

**DISTRIBUTED PARTICLE SWARM OPTIMIZER FOR  
WIRELESS SENSOR NETWORKS**

BY

SAMEER HUSAIN ARASTU

A Thesis Presented to the  
DEANSHIP OF GRADUATE STUDIES

**KING FAHD UNIVERSITY OF PETROLEUM & MINERALS**

DHAHRAN, SAUDI ARABIA

In Partial Fulfillment of the  
Requirements for the Degree of

**MASTER OF SCIENCE**

In

TELECOMMUNICATION ENGINEERING

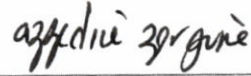
MAY 2012

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS  
DHAHRAN 31261, SAUDI ARABIA

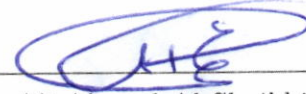
DEANSHIP OF GRADUATE STUDIES

This thesis, written by **Sameer Husain Arastu** under the direction of his thesis advisor and approved by his thesis committee, has been presented to and accepted by the Dean of Graduate Studies, in partial fulfillment of the requirements for the degree of **MASTER OF SCIENCE IN TELECOMMUNICATION ENGINEERING**.

Thesis Committee



Dr. Azzedine Zerguine (Advisor)



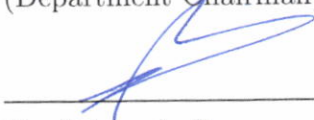
Dr. Ali Ahmad Al-Shaikhi (Member)



Dr. Ali T. Al-Awami (Member)



Dr. Ali Ahmad Al-Shaikhi  
(Department Chairman)



Dr. Salam A. Zummo  
(Dean of Graduate Studies)



Date:



*Dedicated to my parents*

# ACKNOWLEDGEMENTS

*I begin with the name of Allah, the Most Merciful and the Most Beneficent*

All praise is due to Allah, the source of knowledge, blessings and strength. I acknowledge His infinite mercy and grace in making this work a success. And may Allah send His eternal peace and continuous blessings upon his final messenger Muhammad (PBUH), who is the inspiration of our lives, and upon his family and his companions. The successful completion of this work was made possible by a number of contributions from different persons, to whom I wish to express my due gratitude.

I am extremely grateful to King Fahd University of Petroleum and Minerals for giving me the opportunity to carry out this work under the guidance of a scholarly faculty, generous research resources, and a stimulating work environment. My sincere gratitude and thanks go to my thesis advisor Dr. Azzedine Zerguine. I am thankful to him for his meticulous attention, useful guidance, patience and for the multitudes of favors I have taken from him. I would like to extend my appreciation to my thesis committee members Dr. Ali Ahmad Al-Shaikhi and Dr. Ali T Al-Awami for their guidance and cooperation. I would like to especially thank Dr. Mohammad Omer bin Saeed for his patience in guiding me during the initial stages of my thesis work and for his continuous guidance and support.

I would like to thank my friends and colleagues for their support and awareness they have given me since I arrived at KFUPM. We certainly enjoyed doing our thesis during the same time, sharing our experiences, spending long nights together doing our thesis work and celebrating our achievements. My deepest gratitude

goes to my resilient and loving parents and family. I am eternally indebted to my parents for all their prayers, concern, love and understanding.

# TABLE OF CONTENTS

<b>ACKNOWLEDGEMENTS</b>	<b>iv</b>
<b>LIST OF FIGURES</b>	<b>ix</b>
<b>LIST OF TABLES</b>	<b>xiii</b>
<b>THESIS ABSTRACT (ENGLISH)</b>	<b>xiv</b>
<b>THESIS ABSTRACT (ARABIC)</b>	<b>xvi</b>
<b>NOMENCLATURE</b>	<b>xvii</b>
<b>CHAPTER 1. INTRODUCTION</b>	<b>1</b>
1.1 Background . . . . .	2
1.1.1 Wireless Sensor Networks . . . . .	2
1.1.2 Particle Swarm Optimization Algorithm . . . . .	4
1.2 Literature Review . . . . .	9
1.3 Thesis Contributions . . . . .	15
1.4 Thesis Layout . . . . .	16
<b>CHAPTER 2. DISTRIBUTED ESTIMATION PROBLEM AND</b>	

<b>ALGORITHM FORMULATION</b>	<b>17</b>
2.1 System Model . . . . .	18
2.2 Problem Statement . . . . .	20
2.3 Algorithm Formulation . . . . .	20
<b>CHAPTER 3. PROPOSED ALGORITHMS</b>	<b>25</b>
3.1 Diffusion Particle Swarm Optimization Algorithm with Variable Inertia Weight (DPSO-VIW) . . . . .	26
3.2 Diffusion Particle Swarm Optimization Algorithm with Variable Constriction Factor (DPSO-VCF) . . . . .	26
3.3 Diffusion Modified Particle Swarm Optimization (DMPSO) Algo- rithm . . . . .	28
3.4 Diffusion Particle Swarm Optimization - Least Mean Squares (DPSO- LMS) Algorithm . . . . .	30
<b>CHAPTER 4. SENSITIVITY ANALYSIS OF THE PROPOSED ALGORITHMS</b>	<b>40</b>
4.1 Simulation Setup . . . . .	41
4.2 Sensitivity analysis on the swarm parameters of DPSO-VIW algo- rithm . . . . .	42
4.3 Sensitivity analysis on swarm parameters of DPSO-VCF algorithm	43
4.4 Sensitivity analysis on swarm parameters of DMPSO algorithm .	44
4.5 Sensitivity analysis on the network parameters for all the proposed algorithms . . . . .	45
<b>CHAPTER 5. COMPARISON OF THE PROPOSED ALGORITHMS</b>	<b>68</b>
5.1 Comparison of cooperative and non-cooperative PSO algorithm .	69

5.2	Comparison of algorithms in training phase . . . . .	70
5.3	Comparison of the algorithms in testing phase . . . . .	71
5.4	Comparison of the algorithms in steady state . . . . .	72
5.5	Comparison of computational complexity . . . . .	72
<b>CHAPTER 6. CONCLUSION</b>		<b>88</b>
6.1	Contributions . . . . .	88
6.2	Future Work . . . . .	89
<b>REFERENCES</b>		<b>91</b>
<b>VITAE</b>		<b>99</b>



# LIST OF FIGURES

1.1	Execution steps of basic PSO algorithm . . . . .	10
2.1	A sensor network of S nodes . . . . .	18
2.2	System model diagram . . . . .	19
3.1	Execution steps of DPSO-VIW algorithm. . . . .	36
3.2	Execution steps of DPSO-VCF algorithm. . . . .	37
3.3	Execution steps of DMPSO algorithm. . . . .	38
3.4	Execution steps of DPSO-LMS algorithm. . . . .	39
4.1	A 20 node network . . . . .	42
4.2	MSD curves of DPSO-VIW algorithm for different acceleration constant values at 20 dB SNR. . . . .	47
4.3	MSD curves of DPSO-VIW algorithm for different values of inertia weight at 20 dB SNR. . . . .	48
4.4	MSD curves of DPSO-VIW algorithm for different values of velocity constraint factor at 20 dB SNR. . . . .	49
4.5	MSD curves of DPSO-VCF algorithm for different values of acceleration constant at 20 dB SNR. . . . .	50

4.6	MSD curves of DPSO-VCF algorithm for different values of con- striction factor limit at 20 dB SNR. . . . .	51
4.7	MSD curves of DPSO-VCF algorithm for different values of velocity constraint factor at 20 dB SNR. . . . .	52
4.8	MSD curves of DMPSO algorithm for different acceleration con- stant values at 20 dB SNR. . . . .	53
4.9	MSD curves of DMPSO algorithm for different values of slope con- stants at 20 dB SNR. . . . .	54
4.10	MSD curves of DMPSO algorithm for different values of velocity constraint factor at 20 dB SNR. . . . .	55
4.11	MSD curves of DPSO-VIW algorithm for different network size at 20 dB SNR. . . . .	56
4.12	MSD curve of DPSO-VIW algorithm for different particle swarm size at 20 dB SNR. . . . .	57
4.13	MSD curve of DPSO-VIW algorithm for different input data win- dow size at 20 dB SNR. . . . .	58
4.14	MSD curves of DPSO-VCF algorithm for different network size at 20 dB SNR. . . . .	59
4.15	MSD curves of DPSO-VCF algorithm for different particle swarm size at 20 dB SNR. . . . .	60
4.16	MSD curves of DPSO-VCF algorithm for different input data win- dow size at 20 dB SNR. . . . .	61
4.17	MSD curves of DMPSO algorithm for different network size at 20 dB SNR. . . . .	62

4.18	MSD curves of DMPSO algorithm for different particle swarm size at 20 dB SNR. . . . .	63
4.19	MSD curves of DMPSO algorithm for different input data window size at 20 dB SNR. . . . .	64
4.20	MSD curves of DPSO-LMS algorithm for different network size at 20 dB SNR. . . . .	65
4.21	MSD curves of DPSO-LMS algorithm for different particle swarm size at 20 dB SNR. . . . .	66
4.22	MSD curves of DPSO-LMS algorithm for different input data window size at 20 dB SNR. . . . .	67
5.1	Comparison of MSD curves of NCPSO-VIW (k=5) algorithm with DPSO-VIW (k=5) algorithm at network size of 20 nodes . . . . .	74
5.2	Comparison of MSE curves of DPSO-VIW (k=5) algorithm with NCPSO-VIW (k=5) at 20 dB SNR and at network size of 20 nodes.	75
5.3	Comparison of MSD curves of NCPSO-VIW algorithm (k=100) with DPSO-VIW algorithm (k =5) at network size of 20 nodes. . . . .	76
5.4	Comparison of MSE curves of NCPSO-VIW algorithm (k=100) with DPSO-VIW algorithm (k=5) at 20 dB SNR and at network size of 20 nodes. . . . .	77
5.5	Comparison of MSD curves of different algorithms at 10 dB SNR and at network size of 20 nodes . . . . .	78
5.6	Comparison of MSE curves of different algorithms at 10 dB SNR and at network size of 20 nodes. . . . .	79
5.7	Comparison of MSD curves of different algorithms at 20 dB SNR and at network size of 20 nodes . . . . .	80

5.8	Comparison of MSE curves of different algorithms at 20 dB SNR and at network size of 20 nodes. . . . .	81
5.9	Comparison of testing phase of different algorithms at 10 dB SNR and at network size of 20 nodes. . . . .	82
5.10	Comparison of testing phase of different algorithms at 20 dB SNR and at network size of 20 nodes. . . . .	83
5.11	Comparison of steady state MSD of different algorithms at 10 dB SNR and at network size of 20 nodes. . . . .	84
5.12	Comparison of steady state MSE of different algorithms at 10 dB SNR and at network size of 20 nodes. . . . .	85
5.13	Comparison of steady state MSD of different algorithms at 20 dB SNR and at network size of 20 nodes. . . . .	86
5.14	Comparison of steady state MSE of different algorithms at 20 dB SNR and at network size of 20 nodes. . . . .	87

# LIST OF TABLES

4.1	Optimum parameter values of DPSO-VIW algorithm . . . . .	43
4.2	Optimum parameter values of DPSO-VCF algorithm . . . . .	44
4.3	Optimum parameter values of DMPSO algorithm . . . . .	45
5.1	Steady state MSD and MSE values of DPSO-LMS compared to other algorithms at 10-dB and 20-dB SNR . . . . .	71
5.2	Number of computations required by the proposed algorithms . .	72
5.3	Computational cost of the proposed algorithms . . . . .	73

# THESIS ABSTRACT

**Name:** SAMEER HUSAIN ARASTU.  
**Title:** DISTRIBUTED PARTICLE SWARM OPTIMIZER FOR WIRELESS SENSOR NETWORKS.  
**Degree:** MASTER OF SCIENCE.  
**Major Field:** TELECOMMUNICATION ENGINEERING.  
**Date of Degree:** May, 2012.

*In this work, a diffusion particle swarm optimization (DPSO) algorithm is proposed to cooperatively estimate a monitored parameter by the sensor nodes in an ad-hoc wireless sensor network (WSN). Here, every sensor node of a wireless sensor network is equipped with a PSO algorithm to estimate a parameter of interest. A novel diffusion scheme is used to cooperatively estimate the parameter by sharing the local best particle and the corresponding particle error value to the neighboring nodes. The performance of the DPSO algorithm is improved by applying different enhancements to the PSO algorithm. Therefore, different types of the DPSO algorithm proposed are: the DPSO algorithm with variable inertia weight (DPSO-VIW), the DPSO algorithm with variable constriction factor (DPSO-VCF), the diffusion modified PSO (DMPSO) algorithm, and a hybrid DPSO-LMS (DPSO-LMS) algorithm. The simulation results reports a great improvement brought about by the DPSO algorithms over the non-cooperative PSO-LMS (NCPSO-LMS) algorithm, the diffusion least-mean-squares (DLMS) algorithm and the diffusion recursive-least-squares (DRLS) algorithm.*

*Keywords : wireless sensor network (WSN), particle swarm optimization (PSO) algorithm, least mean squares (LMS) algorithm, recursive least squares (RLS) algorithm, diffusion protocol, inertia weight, constriction factor.*

## خلاصة الرسالة

الإسم :	سمير حسين أرسطو
عنوان الرسالة :	محسّن سرب الجسيمات الموزعة للشبكات الإستشعارية اللاسلكية
الدرجة العلمية :	ماجستير
التخصص :	هندسة الإتصالات
تاريخ التخرج :	فبراير 2012 م

في هذا العمل، تم إقتراح خوارزمية لتحقيق أمثلية إنتشار سرب الجسيمات (DPSO) لتقدير (بشكل تعاوني) عامل متغير مراقب بواسطة عقد إستشعارية في شبكة إستشعار لاسلكية (WSN). كل عقدة إستشعارية مجهزة بخوارزمية (DPSO) لتقدير المعامل المطلوب. تم إستخدام مخطط إنتشار جديد لتقدير المعامل بمشاركة أفضل جسيم محلي والقيمة الخاطئة للجسم الموافق للعقد المجاورة. تم تحسين أداء الخوارزمية (DPSO) بتطبيق تعزيزات مختلفة للخوارزمية (PSO). لذلك، الأنواع المختلفة المقترحة من خوارزمية (DPSO) هي: خوارزمية (DPSO) بواسطة معامل قصور ذاتي متغير (DPSO-VI), خوارزمية (DPSO) مع معامل إنقباض متغير (DPSO-VCF), خوارزمية (PSO) الإنتشار المعدلة (DMPSO), وأخيراً، خوارزمية مهجنة (DPSO-LMS). نتائج المحاكاة أظهرت تحسينات كبيرة بواسطة خوارزمية (DPSO) مقارنة مع الخوارزميات غير التعاونية (NCP-SO-LMS), خوارزمية إنتشار مربعات الوسط الصغرى (DLMS), وأيضاً خوارزمية إنتشار المربعات الصغرى التكرارية (DRLS).

**كلمات دلالية :** شبكة إستشعار لاسلكية (WSN), خوارزمية أمثلية إنتشار السرب (PSO), خوارزمية إنتشار مربعات الوسط الصغرى (LMS), خوارزمية المربعات الصغرى التكرارية (RLS), بروتوكول الإنتشار, معامل القصور الذاتي, معامل الإنقباض.



# NOMENCLATURE

## Abbreviations

FC	:	Fusion Center
WSN	:	Wireless Sensor Network
EA	:	Evolutionary algorithm
LMS	:	Least mean squares algorithm
RLS	:	Recursive least squares algorithm
PSO	:	Particle swarm optimization algorithm
DLMS	:	Diffusion least mean squares algorithm
DRLS	:	Diffusion recursive least squares algorithm
DPSO	:	Diffusion particle swarm optimization algorithm
DPSO-VIW	:	Diffusion particle swarm optimization algorithm with variable inertia weight
DPSO-VCF	:	Diffusion particle swarm optimization algorithm with variable constriction factor
DMPSO	:	Diffusion modified particle swarm optimization algorithm
DPSO-LMS	:	Diffusion particle swarm optimization - least mean squares algorithm
MSD	:	Mean Square Deviation
MSE	:	Mean Square Error

## Notations

$S$	:	number of sensor nodes
$s$	:	node number
$p$	:	number of neighbor nodes for node $s$
$t$	:	iteration number
$\mathbf{w}_0$	:	unknown parameter
$\mathbf{d}_s$	:	measurement vector at node $s$
$\mathbf{v}_s$	:	white noise vector at node $s$
$\mathbf{u}_s$	:	input data vector at node $s$
$U_s$	:	input data matrix at node $s$
$J_{s,i}$	:	mean square error of $i^{th}$ particle at node $s$
$M$	:	particle dimension size
$N$	:	input data block size
$k$	:	particle population size at a node
$X_{s,i}$	:	$i^{th}$ particle position at node $s$
$x_{s,i,j}$	:	position of $i^{th}$ particle in $j^{th}$ dimension at node $s$
$V_{s,i}$	:	velocity of $i^{th}$ particle at node $s$
$v_{s,i,j}$	:	velocity of $i^{th}$ particle in $j^{th}$ dimension at node $s$
$X_{s,i}^*$	:	particle best of $i^{th}$ particle at node $s$
$X_s^{**}$	:	local best particle at node $s$
$J_{s,i}^*$	:	particle best error of $i^{th}$ particle at node $s$
$J_s^{**}$	:	local best error at node $s$
$v_{max}$	:	maximum allowable value for $v_{s,i,j}$

$x_{max}$	:	maximum allowable value for $x_{s,i,j}$
$x_{min}$	:	minimum allowable value for $x_{s,i,j}$
$c_1$	:	acceleration constant for cognitive part of velocity update equation
$c_2$	:	acceleration constant for social part of velocity update equation
$iw$	:	inertia weight
$\alpha$	:	inertia weight update factor
$v_c$	:	velocity constraint factor
$K(t)$	:	variable constriction factor at time t
$k_{min}(t)$	:	minimum value of constriction factor at time t
$k_{max}(t)$	:	maximum value of constriction factor at time t
$iw_{s,i}$	:	inertia weight of $i^{th}$ particle at node s
$\Delta J_{s,i}(t)$	:	error gradient of $i^{th}$ particle at node s at time t
$S_l$	:	slope constant
$\mu$	:	step-size
$.^T$	:	transpose

# CHAPTER 1

## INTRODUCTION

In recent years, the rapid growth in wireless communication and electronic industry has enabled the development of power-efficient, low-cost and multi-functional wireless sensor networks [1]. This research area has become quite demanding as WSN are being used in numerous applications like environment and habitat monitoring, structural health monitoring, health care, home automation, traffic surveillance, just to name a few. These applications commonly require to estimate certain parameters such as temperature, concentration of chemicals, pressure, speed and position of target object.

The sensor nodes can perform multiple operations like data acquisition from the surrounding physical media, signal processing tasks, control signaling with the central node or with the neighboring nodes and communicate relevant data collected through wireless transceivers. Usually in a WSN there is a group of sensors nodes in target sensing areas like battle fields, forests etc with limited

communication and power capability. In such environments, it becomes difficult to replenish the resources like the battery power of the sensor node, therefore the available resources have to be utilized efficiently. The limited resources can be optimally utilized in a distributed network as it reduces multi hop or long range communication required in the centralized network to send the data to the central nodes for data processing.

By adapting a distributed processing in ad-hoc sensor network the data transmission to the central nodes can be reduced and thus the available resources can be used in an optimal manner. There has been extensive research done and several adaptive filtering algorithms [13]-[20] proposed to exploit the spatial and temporal diversity of the sensor network to improve performance of the network.

## **1.1 Background**

### **1.1.1 Wireless Sensor Networks**

A wireless sensor network is a collection of sensor nodes placed in a pre-determined or in a random manner in the sensing location. In a sensor network the sensors coordinate among themselves to form a communication network which is either single hop or multi-hop or a hierarchical network with several cluster or cluster heads. In a centralized network the signal processing takes place at the central location known as the fusion center (FC) and in an ad-hoc network the processing

happens among sensor nodes. In the former case the sensor nodes do not process the data collected and at the sensor node the data is quantized, encoded and transmitted over the wireless channel to the fusion center which has a larger processing and storage capability than the sensor nodes. The data collected from all the nodes is processed at the fusion center and the parameters are estimated.

In the latter case each sensor nodes process the information collected from the neighboring nodes which are within the communication range. The collected information is combined at the sensor node and required signal processing is done to estimate the parameter of interest. This process is repeated over several iterations, with each iteration improving the previously estimated result. As each sensor does the signal processing independently its estimates may vary with respect to its neighboring sensor nodes. So by using an interactive algorithm all the sensors can be made to reach a mutually agreeable estimate. The estimate can be made to reach only asymptotically but the network behaves in a self organized manner without the need of the fusion center. The Fusion center (FC) based topologies benchmark the performance of all ad-hoc network based topologies implemented using WSNs. As all the data from all the nodes is available at the a central location for processing, a common estimate can be derived for all the nodes, but in case of ad-hoc network the estimated evaluated at each node spreads via single hop exchange among the neighboring nodes, which results in some time delay till all the nodes reach to a common parameter estimate. Thus

the quality of the estimates is lower when compared to the FC network.

But there is some major limitations in fusion center topologies like if the FC fails then the whole network will fail. The sensor nodes located far apart consumes more power for communicating with the fusion center. This problem can be mitigated with multi-hop system but at the cost of additional complexity to the system which will also consume additional power due to the complex system. In spite of alternative solutions the FC network is not as robust ad-hoc network. On the other hand, these limitations do not apply to the ad-hoc network; if any node fails there is some performance degradation but it does not affect the functioning of entire network. Each sensor node communicates only with its nearby neighboring node, so the power consumption is lower and battery life is longer in comparison with the sensor nodes in FC topology network.

### **1.1.2 Particle Swarm Optimization Algorithm**

Particle swarm optimization is a modern heuristic algorithm, which belongs to the category of swarm intelligence methods. It was first introduced by Kennedy and Eberhart [2] and was initially simulated on a simplified social system where each particle position is assumed to be a state of mind with particular setting of abstract variables which represents the individual beliefs and attitude [3]. The movement of the particles represents the change of these abstract variables. The swarms evaluate their position through external stimuli and compare it with ex-

isting knowledge and replace the existing values with the best fit. There are three important properties of human and animal social behavior, i.e., evaluation, comparison and imitation used by particle swarm to adapt to the environmental changes [3].

The particle swarm is an ad-hoc system where each particle is on its own and acts upon its local information, but as a group of particles, the swarm of particles is capable of self organizing to perform complex task. Due to inter communication among the particles, formation of complex structures is possible at swarm level, which helps in solving complex optimization problems.

Kennedy and Eberhart laid down five basic principle of Swarm Intelligence [2] and the PSO algorithm follows all these principles. These five basic principles are:

Proximity: The swarm should be able to perform simple space and time calculations.

Quality: The swarm should be able to react to quality factors in the environment.

Diverse response: The swarm should not hold on to activities along excessive narrow channels.

Stability: The swarm should not be very sensitive to environmental changes.

Adaptability: The swarm should be able to adapt to the environmental stimuli when it is worth the computational price.

The PSO algorithm is a robust and fast algorithm and can provide better so-



lutions to nonlinear, not differential, multi-modal problems. It has been applied in wide variety of scientific field [4] due to its high-quality solution, lower computational cost than other evolutionary algorithms and also stable convergence characteristics. The Evolutionary algorithm (EA) uses genetic operations like selection, mutation and crossover to search for the global minimum, whereas for PSO algorithm, position update and velocity update equations are used by the particles at each iteration to update their position. At every iteration each particle learns from its own experience and also from its neighbors experience and adjust its position in the search space. Unlike in EA algorithm where particles are replaced at each generation, in PSO algorithm the particles stay alive and are restricted to move within the search space for the whole run. Another major difference is that , EA algorithm does competitive search whereas as PSO algorithm does a cooperative search for optimal solution.

PSO algorithm is a population based algorithm and is used to optimize continuous and real-valued function in the  $m$ -dimensional space  $\mathbb{R}_m$ . The population is also known as swarm, consists of particles which move in predefined search space. A swarm consists of group of particles and each particle position in the search space represents a potential solution to the problem. The PSO algorithm selects the optimal solution through iterative and probabilistic modification of the existing solutions. The commonly used PSO algorithm [3] is discussed here. Consider a swarm population size of  $k$  particles. A particle is considered as a point in the

$m$  - dimensional search space. The different elements of PSO are discussed as follows,

**Particle position**  $X_i(t)$ : The particle is the potential solution represented by an  $m$ -dimensional real-valued vector. The  $i^{th}$  particle at time  $t$  is given as

$X_i(t) = (x_{i,1}(t), x_{i,2}(t), x_{i,3}(t), \dots, x_{i,M}(t))$ , where  $x_{i,j}(t)$  is the position of the  $i^{th}$  particle in the  $j^{th}$  dimension.

**Particle velocity**  $V_i(t)$ : It is the velocity of moving particles and is represented by  $m$ -dimensional real-valued vector. The  $i^{th}$  particle velocity at time  $t$  is given as  $V_i(t) = (v_{i,1}(t), v_{i,2}(t), \dots, v_{i,M}(t))$ , where  $v_{i,j}(t)$  is the velocity of the  $i^{th}$  particle in the  $j^{th}$  dimension.

**Inertia weight**  $iw(t)$ : It is parameter to control the effect of previous particle velocity on the current velocity. It influences the global and local exploration abilities of the particles.

**Particle best**  $X^*(t)$ : As the particle moves through the search space it evaluates its error value at every position and compares it with its individual best error value attained at any time up to the current time. The position of the particle corresponding to best error value is saved as particle best (pbest). The error for every particle position is calculated using the objective function  $J$ , if  $J_i(t) < J_i^*(t - 1)$ ,  $i = 1, 2, \dots, k$ ; then pbest is updated as  $X_i^*(t) = X_i(t)$  corresponding to the error  $J_i^*(t) = J_i(t)$  else the values are updated for current time as  $X_i^*(t) = X_i^*(t - 1)$  and  $J_i^*(t) = J_i^*(t - 1)$ .

**Global best  $X^{**}(t)$ :** It is best position among all the pbest of the particles and the global best (gbest) at time  $t$  is evaluated as follows. First search for the minimum error value among all  $J_i^*(t), i = 1, 2, \dots, k$  and denote the minimum error as  $J_{min}(t)$  and the corresponding particle as  $X_{min}(t)$ . If  $J_{min}(t) < J^{**}(t-1)$  then update global best as  $X^{**}(t) = X_{min}(t)$  and  $J^{**}(t) = J_{min}(t)$  else update  $J^{**}(t) = J^{**}(t-1)$  and  $X^{**}(t) = X^{**}(t-1)$ . The global best is expressed as  $X^{**}(t) = (x_1^{**}(t), x_2^{**}(t), \dots, x_m^{**}(t))$ ,

**Stopping criteria:** The search for global minimum needs to terminate either after attaining the global minimum or closer to it or after completing pre-specified number of iterations, so that the algorithm works within the feasible computational cost. The search for global minimum in the search space is terminated if any of the following condition is satisfied.

1. There is no change in global best for pre-specified iterations.
2. The maximum number of allowable iterations is reached.

The particle velocity is constrained to improve the local exploration of the problem space. The particle velocity in  $j^{th}$  dimension is limited as:  $v_{max} = v_c x_{max}$ , where  $v_c$  is the velocity constraint factor.

The particles positions and velocities are updated using the following equation,

velocity update for  $i = 1,2,3,\dots,P$  and  $j = 1,2,\dots,M$

$$\begin{aligned} v_{i,j}(t+1) = & iw v_{i,j}(t) \\ & +c_1r_1(x_{i,j}^*(t) - x_{i,j}(t)) \\ & +c_2r_2(x_j^{**}(t) - x_{i,j}(t)) \end{aligned} \tag{1.1}$$

position update,

$$x_{i,j}(t+1) = x_{i,j}(t) + v_{i,j}(t+1) \tag{1.2}$$

The above execution steps of PSO algorithm is given in the Fig. 1.1 as follows:

## 1.2 Literature Review

In target sensing areas like battle fields, forests etc it is not feasible to replenish the resources of the sensor node easily, so the available resources have to be utilized in an optimal way. Distributed computing has attracted many researchers to apply to WSNs, as it enables low-cost estimation of the parameters and also its robustness to node failure. Distributed parameter estimation can be done either by cooperatively estimating the parameter of interest by data sharing among the neighboring sensor nodes or by non-cooperative estimation where the sen-

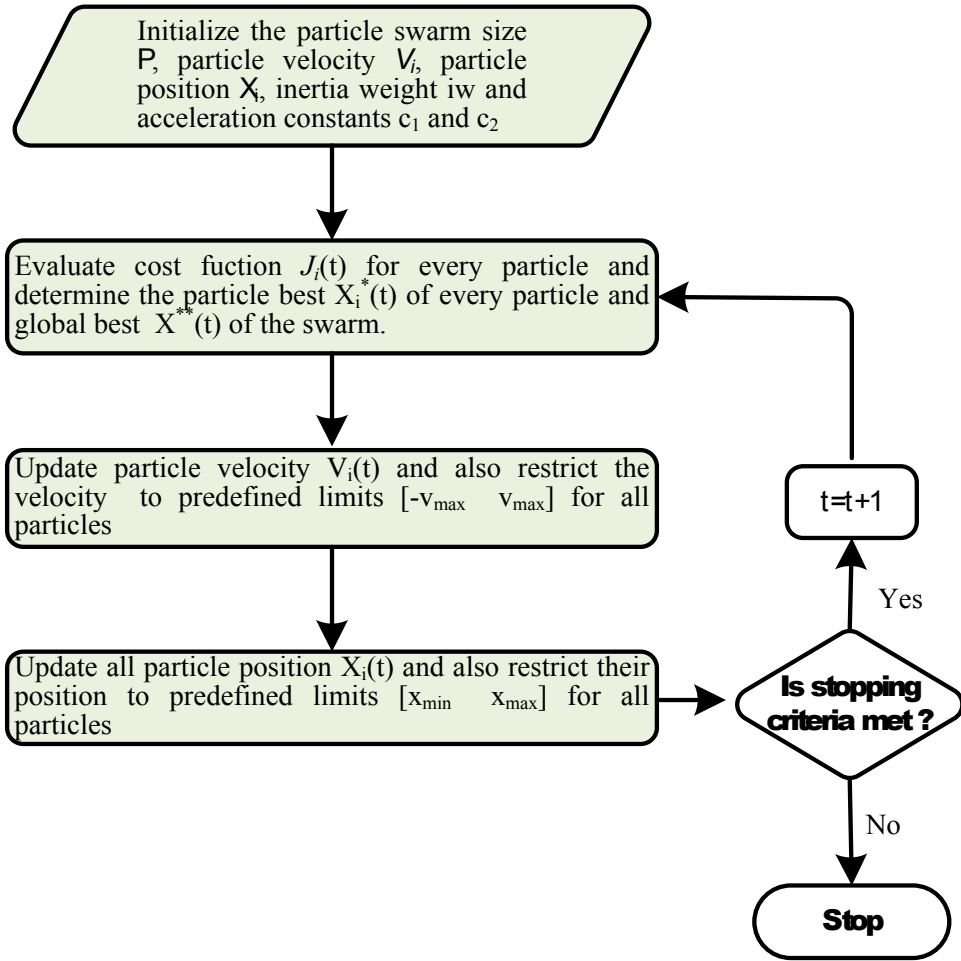


Figure 1.1: Execution steps of basic PSO algorithm

Each node independently estimates the parameter without any data sharing to its neighboring nodes. An improved performance can be achieved by collaboration in a cooperative distributed network because it exploits the spatial and temporal diversity of the network to reduce the estimation error whereas the non-cooperative network can only exploit the temporal diversity of the network. A lot of research is carried out on consensus-based distributed signal processing [5] - [12]. In all the suggested schemes, the data is collected by sensor at once and then consensus

is reached by locally sharing messages. A theoretical framework for analysis of consensus algorithms and different areas of application has been well illustrated in [12]. The drawback of consensus-based algorithms is that they are not robust enough to tackle the problem of estimating time-varying signals or dynamic systems. Recently, distributed parameter estimation for dynamic systems such as ad-hoc WSNs have received lot of attention. In [13] the authors have proposed a diffusion scheme for distributed Kalman filtering where the data between neighboring sensors is diffused before each sensor updates its own estimate using a Kalman filter, thus improving the overall performance of the system. Further improvement to potential problems have been suggested in [14] - [15].

A sequential scheme was suggested, where the information circulated through a topological cycle and LMS adaptive filters are used at each sensor to adapt to variations in the signal statistics [16]. The network adopts a Hamiltonian cycle where the sensor node updates the weight using its local data and sends the updated weights to the next sensor in the cycle to get the next better quality estimate of the weights. The sensor is able to account for any time variations in the process as the sensors use newly acquired data at each iteration to update the estimate. This distributed scheme provides a faster convergence of solution than a centralized system and also attains a low steady-state error at a lower computational complexity. However, if any sensor node in the cycle fails then the network is broken and would stop functioning until the cycle is restored. In [17]

the author suggested a solution to the node failure problem but the computational complexity of the algorithm is increased at the cost of performance degradation. To overcome the drawbacks of the incremental algorithm especially the topological constraints in [16] and also to fully exploit the distributed nature of the network a new algorithm was proposed known as the diffusion LMS algorithm (DLMS) [18]. The computational cost was higher but was able to take advantage of temporal and spatial diversity of the network. In diffusion LMS the network topology was such that every sensor is able to communicate with its nearby sensor nodes. So the sensor received the updated weight from all the neighboring sensors and did a convex combination of the received local estimates. Using LMS recursion the sensor updates the local estimate and this process is repeated for each sensor. In [19] the author suggested an improved version of this algorithm where the process is reversed, that is, first the LMS recursion updates the local estimate at each sensor and then the convex combination of the estimates of the nearby neighbors is taken. To improve the convergence speed RLS algorithm is suggested in place of LMS algorithm but at the cost of higher computational complexity [20]. These adaptive algorithms does not have the hill climbing capability to avoid the local minima and are held in the local minimum of the search space.

Particle swarm optimization technique is simple and effective and has been used by researchers in addressing WSNs problems such as optimal deployment of the sensor node to achieve desired coverage, connectivity and energy efficiency

with minimal number of sensor nodes [21], location identification of the sensor nodes with respect to pre-determined location [22], energy efficient clustering of sensor nodes to have minimal communication with the fusion centers [23] and data aggregation of voluminous distributed data of large scale deployed sensor nodes in an optimal way [24].

The WSNs faces some technical challenges such as dynamic topology, dense ad-hoc deployment, spatial distribution and limitations in memory, bandwidth, computational resources and energy. As traditional optimization techniques such as linear, non-linear quadratic programming, Newton based techniques and interior point method requires enormous computational efforts. The PSO algorithm is found to be more suitable in addressing these issues because of its inherent advantages such as ease of hardware and software implementation, available guidelines for choosing its parameters, ability to overcome local minimum problem and faster convergence than other heuristic algorithms such as genetic algorithm, differential evolution and bacterial foraging algorithm.

PSO has been applied to a large number of problems and systems as it is able to optimize a wide array of functions of different types. In [25] the author proposed a hybrid-PSO, which is a combination of Evolutionary algorithm with the basic PSO. Similar scheme has been suggested to integrate the PSO and Genetic Algorithm (GA) methods in parallel and in series. Both the algorithms were able to perform better than standard PSO algorithm on a series of benchmark



functions [26]. To achieve global minimum quickly and without premature convergence, proper tuning of the PSO algorithm parameters is required. This matter has been discussed and the effects of the tuning and parameter values are studied in numerous papers [27] - [31].

The idea of sharing information with the neighboring particles was first suggested by Kennedy [32], to enhance the search space and achieve better convergence. The entire swarm is divided into smaller swarm topologies and each swarm evaluates its local best. Unlike the standard PSO, particle position is updated using the local best instead of the global best. A cooperative particle swarm optimizer (CPSO) which reduced the convergence time significantly was discussed in [33], [34]. In this technique the solution vector of  $n$  dimension is split into  $n$  one dimensional vector and each sub-vector is optimized using a separate PSO algorithm. The global solution of all the swarms was concatenated to get the final solution vector. A hybrid CPSO which is a combination of standard PSO and CPSO was proposed to improve the convergence time. An extended PSO (EPSO) algorithm was suggested in [35]. In this algorithm the particle velocity is calculated using both local as well as global best positions of the particles velocity at each iteration. The advantages of both global best and local best are combined but needs further tuning of weight assignments ( $c$ 's), and topology for local best. Kennedy in his paper [36] has discussed the good practices to be adopted and the bad practices to be ignored. He gave an informal discussion of the algorithm and

its different parameters and emphasizes that the real research goal is not to make the algorithm more complicated. In fact, the goal is to strip it down to its essentials, at least while this paradigm is still young, and avoid suboptimal methods. The drawbacks of these earlier proposed PSO algorithms are that they require large particle population size and are more suitable for centralized networks due to higher computational complexity.

### **1.3 Thesis Contributions**

In the earlier proposed PSO algorithms for WSN networks [37], the particle population size and the data window size used at every node for estimating the parameter were quite large, which increased the computational complexity at the node. By using distributed estimation techniques, particle population size and input data window size used in PSO algorithm can be reduced. Thus leading to reduction in overall computational complexity of the network. A novel diffusion scheme is proposed for data sharing among the sensor nodes. The four different distributed estimation algorithms proposed in this thesis are as follows: First, in DPSO-VIW algorithm, PSO algorithm uses a linearly decreasing inertia weight instead of a constant inertia weight to improve the global and local exploration capability of the particle [38]. Next, in DPSO-VCF algorithm, a linearly decreasing constriction factor suggested in [39] is used to improve the convergence

speed and performance. Then in DMPSO algorithm, inertia is made the function of the change in error value of the particle as proposed in [40]. Finally in DPSO-LMS algorithm a hybrid technique is proposed, which combines the advantages of the PSO algorithm and LMS algorithm with some increase in computational complexity but it has shown a good improvement in performance.

## 1.4 Thesis Layout

The remainder of this thesis is organized as follows. Chapter 2 describes the system model, the problem statement and the proposed algorithm. Chapter 3 explains the different enhancements done to the proposed algorithm to improve the network performance. In Chapter 4, all the proposed algorithms are simulated under a common experimental setup and a sensitivity analysis is carried out on different network and particle swarm parameters to identify the optimum values. Then in Chapter 5, the performance of all the proposed algorithms are compared with each other and also with DLMS and DRLS algorithms and conclusions are drawn on their performance and computational complexity. Finally in Chapter 6 the thesis contributions and future work is stated.

## CHAPTER 2

# DISTRIBUTED ESTIMATION PROBLEM AND ALGORITHM FORMULATION

In this work, different algorithms have been developed to address the problem of distributed estimation of monitored parameter in the WSNs. This chapter begins with the overview of the system model of a WSN, followed by the problem statement and finally an algorithm is formulated as a solution for the stated problem.

## 2.1 System Model

Consider a network of  $S$  sensor nodes randomly distributed in a normalized area of  $(1 \times 1)$  square units as shown in the Fig. 2.1. The nodes are placed in such a way that every node has some sensors in close proximity and each node is interconnected only to its neighboring nodes. Each node forms a communication link with its neighbors to share information in a single hop. The communication range  $r$  is set based on the amount of transmitting power each node is allowed. So the nodes that are within the range  $r$  of any node  $s$  comprise the neighbors of that node. It is also assumed that the communication between nodes is noise free.

Consider a system as shown in Fig. 2.2. The system block is considered at every

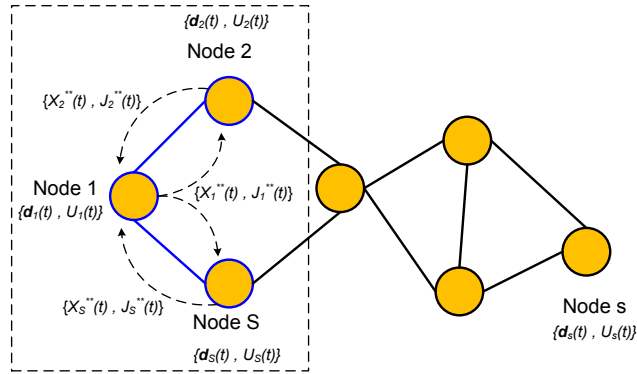


Figure 2.1: A sensor network of  $S$  nodes

sensor node where an unknown system parameter  $\mathbf{w}_0$  is estimated. The unknown system parameter  $\mathbf{w}_0$  is represented by a column vector of order  $(M \times 1)$ . The

input and output of the system at time  $t$  is defined by  $\mathbf{U}_s(t)$  and  $\mathbf{d}_s(t)$ , respectively. The input data matrix  $\mathbf{U}_s(t)$  is of order  $(N \times M)$  and is a group of row vectors  $\mathbf{u}_s(t)$  of order  $(1 \times M)$  formed using the input data block as follows:

$$\mathbf{U}_s(t) = \text{col}\{\mathbf{u}_s(t - N + 1), \mathbf{u}_s(t - N + 2), \dots, \mathbf{u}_s(t)\} \quad (2.1)$$

and output  $\mathbf{d}_s(t)$  is a column vector of length  $N \times 1$  and is expressed as follows:

$$\mathbf{d}_s(t) = \mathbf{U}_s(t)\mathbf{w}^o + \mathbf{v}_s(t), \quad (2.2)$$

where  $\mathbf{v}_s(t)$  is an additive white noise vector at every node.

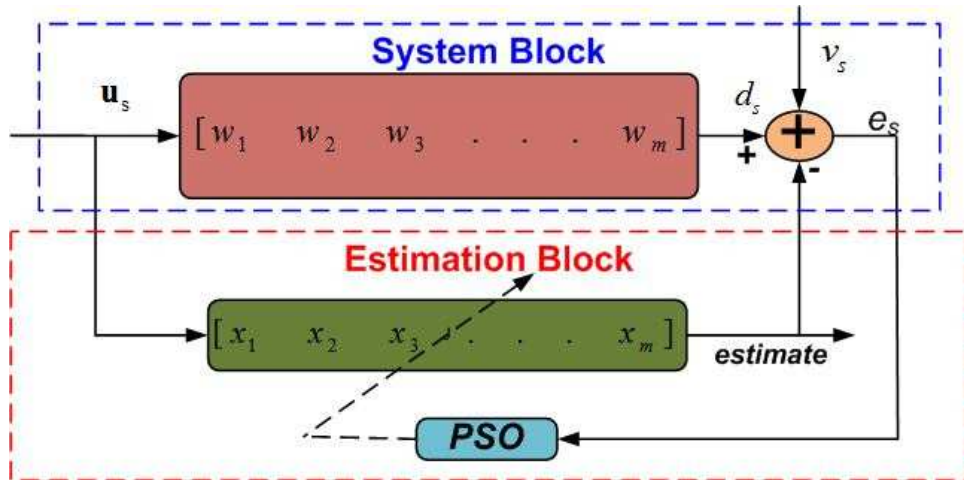


Figure 2.2: System model diagram

## 2.2 Problem Statement

The purpose of the nodes in the network is to estimate the value of a certain parameter of interest  $\mathbf{w}_0$ . The simplest solution to this estimation problem is for each node to estimate the unknown vector using only its own set of data. Such a case is termed as the no cooperation case as the nodes are not communicating with each other. In a non-cooperative sensor network, only temporal diversity of the network is exploited. The spatial diversity of the nodes is not being utilized here and so this case is counter productive as the poor performance of the nodes with low SNR will result in poor performance of the network. In order to exploit the spatial diversity of the network, the information about the estimated parameter at every node should be shared. Therefore by receiving better parameter estimate from the neighboring node the performance of poor performing node can be improved and thus the overall performance of the network can be improved.

## 2.3 Algorithm Formulation

A distributed PSO algorithm is proposed to improve the performance of the network by exploiting both the spatial and temporal diversity of the network. A PSO algorithm is considered at every node  $s$  to estimate the desired parameter  $\mathbf{w}^o$  and for data sharing, a novel data diffusion scheme is proposed. In the proposed diffusion scheme the sensor nodes share its local best particle position  $X_s^{**}$

and the corresponding local best error  $J_s^{**}$  with its neighboring nodes as shown in Fig. 2.1. The information shared from the neighboring nodes is used to reduce the estimation error at every node.

Generally a PSO algorithm works efficiently for batch type optimization problems where the entire input data is available off line. But in an online processing scenario the entire input data is not available, so an input data block of size  $N$  is taken at every iteration. The data window slides at every iteration by one step which will add a new data point and exclude the oldest data point in the data window so that the window size remains constant.

In the Fig 2.2, the estimation block uses a PSO algorithm at each node to minimize error given by the objective function as:

$$J_{s,i}(t) = [||\mathbf{d}_s(t) - U_s(t)X_{s,i}(t)||^2]/N, \quad (2.3)$$

where  $X_{s,i}(t)$  is the  $i^{th}$  particle position vector of node  $s$ . This objective function defines the search space and the position of every particle in the search space is assumed to be the potential estimate of the unknown parameter  $\mathbf{w}_0$ . The computational complexity is also lowered by using smaller swarm size at every node in comparison to the non-cooperative PSO algorithms used in the WSNs.

The steps of the proposed algorithm are as follows:

1. **Initialization:** At  $t = 0$ , initialize  $k$  particles  $X_{s,i}(0), i = 1, 2, \dots, k$  of



dimension  $M$  at each node, where

$X_{s,i}(0) = [x_{s,i,1}(0), x_{s,i,2}(0), \dots, x_{s,k,M}(0)]$ . The coefficients  $x_{s,i,j}(0), j = 1, 2, \dots, M$  of every particle is uniformly distributed in the range  $[x_{min}, x_{max}]$ . Similarly, initialize the velocities  $V_{s,i}(0), i = 1, 2, \dots, k$  of all the node particles, where  $V_{s,i}(0) = [v_{s,i,1}(0), v_{s,i,2}(0), \dots, v_{s,k,M}(0)]$ . The velocity coefficients  $v_{s,i,j}(0)$  are uniformly distributed in the range  $[-v_{max}, v_{max}]$ . The velocity coefficients are limited in a certain range to explore the search space more effectively and the maximum velocity coefficient  $v_{max}$  is defined as [42]

$$v_{max} = v_c x_{max}, \quad (2.4)$$

where  $v_c$  is the velocity constraint factor.

2. **Particle error calculation:** Calculate the estimation error for every particle using the objective function given in (2.3).
3. **Particle best position:** Only in first iteration ( $t = 0$ ), set the particle best position  $X_{s,i}^*(0)$  to the current position of the particle  $X_{s,i}(0)$  and particle best error  $J_{s,i}^*(0)$  to the corresponding particle error value  $J_{s,i}(0)$ . For  $t > 0$ , check: If  $J_{s,i}(t) < J_{s,i}^*(t-1), i = 1, 2, \dots, k$  then set  $J_{s,i}^*(t) = J_{s,i}(t), X_{s,i}^*(t) = X_{s,i}(t)$  and continue; else set  $J_{s,i}^*(t) = J_{s,i}^*(t-1), X_{s,i}^*(t) = X_{s,i}^*(t-1)$  and continue.

4. **Local best particle position:** Search for the minimum among all particle best error  $J_{s,i}^*(t), i = 1, 2, \dots, k$  and assign it to  $J_{s,min}(t)$ , then set  $X_{s,min}(t)$  to the particle position corresponding to the error  $J_{s,min}(t)$ . If  $t > 0$  and  $J_{s,min}(t) < J_s^{**}(t-1)$ , then update local best particle error as  $J_s^{**}(t) = J_{s,min}(t)$  and local best particle position as  $X_s^{**}(t) = X_{s,min}(t)$  and continue; else set  $J_s^{**}(t) = J_s^{**}(t-1), X_s^{**}(t) = X_s^{**}(t-1)$  and continue.
  
5. **Diffusion:** If a node has  $p$  neighboring nodes including itself then share its local best particle error  $J_s^{**}(t)$  and corresponding local best particle position  $X_s^{**}(t)$  to its  $p-1$  neighboring nodes. Using the error values received from its neighbors, as shown in Fig. 2.1, identify the minimum local best error among itself and  $p-1$  neighboring nodes and set  $J_s^{**}(t) = \min(J_s^{**}(t), J_1^{**}(t), \dots, J_{p-1}^{**}(t))$  and then update the local best particle position  $X_s^{**}(t)$  to the particle position corresponding to the error  $J_s^{**}(t)$ .
  
6. **Velocity update:** For the next iteration update the particle velocity using the current particle velocity, the local best particle position  $X_{s,i}^*(t)$  and particle best position  $X_{s,i}^{**}(t)$ . The  $i^{th}$  particle velocity coefficient in the  $j^{th}$

dimension is updated according to

$$\begin{aligned}
v_{s,i,j}(t+1) &= iw(t)v_{s,i,j}(t) \\
&+ c_1 r_1 (x_{s,i,j}^*(t) - x_{s,i,j}(t)) \\
&+ c_2 r_2 (x_{s,j}^{**}(t) - x_{s,i,j}(t)), \tag{2.5}
\end{aligned}$$

where  $c_1$  and  $c_2$  are acceleration constants and  $r_1$  and  $r_2$  are uniformly distributed random numbers in  $[0, 1]$ .

7. **Position update:** Using the updated velocities, then update the particle position according to:

$$x_{s,i,j}(t+1) = x_{s,i,j}(t) + v_{s,i,j}(t+1). \tag{2.6}$$

Goto step 11.

8. **Stopping criteria:** If the maximum number of allowable iterations is reached then stop; else continue.

9. **Time update:** Update the time counter  $t = t + 1$ .

Goto step 2.

Repeat steps 2 to 12 at every node  $s, s = 1, 2, \dots, S$ .

## CHAPTER 3

# PROPOSED ALGORITHMS

The PSO algorithm is mainly affected by stagnancy of particles at the local minimum. To overcome the problem of stagnancy, different enhancements done to algorithm proposed in Chapter 2. The four different enhancements of the proposed algorithm is illustrated in this chapter. This chapter starts with the detail description of the DPSO-VIW algorithm, followed by description of the DPSO-VCF algorithm, then DMPSO algorithm and finally a hybrid algorithm DPSO-LMS algorithm is described.

### **3.1 Diffusion Particle Swarm Optimization Algorithm with Variable Inertia Weight (DPSO-VIW)**

A linearly decreasing inertia weight proposed in [38] is used in the proposed algorithm in Chapter 2. The inertia weight function is defined in (3.1).

$$iw(t) = \alpha iw(t - 1) \quad (3.1)$$

where  $\alpha$  is the weight decreasing factor and  $iw(t)$  is inertia weight at time  $t$ . The execution steps of the DPSO-VIW algorithm is shown in the Fig. 3.1

### **3.2 Diffusion Particle Swarm Optimization Algorithm with Variable Constriction Factor (DPSO-VCF)**

This algorithm uses a time varying factor to update the particle velocity in order to guarantee the convergence of the PSO algorithm. The PSO algorithm with a constriction factor was initially proposed in [41] and [42] and was later used

in many applications because of its better performance than the standard PSO algorithm. A time dependent constriction factor was later proposed in [39] for non-linear system, where the constriction factor was varied at every iteration by varying  $k_c$  as shown in (3.5). In the proposed algorithm, the velocity update equation (3.9) is modified by introducing a variable constriction factor  $K$  suggested in [39]. The modified velocity update equation is shown in (3.2) .

$$\begin{aligned}
v_{s,i,j}(t) = & K(t) (v_{s,i,j}(t-1) \\
& + c_1 r_1 (x_{s,i,j}^*(t-1) - x_{s,i,j}(t-1)) \\
& + c_2 r_2 (x_{s,j}^{**}(t-1) - x_{s,i,j}(t-1)))
\end{aligned} \tag{3.2}$$

where  $K$  is given as

$$K(t) = \frac{k_c(t)}{|2 - \Phi - \sqrt{\Phi^2 - 4\Phi}|}, \tag{3.3}$$

where

$$\Phi = c_1 + c_2, \Phi > 4 \tag{3.4}$$

and

$$k_c(t) = k_{min} + (k_{max} - k_{min}) \frac{R-t}{R-1} \tag{3.5}$$

where the variable  $R$  is the maximum number of iterations and  $t$  is the current

iteration.

The proposed modification of  $K$  can be intuitively explained by noting that as the particle gets closer to global minimum, it undergoes a process similar to a "cooling" one which results in a stabilizing effect on the swarm and which therefore calls for the use of a lower value of the constriction factor. The general DPSO algorithm proposed in Chapter 2 is used and a velocity update with constriction factor is applied as given in (3.2). The execution steps of DPSO-VCF algorithm is shown in the Fig. 3.2

### **3.3 Diffusion Modified Particle Swarm Optimization (DMPSO) Algorithm**

In this algorithm the speed and efficiency of the search is improved by independently adjusting the inertia weight of each particle according to the change in the error value of that particle. The inertia weight is made adaptable i.e. it is either maintained at same value or changed when a better fit position is encountered in order to move the particle more closer to the favorable position. If the particle does not attain a lower estimation error, its inertia influence is reduced. This modification, however, does not prevent the hill climbing capabilities of PSO, it merely increases the influence of potentially fruitful inertia directions, while decreasing the influence of potentially unfavorable inertia directions. The inertia

weight function suggested in [40] is shown as:

$$iw_{s,i}(t) = \frac{1}{\left(1 + e^{\frac{-\Delta J_{s,i}(t)}{S_l}}\right)}, \quad (3.6)$$

where  $iw_{k,i}(t)$  is the inertia weight of the  $i^{th}$  particle of node  $k$ ,  $\Delta J_{k,i}(t)$  is the change in particle error between the current and last generation, and  $S_l$  is the slope constant used to adjust the transition slope based on the expected error range. This relation limits the inertia weight to the interval (0,1), with the midpoint of 0.5 corresponding to zero change in error. Consequently, increase in error will lead to inertia weight larger than the recommended fixed experimental value of 0.5, and decrease in error will lead to inertia weight smaller than 0.5. The DMPSO algorithm execution steps is shown Fig. 3.3



### 3.4 Diffusion Particle Swarm Optimization - Least Mean Squares (DPSO-LMS) Algorithm

The performance can be further enhanced if PSO is hybridized with another algorithm e.g., LMS algorithm [43]. In [44], a hybrid PSO-LMS algorithm was proposed to overcome the problem of stagnation of swarm particles in the search space. In this algorithm, there is no clear indication when to increase the influence of the LMS component to prevent stagnation of the particles in the learning process and therefore there is an overhead to select the appropriate scaling factors to control the effect of the two algorithms on the particle's position update. This difficulty can be overcome by separately using both algorithms at different time periods and thus avoiding the usage of an additional factor to control the effect either algorithm on the particle position update.

Initially, a PSO algorithm with inertia weight update function proposed in [40] can be used globally explore the search space and converge to the solution faster. As soon as the global best of the swarm becomes stagnant the particle position can be updated using an LMS recursion to effectively search for the solution locally in the search space. In the earlier proposed PSO algorithms for WSN network [37], the sensor nodes estimated the unknown parameter non-cooperatively and used large particle population size at every node, which increased the computational complexity. Hence, in this work, cooperative estimation is carried out using a

diffusion scheme which reduces the particle population size at every node, leading to a substantial reduction in computational complexity. Initially the estimation error is minimized using the PSO algorithm. As the particles comes closer to the global minimum of the objective function (2.3), the local best of the node stagnates due to lack of finer search capability of the PSO algorithm. This drawback is overcome by updating the particle position using the LMS recursion with a smaller step size. The steps of the proposed algorithm are as follows:

1. **Initialization:** At  $t = 0$ , initialize  $k$  particles  $X_{s,i}(0), i = 1, 2, \dots, k$  of dimension  $M$  at each node, where

$X_{s,i}(0) = [x_{s,i,1}(0), x_{s,i,2}(0), \dots, x_{s,i,M}(0)]$ . The coefficients  $x_{s,i,j}(0), j = 1, 2, \dots, M$  of every particle is uniformly distributed in the range  $[x_{min}, x_{max}]$ . Similarly, initialize the velocities  $V_{s,i}(0), i = 1, 2, \dots, k$  of all the node particles, where  $V_{s,i}(0) = [v_{s,i,1}(0), v_{s,i,2}(0), \dots, v_{s,i,M}(0)]$ . The velocity coefficients  $v_{s,i,j}(0)$  are uniformly distributed in the range  $[-v_{max}, v_{max}]$ . The velocity coefficients are limited in a certain range to explore the search space more effectively and the maximum velocity coefficient  $v_{max}$  is defined as [42]

$$v_{max} = v_c x_{max}, \quad (3.7)$$

where  $v_c$  is the velocity constraint factor.

2. **Particle error calculation:** Calculate the estimation error for every par-

ticle using the objective function given in (2.3).

3. **Particle best position:** Only in first iteration ( $t = 0$ ), set the particle best position  $X_{s,i}^*(0)$  to the current position of the particle  $X_{s,i}(0)$  and particle best error  $J_{s,i}^*(0)$  to the corresponding particle error value  $J_{s,i}(0)$ . For  $t > 0$ , check: If  $J_{s,i}(t) < J_{s,i}^*(t-1)$ ,  $i = 1, 2, \dots, k$  then set  $J_{s,i}^*(t) = J_{s,i}(t)$ ,  $X_{s,i}^*(t) = X_{s,i}(t)$  and continue; else set  $J_{s,i}^*(t) = J_{s,i}^*(t-1)$ ,  $X_{s,i}^*(t) = X_{s,i}^*(t-1)$  and continue.
4. **Local best particle position:** Search for the minimum among all particle best error  $J_{s,i}^*(t)$ ,  $i = 1, 2, \dots, k$  and assign it to  $J_{s,min}(t)$ , then set  $X_{s,min}(t)$  to the particle position corresponding to the error  $J_{s,min}(t)$ . If  $t > 0$  and  $J_{s,min}(t) < J_s^{**}(t-1)$ , then update local best particle error as  $J_s^{**}(t) = J_{s,min}(t)$  and local best particle position as  $X_s^{**}(t) = X_{s,min}(t)$  and continue; else set  $J_s^{**}(t) = J_s^{**}(t-1)$ ,  $X_s^{**}(t) = X_s^{**}(t-1)$  and continue.
5. **Diffusion:** If a node has  $p$  neighboring nodes including itself then share its local best particle error  $J_s^{**}(t)$  and corresponding local best particle position  $X_s^{**}(t)$  to its  $p-1$  neighboring nodes. Using the error values received from its neighbors, as shown in Fig.2.1, identify the minimum local best error among itself and  $p-1$  neighboring nodes and set  $J_s^{**}(t) = \min(J_s^{**}(t), J_1^{**}(t), \dots, J_{p-1}^{**}(t))$  and then update the local best particle position  $X_s^{**}(t)$  to the particle position corresponding to the error  $J_s^{**}(t)$ .

6. **inertia weight update:** Update the inertia weight according to [40]:

$$iw_{s,i}(t) = \frac{1}{\left(1 + e^{\frac{-\Delta J_{s,i}(t)}{s_i}}\right)}, \quad (3.8)$$

7. **Stagnancy test:** If the local best  $X_s^{**}(t)$  of the node is not same as the prior value then goto step 8 else goto step 10

8. **Velocity update:** For the next iteration update the particle velocity using the current particle velocity, the local best particle position  $X_{s,i}^*(t)$  and particle best position  $X_{s,i}^{**}(t)$ . The  $i^{th}$  particle velocity coefficient in the  $j^{th}$  dimension is updated according to

$$\begin{aligned} v_{s,i,j}(t+1) &= iw_{s,i}(t) v_{s,i,j}(t) \\ &\quad + c_1 r_1 (x_{s,i,j}^*(t) - x_{s,i,j}(t)) \\ &\quad + c_2 r_2 (x_{s,i,j}^{**}(t) - x_{s,i,j}(t)), \end{aligned} \quad (3.9)$$

where  $c_1$  and  $c_2$  are acceleration constants and  $r_1$  and  $r_2$  are uniformly distributed random numbers in  $[0, 1]$ .

9. **Position update:** Using the updated velocities, then update the particle position according to:

$$x_{s,i,j}(t+1) = x_{s,i,j}(t) + v_{s,i,j}(t+1). \quad (3.10)$$

Goto step 11.

10. **Position update (LMS):** As the particles comes closer to the global minima of the objective function (2.3), the local best fitness value of the node stagnates due to lack of finer search capability of the PSO algorithm. This drawback is overcome by updating the particle position using the LMS recursion with a smaller step size. The particle position update is done using the LMS algorithm

$$X_{s,i}(t+1) = X_{s,i}(t) + \mu[d_s(t) - \mathbf{u}_s(t)X_{s,i}(t)]\mathbf{u}_s(t), \quad (3.11)$$

where  $\mu$  is the step size,  $\mathbf{u}_s(t)$  is the input data vector as shown in (2.1) and  $d_s(t)$  is the output of the unknown system obtained as follows:

$$d_s(t) = \mathbf{u}_s(t)\mathbf{w}^o + v_s(t), \quad (3.12)$$

where  $v_s(t)$  is white Gaussian noise and  $\mathbf{w}^o$  is the unknown system parameter vector. At every iteration the particle position update is performed for all the input vectors  $\mathbf{u}_s(t)$  in input data matrix  $U_s(t)$  sequentially.

11. **Stopping criteria:** If the maximum number of allowable iterations is reached then stop; else continue.
12. **Time update:** Update the time counter  $t = t + 1$ .

Goto step **2**.

Repeat steps 2 to 12 at every node  $s, s = 1, 2, \dots, S$ . Finally, Fig. 3.4 summarizes the execution steps of the diffusion PSO-LMS algorithm.

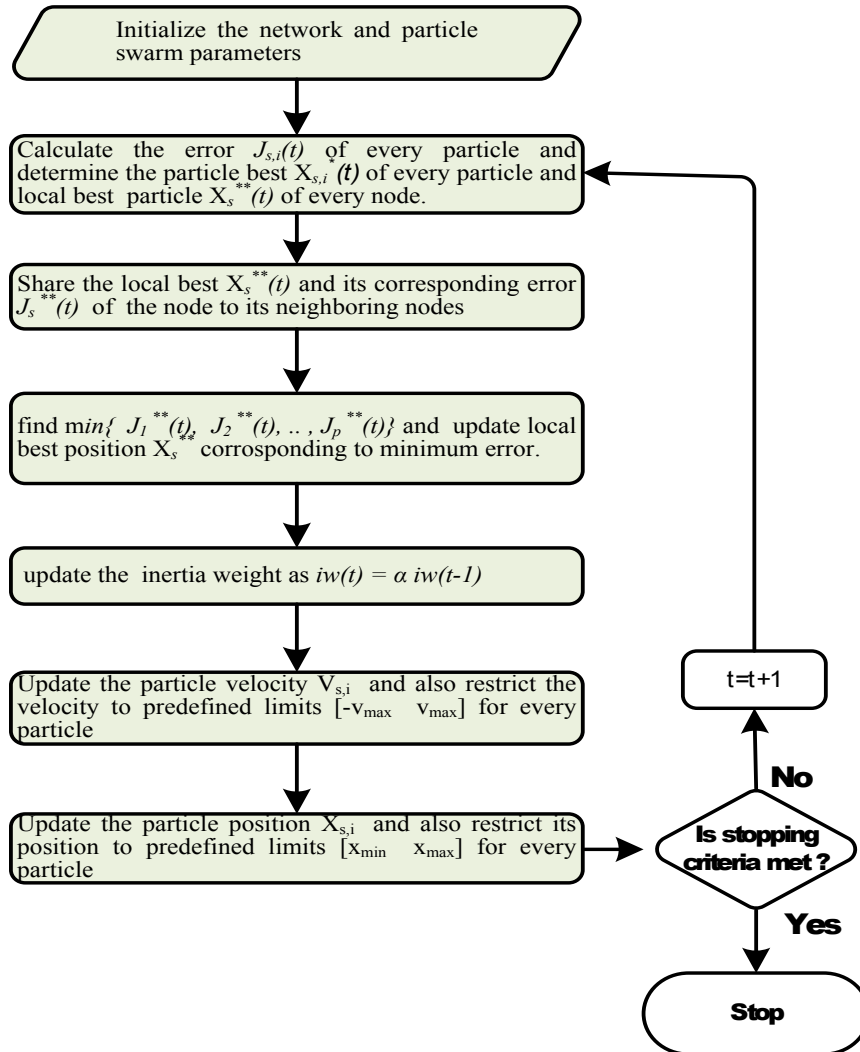


Figure 3.1: Execution steps of DPSO-VIW algorithm.

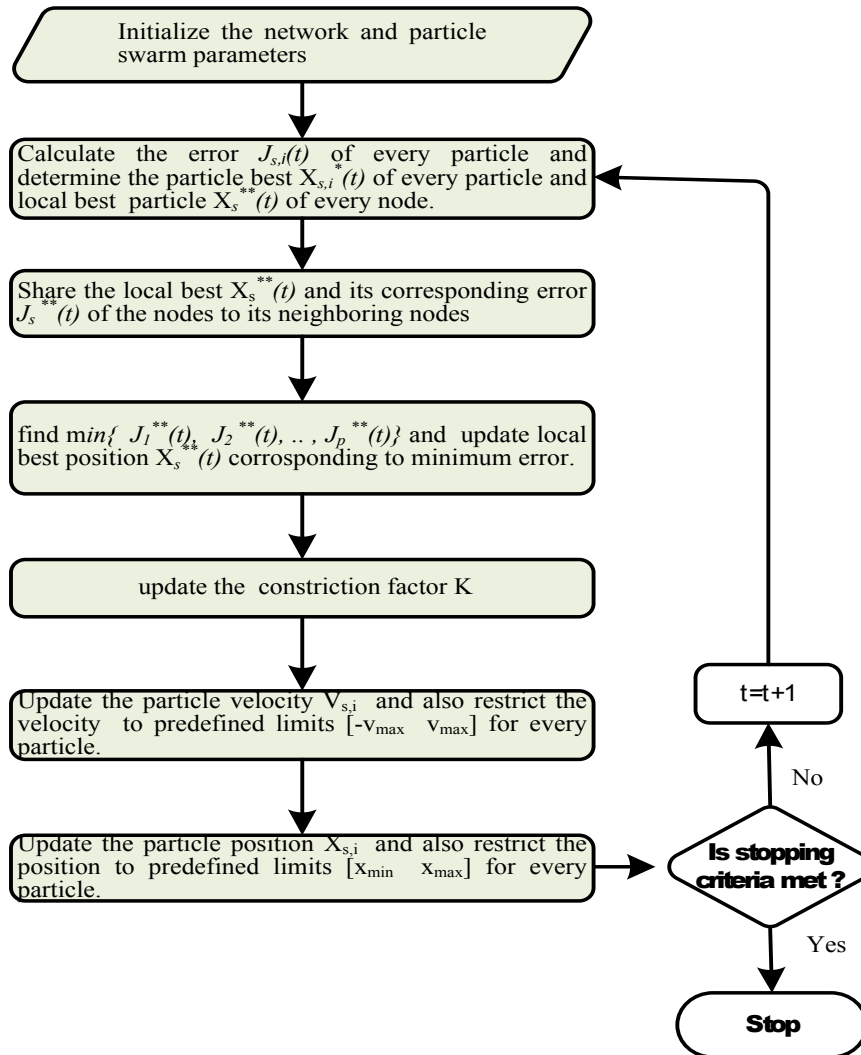


Figure 3.2: Execution steps of DPSO-VCF algorithm.



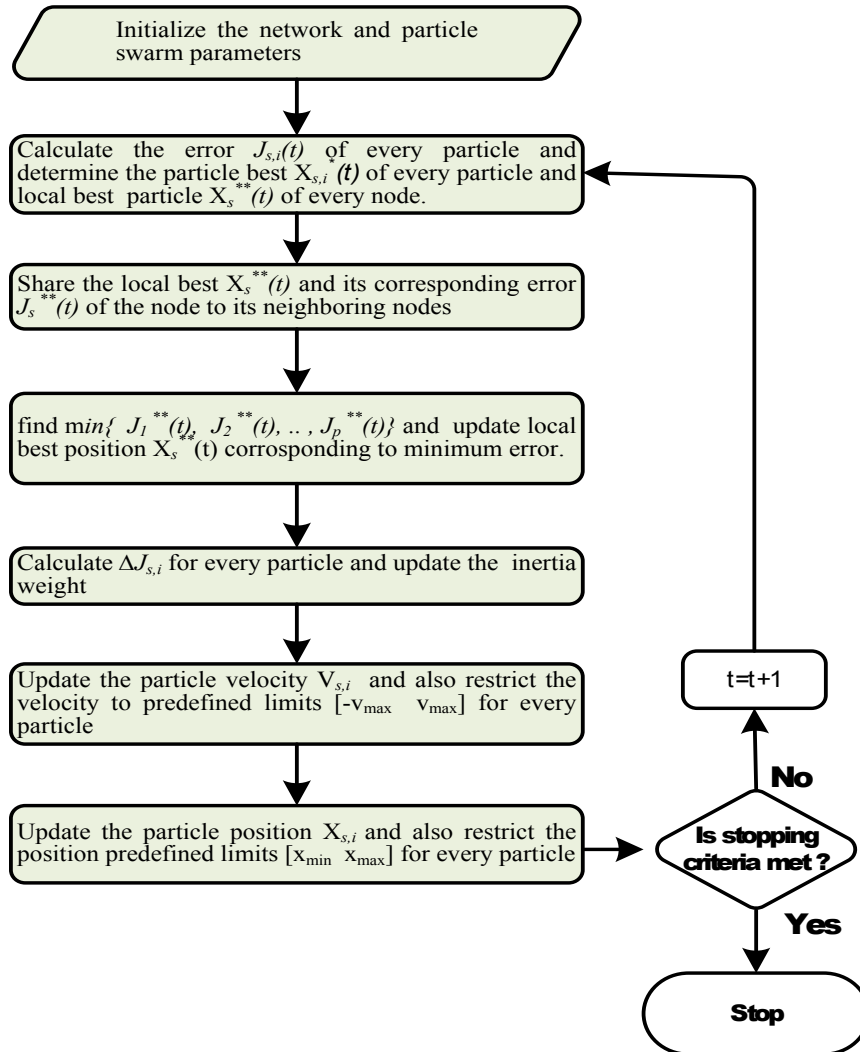


Figure 3.3: Execution steps of DMPSO algorithm.

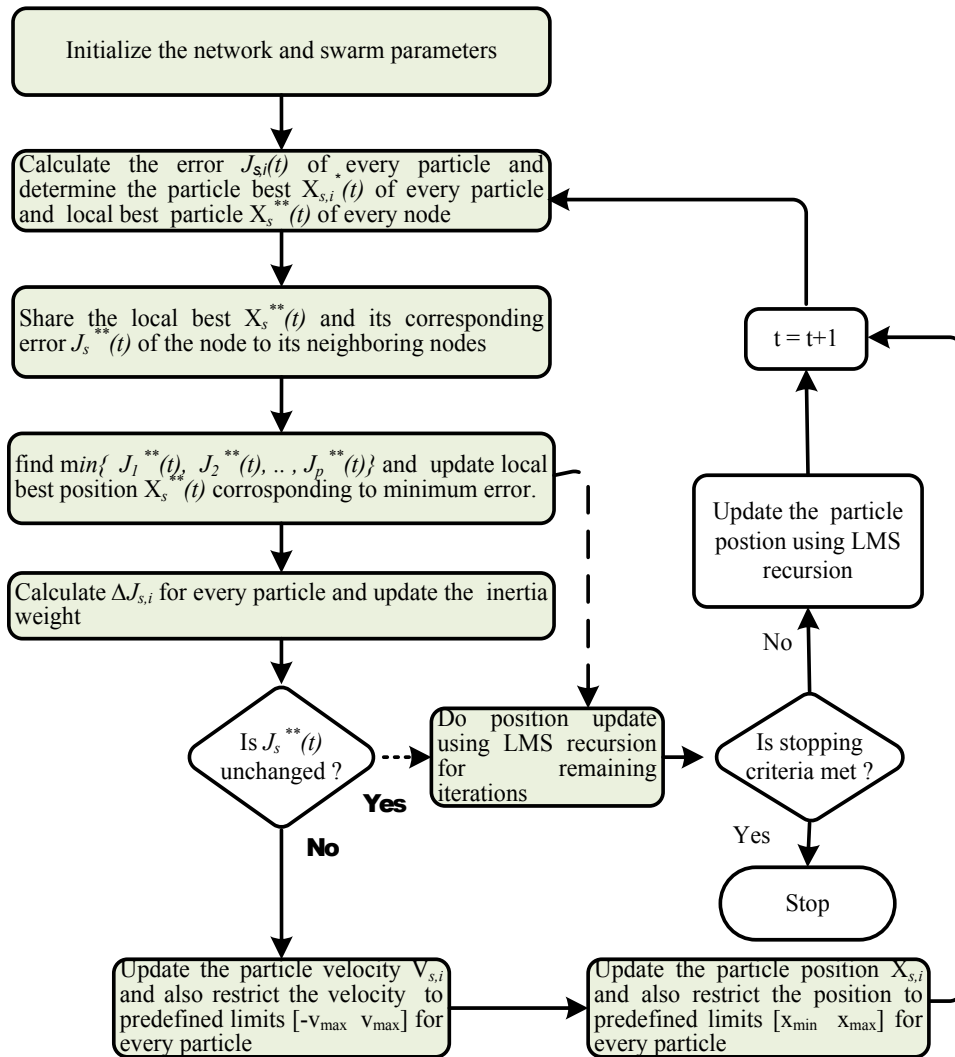


Figure 3.4: Execution steps of DPSO-LMS algorithm.

## CHAPTER 4

# SENSITIVITY ANALYSIS OF THE PROPOSED ALGORITHMS

In this chapter all the proposed algorithms are simulated using a common simulation setup. The MSE and MSD curves are plotted to evaluate the performance of all the algorithms proposed in Chapter 3. Sensitivity analysis of all the proposed algorithms is performed on different network and particle swarm parameters and optimum swarm parameter values are identified. This chapter begins with the simulation setup, followed by the sensitivity analysis on the swarm parameters of DPSO-VIW, DPSO-VCF and DMPSO algorithm. Then in the next

section, the sensitivity analysis on different network parameters such as network size, particle swarm size and input data window size is performed for all the four proposed algorithm.

## 4.1 Simulation Setup

A wireless sensor network is setup with network size of  $S = 20$  sensor nodes as shown in the Fig.4.1, with each node sharing its data with other neighboring nodes in its communication range. A correlated input data block size of  $N = 10$  is used at every iteration and a new data point is replaced with the oldest data value at every iteration as explained previously in System model section of Chapter 2. The noise is assumed to be white. The unknown vector  $\mathbf{w}_0$  ( $M \times 1$ ) is initialized as  $col \{1, 1, \dots, 1\} / \sqrt{M}$ , and the tap size is set to  $M = 4$ . At every node a particle swarm size of  $k = 5$  particles is initialized. The dimension coefficients of the particles is assumed to be uniformly distributed in the range of  $[0, 1]$ . For performance evaluation the mean-square-deviation (MSD) and mean-square-error (MSE). The MSD is defined as the mean squared error between the estimated parameter  $X_s^{**}$  and the unknown parameter  $w_o$  and is given as:

$$MSD = E\|\mathbf{w}_o - X_s^{**}(t)\|^2. \quad (4.1)$$

and mean-square-error(MSE) is calculated at every node using the local best

particle position  $X_s^{**}$  using the equation given as:

$$MSE = [||\mathbf{d}_s - \mathbf{U}_s(t)X_s^{**}(t)||^2]/N \quad (4.2)$$

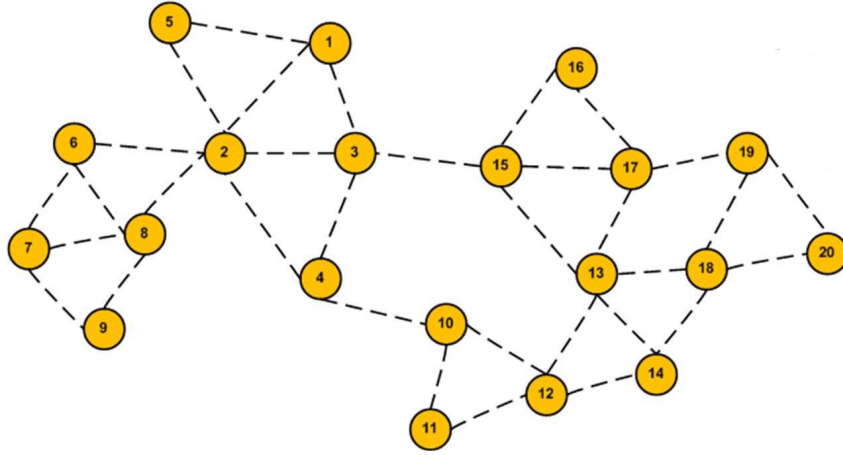


Figure 4.1: A 20 node network

## 4.2 Sensitivity analysis on the swarm parameters of DPSO-VIW algorithm

The sensitivity analysis is performed for different swarm parameters such as acceleration constants  $c_1$  and  $c_2$ , inertia weight  $iw$  and velocity constraint factor  $v_c$  and the results are shown for all these parameters in Fig. 4.2, Fig. 4.3 and Fig. 4.4 respectively. The acceleration constants are varied in range of 0.1 to 2 in steps of 0.1, velocity constraint factor  $v_c$  from 0.1 to 0.6 in steps of 0.05, inertial weight

constant  $iw$  from 0.1 to 2 in steps of 0.1. Using the sensitivity analysis results, the optimum values of swarm parameters are identified and acceleration constants  $c_1$  and  $c_2$ , the inertia weight  $iw$  and the velocity constraint factor  $v_c$  are set to the values given in table 4.1.

parameters	opt. values
$c_1$	1.2
$c_2$	1.2
$iw$	0.8
$v_c$	0.2

Table 4.1: Optimum parameter values of DPSO-VIW algorithm

### 4.3 Sensitivity analysis on swarm parameters of DPSO-VCF algorithm

The sensitivity analysis is carried out for different parameters of DPSO-VCF algorithm using the same the network size, the population size of the particles, the input data window size and the Signal-to-Noise ratio (SNR) used for the DPSO algorithm. The acceleration constants are varied in range of 0.1 to 2 in steps of 0.1, velocity constraint factor  $v_c$  from 0.1 to 0.6 in steps of 0.05 and the constriction factor  $k_{min}$  and  $k_{max}$  is varied in steps of 2. The sensitivity analysis of different parameters such as acceleration constants  $c_1$  and  $c_2$ , the velocity constraint factor  $v_c$  and the constriction factor  $k_{min}$  and  $k_{max}$  are shown in Fig. 4.5, Fig. 4.7 and Fig. 4.6 respectively. Thus the the swarm parameters are set to the optimum

values obtained from the analysis i.e. the acceleration constants  $c_1$  and  $c_2$ , the velocity constraint factor  $v_c$  and the constriction factor  $k_{min}$  and  $k_{max}$  are set to the values shown in table 4.2.

parameters	opt. values
$c_1$	2.5
$c_2$	2.5
$k_{min}$	2
$k_{max}$	5
$v_c$	0.5

Table 4.2: Optimum parameter values of DPSO-VCF algorithm

## 4.4 Sensitivity analysis on swarm parameters of DMPSO algorithm

The sensitivity analysis for different parameters such as acceleration constants  $c_1$  and  $c_2$ , slope constant  $S_l$  and velocity constraint factor  $v_c$  as shown in the Fig. 4.8, Fig. 4.9 and Fig. 4.10 respectively. The swarm parameters are set to the optimum values obtained from the sensitivity analysis of the swarm parameter i.e. the acceleration constants  $c_1$  and  $c_2$ , the slope constant  $S_l$  and the velocity constraint factor  $v_c$  are set the values shown in the table 4.3.

parameters	opt. values
$c_1$	1.8
$c_2$	1.8
$S_l$	1.2
$v_c$	0.1

Table 4.3: Optimum parameter values of DMPSO algorithm

## 4.5 Sensitivity analysis on the network parameters for all the proposed algorithms

Using the optimum swarm parameters given in table 4.1, 4.2, 4.3 for DPSO-VIW, DPSO-VCF, DMPSO and DPSO-LMS respectively, MSD curves are plotted for different network size  $S$ , particle size  $k$  and input data window size  $N$ . In the first scenario, for all the four proposed algorithm, the network size  $S$  is increased in steps of 5 nodes ranging from 5 to 100 and the MSD curves are plotted as shown in the Fig. 4.11, Fig. 4.14, Fig. 4.17 and Fig. 4.20. From the figures it is inferred that, as the network size of sensor nodes in a given area increases, the number of neighbors to a node also increases. Thus more information is shared among the nodes which increases the performance but at the cost of high data processing at every node, since more information is received from the neighboring nodes. In the second scenario the particle population size  $k$  is varied in step of 1 particle ranging from 2 to 20 particles. In the Fig. 4.12, Fig. 4.15, Fig. 4.18, Fig. 4.21 it is shown that as the particle size increases the performance also increases because the increase in the particle density increases the chance of finding the



global minimum easily as larger size of swarm can do an extensive search in the search space and can also help the solution to converge faster. In the final scenario, the data window size is increased in steps of 5 data points ranging from 5 to 100 . In the Fig. 4.13, Fig. 4.16, Fig. 4.19, Fig. 4.22 it is shown that as as the input data window size increases the performance also improves. A large input data window means the estimation error is averaged over larger number of error values which leads to much better error approximation and hence the network performance increases.

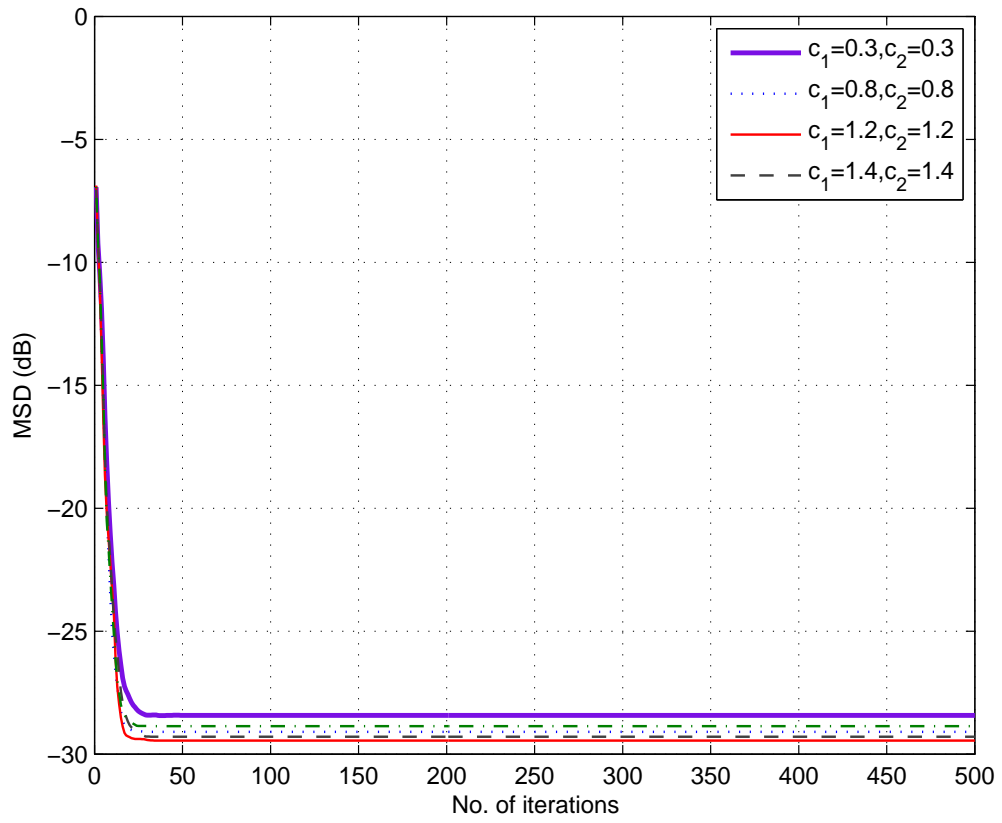


Figure 4.2: MSD curves of DPSO-VIW algorithm for different acceleration constant values at 20 dB SNR.

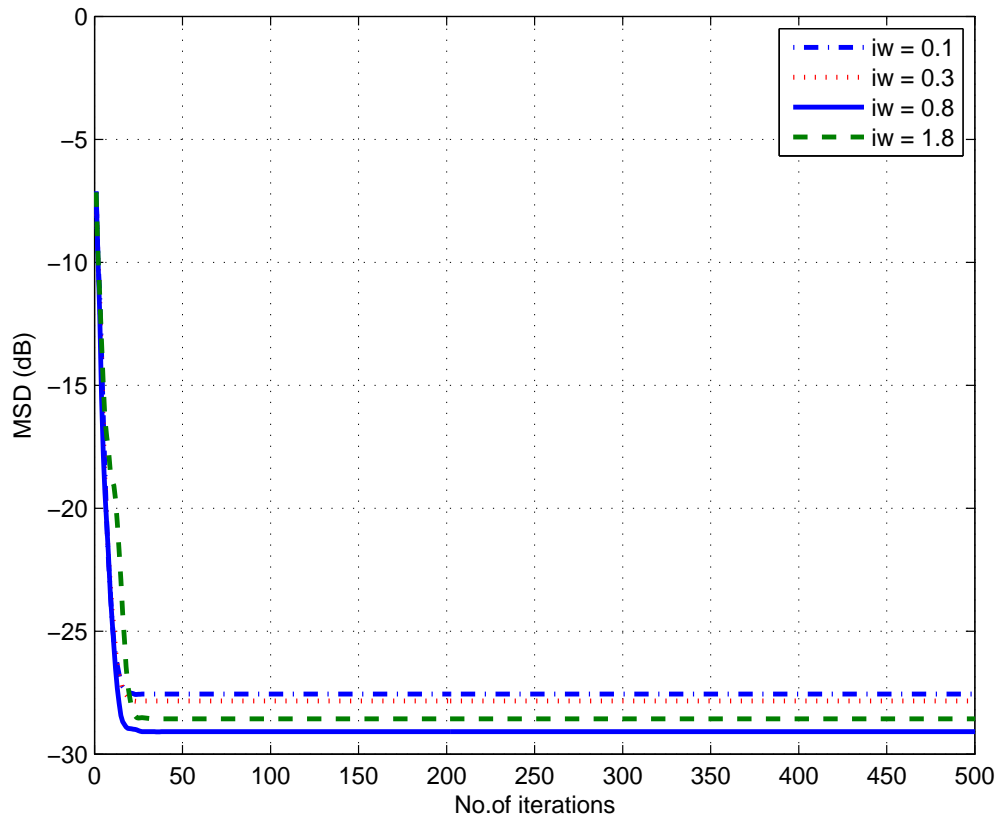


Figure 4.3: MSD curves of DPSO-VIW algorithm for different values of inertia weight at 20 dB SNR.

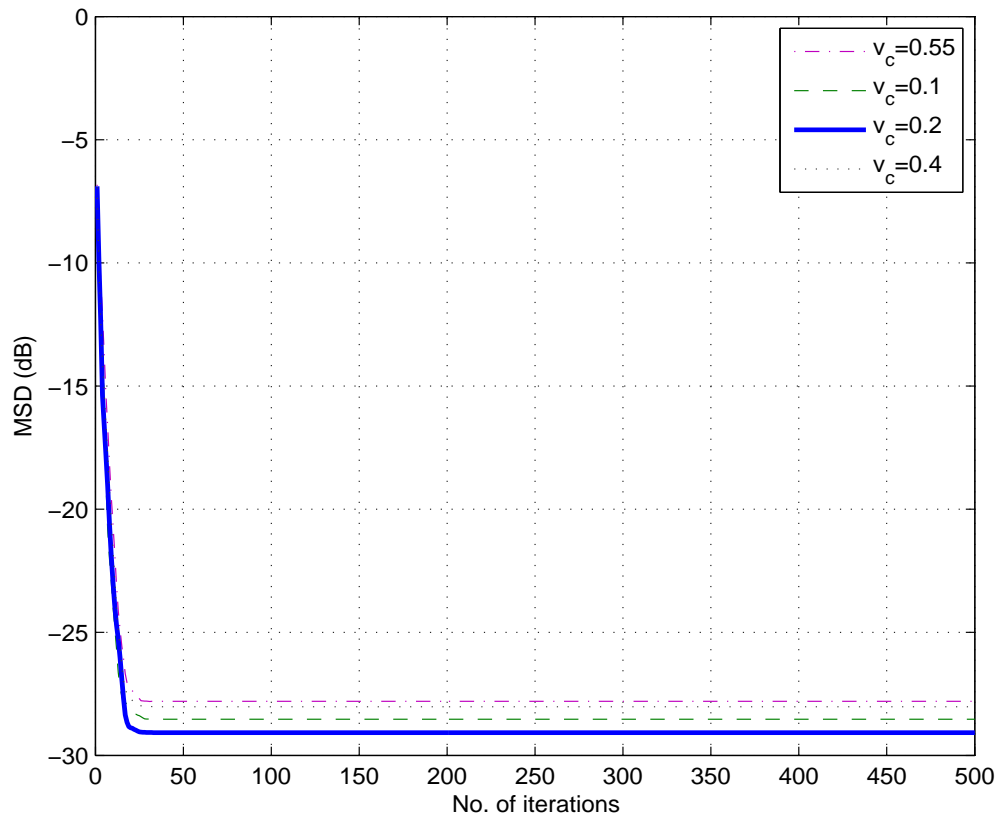


Figure 4.4: MSD curves of DPSO-VIW algorithm for different values of velocity constraint factor at 20 dB SNR.

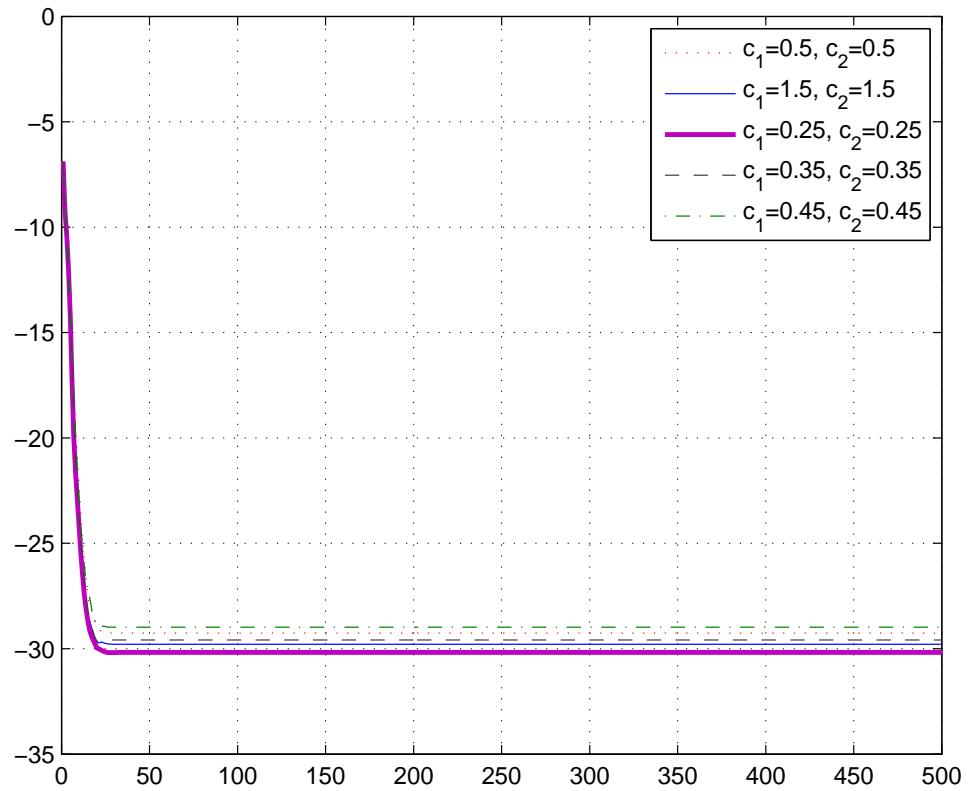


Figure 4.5: MSD curves of DPSO-VCF algorithm for different values of acceleration constant at 20 dB SNR.

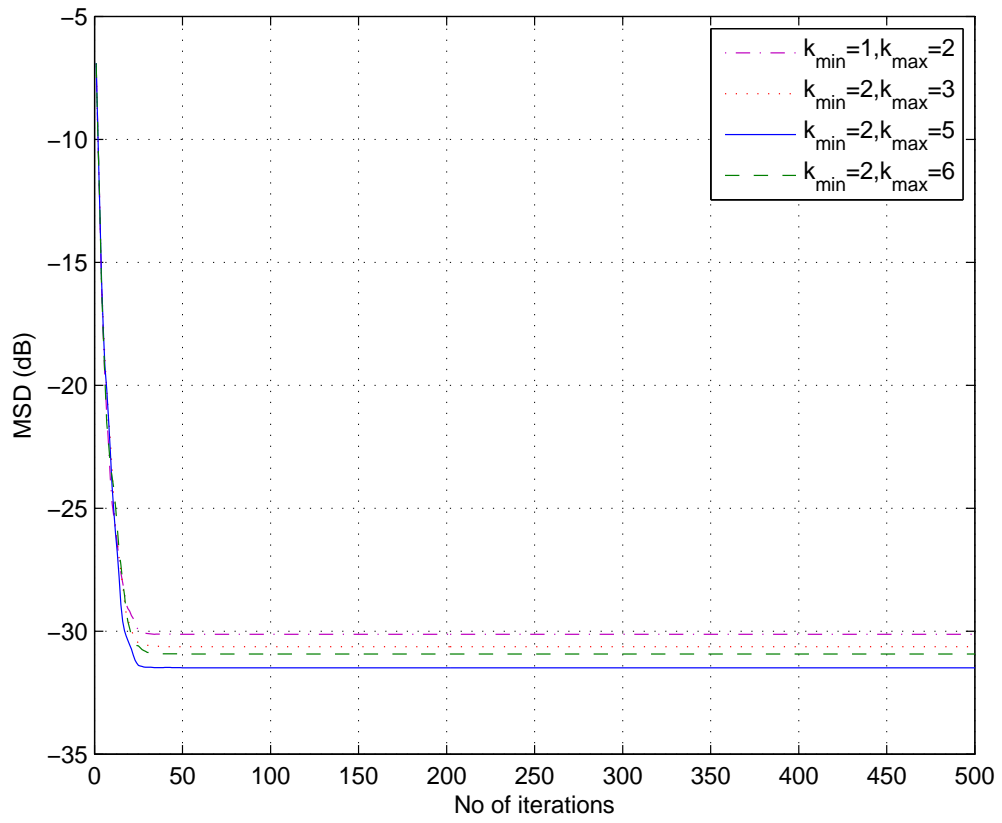


Figure 4.6: MSD curves of DPSO-VCF algorithm for different values of constriction factor limit at 20 dB SNR.

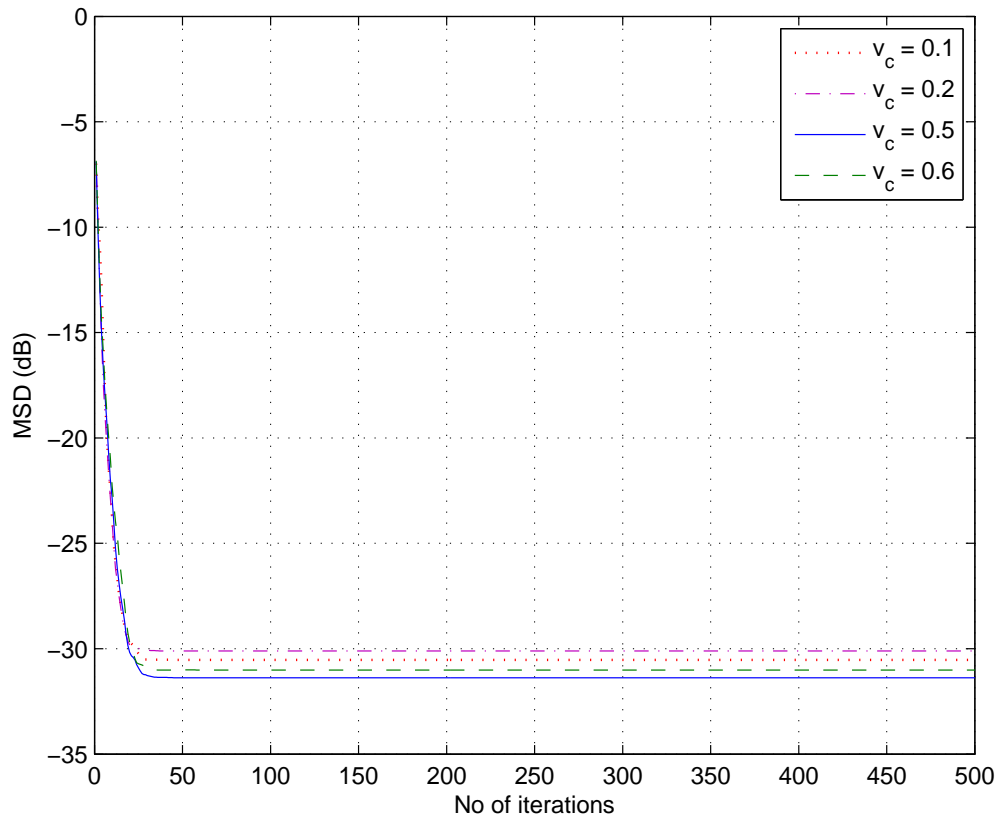


Figure 4.7: MSD curves of DPSO-VCF algorithm for different values of velocity constraint factor at 20 dB SNR.

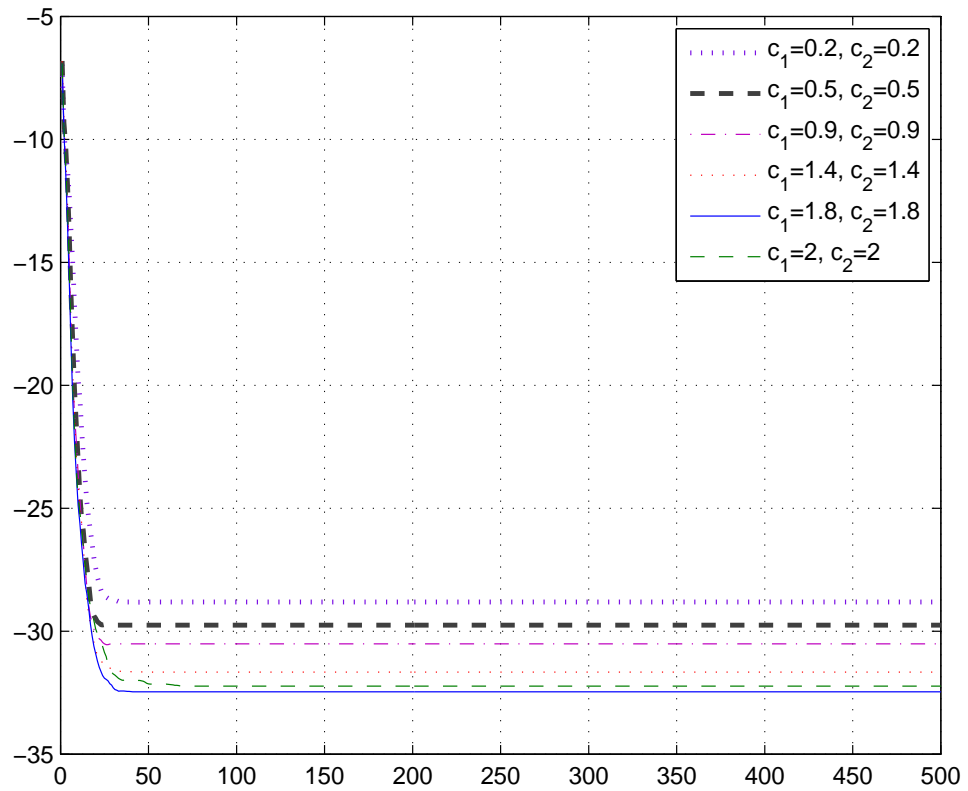


Figure 4.8: MSD curves of DMPSO algorithm for different acceleration constant values at 20 dB SNR.



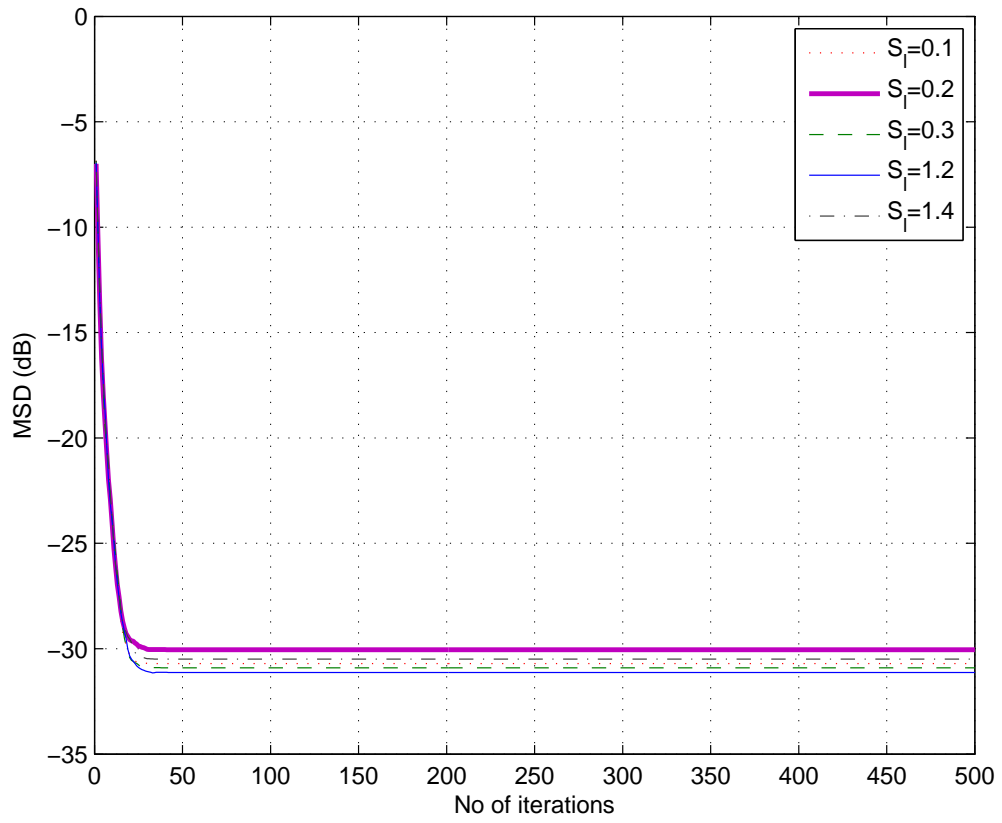


Figure 4.9: MSD curves of DMPSO algorithm for different values of slope constants at 20 dB SNR.

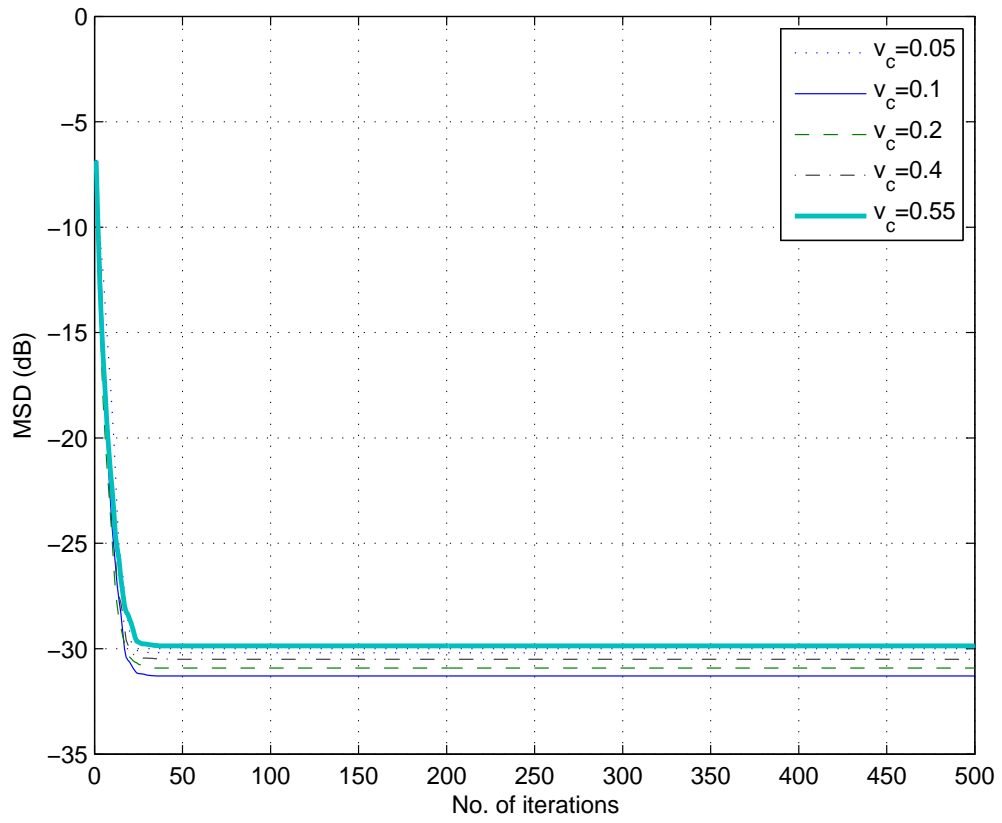


Figure 4.10: MSD curves of DMPSO algorithm for different values of velocity constraint factor at 20 dB SNR.

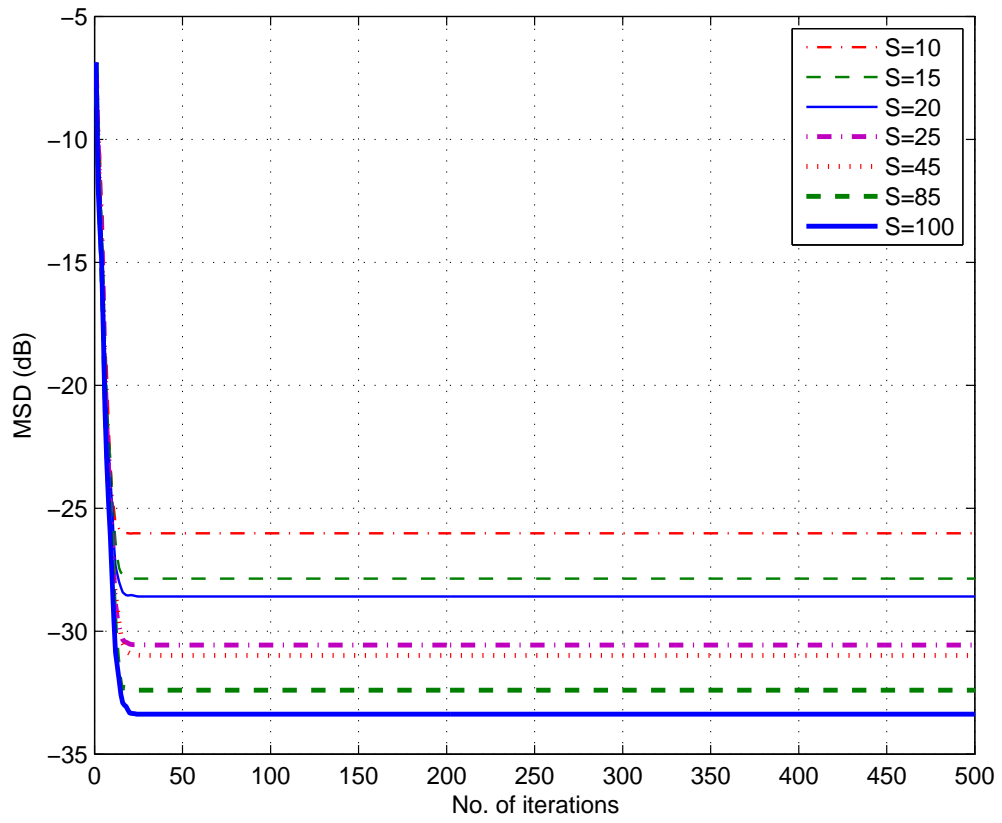


Figure 4.11: MSD curves of DPSO-VIW algorithm for different network size at 20 dB SNR.

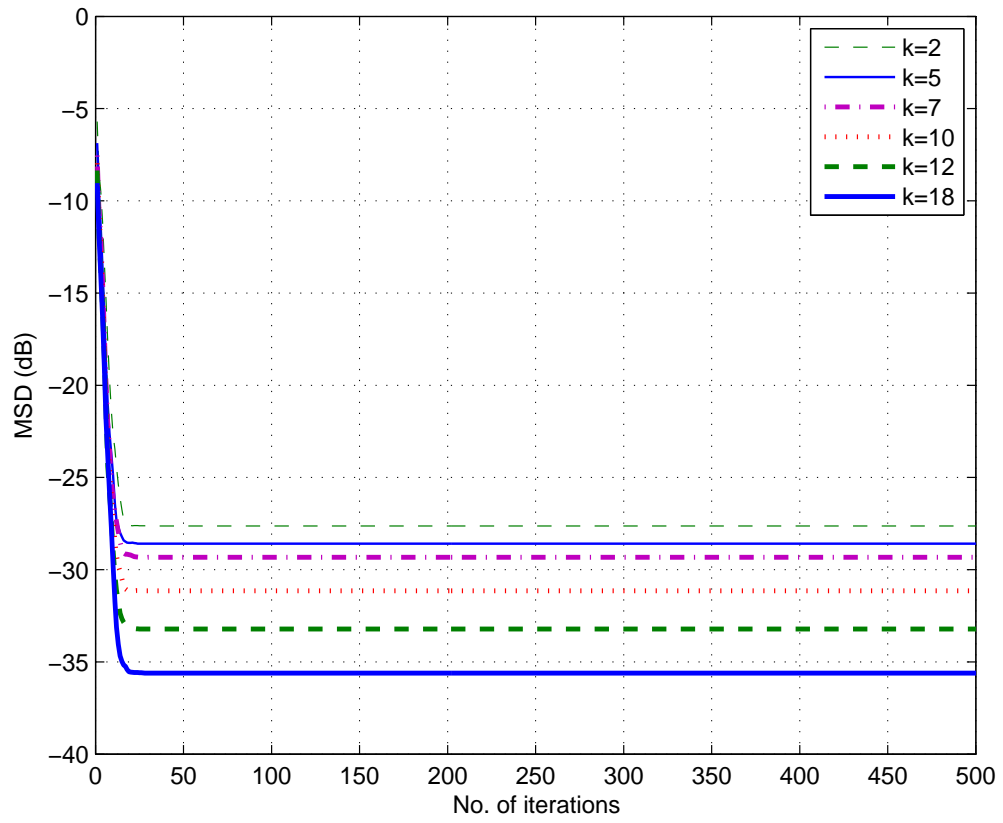


Figure 4.12: MSD curve of DPSO-VIW algorithm for different particle swarm size at 20 dB SNR.

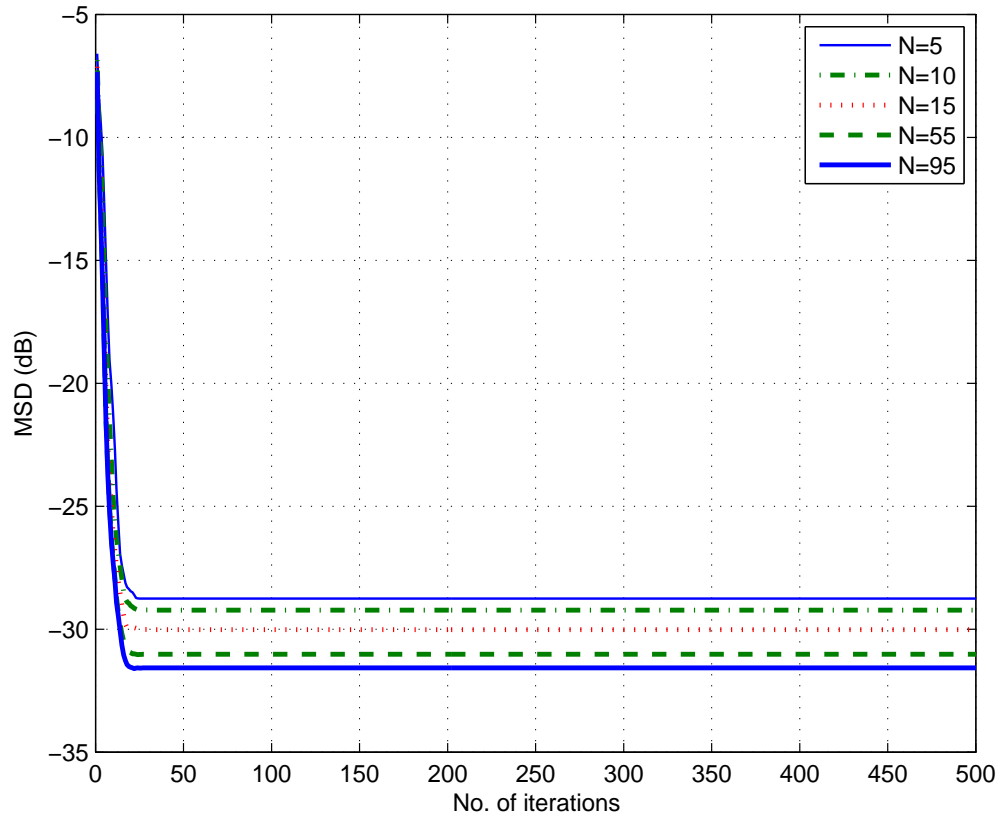


Figure 4.13: MSD curve of DPSO-VIW algorithm for different input data window size at 20 dB SNR.

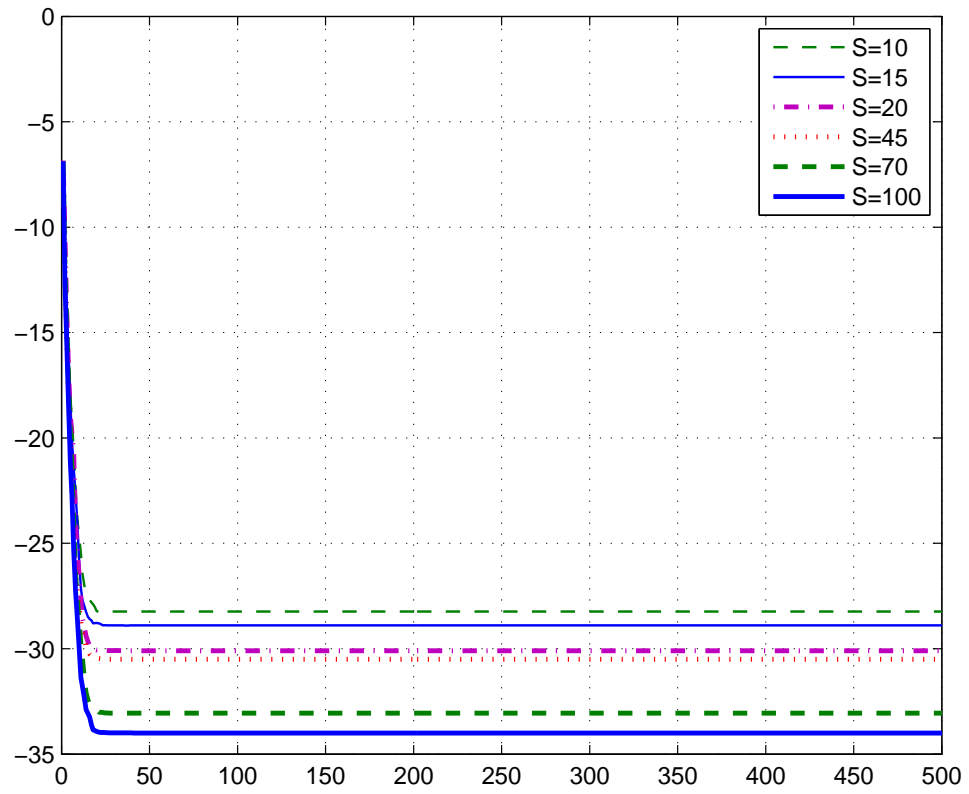


Figure 4.14: MSD curves of DPSO-VCF algorithm for different network size at 20 dB SNR.

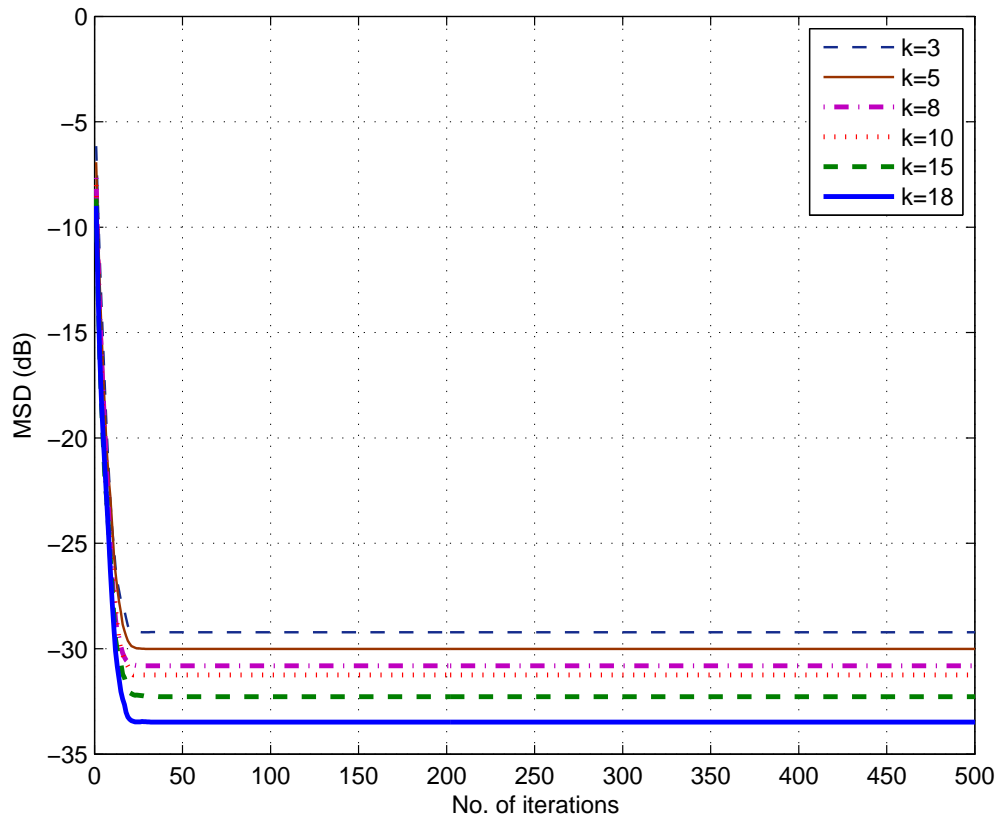


Figure 4.15: MSD curves of DPSO-VCF algorithm for different particle swarm size at 20 dB SNR.

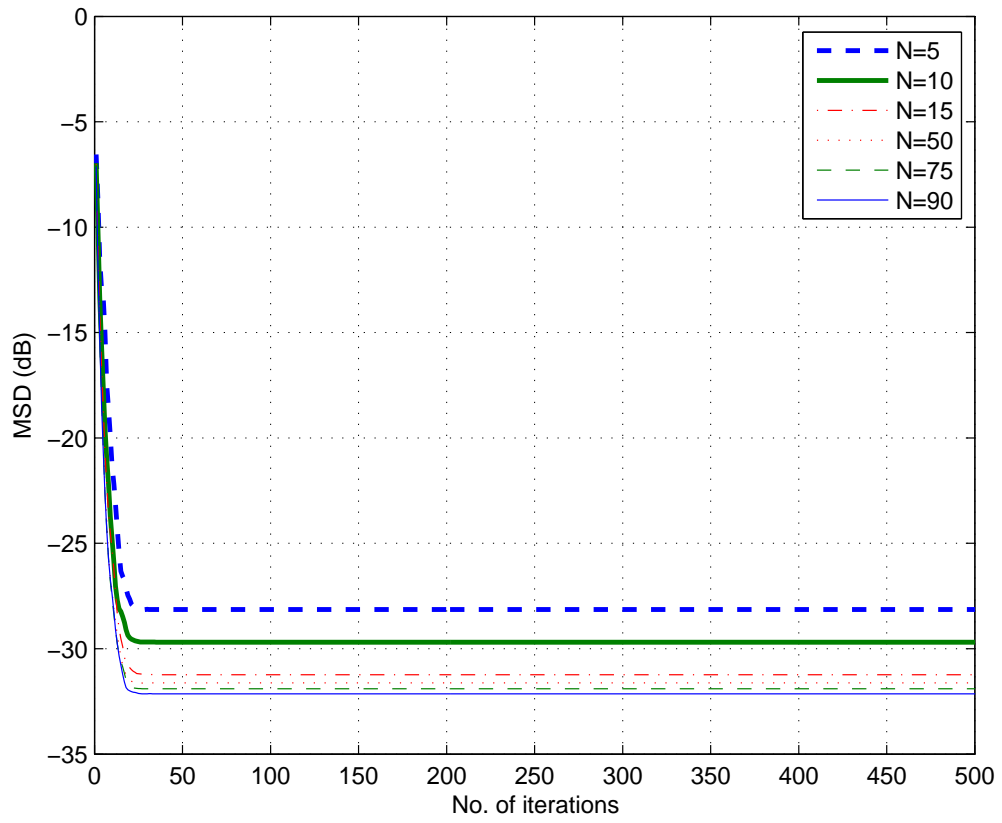


Figure 4.16: MSD curves of DPSO-VCF algorithm for different input data window size at 20 dB SNR.



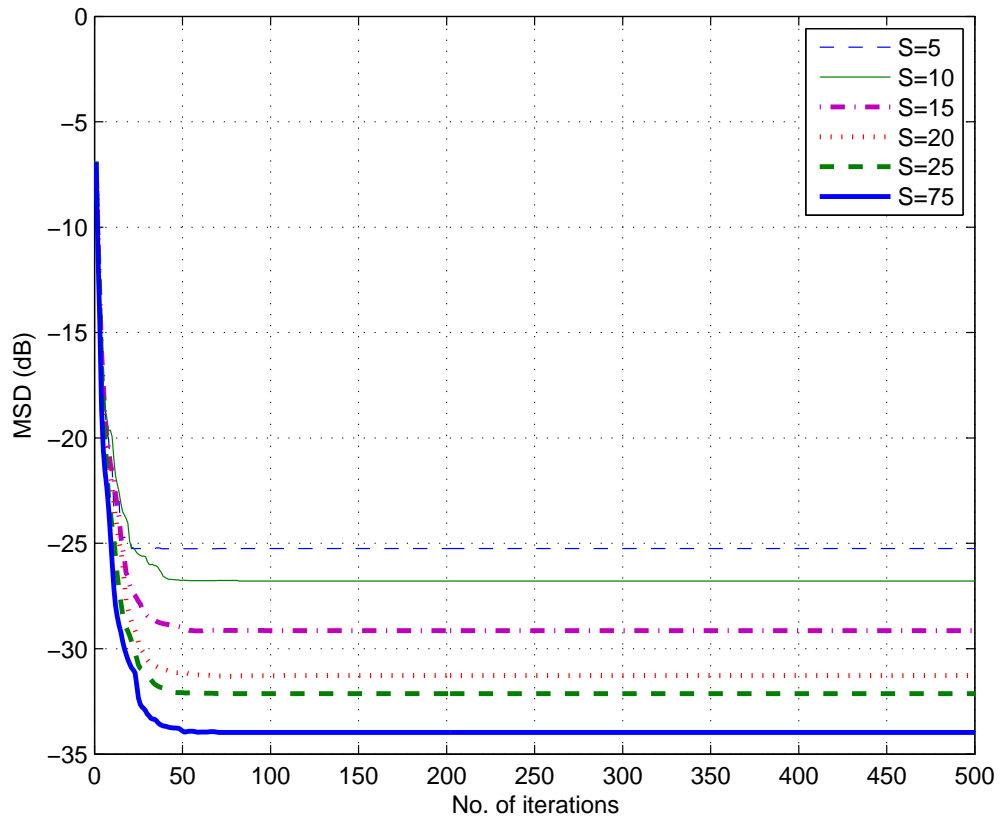


Figure 4.17: MSD curves of DMPSO algorithm for different network size at 20 dB SNR.

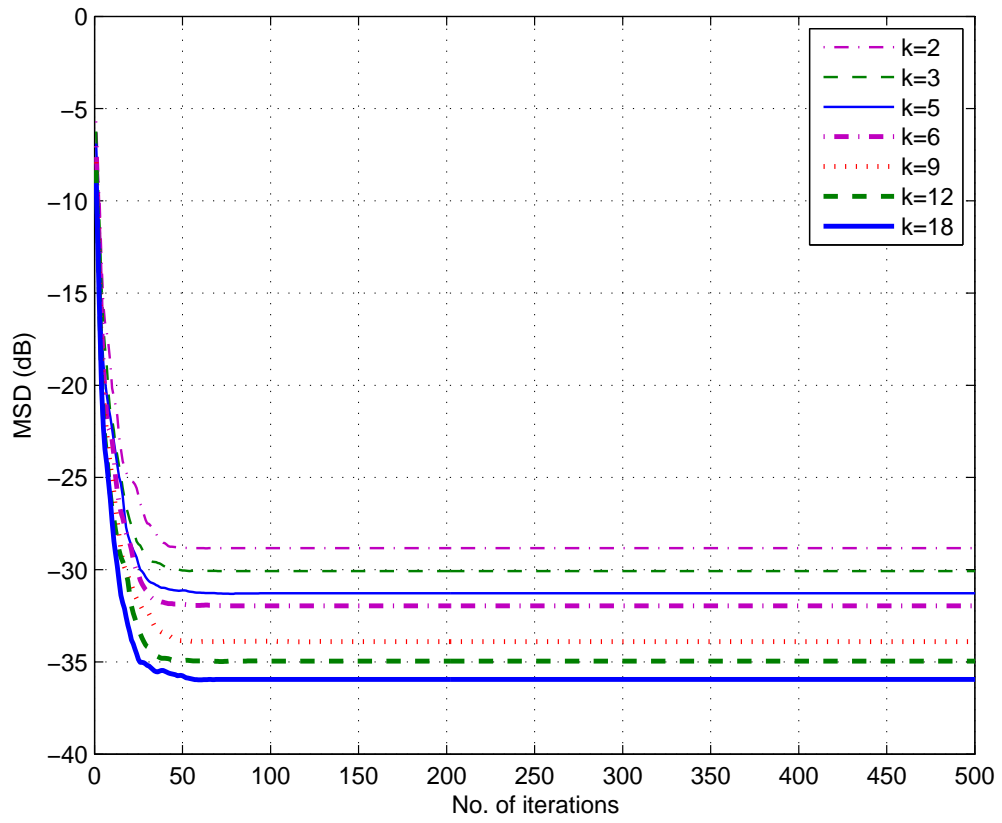


Figure 4.18: MSD curves of DMPSO algorithm for different particle swarm size at 20 dB SNR.

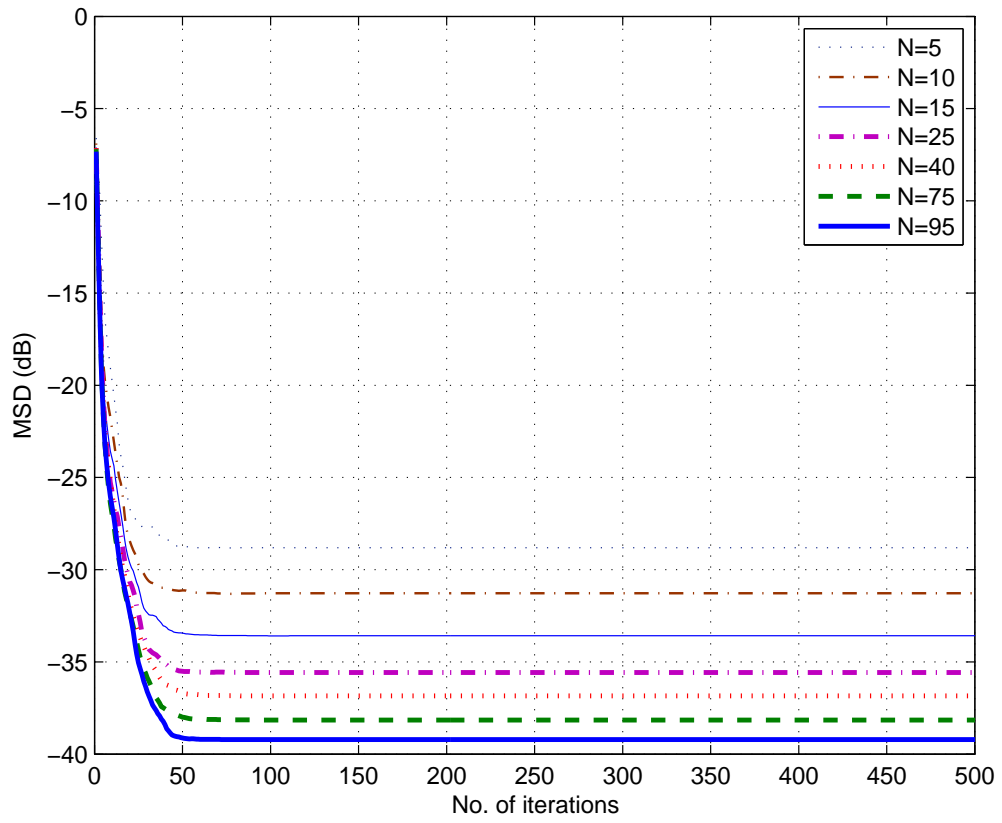


Figure 4.19: MSD curves of DMPSO algorithm for different input data window size at 20 dB SNR.

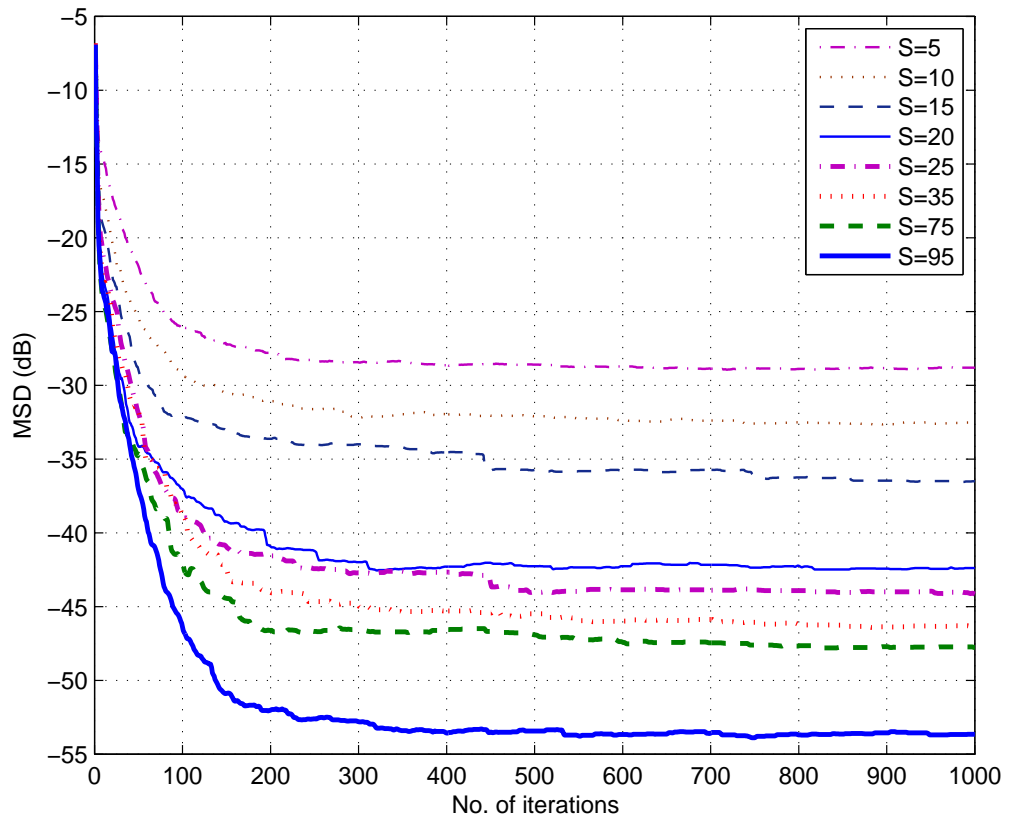


Figure 4.20: MSD curves of DPSO-LMS algorithm for different network size at 20 dB SNR.

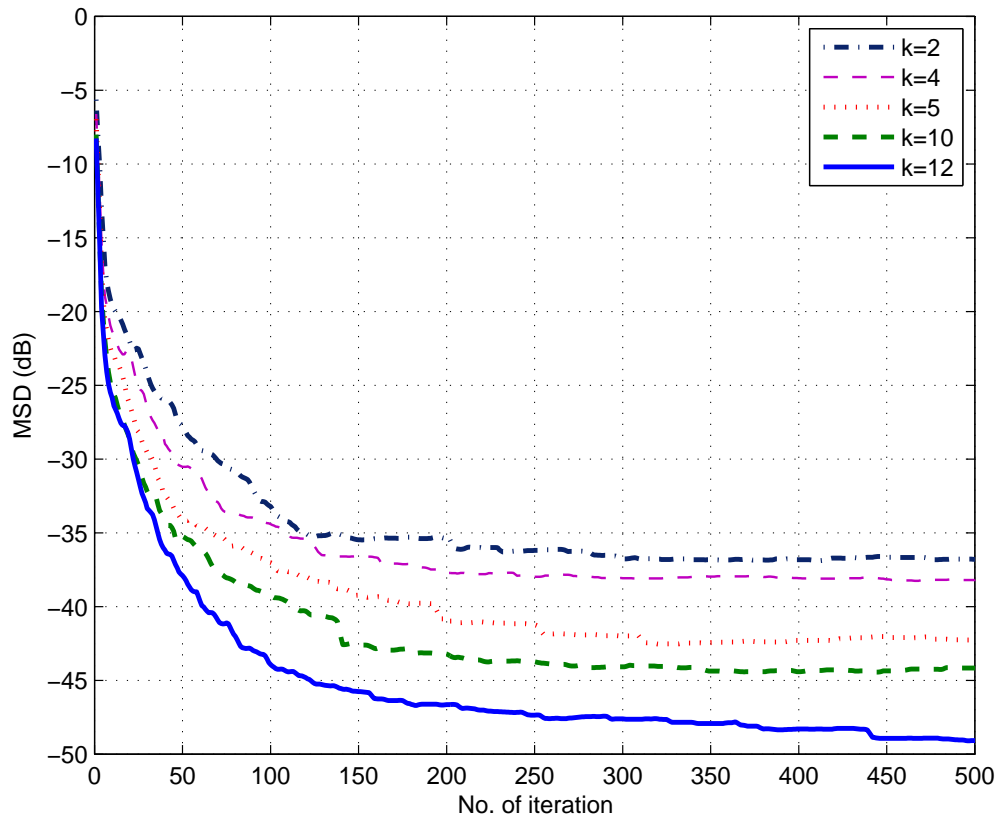


Figure 4.21: MSD curves of DPSO-LMS algorithm for different particle swarm size at 20 dB SNR.

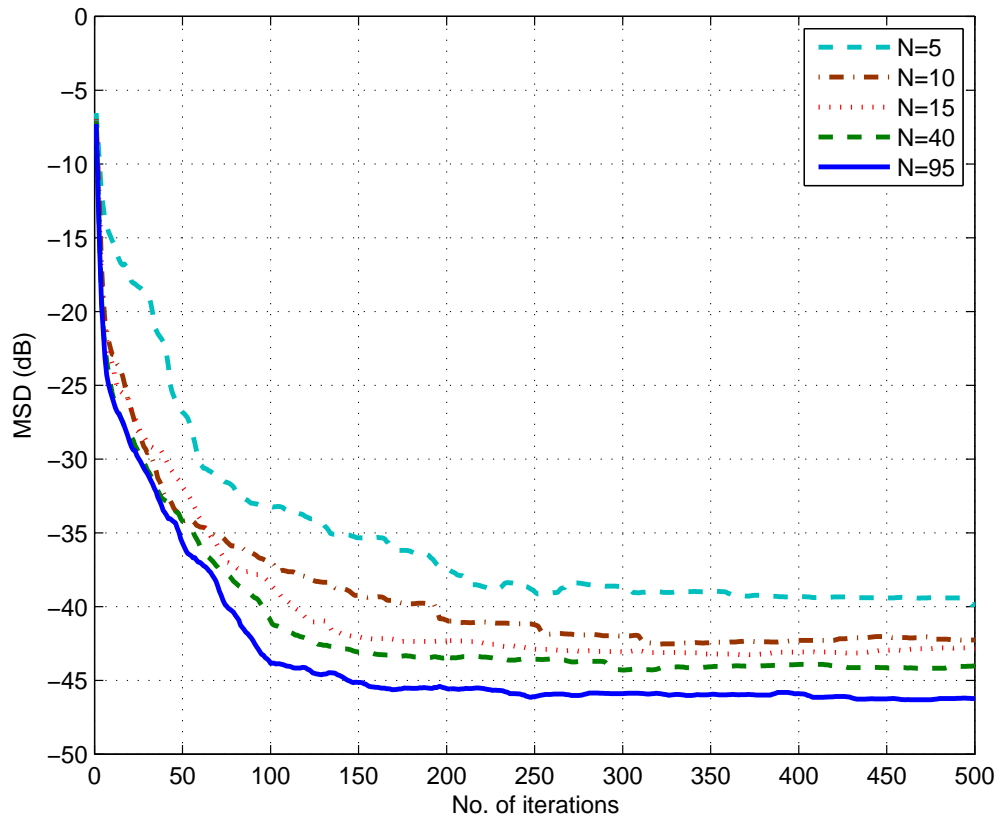


Figure 4.22: MSD curves of DPSO-LMS algorithm for different input data window size at 20 dB SNR.

## CHAPTER 5

# COMPARISON OF THE PROPOSED ALGORITHMS

In this chapter the performance curves of the proposed algorithms are compared with each other and also with the non-cooperative PSO algorithm, the DLMS algorithm and the DRLS algorithm. This chapter begins with the comparison of performance curves of cooperative and non-cooperative PSO algorithm. Then, the performance curves of all the proposed algorithms, DLMS and DRLS algorithms are compared in both training and testing phase of the algorithm. Further, in the next section the steady state error values at each node is compared. In the final section the computational complexity of all the proposed algorithms is calculated and compared.

## 5.1 Comparison of cooperative and non-cooperative PSO algorithm

In Fig. 5.1 and Fig. 5.2, the MSD and MSE curve of the DPSO-VIW algorithm and NCPSO-VIW algorithm are compared. Figure 5.1 depicts the performance of the DPSO-VIW algorithm as compared to that of the non-cooperative PSO algorithm (NCPSO-VIW). As can be seen from this figure, a 15 dB improvement is brought about by the DPSO-VIW algorithm over the NCMPSO-VIW algorithm at both 10 dB and 20 dB SNR. The poor performance of the NCMPSO-VIW algorithm is due to its convergence to a local minimum. The cooperative estimation improves the performance as the proposed algorithm exploits both the spatial and temporal diversity of the network.

The DPSO-VIW algorithm also reduces the computational complexity and improves the performance of PSO algorithm by using smaller swarm size at sensor nodes. This is demonstrated by a scenario where 100 particles at each node is used in a non-cooperative sensor network and 5 particles at each node is used in a cooperative sensor network. For the non-cooperative network the PSO-VIW algorithm is used and for cooperative network DPSO-VIW algorithm is used. The results are reported in Fig. 5.3 and Fig. 5.4. This figure shows that by doing cooperative estimation the computational complexity can be reduced without any performance degradation of the network. This shows that using small population



size of swarms at every node the computational complexity is reduced 20 times without degradation in performance of the DPSO algorithm. Thus cooperative estimation reduces the computational complexity and improve the performance of PSO-VIW algorithm by employing smaller swarm size at sensor nodes.

## 5.2 Comparison of algorithms in training phase

The performance of previously proposed diffusion algorithms such as diffusion LMS [18] and diffusion RLS [20] algorithms are compared with the newly proposed diffusion algorithms such as DPSO-VIW, DPSO-VCF, DMPSO and DPSO-LMS. The MSD and MSE curves for both 10 dB and 20 dB signal-to-noise (SNR) ratio are plotted in Fig. 5.5, Fig. 5.6, Fig. 5.7 and Fig. 5.8 respectively.

The MSD curves in the Fig. 5.5 and Fig. 5.7 shows that DPSO-VIW algorithm performs about 5 dB better than DRLS algorithm, DPSO-VCF algorithm performs about 2-dB better than DPSO-VIW algorithm, DMPSO algorithm performs about 3.5 dB better than DPSO-VCF algorithm and DPSO-LMS algorithm outperforms all the algorithms by a wide margin.

The MSE curves in the Fig. 5.6 and Fig. 5.8 shows about 0.5 dB better performance for DPSO-VIW algorithm than DRLS algorithm, 0.3 dB better performance for DPSO-VCF algorithm than DPSO-VIW algorithm, DPSO-LMS algorithm shows about 0.5 dB better performance than DPSO-VCF algorithm and

and DMPSO algorithm. The steady state MSD and MSE values at both 10-dB and 20-dB SNR are summarized in the table 5.1 for better comparison of the error values. The DPSO-LMS algorithm outperforms DLMS algorithm and DRLS algorithm at the cost of higher computational complexity.

Algorithms	SNR = 10-dB		SNR = 20-dB	
	MSD (dB)	MSE (dB)	MSD (dB)	MSE (dB)
DLMS	-10	-9	-20	-18.85
DRLS	-14	-9.4	-24	-19.4
DPSO-VIW	-19	-9.6	-29	-19.7
DPSO-VCF	-21	-9.8	-31	-19.9
DMPSO	-22.5	-9.9	-33.5	-20
DPSO-LMS	-31	-10.1	-41.5	-20.4

Table 5.1: Steady state MSD and MSE values of DPSO-LMS compared to other algorithms at 10-dB and 20-dB SNR

### 5.3 Comparison of the algorithms in testing phase

The parameter estimated by the proposed algorithms are tested for varying input data. The Fig. 5.9 and Fig. 5.10 shows MSE curves obtained in the testing phase for all the proposed algorithms at both 10 dB and 20 dB SNR respectively. When compared to the steady state error values shown in Fig. 5.6 and Fig 5.8, all the proposed algorithm in testing phase shows similar performance. This confirms the accuracy of the estimated parameter value and functioning of the proposed algorithms.

## 5.4 Comparison of the algorithms in steady state

The steady state MSD and MSE values at each node in the network is calculated for proposed algorithms using simulation setup given in Chapter 4. As shown in Fig 5.11, 5.13, Fig 5.12 and 5.14 the steady state MSD and MSE values are plotted at both 10 dB and 20 dB SNR for the DLMS, DRLS and the newly proposed algorithms. The figures shows that MSD remains constant at all nodes for all the diffusion PSO algorithms whereas for DLMS algorithm and DRLS algorithm there is about 5-dB of error variation at the nodes.

## 5.5 Comparison of computational complexity

The computational complexity of all the proposed algorithms is evaluated by computing the number of multiplications and additions required at each node for all the proposed algorithms. The approximate number of multiplications ( $N_m$ ) and additions ( $N_a$ ) for every iteration of the proposed algorithms at a node is given in the table as: In order to compare the computational complexity of different

Algorithms	$N_m$	$N_a$
NCPSO-VIW	$k(N(M + 1) + 3M) + 5$	$k(N(M - 1) + N + 3M) + (2k + 4Mk + 1)$
DPSO-VIW	$k(N(M + 1) + 3M) + 5$	$k(N(M - 1) + N + 3M) + (2k + 4Mk + p + 1)$
DPSO-VCF	$k(N(M + 1) + 3M) + 12$	$k(N(M - 1) + N + 3M) + (2k + 4Mk + p + 8)$
DMP SO	$k(N(M + 1) + 3M) + 7$	$k(N(M - 1) + N + 3M) + (2k + 4Mk + p + 3)$
DPSO-LMS	$k(N(2M + 1)) + 7$	$k(2NM) + (2k + 4Mk + p + 3)$

Table 5.2: Number of computations required by the proposed algorithms

proposed algorithms, network parameter values used in prior simulations are used. Therefore the input window size of  $N = 10$ , particle swarm size of  $k = 5$ , particle dimension of  $M = 4$  and number of neighboring nodes  $p = 4$  are substituted in the equations given in table 5.2 to compute the number of Multiplication and number of additions required by each of the proposed algorithm at a node. The calculated computations for each of the proposed algorithm are listed in table 5.3. The values in the table 5.3 shows that the computational complexity of all the proposed algorithms remains more or less the same except for the DPSO-LMS algorithms which requires some additional computations. However DPSO-LMS algorithm outperforms other proposed algorithms by a greater margin as shown in the Fig 5.8.

	NCPSO-VIW	DPSO-VIW	DPSO-VCF	DMPSO	DPSO-LMS
$N_m$	315	315	322	317	457
$N_a$	351	355	362	357	497

Table 5.3: Computational cost of the proposed algorithms

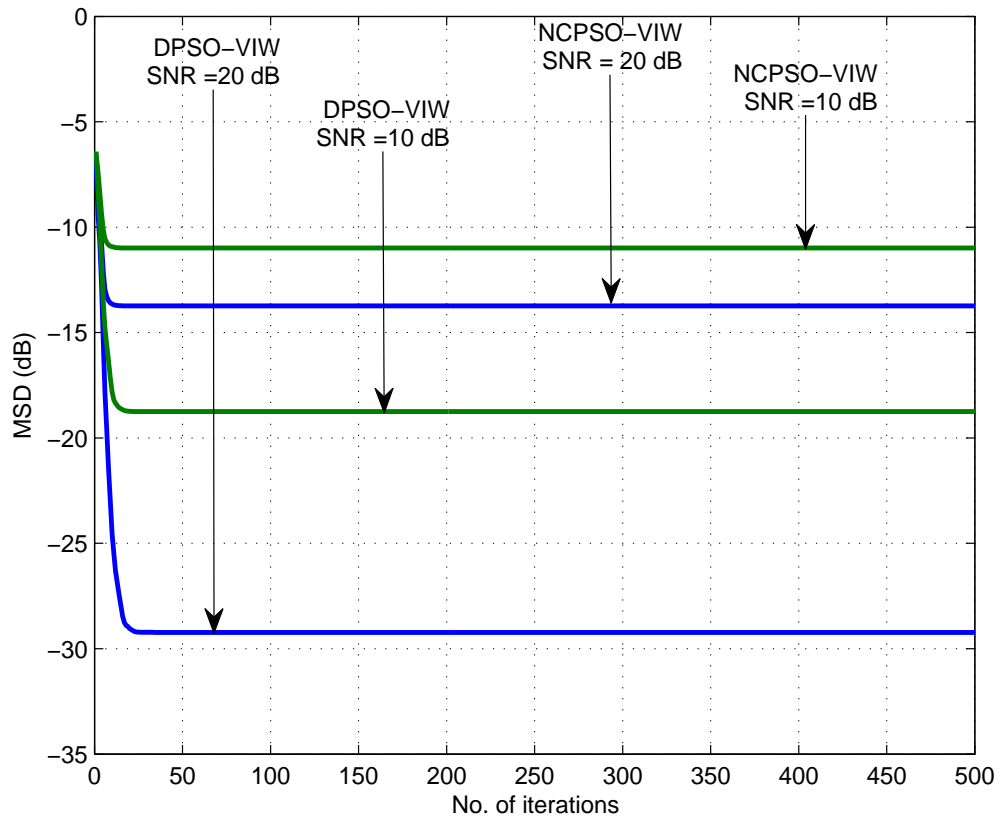


Figure 5.1: Comparison of MSD curves of NCPSO-VIW ( $k=5$ ) algorithm with DPSO-VIW ( $k=5$ ) algorithm at network size of 20 nodes

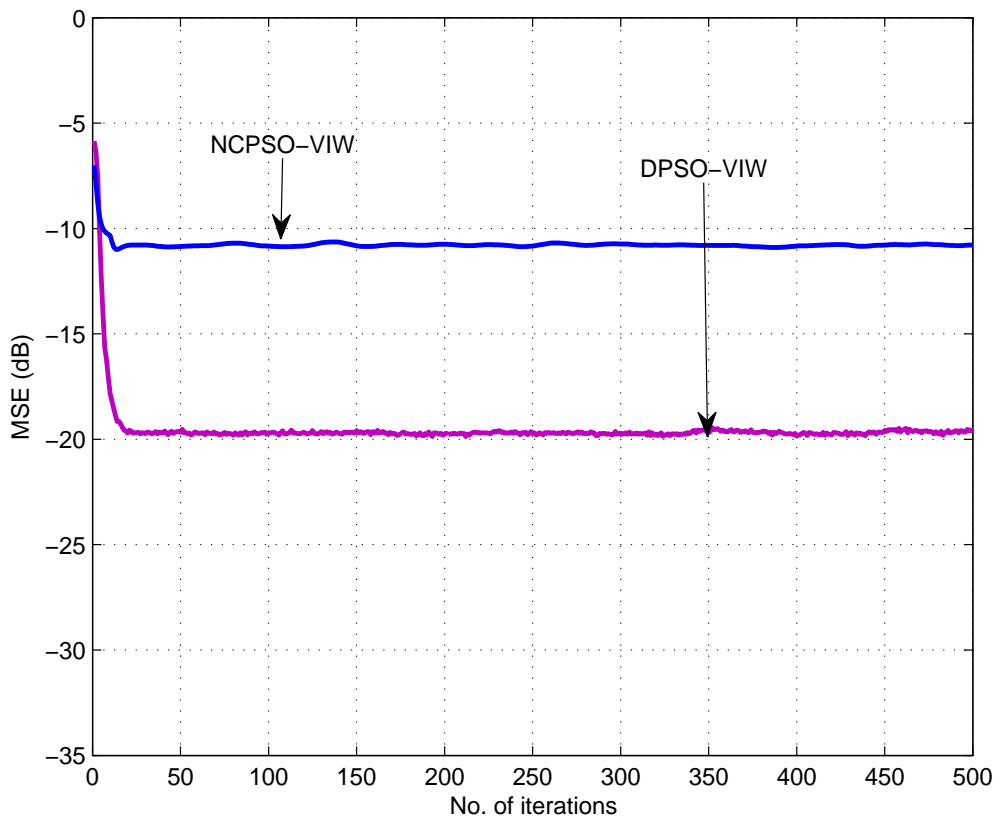


Figure 5.2: Comparison of MSE curves of DPSO-VIW ( $k=5$ ) algorithm with NCPSO-VIW ( $k=5$ ) at 20 dB SNR and at network size of 20 nodes.

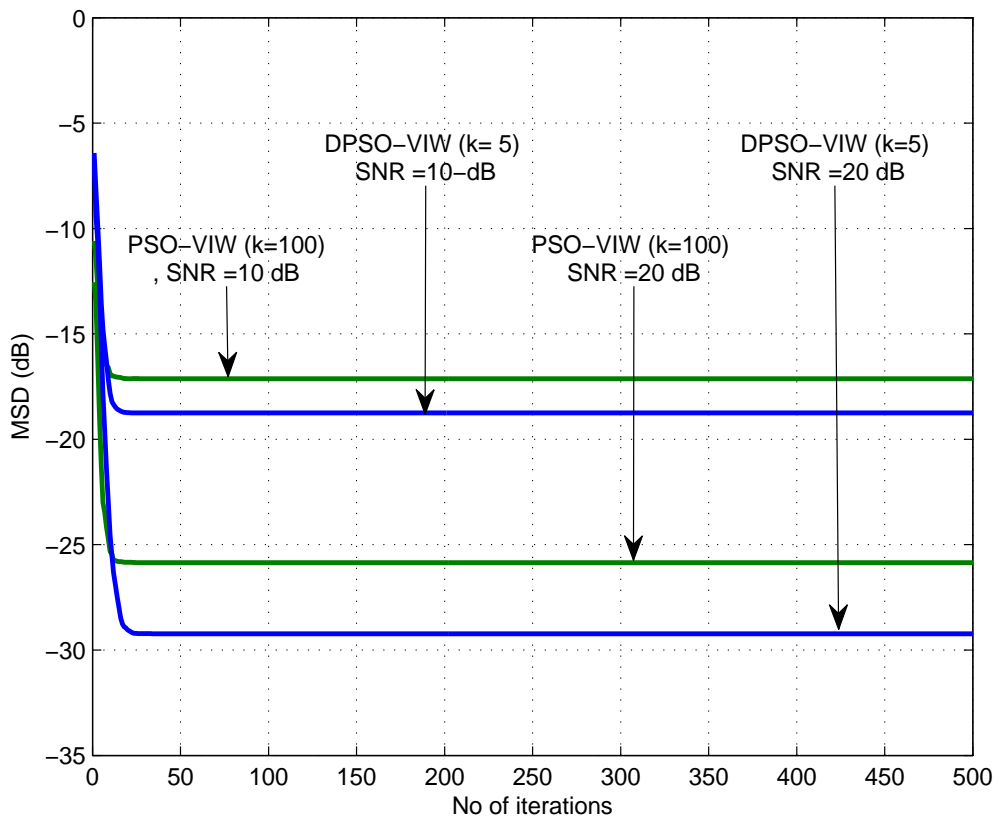


Figure 5.3: Comparison of MSD curves of NCP SO-VIW algorithm ( $k=100$ ) with DPSO-VIW algorithm ( $k=5$ ) at network size of 20 nodes.

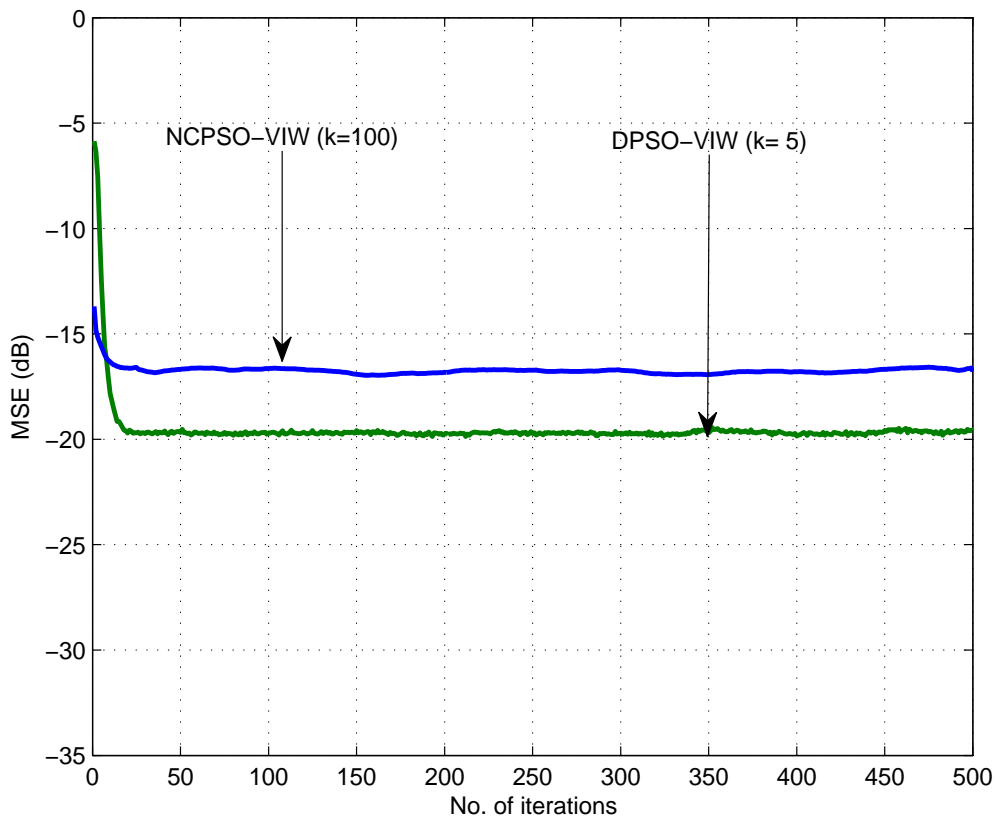


Figure 5.4: Comparison of MSE curves of NCP SO-VIW algorithm ( $k=100$ ) with DP SO-VIW algorithm ( $k=5$ ) at 20 dB SNR and at network size of 20 nodes.



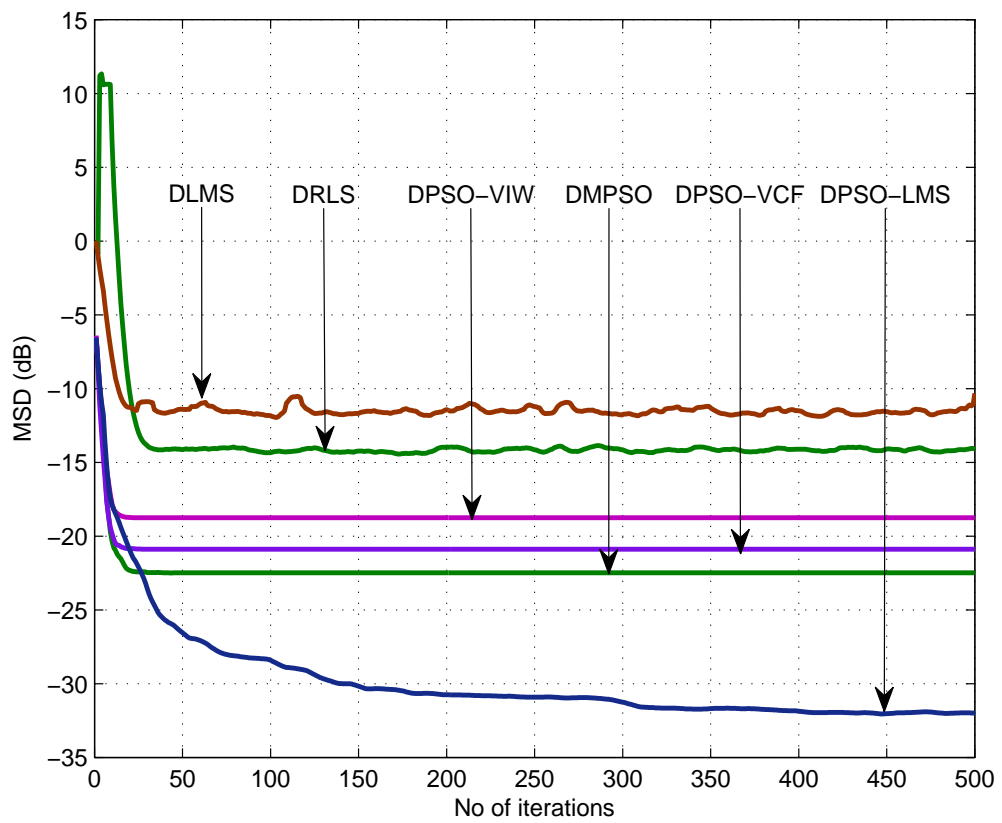


Figure 5.5: Comparison of MSD curves of different algorithms at 10 dB SNR and at network size of 20 nodes

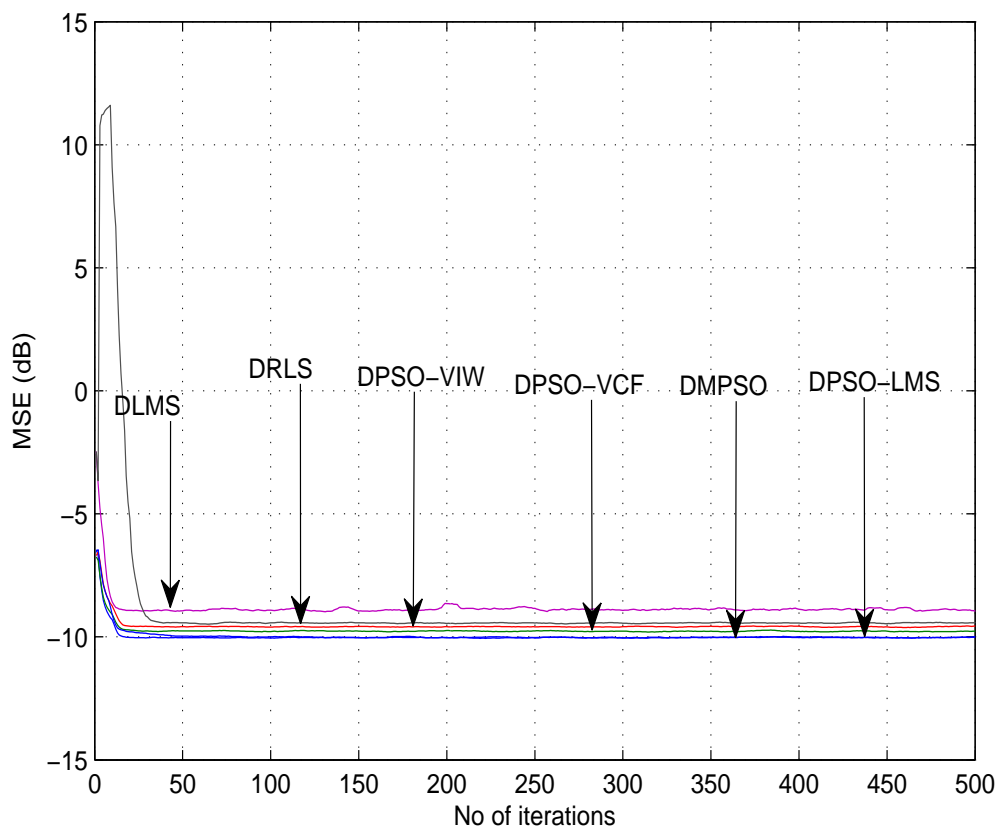


Figure 5.6: Comparison of MSE curves of different algorithms at 10 dB SNR and at network size of 20 nodes.

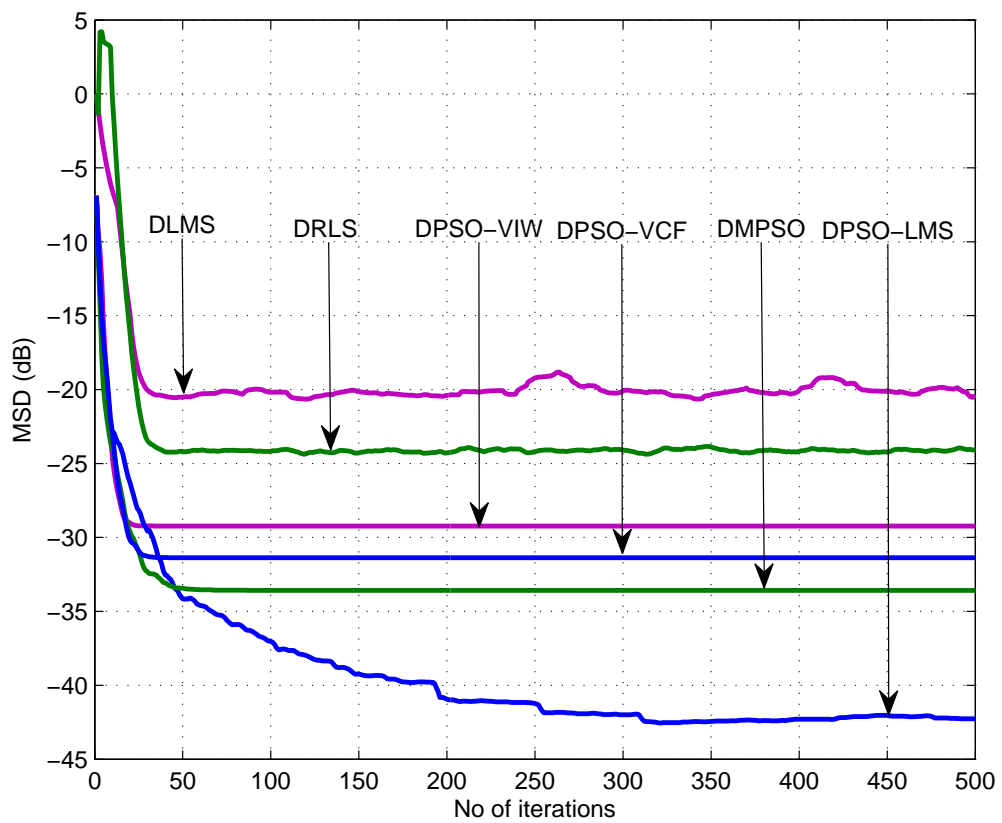


Figure 5.7: Comparison of MSD curves of different algorithms at 20 dB SNR and at network size of 20 nodes

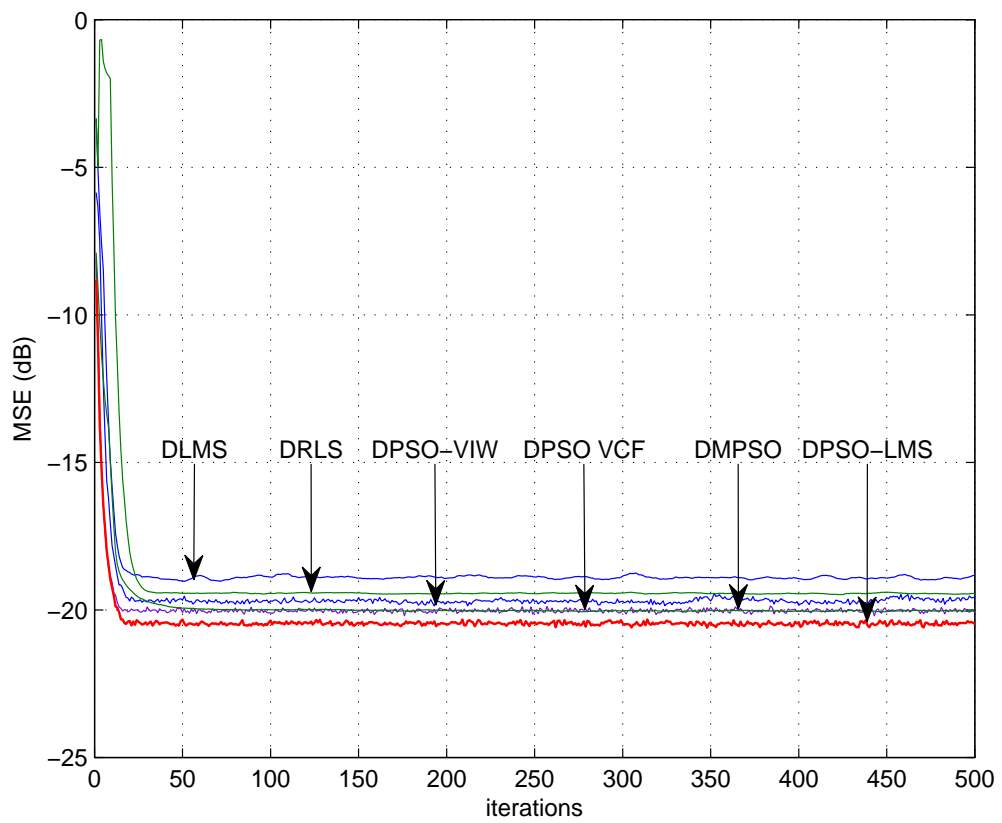


Figure 5.8: Comparison of MSE curves of different algorithms at 20 dB SNR and at network size of 20 nodes.

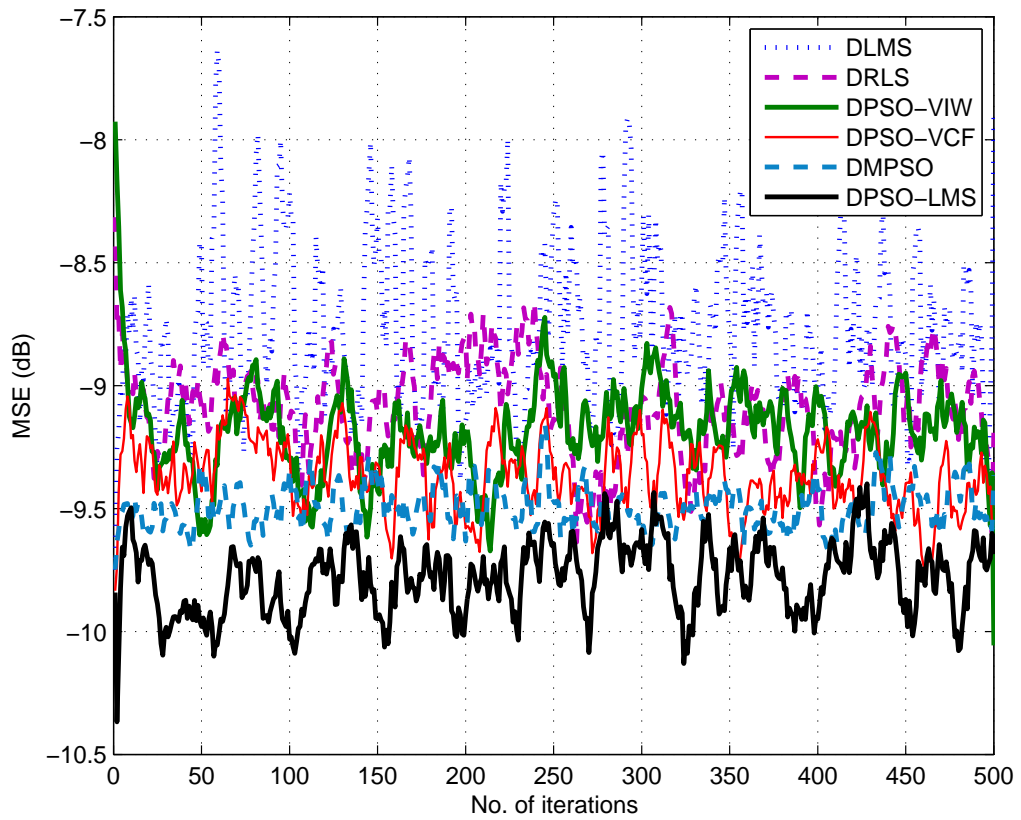


Figure 5.9: Comparison of testing phase of different algorithms at 10 dB SNR and at network size of 20 nodes.

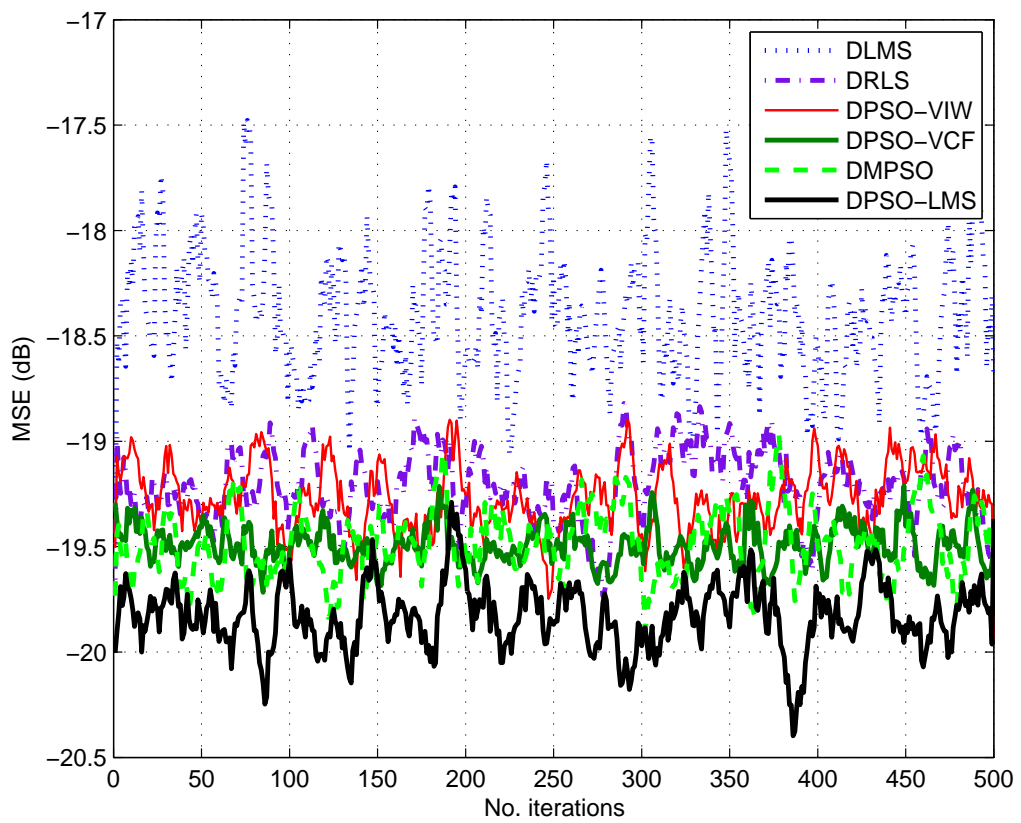


Figure 5.10: Comparison of testing phase of different algorithms at 20 dB SNR and at network size of 20 nodes.

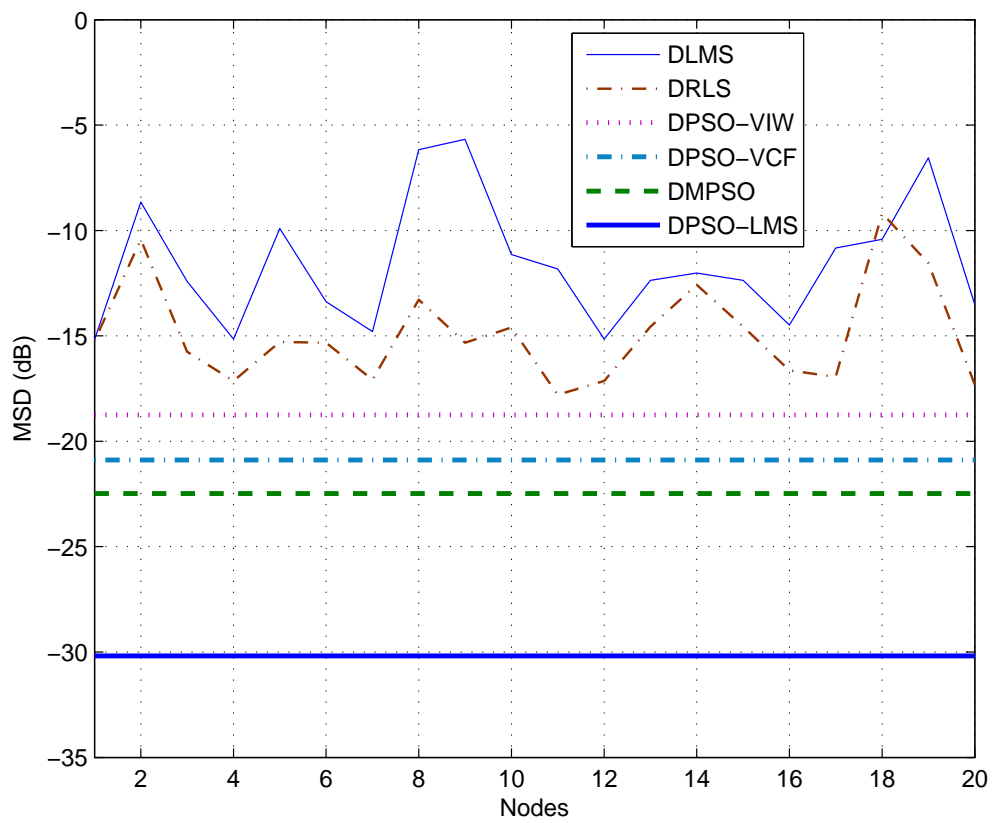


Figure 5.11: Comparison of steady state MSD of different algorithms at 10 dB SNR and at network size of 20 nodes.

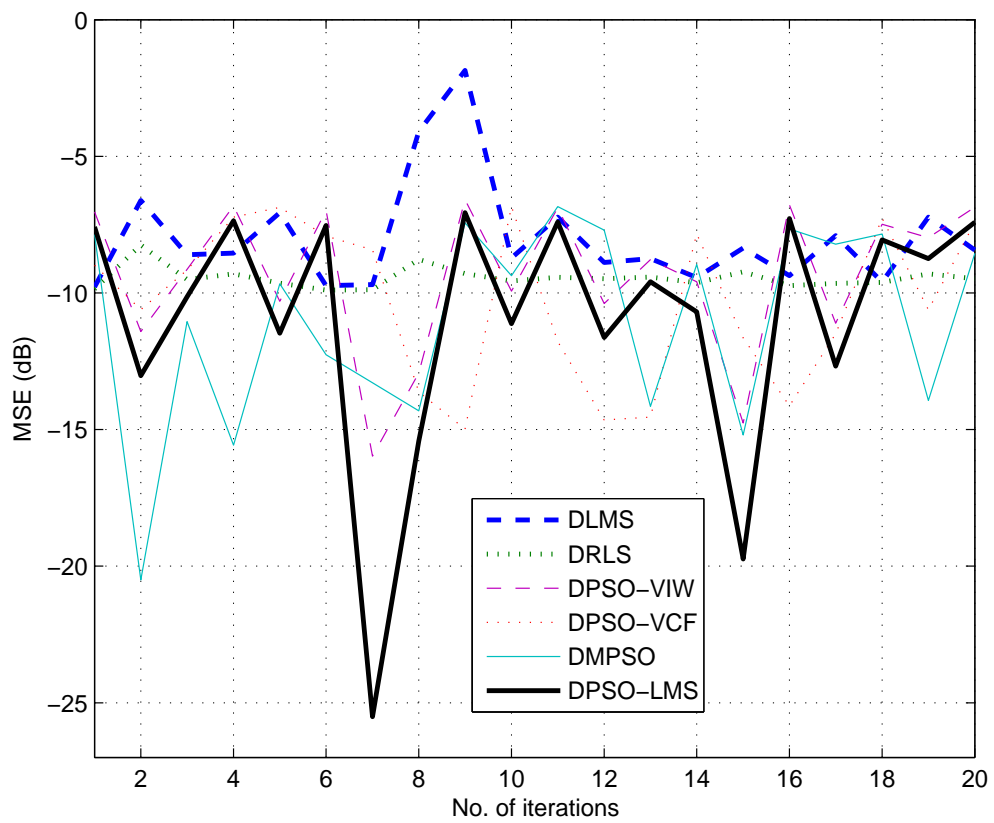


Figure 5.12: Comparison of steady state MSE of different algorithms at 10 dB SNR and at network size of 20 nodes.



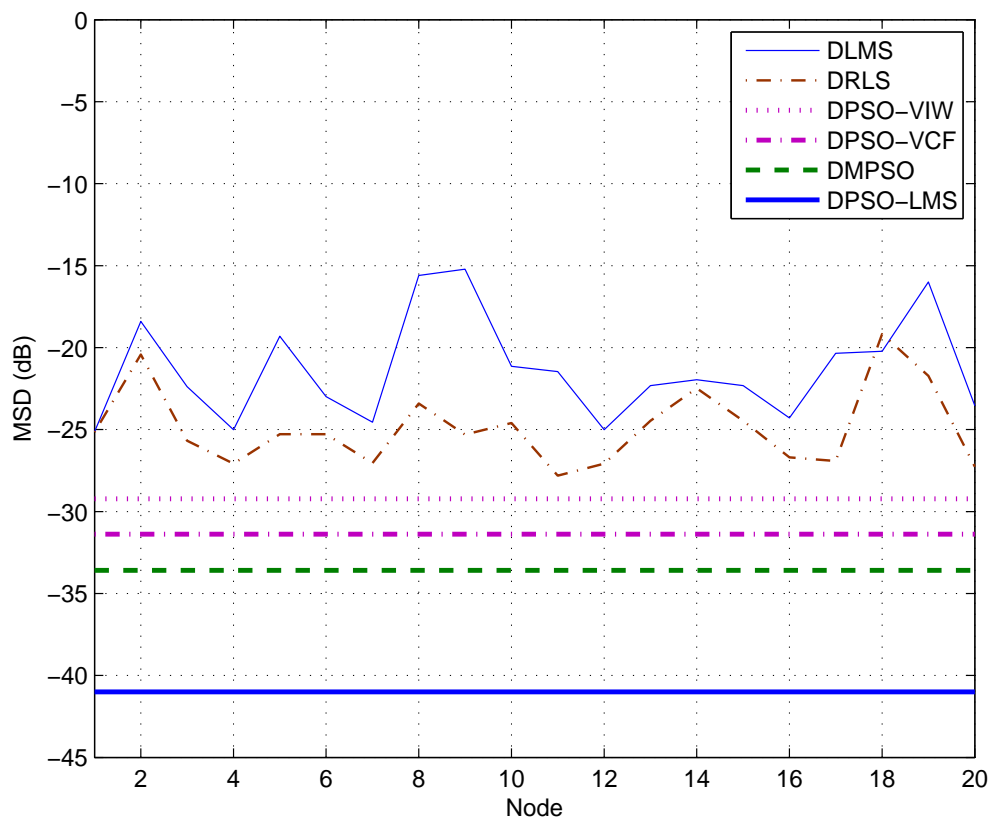


Figure 5.13: Comparison of steady state MSD of different algorithms at 20 dB SNR and at network size of 20 nodes.

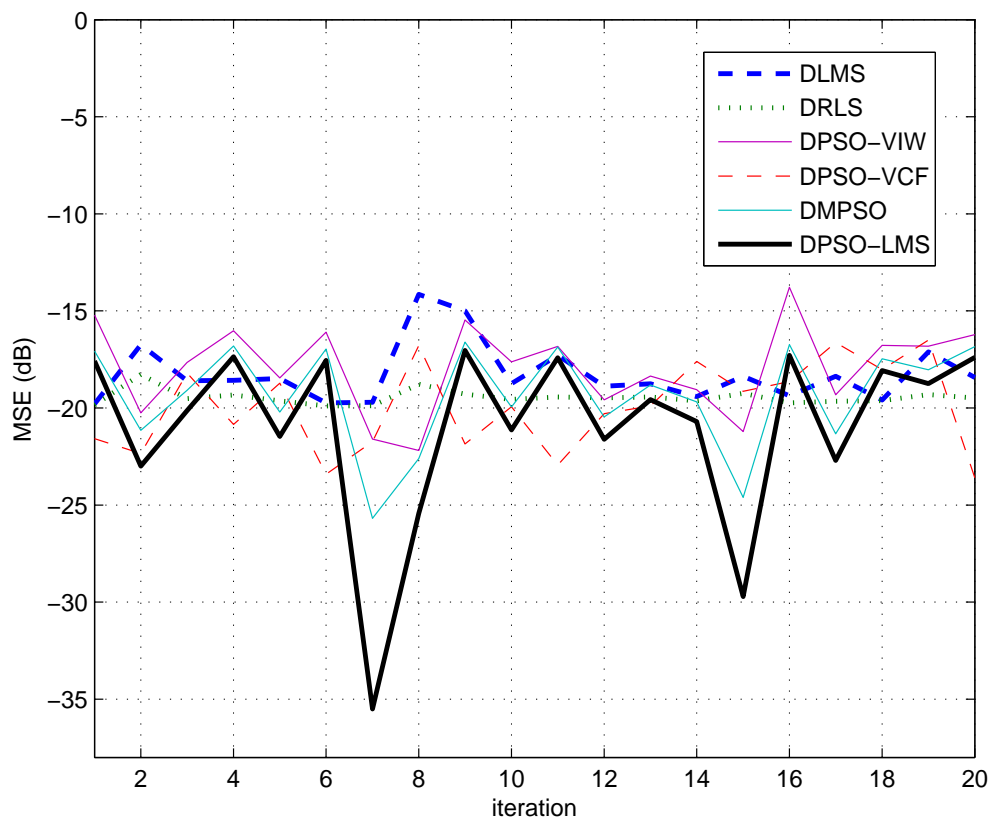


Figure 5.14: Comparison of steady state MSE of different algorithms at 20 dB SNR and at network size of 20 nodes.

## CHAPTER 6

# CONCLUSION

### 6.1 Contributions

A distributed estimation algorithm is proposed which uses a PSO algorithm at every sensor node for parameter estimation and uses a diffusion scheme for cooperative estimation by exploiting the spatial and temporal diversity of the network. The proposed algorithm reduces the computational complexity by using smaller particle swarm size and input data window size compared to prior proposed PSO algorithms. The cooperative estimation improves the performance of the algorithm when compared non-cooperative algorithms. The performance of the proposed diffusion PSO algorithm is further improved by applying different particle position update equations to the proposed diffusion PSO algorithm. The

four different variants of DPSO algorithm are as follows.

1. In the DPSO-VIW algorithm, the inertia weight factor in the velocity equation is varied linearly to enhance the local and global search capability of the algorithm. The results showed better performance than the earlier proposed diffusion adaptive algorithms.
2. In DPSO-VCF, a variable constriction factor in the proposed diffusion PSO is used to control the velocity of the particle, the results showed faster and better convergence of the steady state error.
3. In DMPSO algorithm, the inertia weight is varied proportional to the change in particle error to improve the performance further.
4. In the hybrid DPSO-LMS algorithm, the advantages of both PSO and LMS algorithm were used fruitfully. This algorithm outperformed all the previously proposed algorithms but at the cost of increase in computational complexity.

## **6.2 Future Work**

In future work, these proposed algorithms can be applied to complex optimization problems in WSNs. The proposed algorithms can also used for non-linear systems as PSO is known to perform better than other adaptive and evolutionary

algorithms.

The computational complexity of the algorithm can be reduced by reducing the particle swarm size.

Finally, new hybrid algorithms can be implemented by using other adaptive algorithms.

# REFERENCES

- [1] I. F. Akyildiz, "A survey on sensor networks," *IEEE Comm. Mag.*, vol. 40, no. 8, pp. 102-114, 2002.
- [2] J. Kennedy and R. Eberhart, *Swarm intelligence*, Academic Press, 2001.
- [3] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. of IEEE Int'l Conf. on Neural Networks*, pp. 1942-1948, 1995.
- [4] K. E. Parsopoulos, M. N. Vrahatis, "On the computation of all global minimizers through particle swarm optimization," *IEEE Transactions on Evolutionary Computation*, vol.8, no.3, pp. 211- 224, June 2004.
- [5] I. D. Schizas, A. Ribeiro, G. B. Giannakis, "Consensus in Ad Hoc WSNs With Noisy LinksPart I: Distributed Estimation of Deterministic Signals," *IEEE Transactions on Signal Processing*, vol.56, no.1, pp.350-364, Jan. 2008
- [6] I. D. Schizas, G. B. Giannakis, S. I. Roumeliotis, and A. Ribeiro, "Consensus in Ad Hoc WSNs With Noisy LinksPart II: Distributed Estimation and

- Smoothing of Random Signals,” IEEE Transactions on Signal Processing, vol. 56, no. 4, pp. 1650-1666, Apr. 2008.
- [7] A. Jadbabaie and A. S. Morse, “Coordination of groups of mobile autonomous agents using nearest neighbor rules,” IEEE Transactions on Automatic Control, vol. 48, no. 6, pp. 988-1001, June 2003.
- [8] L. Xiao and S. Boyd, “Fast linear iterations for distributed averaging,” Systems and Control Letters, vol. 53, no. 1, pp. 65-78, Sept. 2004.
- [9] S. Barbarossa, G. Scutari, and T. Battisti, “Distributed signal subspace projection algorithms with maximum convergence rate for sensor networks with topological constraints,” in IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP). 2009, pp. 2893-2896, IEEE.
- [10] S. Barbarossa, G. Scutari, and T. Battisti, “Cooperative sensing for cognitive radio using decentralized projection algorithms,” in IEEE 10th Workshop on Signal Processing Advances in Wireless Communications, SPAWC. 2009, pp. 116-120, IEEE.
- [11] L. Xiao, S. Boyd, and S. Lall, “A space-time diffusion scheme for peer-to-peer least-squares estimation,” in Proceedings of the 5th international conference on Information processing in sensor networks. 2006, pp. 168-176, ACM.

- [12] R. Olfati-Saber, J. Alex Fax, and Richard M. Murray, "Consensus and Cooperation in Networked Multi-Agent Systems," *Proceedings of the IEEE*, vol. 95, no. 1, pp. 215-233, Jan. 2007.
- [13] F. S. Cattivelli, C. G. Lopes, and A. H. Sayed, "Diffusion strategies for distributed Kalman filtering: formulation and performance analysis," in *Proc. Workshop on Cognitive Inf. Process.*, pp. 36-41, 2008.
- [14] F. S. Cattivelli and A. H. Sayed, "Diffusion mechanisms for fixed point distributed Kalman smoothing," in *Proc. EUSIPCO*, 2008, number 3, pp. 1-4.
- [15] R. Olfati-Saber, "Distributed Kalman filtering for sensor networks," *IEEE Conference on Decision and Control*, pp. 5492-5498, 2007
- [16] C. G. Lopes and A. H. Sayed, "Incremental adaptive strategies over distributed networks," *IEEE Trans. on Signal Processing*, vol. 55, pp. 4064-4077, Aug. 2007.
- [17] C. G. Lopes and A. H. Sayed, "Randomized incremental protocols over adaptive networks," *IEEE ICASSP*, pp. 3514-3517, 2010.
- [18] C. G. Lopes and A. H. Sayed, "Diffusion least-mean squares over adaptive networks: formulation and performance analysis," *IEEE Trans. on Signal Process.*, vol. 56, no. 7, pp. 3122-3136, July 2008.



- [19] Federico S. Cattivelli and A. H. Sayed, "Diffusion LMS algorithms with information exchange," 42nd Asilomar Conference on Signals, Systems and Computers, no. 8, pp. 251-255, Oct. 2008.
- [20] F. S. Cattivelli, C. G. Lopes, and A. H. Sayed, "Diffusion Recursive least squares for Distributed Estimation Over Adaptive Networks," IEEE Trans. on Signal Process., vol. 56, no. 5, pp. 1865-1877, May 2008.
- [21] Z. Bojkovic and B. Bakmaz, "A survey on wireless sensor networks deployment," WSEAS Trans. on Comm., vol. 7, no. 12, pp. 1172-1181, 2008.
- [22] A. Gopakumar and L. Jacob, "Localization in wireless sensor networks using particle swarm optimization," in Proc. of IET Int'l Conf. on Wireless, Mobile and Multimedia Networks, pp. 227-230, 2008.
- [23] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-efficient communication protocol for wireless microsensor networks," in Proc. of 33rd Annual Hawaii Int'l Conf. on System Sciences, vol. 2, 2000.
- [24] R. Rajagopalan and P.K. Varshney, "Data-aggregation techniques in sensor networks: A survey," IEEE Comm. Surveys Tuts., vol. 8, no. 4, pp. 48-63, Fourth Quarter 2006.

- [25] M. Lejbjerg, T. K. Rasmussen, and T. Krink, "Hybrid particle swarmv optimiser with breeding and subpopulations," Proceedings of the Third Genetic and Evolutionary Computation Conference, pages 469-476, 2001.
- [26] X. H. Shi, Y. H. Lu, C. G. Zhou, H. P. Lee, W. Z. Lin, and Y. C. Liang, "Hybrid evolutionary algorithms based on PSO and GA," The 2003 Congress on Evolutionary Computation, 2393-2399, 1996.
- [27] Y. H. Shi and R. C. Eberhart, "Parameter selection in particle swarm optimization," Evolutionary Programming VII, Springer, Lecture Notes in Computer Science, 1447:591-600, 1998.
- [28] Y. H. Shi and R. C. Eberhart, "Empirical study of particle swarm optimization," Proceedings of the IEEE International Conference on Evolutionary Computation, pages 1945-1950, 1999.
- [29] A. Carlise and G. Dozier, "An off-the shelf pso," Proceedings of the Particle Swarm Optimization Workshop, pages 1-6, 2001.
- [30] P. J. Angeline, "Evolutionary optimization versus particle swarm optimization: Philosophy and performance differences," Lecture Notes in Computer Science, 1447:601-610, 1998. Book Title: Evolutionary Programming VII, ISBN 978-3-540-64891-8.

- [31] A. El-Gallad, M. El-Hawary, A. Sallam, and A. Kalas, "Enhancing the pso via proper parameters selection," Proceedings of the IEEE Canadian Conference on Electrical and Computer Engineering, 2:792-797, 2002.
- [32] J. Kennedy, "Small world sand mega-minds: Effects of neighborhood topology on particle swarm performance," Proceedings of the 1999 IEEE Congress of Evolutionary Computation, 3:1931-1938, 1999.
- [33] F. van den Bergh and A. P. Engelbrecht, "Cooperative learning in neural networks using particle swarm optimizers," Annual Research Conference of South African Institute of Computer Scientists and Information Technologists 2000, 26:84-90, 2001.
- [34] F. Van den Bergh and A. P. Dngelbrecht, "A cooperative approach to particles swarms optimization," IEEE Transactions on Evolutionary Computing, 8(3):255-239, 2004.
- [35] J. Xu and Z. Xin, "An extended particle swarm optimizer," Proceedings of the 19th IEEE International Parallel and Distributed Processing Symposium, 2005.
- [36] J. Kennedy, "Some issues and practices for the particle swarms," Proceedings of the 2007 IEEE Swarm Intelligence Symposium, pages 162-169, 2007

- [37] R. V. Kulkarni and G. K. Venayagamoorthy, "Particle Swarm Optimization in Wireless-Sensor Networks: A Brief Survey," *IEEE Trans. on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, vol.41, no.2, pp.262-267, March 2011
- [38] Y. H. Shi and R. Eberhart, "A modified particle swarm optimizer," in *Proc. of IEEE Int'l Conf. on Evolutionary Computation*, pp. 69-73, 1998.
- [39] Ali T. Al-Awami, Azzedine Zerguine, Lahouari Cheded, Abdelmalek Zidouri, Waleed Saif, "A new modified particle swarm optimization algorithm for adaptive equalization," *Digital Signal Processing*, Volume 21, Issue 2, Pages 195-207. March 2011.
- [40] D. J. Krusienski and W. K. Jenkins, "A modified particle swarm optimization algorithm for adaptive filtering," *IEEE Int'l Symposium on Circuits and Systems*, vol., no., pp.4 pp.-140, 21-24 May 2006
- [41] M. Clerc. "The swarm and the queen: Towards a deterministic and adaptive particle swarm optimization," *Proceedings of the 1999 Congress on Evolutionary Computation*, pages 1951-1957, 1999.
- [42] R. Eberhart and Y. Shi, "Comparing inertia weights and constriction factors in particle swarm optimization," *Proceedings of the Congress on Evolutionary Computing*, pages 84-89, 2000

- [43] A.H. Sayed, *Fundamentals of Adaptive Filtering*. New York: Wiley, 2003.
- [44] D. J. Krusienski and W. K. Jenkins, "A particle swarm optimization-least mean squares algorithm for adaptive filtering," *Signals, Systems and Computers*, 2004. Conference Record of the Thirty-Eighth Asilomar Conference on , vol.1, no., pp. 241- 245 Vol.1, 7-10 Nov. 2004.

# VITAE

- Sameer Husain Arastu.
- Born in Hyderabad, India on June 3, 1985.
- Received Bachelor of Engineering (BE) degree in Electronics and Communication Engineering from Osmania University in June 2007
- Joined Mahindra Satyam company as a Software Engineer in June 2007
- Joined King Fahd University of Petroleum and Minerals in February 2010 as a Research Assistant in Electrical Engineering Department.
- **Contact details:**
- **Present Address:** Department of Electrical Engineering, King Fahd University of Petroleum and Minerals, P.O. Box 8657 , Dhahran 31261, Saudi Arabia.
- **E-mail Address:** sameerh@kfupm.edu.sa
- **Permanent Address:** H.No 11-5-321, Red Hills, Hyderabad -500004, India
- **E-mail Address:** arastu.ece04@gmail.com