONCLUSION

Here we found three node layered model generate a limit cycle for some condition. We have marked out some attribute for asymptotically stable in equilibrium In this type vector couple for L library memories conditions to seem L (N + P) linear equations. This L (N + P) equation describe in (N+P) + (2NP) piece wise-linear inequalities in unknowns entries of W. This data and results is useful in the hardware implementations of recurrent neural network of the BAM. Network is shown to be trained within 1.8 second of the given signal.

5. REFERENCES

[1] A. Ruiz, D. H. Owens, and S. Townley "Existence, learning, and replication of periodic motions in recurrent neural network". IEEE Trans. Neural Network.Vol. 9. no 4, pp 651- 661. Jul,1998.

[2] A. Cichoki and R. Unbehauen, "Neural networks for solving systems of linear equation and related problem," IEEE Trans. Comput Syst., Vol. 39, pp. 124-138, 1992.

[3] B. Kosko, "Bidirectional associative memories," IEEE Trans. System Men Cybern., Vol. SMC-18, No.1, pp.49-60,1988.

[4] B. Kosko, "Unsupervised learning is noise," IEEE Trans. Neural Networks, Vol. l, No1, pp.44-57, Mar,1990.

[5] B. Kosko, "Structural stability of unsupervised learning in feedback neural networks," IEEE Trans. Automatic Control, Vol. 36, No7, pp.785-792, July,1991.

[6] D.W. Tank, and J.J.Hopfield, "Simple neural optimization networks: An A/D converter, signal decision circuit, and a linear programming circuit," IEEE Trans. Circuit syst., Vol.CS-33, pp.533-541,1986.

[7] J. Wang, "Recurrent neural networks for solving linear matrix equation," Comput. Math. With Applicat., Vol.26, No.9 pp.23-34,1993.

[8] L.X. Wang, and J.M. Mendel, "Three dimensional structured network for matrix equation solving," IEEE Trans. Comput, Vol 40, pp.1337-1345,1991.

[9] J.Wang.and.G Wu,"Recurrent neural networks for synthesizing linear control system via pole assignment," Int. J. Syst. Sci., Vol 26, No.12, pp.2369-2382, 1985.

[10] A. Tiya, and,Y Abu-Mostafa. "A method for the associative storage of analog vectors," in advance in neural information processing system II, (D.S. Touretzky ed.) Morgan Kaufmann Publishers Inc., pp 590-595,1990.

[11] Y .Yang, Y. Wang X. Yuan "Bidirectional Extreme Learning Machine for Regression Problem and Its Learning Effectiveness," IEEE Trans. Neural Networks, Vol. 23.Issue 9. pp.1498 – 1505 Sept. 2012.

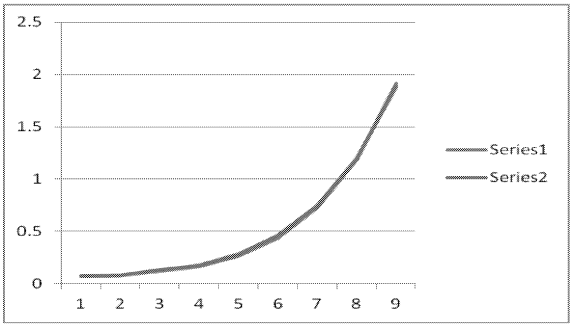
[12] Sangwoo Moon Hairong Qi "Hybrid Dimensionality Reduction Method Based on Support Vector Machine and Independent Component Analysis," IEEE Trans. Neural Networks, Volume: 23. Issue 5, pp.749 – 761 May 2012.

Appendix

Table -1

March of ξ = .25, $X_1(t)$, $X_2(t)$, $X_3(t)$, V_1, V_2, for $\dot{W}_1$, $\dot{W}_2$ in the beginning time t = 0.0 second to 1.8 second with the beginning weight $W_1 = 0.5$ and condition $W_1 = W_2$, and $\tanh(X_3(t)) = \sinh(t)$

t	$X_1(t)$	$X_2(t)$	$X_3(t)$	V_1	V_2	$\dot{W}_1$	$\dot{W}_2$
0.2	-1.2195	- 0.8187	-0.63186	-1.1347	-1.2195	0.0718	0.0771
0.4	-1.2426	-0.6703	-0.5288	-1.1302	-1.2426	0.0798	0.877
0.6	-1.2499	-0.5488	-0.4990	-1.1286	-1.2499	0.1233	0.1366
0.8	-1.2437	-0.4493	-0.3645	-1.1299	-1.2437	0.1604	0.1766
1.0	-1.2280	-0.3678	-0.3045	-1.1329	-1.2280	0.2624	0.2845
1.2	-1.2068	-0.3012	-0.2578	-1.1372	-1.2068	0.4377	0.4645
1.4	-1.1832	-0.2465	-0.2210	-1.1420	-1.1832	0.7257	0.75198
1.6	-1.1595	-0.2019	-0.1929	-1.1469	-1.1595	1.1873	1.2004
1.8	-1.1369	-0.1653	-0.1712	-1.1517	-1.1363	1.9137	1.889

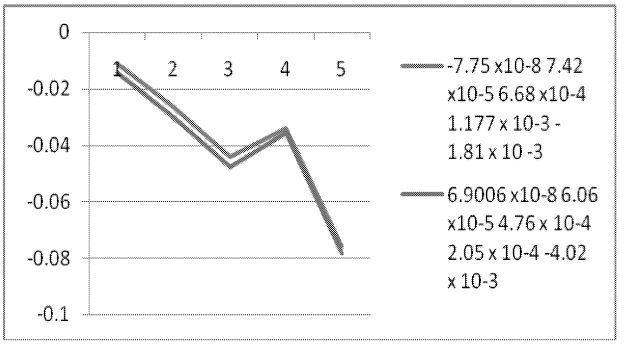


$\dot{W}_1$ and $\dot{W}_2$ data shown by the above diagram of the line 1 and line 2.

Table-2

March of ξ = .25, $X_1(t), X_2(t), X_3(t), V_1, V_2$, for $\dot{W}_1$, $\dot{W}_2$ in the beginning time t = 0.0 second to 1.8 second with the beginning weight $W_1 = 0.5$ and condition $W_1 = W_2$, and $\tanh(X_3(t)) = \sinh(X_3(t))$

$X_1(t)$	$X_2(t)$	$X_3(t)$	V_1	V_2	$\dot{W}_1$	$\dot{W}_2$
0.0	0.1	0.0448	1.0	-0.8904	$-7.75/10^8$	$6.900/10^8$
0.2	0.3	0.2213	-0.763	7-0.6238	$7.42/10^5$	$6.06/10^5$
0.4	0.5	0.3864	-0.475	-0.3243	$6.68/10^4$	$4.76/10^4$
0.6	0.7	0.5322	- 0.1745	-0.0304	$1.177/10^3$	$2.05/10^4$
0.8	0.9	0.6540	0.1049	0.2292	$-1.81/10^3$	$-4.02/10^3$
1.0	1.1	0.7506	0.3415	0.4411	-0.0111	-0.0144
1.2	1.3	0.8238	0.5285	0.6042	-0.0262	-0.0300
1.4	1.5	0.8775	0.6691	0.7243	-0.0439	-0.0475
1.6	1.7	0.801	0.7710	0.8103	-0.0338	-0.03553
1.8	1.9	0.9425	0.8432	0.8705	-0.07570	-0.07815



$\dot{W}_1$ and $\dot{W}_2$ data shown by the above diagram of the blue line 1 and red line 2.

COMPUTER SIMULATION

```
import java.io.*;
class Test
{
public static void main(String s[ ] ) throws IOException
        {
             float w1,w2,v1,v2,x1,x3,yt,t,x2,w;
             float neeta;
             int ut;
             w =0.5f;
             neeta = 0.25f;
BufferedReader obj = new BufferedReader(new InputStreamReader(System.in));
 System.out.println("|**********************|");
            System.out.println("|   1. sinh(t)                        |");
             System.out.println("|   2. cosh(t)                        |");
             System.out.println("|   3. tan(t)                         |");
             System.out.println("|**********************|");
             System.out.println("Enter The Value For ut from 1 to 3 :- ");
             ut =Integer.parseInt(obj.readLine());
             System.out.println("ut is :- "+ut);
            int flag=0;
             switch(ut)
             {
                    case 1:
                           flag=1;
                           break;
                    case 2:
                           flag=2;
                           break;
                    case 3:
                           flag=3;
                           break;
             }
             t=0.0f;
             float d1,d2,d3,d4,d5,d6,d7,d8,d9;
             if(flag==1)
             {
                while(t<=2.2)
                {
                     System.out.println();
                     x2=(float) (Math.sinh(t) - Math.cosh(t));
                     System.out.print("X2="+x2);
                     x1 = (float)(Math.tanh(x2) - (1-(Math.tanh(x2) * Math.tanh(x2))));
                     v2=x1;
                     System.out.print("  X1="+x1);
                     //************
                          d1=(float)Math.tanh(x1);
                          d2=(float)(1- (d1*d1));
                          v1 = d1-d2;
                         System.out.print(" V1="+v1);
                     //*****************
                     System.out.print("     V2="+v2);
```

```
                    // ******************************
                        d5 =(float) (Math.tanh(x1)+Math.tanh(x2));
                        d6 =(float) (( Math.tanh(x2)*Math.tanh(x2) ) - (
Math.tanh(x1)*Math.tanh(x1)));
                        x3 =(float)((d5-d6)*w);
                        System.out.print(" X3="+x3);
                    //*******************************
                        d7 = (float)(Math.tanh(x3) - Math.sinh(t));
                        d8 = (float)(1 - ((Math.tanh(x3)*Math.tanh(x3) )));
                        d9 = (float) (d8- Math.cosh(t));
                        w1 = (float)((-neeta)*d7*d9*v1);
                        System.out.print(" W1="+w1);
                    // *******************************
                //*******************************
                        w2 = (float)((-neeta)*d7*d9*v2);
                        System.out.print(" W2="+w2);
                    // *******************************
                     t=t+0.2f;
                }
            }
            else if(flag==2)
            {
                while(t<=2.2)
                {
                     System.out.println();
                    x2 = (float)(Math.cosh(t) -  Math.sinh(t));
                     System.out.print("X2="+x2);
                      x1 =(float) (Math.tanh(x2) - (1-(Math.tanh(x2) * Math.tanh(x2))));
                      v2 = x1;
                     System.out.print(" X1="+x1);
                   //************
                        d1=(float)Math.tanh(x1);
                        d2=(float)(1- (d1*d1));
                        v1 = d1-d2;
                        System.out.print(" V1="+v1);
                    //*****************
                     System.out.print(" V2="+v2);
                  // *******************************
                        d5 =(float) (Math.tanh(x1)+Math.tanh(x2));
                        d6 =(float) (( Math.tanh(x2)*Math.tanh(x2) ) - (
Math.tanh(x1)*Math.tanh(x1)));
                        x3 =(float)((d5-d6)*w);
                        System.out.print(" X3="+x3);
                    //*******************************
                    //*******************************
                        d7 = (float)(Math.tanh(x3) - Math.cosh(t));
                        d8 = (float)(1 - ((Math.tanh(x3)*Math.tanh(x3) )));
                        d9 = (float) (d8- Math.sinh(t));
                        w1 = (float)((-neeta)*d7*d9*v1);
                        System.out.print(" W1="+w1);
                    // *******************************
                //*******************************
```

```
                    w2 = (float)((-neeta)*d7*d9*v2);
                    System.out.print(" W2="+w2);
                // ******************************
                  t=t+0.2f;
              }
          }
          else if(flag==3)
          {
              while(t<=2.2)
              {
                  System.out.println();
                  x2 =(float) (Math.tanh(t) -  (1-(Math.tanh(t)*Math.tanh(t))));
                  System.out.print(" X2="+x2);
                  x1 = (float)(Math.tanh(x2) - (1-(Math.tanh(x2) * Math.tanh(x2))));
                  v2=x1;
                  System.out.print(" X1="+x1);
                  //************
                    d1=(float)Math.tanh(x1);
                    d2=(float)(1- (d1*d1));
                    v1 = d1-d2;
                    System.out.print(" V1="+v1);
                 //*****************
                    System.out.print(" V2="+v2);
                 // ******************************
                    d5 =(float) (Math.tanh(x1)+Math.tanh(x2));
                    d6 =(float) (( Math.tanh(x2)*Math.tanh(x2) ) - (
Math.tanh(x1)*Math.tanh(x1)));
                    x3 =(float)((d5-d6)*w);
                    System.out.print(" X3="+x3);
                 //*********************************
                 //*********************************
                    d7 = (float)(Math.tanh(x3) - Math.tanh(t));
                    d8 = (float)(1 - ((Math.tanh(x3)*Math.tanh(x3) )));
                    d9 = (float) (d8- (1-(Math.tanh(t)*Math.tanh(t))));
                    w1 = (float)((-neeta)*d7*d9*v1);
                    System.out.print(" W1="+w1);
                // ********************************
               //*********************************
                    w2 = (float)((-neeta)*d7*d9*v2);
                    System.out.print(" W2="+w2);
                // ******************************
                   t=t+0.2f;
              }
          }
          else
           {
               System.out.println("****");
           }

      }

}
```