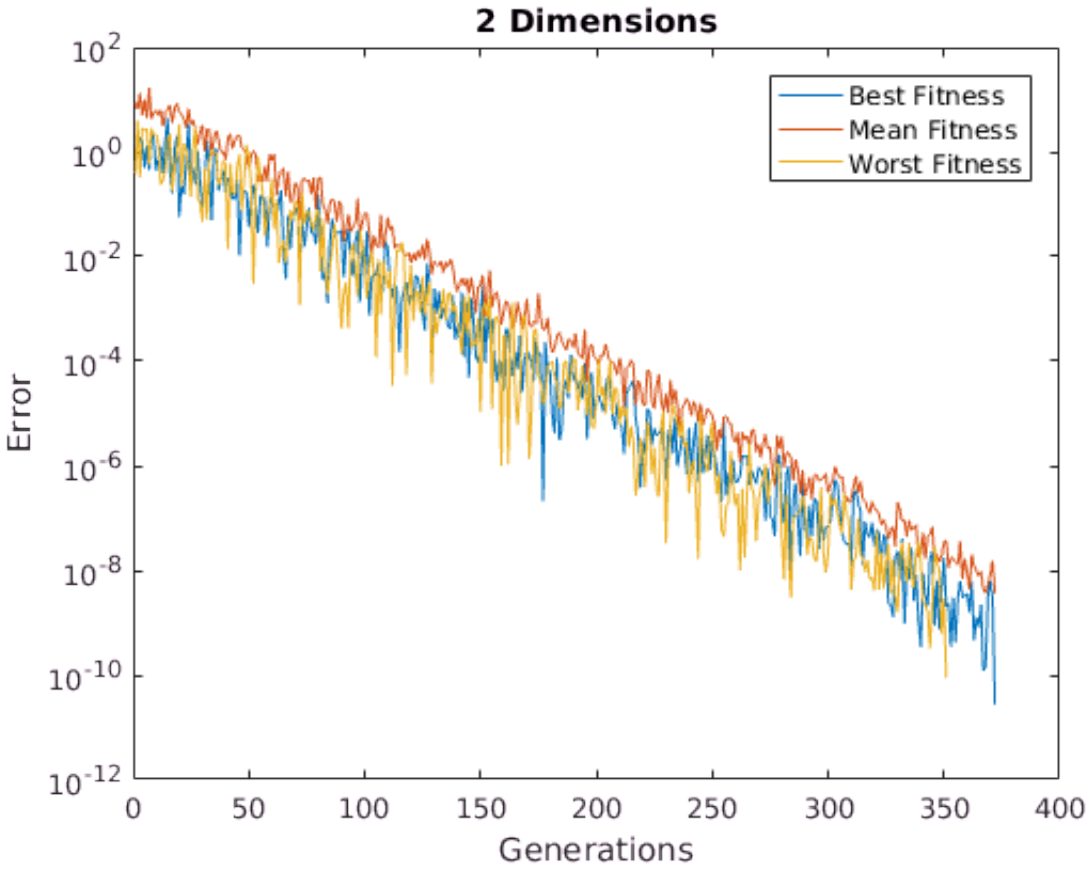
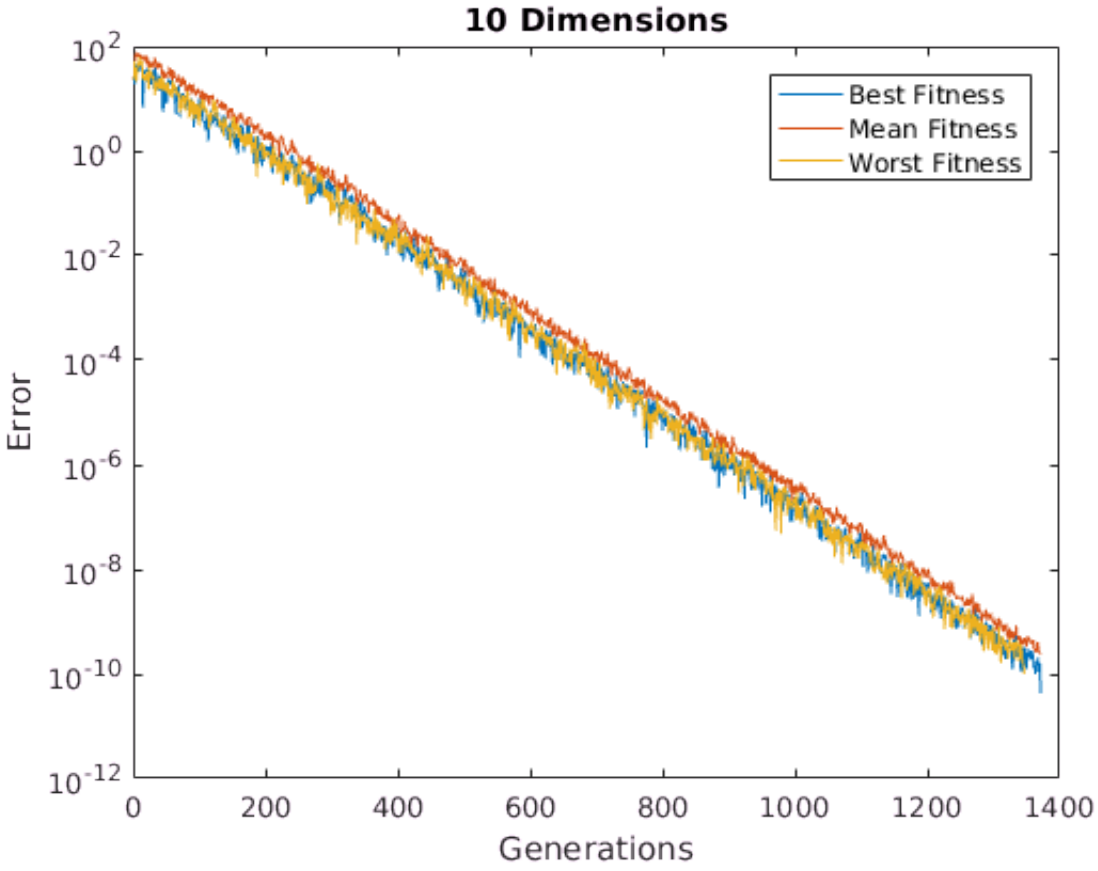
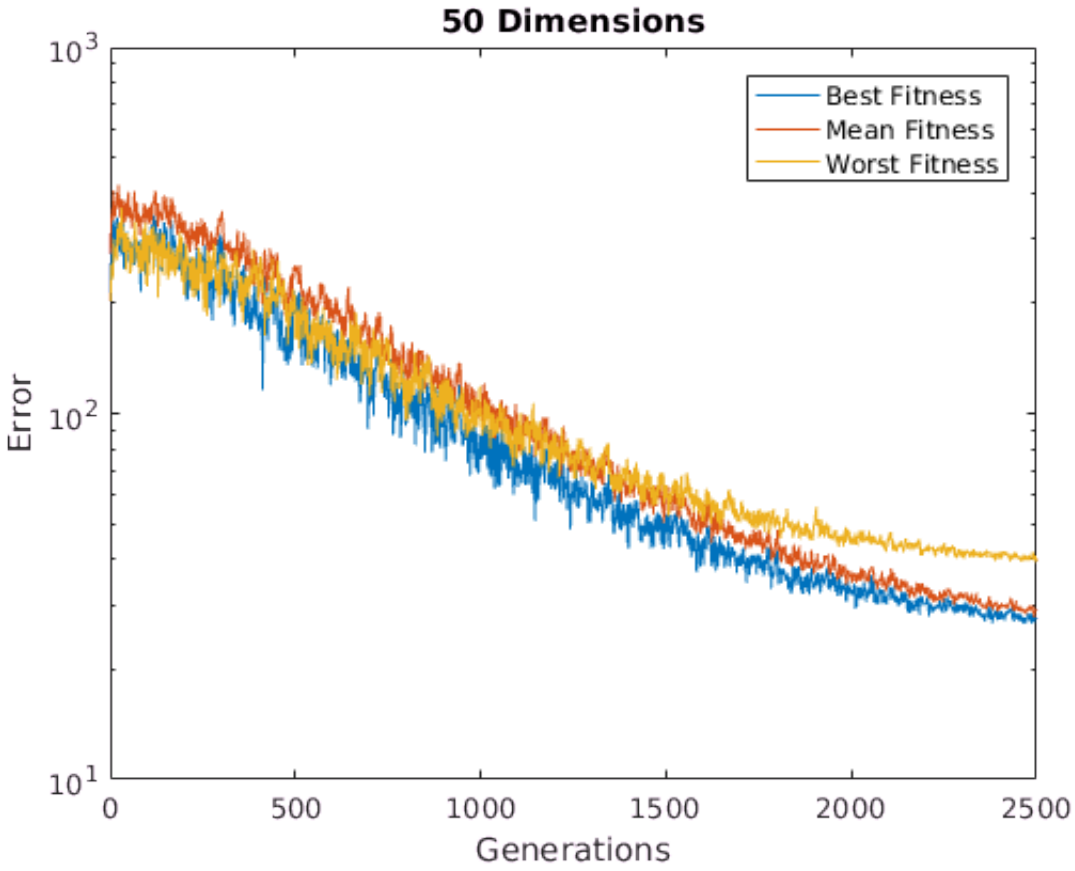
```
clear;
addpath('sample_code');


nParameters(1) = 2;
nParameters(2) = 10;
nParameters(3) = 50;
evaluationFunction = 'evaluationFunction';

for experiment = 1:3
    clear bestFitness medianFitness worstFitness bestSolution worstSolution
    parameters.title = sprintf('%d Dimensions', nParameters(experiment));
    figure(experiment);clf;
    for run=1:5
        r = cmaes(evaluationFunction, nParameters(experiment), parameters);
        bestFitness{run} = r.bestFitness;
        medianFitness{run} = r.medianFitness;
%         bestFinalFitness(run) = bestFitness{run}(end);
        worstFitness{run} = r.worstFitness;
        bestSolution(run,:) = r.bestSolution;
        worstSolution(run,:) = r.worstSolution;
        bestFinalFitness(run) = bestFitness{run}(end);
        worstFinalFitness(run) = worstFitness{run}(end);
    end
    [x iBest] = sort(bestFinalFitness);
    [x iWorst] = sort(worstFinalFitness);
    semilogy(bestFitness{iBest(1)});
    hold on;
    semilogy(medianFitness{iBest(1)});
    semilogy(bestFitness{iBest(end)});
%     overallBestSolution{experiment} = bestSolution{iBest(1)};
    xlabel('Generations');
    ylabel('Error');
    legend('Best Fitness','Mean Fitness', 'Worst Fitness');
    title(parameters.title);
    bestSolutionFound = bestSolution(iBest(1),:);
    disp(bestSolutionFound);
    bestValue = bestFitness{iBest(1)}(end)
    worstSolutionfound = bestSolution(iBest(end),:);
    disp(worstSolutionfound);
    worstValue = bestFitness{iBest(end)}(end)

end
```

```
   1.0e-06 *

    0.0002    0.3701
bestValue = 2.7171e-11
   1.0e-06 *

    0.1978    0.6362
worstValue = 8.8065e-11
   1.0e-06 *

   -0.1139    0.1497    0.1009    0.1480    0.1546   -0.0623   -0.1362   -0.1635    0.2474    0.1247
bestValue = 4.3087e-11
   1.0e-06 *

   -0.1039   -0.0361    0.1123   -0.0026    0.1316    0.4397    0.3375    0.1071   -0.1016   -0.3603
worstValue = 9.9391e-11
```

```
  Columns 1 through 11

   -0.9616   -0.0170   -0.0331   -0.9907   -1.0009   -1.0114   -0.0432    0.0139   -0.0028    0.0055   -

  Columns 12 through 22

   -0.0018    0.0005   -0.0123    0.0492    1.0117    0.0052    1.0127    0.9670    0.0299   -0.0293

  Columns 23 through 33

    0.9822    0.0015    0.0094   -0.9982   -0.9896   -0.9746    0.9781   -0.0063    0.0077    0.0193

  Columns 34 through 44

   -1.0073    1.0057    0.0219    0.9835    0.0278   -0.0159   -0.0162    1.0057   -0.0322   -0.9842   -

  Columns 45 through 50

    0.0202    1.0066   -0.0255    0.9771    1.0027    0.9868

bestValue = 27.5192

  Columns 1 through 11

    0.0049   -2.0026   -1.0060   -0.0063    0.9950    1.9884    0.0184    0.0041   -0.9965   -1.0013   -

  Columns 12 through 22

    0.0077    0.0124   -0.0012   -0.0339   -0.9787   -0.0259    1.9745   -1.0142    0.0211    0.0131   -

  Columns 23 through 33

    0.0081    0.0051   -1.0149    0.9927    0.0062   -0.9630    0.0396    0.9881    1.0057    0.9849   -

  Columns 34 through 44

    1.0178    0.0334   -0.0394   -0.0117    0.9914   -0.0408    0.9952   -0.9785    0.9775    0.0126

  Columns 45 through 50

    0.0052   -0.0175   -0.0174   -0.0139   -1.9604    0.9640

worstValue = 40.2082
```


2 Dimensions
Best Fitness
Mean Fitness
Worst Fitness
Error
Generations
10^{2}
10^{0}
10^{-2}
10^{-4}
10^{-6}
10^{-8}
10^{-10}
10^{-12}
0
50
100
150
200
250
300
350
400


10 Dimensions
Best Fitness
Mean Fitness
Worst Fitness
Error
Generations
10^{2}
10^{0}
10^{-2}
10^{-4}
10^{-6}
10^{-8}
10^{-10}
10^{-12}
0
200
400
600
800
1000
1200
1400




```
%% Result
% For 2 dimensions this converges very fast and a solution can
% be found already after about 370 generations.
% For 10 Dimensions a good solution could be found after about
% 1350 generations.
% For 50 dimensions even after 2500 no good solution could be
% found. The rate of conversion is much slower in this case.
% Also it can be seen that worst and best fitness for more
% generations are more different
```