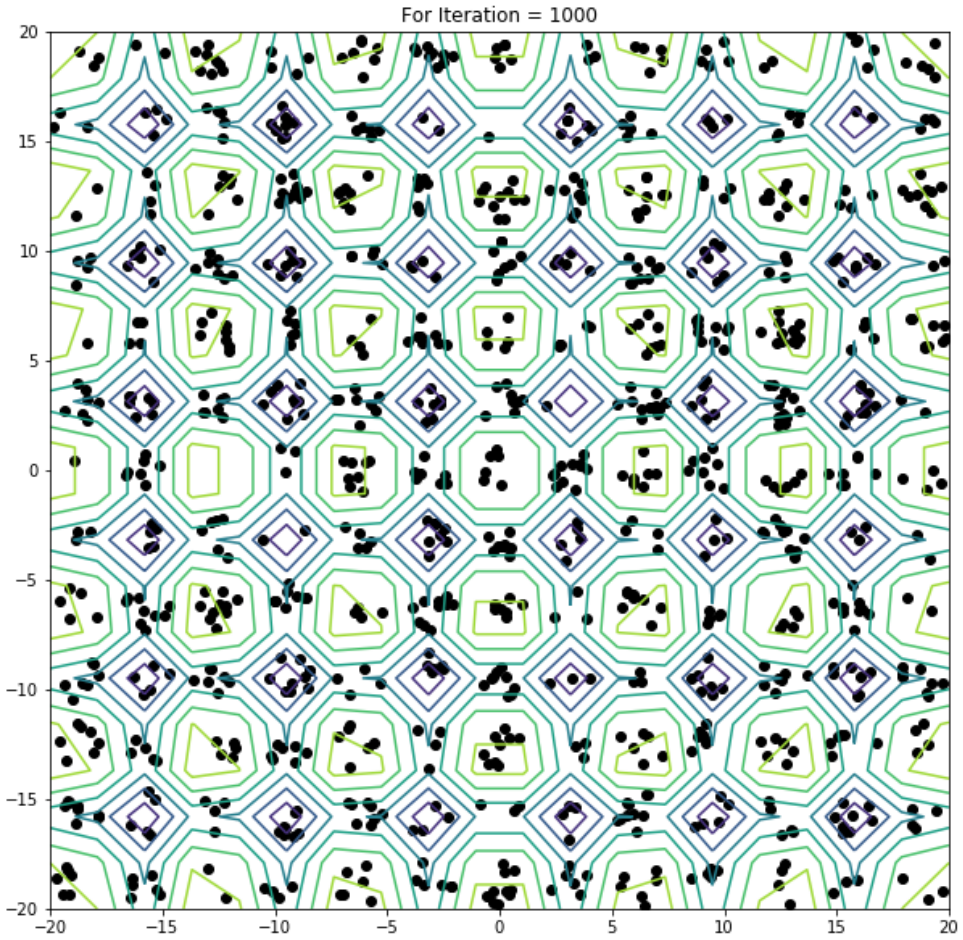
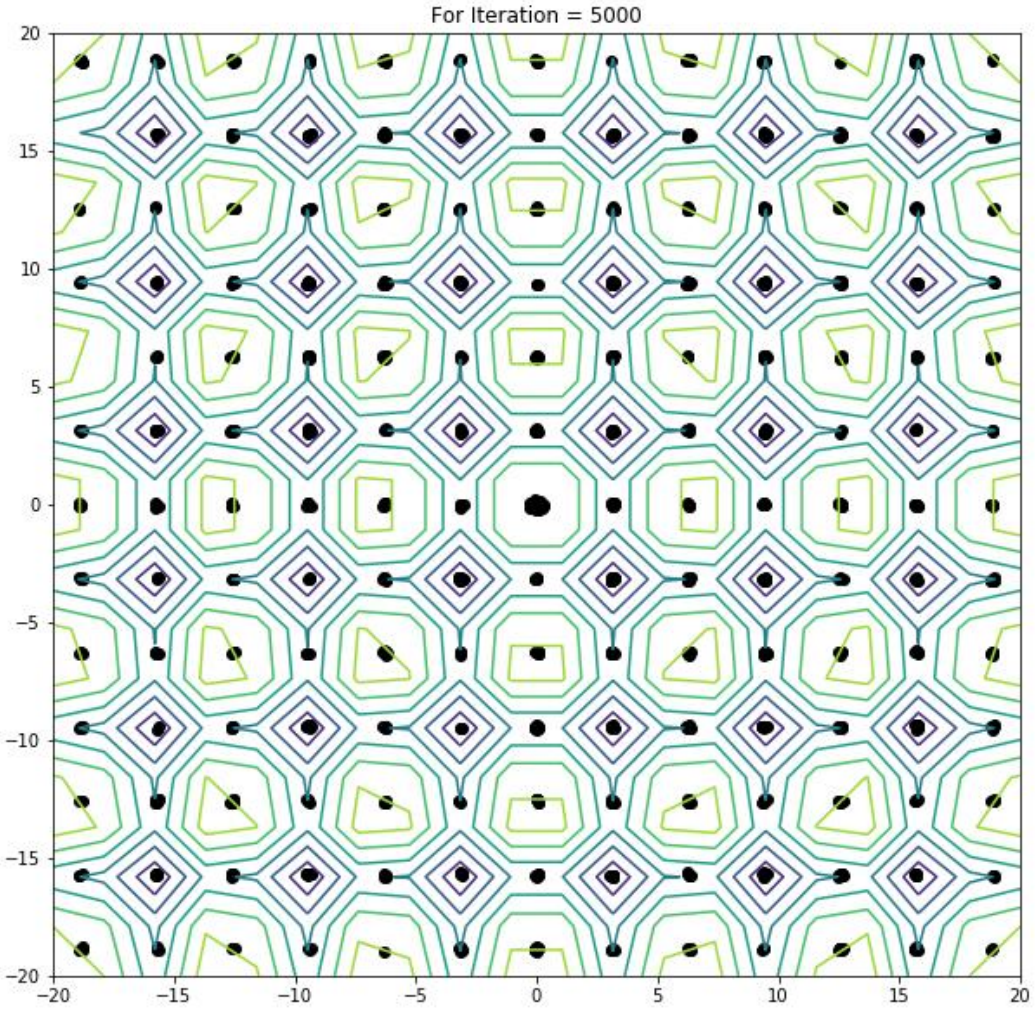
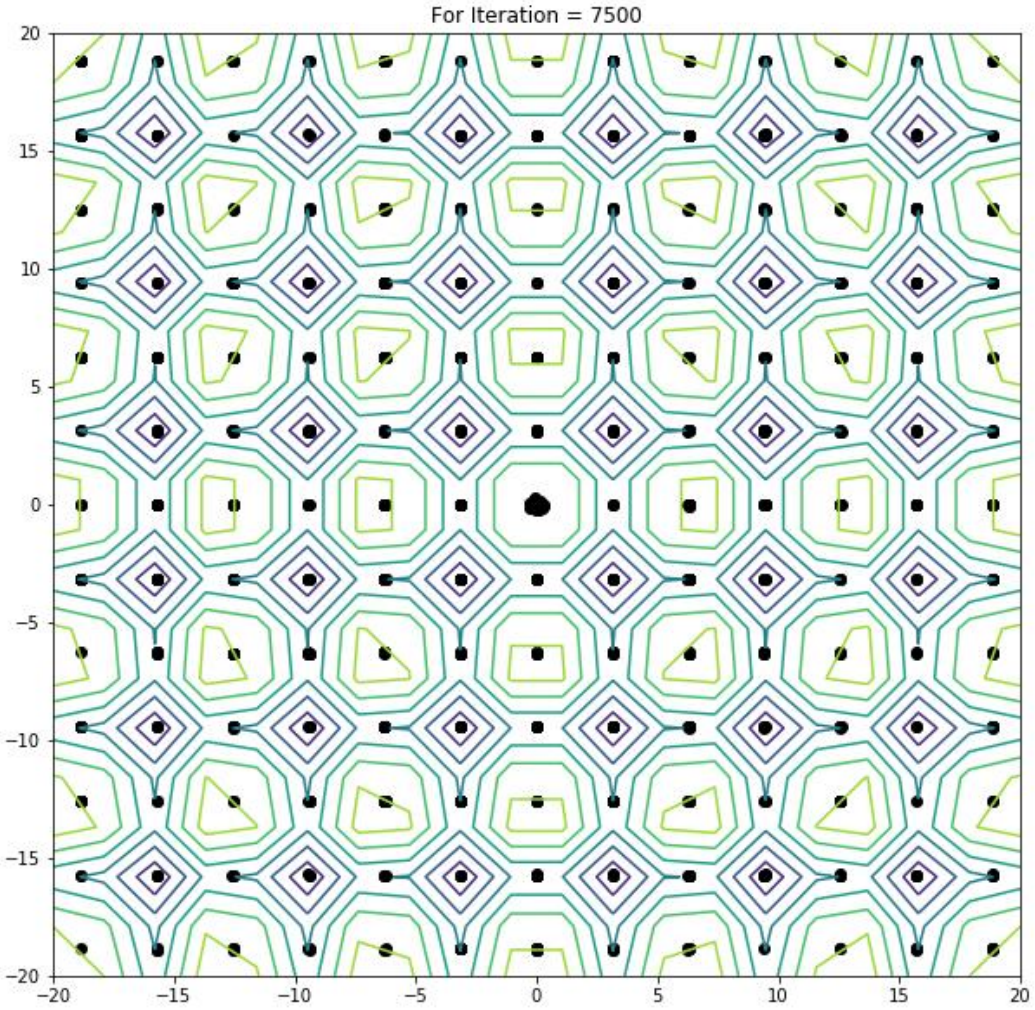
Problem 1:



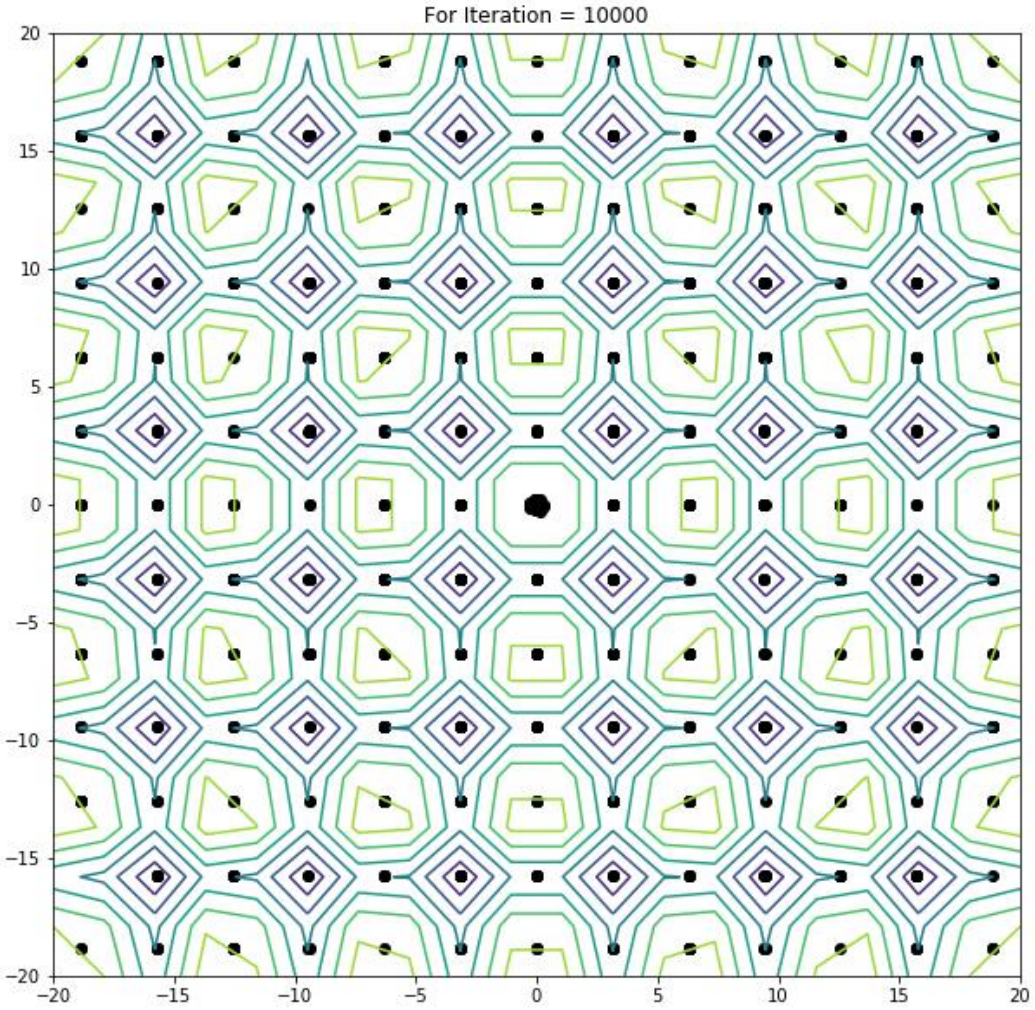

```
Global Min =  1.00045
```




```
Global Min =  0.900796
```




```
Global Min =  0.900016
```

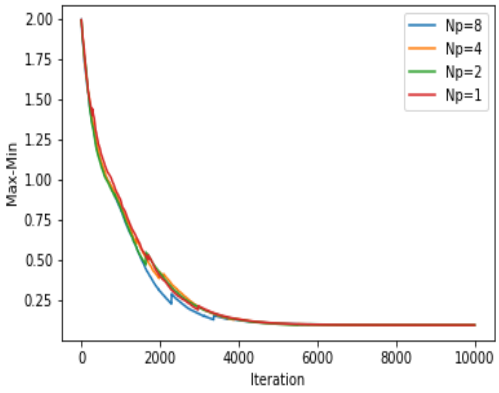


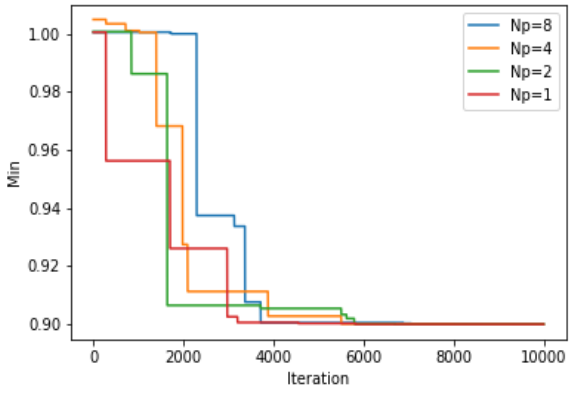

```
Global Min =  0.900003
```

With increase in number of iteration cloud start to converge on global minima as well as local minima. The cluster of point increase at global minima with increase in number of iterations.

Problem 2:

# Processor	Test Case 1	Test Case 2	Test Case 3	Test Case 4	Test Case 5	Average(T)	T_0	T_s
1	0.544855	0.424232	0.529157	0.517849	0.48428	0.50007		
2	0.602979	0.562851	0.589264	0.585762	0.538604	0.5758	0.65	0.8683
4	0.409485	0.540358	0.474506	0.475243	0.4663	0.4731	2.472	1.058
8	0.446656	0.338949	0.566846	0.479065	0.468921	0.459903	3.172	1.0871





I run the process for 5 test case. With increase in Number of iterations, Difference between Maximum and Minimum and Global minimum decrease, which show that cloud is converging. From the Figure generated by using single thread code it can be seen that point converges to global as well as local minima, which is also the case for multithread program. From the table it seems like that using parallel version doesn't speed up the problem much, only a minimal increase in computation cost is seen. However, for this case we have a trivial function to compute. In case where function calculation is costly, we will see the speed up of execution.

To compile I use the following command:

For 8 processor

mpic++ -fopenmp CRS_P.cpp && mpirun -np 8 ./a.out "diff_8" "min_8"

For 4 processor

mpic++ -fopenmp CRS_P.cpp && mpirun -np 4 ./a.out "diff_4" "min_4"

similarly for #1,2.

Note: I used Toan’s code as a reference for MPI communication.