A New Weight Initialization using Statistically Resilient Method and Moore-Penrose Inverse Method for SFANN

Apeksha Mittal, Amit Prakash Singh and Pravin Chandra

University School of Information and Communication Technology, Guru Gobind Singh Indraprastha University, Sector 16C, Dwarka, Delhi(India)-110078

Abstract.
Objectives: Weight Initialization is an important parameter in training Single Hidden Layer Feedforward Artificial Neural Networks (SFANN) and is important for faster convergence. In this paper a Statistically Resilient Moore-Penrose Inverse based Weight Initialization Technique (SRINWIT) for SFANN is proposed to improve the speed of training in SFANN. **Methods/Statistical analysis**: The proposed technique SRINWIT utilizes statistically resilient weight initialization method NWIT for input to hidden weight initialization and Moore-Penrose Inverse method for hidden to output weight initialization. The technique NWIT ensures that the inputs to a neuron are given their own region of activation function, at the same time utilizes the complete (useful) range of activation function. The Moore-Penrose Inverse Method ensures that the weight space is as close as possible to the optimal solution before training.
Findings: The performance of proposed weight initialization technique SRINWIT is compared to Random Weight Initialization Technique (RWIT) and Inverse Weight Initialization Technique (INVWIT) for 10 function approximation problems. Mean of the mean squared error (MMSE) and median of the MSE (MEDIAN) are compared for both training and testing data for all the three weight initialization techniques RWIT, INVWIT and SRINWIT. The results show that the performance of SRINWIT is superior as compared to RWIT and INVWIT for at least these 10 function approximation task.
Application/ Improvements: The application of this technique can be tested to enhance the speed of training in the real world regression problems.

Keywords: Weight Initialization, Artificial Neural Network, Moore-Penrose Inverse, Feedforward Networks

I. INTRODUCTION

Single hidden layer Feedforward Artificial Neural Networks (SFANNs) are shown to be universal approximators[1, 2, 3] i.e. they have capability to approximate any continuous function. Schematic diagram for Single hidden layer Feedforward Artificial Neural Network (SFANN) is given in Fig.1.
The x_j's represents input to the network where $j \in \{1,..N_I\}$ where N_I is the no. of input connected to the network. w_{ij} represent weights from jth input to ith hidden node, $j \in \{1,\dots,N_I\}$ and $i \in \{1,\dots,N_H\}$, N_H is the total no. of hidden nodes. θ_i represent ith hidden threshold, $i \in \{1,\dots,N_H\}$. β_i denote weights between hidden & output nodes, $i \in \{1,\dots,N_H\}$ and γ denotes output threshold.
Input to ith hidden node is given by (1):

$$n_i = \sum_{i=1}^{N_I} w_{ij}x_j + \theta_i \quad (1$$

The activation function used are sigmoidal as they are finite, bounded and differentiable[4, 5]. Hyperbolic tangent function is used at hidden node as activation function.

$$\sigma(x) = \tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \quad (2)$$

The output of *ith* hidden node is given as

$$h_i = \sigma(n_i) \quad (3)$$

Output of the network (ignoring the bias term γ) is given by

$$y = \sum_{i=1}^{N_H} \beta_i h_i \quad (4)$$

The threshold function at the output node in this work is linear.
Weight initialization significantly affects the speed of training in SFANN[6-12]. A proper method of weight initialization is required for better convergence in SFANN. Conventionally, small uniform random values are used to initialize all the weights from input to hidden layer and hidden to output layer, to break symmetry so that after training process, the weights can achieve different values[13, 14]. In this paper input to hidden weight are first initialized with small random values and hidden to output weights are initialized using Moore-Penrose inverse. This

technique is named INVWIT. A similar technique was proposed by Yam and Chow[9]. INVWIT is compared to conventional Random Weight Initialization Technique (RWIT) for 10 function approximation task and is expected to give better results than RWIT.

A statistically resilient method (NWIT) for initializing weights was proposed by Mittal et al.[12] ensuring inputs to neurons are in the useful (active) range of activation function is in the active region of activation function and utilizes the complete (useful) range of the activation function. In this paper a new Statistically Resilient Moore-Penrose Inverse based weight initialization technique (SRINWIT) for SFANN is proposed, in which input to hidden weights are initialized using NWIT and hidden to output weights are initialized using INVWIT. SRINWIT is expected to give better results than both RWIT and INVWIT.

II. INVERSE WEIGHT INITIALIZATION TECHNIQUE (INVWIT)

In INVWIT, small random values are used to initialize all input to hidden weights. The hidden to output weights are computed using Moore-Penrose Inverse as explained.

Rewriting Eq.4 as:

$$Y = H\beta \quad (5)$$

where H is vector that gives output of hidden nodes as:

$$H = \begin{bmatrix} \sigma(w_{11}x_1 + \theta_1) & \dots & \sigma(w_{H1} + \theta_H) \\ \vdots & & \vdots \\ \sigma(w_{1I}x_I + \theta_1) & \dots & \sigma(w_{HI} + \theta_H) \end{bmatrix} \quad (6)$$

and β is $H \times 1$ hidden to output weight matrix as:

$$\beta = \begin{bmatrix} \beta_1 \\ \vdots \\ \beta_H \end{bmatrix} \quad (7)$$

For the purpose of initialization of weights, σ is assumed to be linear. Thus hidden to output weight matrix β is computed as in Eq.8

$$\beta = H^{\dagger}Y \quad (8)$$

where $H^{\dagger}$ denotes the Moore-Penrose inverse of matrix H.

III. STATISTICALLY RESILIENT INVERSE WEIGHT INITIALIZATION TECHNIQUE (SRINWIT)

In this paper a Statistically Resilient Moore-Penrose Inverse based Weight Initialization Technique (SRINWIT) is proposed. In SRINWIT, all input to hidden weights are initialized using technique similar to NWIT proposed by Mittal et al.[12]. Thus all input to hidden weights are trained such that the all input to the neuron lies in the useful range of activation function. It also ensures that the complete (useful) range of activation function is used. Algorithm NWIT was established to be better than conventional Random Weight Initialization for function approximation problems[12]. A constant value G is defined so that for range $(-G, G)$

$$(\sigma(x))' \geq 0.05$$

The value of G is 2.1783. Thus the weights are initialized in the interval $(-G, G)$ such that active region of activation function is well utilized. This range $(-G, G)$ is then divided into H parts and all the weights to nth hidden are assigned values from the nth part of interval $(-G, G)$. This method ensures that a separate region of activation function is assigned to every hidden node.

Hidden to output weights are initialized by inverse technique as in INVWIT. SRINWIT is expected to perform better than both RWIT and INVWIT in terms of reaching deeper minima.

ALGORITHM SRINWIT

Input : N_I = Total number of Input, N_H = Total number of Hidden Nodes, $INPUT$ = Input Matrix of training sample, $OUTPUT$ = Output matrix of training sample

Output : IW = Weights from input to hidden nodes, HO = Weights from Hidden to output nodes, HT = Threshold at Hidden Node, OT = Threshold at Output Node

1. $MAX = (N_I + 1) \times N_H$;
2. $G = -\log(2\sqrt{5} - \sqrt{19})$
3. $S = 2 \times G/N_H$
4. $W = rand(1, MAX) \times S - S/2$
5. For i ranging from 1 to H
   a. $P = (i-1) \times (N_I + 1) + 1$;
   b. $Q = (i-1) \times (N_I + 1) + N_I$;
   c. $IW(i,1), \dots, IW(i,N_I) = W(1,P), \dots, W(1,Q)$;
   d. $HT(i,1) = W(1, Q+1)$;
   e. $X = \sqrt{IW(i,1)^2 + \cdots + IW(i,N_I)^2 + HT(i,1)^2}$
   f. $IW(i,1), \dots, IW(i,N_I) = IW(i,1), \dots, IW(i,N_I) \times S \times i/X$
   g. $HT(i,1) = HT \times S \times i/X$
6. $MAX = N_H$;
7. $i = MAX + 1$;
8. Add a row of one's below the last row of $INPUT$ and to get $INPUT\_MATRIX$
9. $INPUT\_TO\_HIDDEN\_WEIGHTS = [IW\ HT]$;
10. Add a column of all one's after the last column of $INPUT\_TO\_HIDDEN\_WEIGHTS$ to get $INPUT\_TO\_HIDDEN\_WEIGHT\_MATRIX$.
11. $H = INPUT\_MATRIX * INPUT\_TO\_HIDDEN\_WEIGHT\_MATRIX$
12. $\beta = PINV(H) * OUTPUT$
13. $HO(1,1), \dots, HO(1,N_H) = \beta(1,1), \dots, \beta(1,MAX)$
14. $OT(1,1) = \beta(i)$

IV. EXPERIMENT DESIGN

For function approximation task, the given 10 functions are trained using SFANN.

1. 1-input function from MATLAB sample file *humps.m*

$$y = \frac{1}{(x-0.3)^2 + 0.01} + \frac{1}{(x-0.9)^2 + 0.4} - 6 \quad (9$$

where $x \in [0,1]$

2. 2-input function from MATLAB sample file *peaks.m*

$$y = 3(1-x_1)^2 e^{(-x^2-(x_2+1)^2)} - 10\left(\frac{x_1}{5} - x_1^3 - x_2^5\right) e^{(-x_1^2-x_2^2)} - \left(\frac{1}{3}\right) e^{(-(x_1+1)^2-x_2^2)} \quad (1$$

where $x_1 \in [-3,3]$ and $x_2 \in [-3,3]$.

3. 2-input function from Breiman[15, 16, 17]

$$y = sin(x_1 * x_2) \quad (1$$

where $x_1 \in [-2,2]$ and $x_2 \in [-2,2]$

4. 2-input function from Breiman[15, 16, 17]

$$y = exp(x_1 * sin(\pi * x_2)) \quad (1$$

where $x_1 \in [-1,1]$ and $x_2 \in [-1,1]$

5. 2-input function from Gu et.al.[16, 17]

$$a = 40 * exp(8 * ((x_1 - .5)^2 + (x_2 - .5)^2)) \quad (1$$
$$b = exp(8 * ((x_1 - .2)^2 + (x_2 - .7)^2))$$
$$c = exp(8 * ((x_1 - .7)^2 + (x_2 - .2)^2))$$
$$y = a/(b+c)$$

where $x_1 \in [0,1]$ and $x_2 \in [0,1]$

6. 2-input function from Masters[16, 17]

$$y = (1 + \sin(2x_1 + 3x_2))/(3.5 + \sin(x_1 - x_2)) \quad (1$$

where $x_1 \in [-2,2]$ and $x_2 \in [-2,2]$

7. 2-input function from Maechler et. al. [18, 16, 15]

$$y = 42.659(.1 + x_1(.05 + x_1^4 - 10x_1^2 x_2^2 + 5x_2^4)) \quad (1$$

where $x_1 \in [-5,5]$ and $x_2 \in [-5,5]$

8. 2-input function from Maechler et. al. [18, 16, 15]

$$y = 1.3356[1.5(1-x_1) + \exp(2x_1 - 1)\sin(3\pi(x1 - .6)^2) + \exp\big(3(x_2 - .5)\big)\sin(4\pi(x_2 - .9)^2)] \quad (1$$

where $x_1 \in [0,1]$ and $x_2 \in [0,1]$

9. 2-input function from Maechler et. al. [18, 16, 15]

$$y = 1.9[1.35 + \exp(x_1)\sin(13(x_1 - .6)^2)\exp(-x_2)\sin(7x_2)] \quad ($$

where $x_1 \in [0,1]$ and $x_2 \in [0,1]$

10. 6-input function from Friedman [19]

$$y = 10sin(\pi x_1 x_2) + 20(x_3 - .5)^2 + 10x_4 + 5x_5 + 0x_6 \quad (1$$

where $x_1 \in [-1,1]$ and $x_2 \in [-1,1]$

750 input samples are generated for each function & corresponding output is generated by executing the function. Thus, 750 input-output samples form the training and test data set. Out of these 750 input-output dataset, 250 input-output samples are used for training and remaining 500 are used for testing. During the training process of network, both input-output samples are given to network, whereas for testing only input sample are given and outputs are generated using trained network. This output is compared to actual output to generate error. All the samples in input-output data sets are scaled to $[-1,1]$.

The size of network is summarized in Table 1. The no. of input & output is given in the function definition. N_H is calculated by conducting exploratory experiments for each of the above function by varying N_H from 4 to 20. The optimal value of N_H for which the result was in a way optimal, is considered as value of N_H.

RPROP (Resilient Backpropagation Algorithm) proposed by Reidmiller and Braun[20] is used in training networks. In RPROP, there is direct adaptation of weight step on the basis of local gradient and the current and the previous value of the error of training.

For all of the above 10 function approximation problems, 30 networks are trained using the RWIT, INVWIT and SRINWIT keeping training algorithms, no. of hidden nodes, activation function and other factors as constant. A total of $10 \times 30 \times 2 = 600$ networks are trained & each network is trained for 1000 epochs.

These experiments are conducted using MATLAB R2015a on a 64 bit Intel i5, Microsoft Windows 10 system with 4 GB RAM.

Mean of the mean squared error (MMSE) and median of the MSE (MEDIAN) are reported for both training and testing data for all the three weight initialization techniques RWIT, INVWIT and SRINWIT in Table 2 and 5.

V. RESULTS AND DISCUSSION:

Due to large data volumes of in the experiment, the result summary for both train and test data is reported.

MEAN OF MEAN SQUARE ERROR (MMSE)

The results of MMSE for RWIT, INVWIT and SRINWIT for both train and test data are reported in Table 2.

Train Data. On inspecting values of train data, it is observed that for functions 1, 2, 3, 4, 5, 6, 7, 8, 9

$$\text{MMSE (RWIT)} > MMSE(\text{INVWIT}) > MMSE(SRINWIT)$$

For Function 10,

$$\text{MMSE (RWIT)} > MMSE(\text{SRINWIT}) > MMSE(INVWIT)$$

Table 3 indicates the best technique in terms of MEAN for each function for train data. Based on Table 3 we can summarize the results for test data as follows:

- Performance of SRINWIT is best for 9 functions.
- Performance of INVWIT is best for 1 function.

Test Data. On inspecting values of test data, it is observed that for functions 1, 2, 3, 4, 6, 7, 8 and 9

$$\text{MMSE (RWIT)} > MMSE(\text{INVWIT}) > MMSE(SRINWIT)$$

For Function 5

$$\text{MMSE (RWIT)} > MMSE(\text{SRINWIT}) > MMSE(INVWIT)$$

For Function 10,

$$\text{MMSE (SRINWIT)} > MMSE(\text{RWIT}) > MMSE(INVWIT)$$

Table 4 indicates the best technique in terms of MEAN for each function for test data. Based on Table 4 we can summarize the results for test data as follows:

- Performance of SRINWIT is best for 8 functions.
- Performance of INVWIT is best for 2 functions.

MEDIAN OF MSE (MEDIAN)

The results of MEDIAN for RWIT, INVWIT and SRINWIT for both train and test data are reported in Table 5.

Train Data. On inspecting values of train data, it is observed that for Functions 2, 6, 7 and 8

$$\text{MEDIAN (RWIT)} > MEDIAN(\text{INVWIT}) > MEDIAN(SRINWIT)$$

For Functions 1 and 9

$$\text{MEDIAN (INVWIT)} > MEDIAN(\text{RWIT}) > MEDIAN(SRINWIT)$$

For Function 3

$$\text{MEDIAN (RWIT)} > MEDIAN(\text{SRINWIT}) > MEDIAN(INVWIT)$$

For Functions 4, 5 and 10

$$\text{MEDIAN (SRINWIT)} > MEDIAN(\text{RWIT}) > MEDIAN(INVWIT)$$

Table 6 indicates the best technique in terms of MEDIAN for each function for train data. Based on Table 6 we can summarize the results for test data as follows:

- Performance of SRINWIT is best for 6 functions.
- Performance of INVWIT is best for 4 functions.

In terms of MEDIAN of train data, performance of SRINWIT is best, INVWIT is average and RWIT is worse.

Test Data. On inspecting values of test data, it is observed that for functions 3, 6, 7 and 8

$$\text{MEDIAN (RWIT)} > MEDIAN(\text{INVWIT}) > MEDIAN(SRINWIT)$$

For Function 1, 2 4 and 9

$$\text{MEDIAN (INVWIT)} > MEDIAN(\text{RWIT}) > MEDIAN(SRINWIT)$$

For Function 5

$$\text{MEDIAN (RWIT)} > MEDIAN(\text{SRINWIT}) > MEDIAN(INVWIT)$$

For Function 10

$$\text{MEDIAN (SRINWIT)} > MEDIAN(\text{RWIT}) > MEDIAN(INVWIT)$$

Table 7 indicates the best technique in terms of MEDIAN for each function for test data. Based on Table 7 we can summarize the results for test data as follows

- Performance of SRINWIT is best for 8 functions.
- Performance of INVWIT is best for 2 functions.

VI. CONCLUSION:

In this paper, a Statistically Resilient Moore-Penrose Inverse based Weight Initialization Technique (SRINWIT) for SFANN is proposed in which input to hidden weights are initialized using Statistically Resilient Weight Initialization technique NWIT keeping in mind that the output of hidden neurons lie in the useful range of activation function and utilizes the complete (useful) range of activation function. The hidden to output weights are initialized using Moore-Penrose inverse method. The proposed method SRINWIT is compared to RWIT and INVWIT for 10 function approximation task. On comparing performance of these functions on statistics like MMSE, MEDIAN the performance of SRINWIT is found to be superior to RWIT and INVWIT. We hereby conclude that SRINWIT is better weight initialization technique for SFANN for at least these 10 function approximation task.

ACKNOWLEDGEMENT

Apeksha Mittal would like to thank Media Lab Asia for providing scholarship for research under Visvesvaraya Ph. D. scheme for Electronics and IT.

REFERENCES

[1]. Cybenko G. Approximation by superpositions of a sigmoidal function. *Math. Control, Signals, and Systems*, vol. 2, pp. 303–314, 1989.

[2]. Funahashi K. On the approximate realization of continuous mapping by neural networks. *Neural Networks*, vol. 2, pp. 183–192, 1989.

[3]. Hornik K., Stinchcombe M., and White H. Multilayer feedforward networks are universal approximators. *Neural Networks*, vol. 2, pp. 359–366, 1989.

[4]. Nguyen D. and Widrow B. Improving the learning speed of 2-layer neural networks by choosing initial values of adaptive weights. Proc. Of International Joint Conference on Neural Networks, vol. 3,1990,pp.21-26.

[5]. Kim Y. K. and Ra J.B. Weight Value Initialization for Improving Training Speed in the Backpropagation Network. Proc. Int. Joint Conf. Neural Networks, vol. 3, 1991, pp. 2396–2401.

[6]. Drago G. P. and Ridella S. Statistically controlled activation weight initialization (SCAWI). *IEEE Trans. Neural Networks*, vol. 3, pp. 627–631, July 1992.

[7]. Chen C. L. and Nutter R. S. Improving the training speed of three-layer feedforward neural nets by optimal estimation of the initial weights. Proc. Int. Joint Conf. Neural Networks, vol. 3, 1991, pp. 2063–2068.

[8]. Wessels L. F. A. and Barnard E. Avoiding false local minima by proper initialization of connections. *IEEE Transactions on Neural Networks*, vol. 3, no. 6, pp. 899–905, 1992.

[9]. Yam J. Y. F. and Chow T. W. S. A weight initialization method for improving training speed in feedforward neural network. *Neurocomputing*, vol. 30, pp. 219 – 232, 2000.

[10]. Sodhi S.S., Chandra P. and Tanwar S. A New Weight Initialization Method for Sigmoidal Feedforward Artificial Neural Networks. Proc. Of International Joint Conference on Neural Networks, 2014, pp. 291-298.

[11]. Sodhi S.S. and Chandra P. A Partially Deterministic Weight Initialization Method for SFFANN's. *Advance Computing Conference (IACC), 2014 IEEE International*, pp.1275-1280.

[12]. Mittal A., Chandra P. and Singh A. P. A Statistically Resilient Method of Weight Initialization for SFANN. Proc. of Advances in Computing, Communications and Informatics (ICACCI), IEEE, 2015, pp 1371-1376

[13]. Haykin S. Neural Networks: A Comprehensive Foundations, Engle-wood Cliffs, NJ:Prentice-Hall,1999.

[14]. Rumelhart D. E., Hinton G. E., and Williams R. J. Parallel Distributed Processing: Volume 1 in: Learning internal representations by error propagation, Foundations, D. E. Rumelhart, J. L. McClelland, and The PDP Research Group, Eds. Cambridge: MIT Press, 1987, pp. 318–362.

[15]. Breiman L. The PI method for estimating multivariate functions from noisy data. *Technometrics*, vol. 3, no. 2, pp. 125–160, 1991.

[16]. Cherkassky V., Gehring D. and M¨ulier F. Comparison of adaptive methods for function estimation from samples. *IEEE Transactions on Neural Networks*, vol. 7, no. 4, pp. 969–984, 1996.

[17]. Cherkassky V. and M¨ulier F. Learning from Data – Concepts, Theory and Methods. New York: John Wiley, 1998

[18]. Maechler M., Martin D., Schimert J., Csoppenszky M., and Hwang J. Projection pursuit learning networks for regression. Proc. of the 2nd International IEEE Conference on Tools for Artificial Intelligence, 1990, pp. 350–358.

[19]. Friedman J. H. Multivariate adaptive regression splines (with discussion). Ann. Statist., vol. 19, pp. 1-141, 1991

[20]. Riedmiller M., and Braun H. A direct adaptive method for faster backpropagation learning: The RPROP algorithm. Proceedings of the IEEE International Conference on Neural Networks, 1993, pp. 586–591.

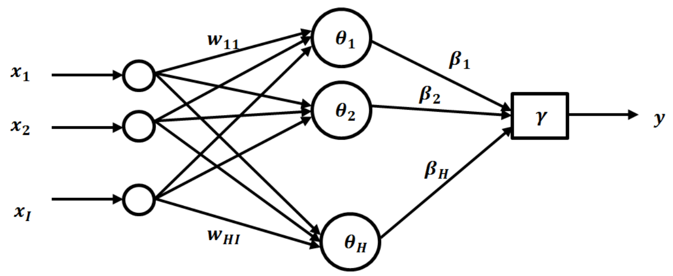

Fig. 1. The schematic diagram for SFANN

Table 1. Network Size Summary

Function	No. of Inputs	No. of hidden nodes	No. of outputs
Fun1	1	7	1
Fun2	2	12	1
Fun3	2	15	1
Fun4	2	15	1
Fun5	2	4	1
Fun6	2	15	1
Fun7	2	15	1
Fun8	2	10	1
Fun9	2	12	1
Fun10	6	10	1

Table 2. Results Summary For Mean Of Mean Square Error For Training And Testing Data Of RWIT & INVWIT. Value Of All Statistics Are Reported $\times 10^{-3}$

	Train Data			Test Data		
Function	RWIT	INVWIT	SRINWIT	RWIT	INVWIT	SRINWIT
1	1.809	1.453	1.037	1.789	1.484	1.032
2	0.312	0.232	0.192	0.523	0.473	0.45
3	13.393	10.683	9.386	17.053	14.275	13.719
4	19.693	17.373	16.447	26.11	25.947	25.319
5	0.191	0.18	0.162	0.319	0.287	0.29
6	49.357	12.895	8.314	54.797	15.752	9.937
7	7.909	5.22	3.604	9.365	7.102	4.84
8	3.962	3.246	2.593	4.232	3.46	2.707
9	6.359	5.71	4.33	8.374	7.88	6.04
10	1.056	0.637	0.647	2.946	2.243	2.986

Table 3. Performance of RWIT, INVWIT and SRINWIT for Functions 1-10 for train data. ✓ indicate that performance of that technique is best for the function (MEAN has least value). N indicates the no. of functions for which the performance is best

Function	1	2	3	4	5	6	7	8	9	10	N
RWIT											0
INVWIT										✓	1
SRINWIT	✓	✓	✓	✓	✓	✓	✓	✓	✓		9

Table 4. Performance of RWIT, INVWIT and SRINWIT for Functions 1-10 for test data. ✓ indicate that performance of that technique is best for the function (MEAN has least value). N indicates the no. of functions for which the performance is best

Function	1	2	3	4	5	6	7	8	9	10	N
RWIT											0
INVWIT					✓					✓	2
SRINWIT	✓	✓	✓	✓		✓	✓	✓	✓		8

Table 5. Results Summary For Median Of Mean Square Error For Training And Testing Data Of RWIT & INVWIT. Values Of All Statistics Are Reported $\times 10^{-3}$

	Train Data			Test Data		
Function	RWIT	INVWIT	SRINWIT	RWIT	INVWIT	SRINWIT
1	1.026	1.063	1.013	1.027	1.052	0.992
2	0.263	0.224	0.179	0.429	0.458	0.426
3	12.955	10.407	12.714	17.144	14.073	13.719
4	20.164	16.879	24.313	25.397	25.544	25.319
5	0.183	0.176	0.248	0.326	0.265	0.29
6	22.277	11.302	10.008	23.892	13.165	9.937
7	6.405	5.1	4.38	8.108	7.12	4.84
8	4.459	2.552	1.686	4.663	2.751	2.707
9	5.65	5.67	3.86	7.471	7.73	5.35
10	1.009	0.617	2.631	2.907	2.192	2.986

Table 6. Performance of RWIT, INVWIT and SRINWIT for Functions 1-10 for train data. ✓ indicate that performance of that technique is best for the function (MEDIAN has least value). N indicates the no. of functions for which the performance is best

Function	1	2	3	4	5	6	7	8	9	10	No
RWIT											0
INVWIT			✓	✓	✓					✓	4
SRINWIT	✓	✓				✓	✓	✓	✓		6

Table 7. Performance of RWIT, INVWIT and SRINWIT for Functions 1-10 for test data. ✓ indicate that performance of that technique is best for the function (MEDIAN has least value). N indicates the no. of functions for which the performance is best

Function	1	2	3	4	5	6	7	8	9	10	N
RWIT											0
INVWIT					✓					✓	2
SRINWIT	✓	✓	✓	✓		✓	✓	✓	✓		8