

E.S. Gopi

Algorithm Collections for Digital Signal Processing Applications using Matlab

TN911.72
G 659

Algorithm Collections for Digital Signal Processing Applications Using Matlab

E.S. Gopi

National Institute of Technology, Tiruchi, India



 Springer



E2008000534

A C.I.P. Catalogue record for this book is available from the Library of Congress.

ISBN 978-1-4020-6409-8 (HB)

ISBN 978-1-4020-6410-4 (e-book)

Published by Springer,
P.O. Box 17, 3300 AA Dordrecht, The Netherlands.

www.springer.com

Printed on acid-free paper

All Rights Reserved

© 2007 Springer

No part of this work may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, electronic, mechanical, photocopying, microfilming, recording or otherwise, without written permission from the Publisher, with the exception of any material supplied specifically for the purpose of being entered and executed on a computer system, for exclusive use by the purchaser of the work.

Algorithm Collections for Digital Signal Processing Applications Using Matlab

*This book is dedicated to
my Wife G.Viji
and my Son V.G.Vasig*

Preface

The Algorithms such as SVD, Eigen decomposition, Gaussian Mixture Model, PSO, Ant Colony etc. are scattered in different fields. There is the need to collect all such algorithms for quick reference. Also there is the need to view such algorithms in application point of view. This Book attempts to satisfy the above requirement. Also the algorithms are made clear using MATLAB programs. This book will be useful for the Beginners Research scholars and Students who are doing research work on practical applications of Digital Signal Processing using MATLAB.

Acknowledgments

I am extremely happy to express my thanks to the Director Dr M.Chidambaram, National Institute of Technology Trichy India for his support. I would also like to thank Dr B.Venkatramani, Head of the Electronics and Communication Engineering Department, National Institute of Technology Trichy India and Dr K.M.M. Prabhu, Professor of the Electrical Engineering Department, Indian Institute of Technology Madras India for their valuable suggestions. Last but not least I would like to thank those who directly or indirectly involved in bringing up this book successfully. Special thanks to my family members father Mr E.Sankara subbu, mother Mrs E.S.Meena, Sisters R.Priyaravi, M.Sathyamathi, E.S.Abinaya and Brother E.S.Anukeerthi.

Thanks

E.S.Gopi

Contents

Preface	xiii
Acknowledgments	xv
Chapter 1 ARTIFICIAL INTELLIGENCE	
1 Particle Swarm Algorithm	1
1-1 How are the Values of 'x' and 'y' are Updated in Every Iteration?	2
1-2 PSO Algorithm to Maximize the Function F(X, Y, Z)	4
1-3 M-program for PSO Algorithm	6
1-4 Program Illustration	8
2 Genetic Algorithm	9
2-1 Roulette Wheel Selection Rule	10
2-2 Example	11
2-2-1 M-program for genetic algorithm	11
2-2-2 Program illustration	13
2-3 Classification of Genetic Operators	15
2-3-1 Simple crossover	16
2-3-2 Heuristic crossover	16
2-3-3 Arith crossover	17
3 Simulated Annealing	18
3-1 Simulated Annealing Algorithm	19
3-2 Example	19
3-3 M-program for Simulated Annealing	23

4 Back Propagation Neural Network	24
4-1 Single Neuron Architecture	25
4-2 Algorithm	27
4-3 Example	29
4-4 M-program for Training the Artificial Neural Network for the Problem Proposed in the Previous Section	31
5 Fuzzy Logic Systems	32
5-1 Union and Intersection of Two Fuzzy Sets	32
5-2 Fuzzy Logic Systems	33
5-2-1 Algorithm	35
5-3 Why Fuzzy Logic Systems?	38
5-4 Example	39
5-5 M-program for the Realization of Fuzzy Logic System for the Specifications given in Section 5-4	41
6 Ant Colony Optimization	44
6-1 Algorithm	44
6-2 Example	48
6-3 M-program for Finding the Optimal Order using Ant Colony Technique for the Specifications given in the Section 6-2	50
Chapter 2 PROBABILITY AND RANDOM PROCESS	
1 Independent Component Analysis	53
1-1 ICA for Two Mixed Signals	53
1-1-1 ICA algorithm	62
1-2 M-file for Independent Component Analysis	65
2 Gaussian Mixture Model	68
2-1 Expectation-maximization Algorithm	70
2-1-1 Expectation stage	71
2-1-2 Maximization stage	71
2-2 Example	72
2-3 Matlab Program	73
2-4 Program Illustration	76
3 K-Means Algorithm for Pattern Recognition	77
3-1 K-means Algorithm	77
3-2 Example	77
3-3 Matlab Program for the K-means Algorithm Applied for the Example given in Section 3-2	78
4 Fuzzy K-Means Algorithm for Pattern Recognition	79
4-1 Fuzzy K-means Algorithm	80
4-2 Example	81
4-3 Matlab Program for the Fuzzy k-means Algorithm Applied for the Example given in Section 4-2	83

5 Mean and Variance Normalization	84
5-1 Algorithm	84
5-2 Example 1	85
5-3 M-program for Mean and Variance Normalization	86
Chapter 3 NUMERICAL LINEAR ALGEBRA	87
1 Hotelling Transformation	87
1-1 Diagonalization of the Matrix 'CM'	88
1-2 Example	88
1-3 Matlab Program	90
2 Eigen Basis	91
2-1 Example 1	91
3 Singular Value Decomposition (SVD)	93
3-1 Example	94
4 Projection Matrix	95
4-1 Projection of the Vector 'a' on the Vector 'b'	95
4-2 Projection of the Vector on the Plane Described by Two Columns Vectors of the Matrix 'X'	96
4-2-1 Example	97
4-2-2 Example 2	98
5 Orthonormal Vectors	100
5-1 Gram-Schmidt Orthogonalization procedure	100
5-2 Example	101
5-3 Need for Orthonormal Basis	101
5-4 M-file for Gram-Schmidt Orthogonalization Procedure	103
6 Computation of the Powers of the Matrix 'A'	103
7 Determination of Kth Element in the Sequence	104
8 Computation of Exponential of the Matrix 'A'	107
8.1 Example	107
9 Solving Differential Equation Using Eigen decomposition	108
10 Computation of Pseudo Inverse of the Matrix	109
11 Computation of Transformation Matrices	111
11-1 Computation of Transformation Matrix for the Fourier Transformation	113
11-2 Basis Co-efficient transformation	115
11-3 Transformation Matrix for Obtaining Co-efficient of Eigen Basis	117
11-4 Transformation Matrix for Obtaining Co-efficient of Wavelet Basis	117
12 System Stability Test Using Eigen Values	118
13 Positive Definite Matrix test for Minimal Location of the Function f (x1, x2, x3, x4...xn)	119

14 Wavelet Transformation Using Matrix Method	119
14-1 Haar Transformation	120
14-1-1 Example	122
14-1-2 M-file for haar forward and inverse transformation	125
14-2 Daubechies-4 Transformation	127
14-2-1 Example	128
14-2-2 M-file for daubechies 4 forward and inverse transformation	131
Chapter 4 SELECTED APPLICATIONS	135
1 Ear Pattern Recognition Using Eigen Ear	135
1-1 Algorithm	135
1-2 M-program for Ear Pattern Recognition	138
1-3 Program Illustration	140
2 Ear Image Data Compression using Eigen Basis	141
2-1 Approach	141
2-2 M-program for Ear Image Data Compression	143
3 Adaptive Noise Filtering using Back Propagation Neural Network	145
3-1 Approach	146
3-2 M-file for Noise Filtering Using ANN	147
3-3 Program Illustration	149
4 Binary Image Rotation Using Transformation Matrix	150
4-1 Algorithm	151
4-2 M-program for Binary Image Rotation with 45 Degree Anticlockwise Direction	152
5 Clustering Texture Images Using K-means Algorithm	152
5-1 Approach	153
5-2 M-program for Texture Images Clustering	155
6 Search Engine Using Interactive Genetic Algorithm	156
6-1 Procedure	156
6-2 Example	158
6-3 M-program for Interactive Genetic Algorithm	160
6-4 Program Illustration	165
7 Speech Signal Separation and Denoising Using Independent Component Analysis	166
7-1 Experiment 1	166
7-2 Experiment 2	167
7-3 M-program for Denoising	169

8 Detecting Photorealistic Images using ICA Basis	170
8-1 Approach	171
8-1-1 To classify the new image into one among the photographic or photorealistic image	171
8-2 M-program for Detecting Photo Realistic Images Using ICA basis	172
8-3 Program Illustration	174
9 Binary Image Watermarking Using Wavelet Domain of the Audio Signal	175
9-1 Example	175
9-2 M-file for Binary Image Watermarking in Wavelet Domain of the Audio Signal	176
9-3 Program Illustration	180
 Appendix	 183
 Index	 189

Chapter 1

ARTIFICIAL INTELLIGENCE

Algorithm Collections

1. PARTICLE SWARM ALGORITHM

Consider the two swarms flying in the sky, trying to reach the particular destination. Swarms based on their individual experience choose the proper path to reach the particular destination. Apart from their individual decisions, decisions about the optimal path are taken based on their neighbor's decision and hence they are able to reach their destination faster. The mathematical model for the above mentioned behavior of the swarm is being used in the optimization technique as the Particle Swarm Optimization Algorithm (PSO).

For example, let us consider the two variables 'x' and 'y' as the two swarms. They are flying in the sky to reach the particular destination (i.e.) they continuously change their values to minimize the function $(x-10)^2+(y-5)^2$. Final value for 'x' and 'y' are 10.1165 and 5 respectively after 100 iterations.

The Figure 1-1 gives the closed look of how the values of x and y are changing along with the function value to be minimized. The minimization function value reached almost zero within 35 iterations. Figure 1-2 shows the zoomed version to show how the position of x and y are varying until they reach the steady state.

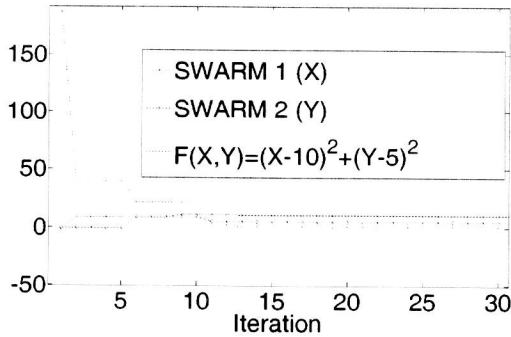


Figure 1-1. PSO Example zoomed version

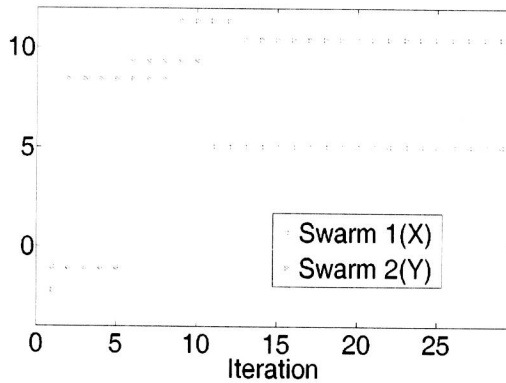


Figure 1-2. PSO Example

1.1 How are the Values of ‘x and y’ are Updated in Every Iteration?

The vector representation for updating the values for x and y is given in Figure 1-3. Let the position of the swarms be at ‘a’ and ‘b’ respectively as shown in the figure. Both are trying to reach the position ‘e’. Let ‘a’ decides to move towards ‘c’ and ‘b’ decides to move towards ‘d’.

The distance between the position ‘c’ and ‘e’ is greater than the distance between ‘d’ and ‘e’. so based on the neighbor’s decision position ‘d’ is treated as the common position decided by both ‘a’ and ‘b’. (ie) the position ‘c’ is the individual decision taken by ‘a’, position ‘d’ is the individual decision taken by ‘b’ and the position ‘d’ is the common position decided by both ‘a’ and ‘b’.

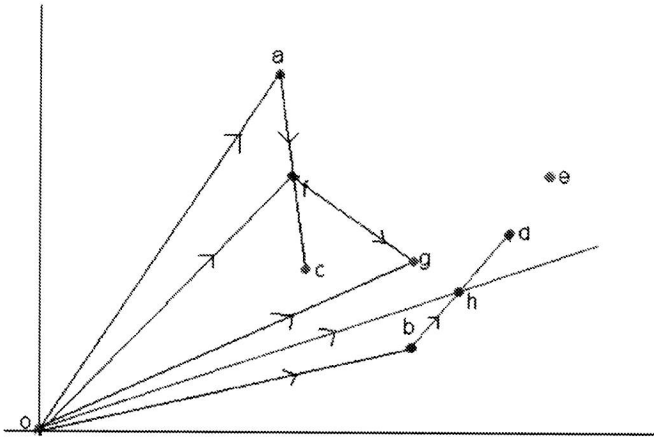


Figure 1-3. Vector Representation of PSO Algorithm

'a' based on the above knowledge, finally decides to move towards the position 'g' as the linear combination of 'oa' , 'ac' and 'ad'. [As 'd' is the common position decided].The linear combination of 'oa' and scaled 'ac' (ie) 'af' is the vector 'of'. The vector 'of' combined with vector 'fg' (ie) scaled version of 'ad' to get 'og' and hence final position decided by 'a' is 'g'.

Similarly, 'b' decides the position 'h' as the final position. It is the linear combination of 'ob' and 'bh'(ie) scaled version of 'bd'. Note as 'd' is the common position decided by 'a' and 'b', the final position is decided by linear combinations of two vectors alone.

Thus finally the swarms 'a' and 'b' moves towards the position 'g' and 'h' respectively for reaching the final destination position 'e'. The swarm 'a' and 'b' randomly select scaling value for linear combination. Note that 'oa' and 'ob' are scaled with 1 (ie) actual values are used without scaling. Thus the decision of the swarm 'a' to reach 'e' is decided by its own intuition along with its neighbor's intuition.

Now let us consider three swarms (A,B,C) are trying to reach the particular destination point 'D'. A decides A', B decides B' and C decides C' as the next position. Let the distance between the B' and D is less compared with A'D and C' and hence, B' is treated as the global decision point to reach the destination faster.

Thus the final decision taken by A is to move to the point, which is the linear combination of OA, AA' and AB'. Similarly the final decision taken

by B is to move the point which is the linear combination of OB, BB'. The final decision taken by C is to move the point which is the linear combination of OC, CC' and CB'.

1.2 PSO Algorithm to Maximize the Function F (X, Y, Z)

1. Initialize the values for initial position a, b, c, d, e
2. Initialize the next positions decided by the individual swarms as a', b', c', d' and e'
3. Global decision regarding the next position is computed as follows. Compute $f(a', b, c, d, e)$, $f(a, b', c, d, e)$, $f(a, b, c', d, e)$, $f(a, b, c, d', e)$ and $f(a, b, c, d, e')$. Find minimum among the computed values. If $f(a', b, c, d, e)$ is minimum among all, the global position decided regarding the next position is a'. Similarly If $f(a, b', c, d, e)$ is minimum among all, b' is decided as the global position regarding the next position to be shifted and so on. Let the selected global position is represented as 'global'
4. Next value for a is computed as the linear combination of 'a', (a'-a) and (global-a) (ie)
 - $nexta = a + C1 * RAND * (a' - a) + C2 * RAND * (global - a)$
 - $nextb = b + C1 * RAND * (b' - b) + C2 * RAND * (global - b)$
 - $nextc = c + C1 * RAND * (c' - c) + C2 * RAND * (global - c)$
 - $nextd = d + C1 * RAND * (d' - d) + C2 * RAND * (global - d)$
 - $nexte = e + C1 * RAND * (e' - e) + C2 * RAND * (global - e)$
5. Change the current value for a, b, c, d and e as nexta, nextb, nextc, nextd and nexte
6. If $f(nexta, b, c, d, e)$ is less than $f(a', b, c, d, e)$ then update the value for a' as nexta, otherwise a' is not changed.

If $f(a, nextb, c, d, e)$ is less than $f(a, b', c, d, e)$ then update the value for b' as nextb, otherwise b' is not changed

If $f(a, b, nextc, d, e)$ is less than $f(a, b, c', d, e)$ then update the value for c' as nextc, otherwise c' is not changed

If $f(a, b, c, \text{nextd}, e)$ is less than $f(a, b, c, d', e)$ then update the value for d' as nextd , otherwise d' is not changed

If $f(a, b, c, d, \text{nexte})$ is less than $f(a, b, c, d, e')$ then update the value for e' as nexte , otherwise e' is not changed

7. Repeat the steps 3 to 6 for much iteration to reach the final decision.

The values for 'c1','c2' are decided based on the weightage given to individual decision and global decision respectively.

Let $\Delta a(t)$ is the change in the value for updating the value for 'a' in t th iteration, then nexta at $(t+1)$ th iteration can be computed using the following formula. This is considered as the velocity for updating the position of the swarm in every iteration.

$$\text{nexta}(t+1) = a(t) + \Delta a(t+1)$$

where

$$\Delta a(t+1) = c1 * \text{rand} * (a' - a) + c2 * \text{rand} * (\text{global} - a) + w(t) * \Delta a(t)$$

' $w(t)$ ' is the weight at t^{th} iteration. The value for 'w' is adjusted at every iteration as given below, where 'iter' is total number of iteration used.

$$w(t+1) = w(t) - t * w(t) / (\text{iter}).$$

Decision taken in the previous iteration is also used for deciding the next position to be shifted by the swarm. But as iteration increases, the contribution of the previous decision is decreases and finally reaches zero in the final iteration.