**PBIL algorithm**

Step 1: A population of solution and a probability vector is generated randomly.

Step 2: The fitness of each individual is evaluated based on the function to be optimized and the individuals are ranked.

The fitness function =

Where,

is the ith bit of embedded watermark and

is the ith bit of extracted watermarks respectively.

Step 3: The probability vector PV[i] is then updated based on learning (either positive or negative learning) from current population

**PV[i] =PV[i]\*(1-LR) + SVB[i]\*LR**

Where, PV[i] is ith bit of probability vector in this generation,

LR is learning rate,

SVB[i] is ith bit of best solution vector of this generation.

Step 4: Mutation probability vector PV[i] is performed with a very low probability.

**PV[i]=PV[i]\*(1-MS)+Random(0 or 1)\*(1-MS)**

Where MS is mutation shift which describes the magnitude to which the PV will be affected by mutation.

Step 5: Create next generation using probability vectors as follows

The probability that a chromosome will have a 1 in ith position is proportional to the value of probability vector’s ith bit. For this we generate a random number in the interval [0, 1] and compare it to PV[i] for each gene of each chromosome.

Steps 2 to 5 are repeated until the population converges to an optimal solution

PBIL can use either positive learning or negative learning to update probability vector. In positive learning the probability vector is moved towards the best solution vector and in negative learning the probability vector moves away from the worst solution vector. Typical value of learning rate lies between 0.1 and 0.4. The mutation probability is low as too high mutation rate prevents population to converge to any optimum solution