

## b. Developing PSO code

```
hold off; clear; close all; clc;
```

Define the eggholder function

```
fun = @(x)(-(x(2)+47)*sin(sqrt(abs((x(1)/2) + (x(2) + 47)))) + ...  
    - x(1)*sin(sqrt(abs(x(1) - (x(2) +47))))))
```

```
fun = function_handle with value:  
    @(x)(-(x(2)+47)*sin(sqrt(abs((x(1)/2)+(x(2)+47))))+-x(1)*sin(sqrt(abs(x(1)-(x(2)+47))))))
```

```
% True value of global optimum (check function works) and for comparison  
% later  
true = fun([512, 404.2319]);
```

Run `x = particleswarm(fun,nvars)` where `nvars` is the number of dimensions.

```
nvars = 2;
```

The bounds are the search domain as given in the wikipedia page

```
lb = [-512,-512];  
ub = [512,512];
```

The options as with all optimisation can be defined. The swarm size is simply the number of particles, and the number of iterations that are carried out is `MaxIterations`, after attempting with default parameters, I set the options to those shown below as it provided a correct solution quickly.

Generally a greater number of iterations and a greater swarm size will result in better results at the cost of computational price.

```
options = optimoptions('particleswarm','SwarmSize',200,MaxIterations=20);  
  
[x, fval] = particleswarm(fun,nvars,lb,ub,options)
```

```
Optimization ended: number of iterations exceeded OPTIONS.MaxIterations.  
x = 1x2  
    512.0000    404.1914  
fval = -959.6388
```

```
error = true - fval
```

```
error = -0.0019
```

Test for many attempts

```
hold on  
for repeat = 1:20  
    fvals = [];  
    for i = 1: 20  
        options = optimoptions('particleswarm','SwarmSize',200,MaxIterations=i);  
        [x, fval] = particleswarm(fun,nvars,lb,ub,options);
```



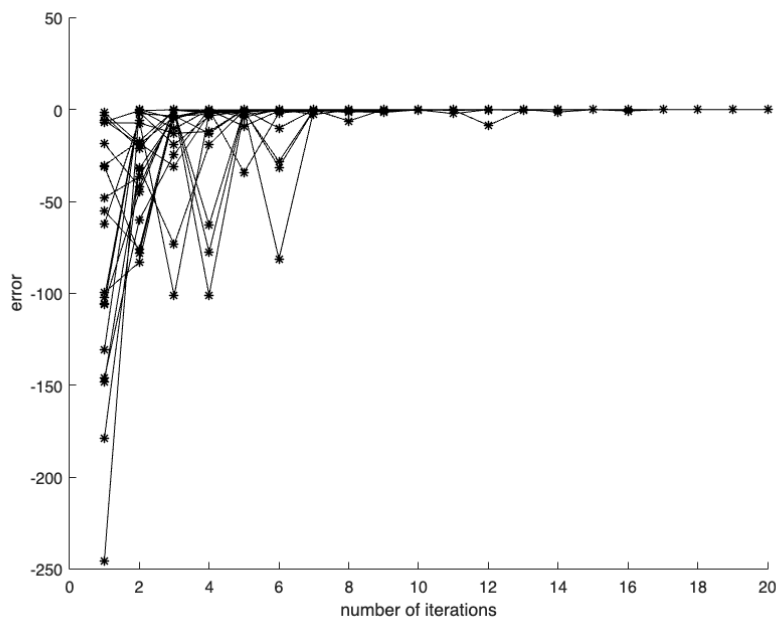










[illegible]

After plotting the optimisation for 20 goes at the same problem you can see the stochastic property of the algorithm and also that it is not always guaranteed to converge in the set time steps. A better measure of convergence would be the change on previous fval estimates which would stop the algorithm from diverging after attaining a zero error.