**# Genetic Algorithm Optimization**

**for Function 1 with D = 2 and D = 10**

clc, clear, close all

rng default

global initial\_flag

options = optimoptions('ga', 'PlotFcn', {@gaplotbestf, @gaplotdistance});

dimensions = [2, 10];

func = 1;

for dim = dimensions

ga\_main\_val = zeros(1, 15);

ga\_main\_exit\_flag = zeros(1, 15);

ga\_main\_op = zeros(1, 15);

for i = 1:15

initial\_flag = 0;

[ga\_x, ga\_val, ga\_exit\_flag, ga\_output] = ga(@(x) benchmark\_func(x, func), dim, [], [], [], [], [], [], [], options);

fname = sprintf('filename\_GA\_Func1\_D%d\_Iter%d.fig', dim, i);

savefig(fname);

% Store results in arrays

ga\_main\_val(i) = ga\_val;

ga\_main\_exit\_flag(i) = ga\_exit\_flag;

if isfield(ga\_output, 'fval')

ga\_main\_op(i) = min(ga\_output.fval);

else

ga\_main\_op(i) = NaN;

end

end

% GA iteration measures for Dimension dim

ga\_val\_max = max(ga\_main\_val);

ga\_val\_min = min(ga\_main\_val);

ga\_val\_mean = mean(ga\_main\_val);

ga\_val\_std = std(ga\_main\_val);

% Store or display these statistics as needed

end

**% Particle Swarm Optimization**

**for Function 1 with D = 2 and D = 10**

clc, clear, close all

rng default

global initial\_flag

options = optimoptions('particleswarm', 'PlotFcn', {@pswplotbestf});

dimensions = [2, 10];

func = 1;

for dim = dimensions

pso\_main\_val = zeros(1, 15);

pso\_main\_exit\_flag = zeros(1, 15);

pso\_main\_op = zeros(1, 15);

for i = 1:15

initial\_flag = 0;

[pso\_x, pso\_val, pso\_exit\_flag, pso\_output] = particleswarm(@(x) benchmark\_func(x, func), dim, [], [], options);

fname = sprintf('filename\_PSO\_Func1\_D%d\_Iter%d.fig', dim, i);

savefig(fname);

% Store results in arrays

pso\_main\_val(i) = pso\_val;

pso\_main\_exit\_flag(i) = pso\_exit\_flag;

if isfield(pso\_output, 'fval')

pso\_main\_op(i) = min(pso\_output.fval);

else

pso\_main\_op(i) = NaN;

end

end

% PSO iteration measures

% ... similar to GA ...

end

**# Pattern Search Optimization**

**for Function 1 with D = 2 and D = 10**

clc, clear, close all

rng default

global initial\_flag

options = optimoptions('patternsearch', 'PlotFcn', {@psplotbestf, @psplotbestx});

dimensions = [2, 10];

func = 1;

for dim = dimensions

ps\_main\_val = zeros(1, 15);

ps\_main\_exit\_flag = zeros(1, 15);

ps\_main\_op = zeros(1, 15);

for i = 1:15

initial\_flag = 0;

[ps\_x, ps\_val, ps\_exit\_flag, ps\_output] = patternsearch(@(x) benchmark\_func(x, func), rand(1, dim), [], [], [], [], [], [], options);

fname = sprintf('filename\_PS\_Func1\_D%d\_Iter%d.fig', dim, i);

savefig(fname);

% Store results in arrays

ps\_main\_val(i) = ps\_val;

ps\_main\_exit\_flag(i) = ps\_exit\_flag;

if isfield(ps\_output, 'fval')

ps\_main\_op(i) = min(ps\_output.fval);

else

ps\_main\_op(i) = NaN;

end

end

% Pattern Search iteration measures

% ... similar to GA ...

end

**# Genetic Algorithm Optimization**

**for Function 2 with D = 2 and D = 10**

clc, clear, close all

rng default

global initial\_flag

options = optimoptions('ga', 'PlotFcn', {@gaplotbestf, @gaplotdistance});

dimensions = [2, 10];

func = 2;

for dim = dimensions

ga\_main\_val = zeros(1, 15);

ga\_main\_exit\_flag = zeros(1, 15);

ga\_main\_op = zeros(1, 15);

for i = 1:15

initial\_flag = 0;

[ga\_x, ga\_val, ga\_exit\_flag, ga\_output] = ga(@(x) benchmark\_func(x, func), dim, [], [], [], [], [], [], [], options);

fname = sprintf('filename\_GA\_Func2\_D%d\_Iter%d.fig', dim, i);

savefig(fname);

% Store results in arrays

ga\_main\_val(i) = ga\_val;

ga\_main\_exit\_flag(i) = ga\_exit\_flag;

if isfield(ga\_output, 'fval')

ga\_main\_op(i) = min(ga\_output.fval);

else

ga\_main\_op(i) = NaN;

end

end

% GA iteration measures for Dimension dim

ga\_val\_max = max(ga\_main\_val);

ga\_val\_min = min(ga\_main\_val);

ga\_val\_mean = mean(ga\_main\_val);

ga\_val\_std = std(ga\_main\_val);

% Store or display these statistics as needed

end

**% Particle Swarm Optimization**

**for Function 2 with D = 2 and D = 10**

clc, clear, close all

rng default

global initial\_flag

options = optimoptions('particleswarm', 'PlotFcn', {@pswplotbestf});

dimensions = [2, 10];

func = 2;

for dim = dimensions

pso\_main\_val = zeros(1, 15);

pso\_main\_exit\_flag = zeros(1, 15);

pso\_main\_op = zeros(1, 15);

for i = 1:15

initial\_flag = 0;

[pso\_x, pso\_val, pso\_exit\_flag, pso\_output] = particleswarm(@(x) benchmark\_func(x, func), dim, [], [], options);

fname = sprintf('filename\_PSO\_Func2\_D%d\_Iter%d.fig', dim, i);

savefig(fname);

% Store results in arrays

pso\_main\_val(i) = pso\_val;

pso\_main\_exit\_flag(i) = pso\_exit\_flag;

if isfield(pso\_output, 'fval')

pso\_main\_op(i) = min(pso\_output.fval);

else

pso\_main\_op(i) = NaN;

end

end

% PSO iteration measures

% ... similar to GA ...

end

**# Pattern Search Optimization**

**for Function 2 with D = 2 and D = 10**

clc, clear, close all

rng default

global initial\_flag

options = optimoptions('patternsearch', 'PlotFcn', {@psplotbestf, @psplotbestx});

dimensions = [2, 10];

func = 2;

for dim = dimensions

ps\_main\_val = zeros(1, 15);

ps\_main\_exit\_flag = zeros(1, 15);

ps\_main\_op = zeros(1, 15);

for i = 1:15

initial\_flag = 0;

[ps\_x, ps\_val, ps\_exit\_flag, ps\_output] = patternsearch(@(x) benchmark\_func(x, func), rand(1, dim), [], [], [], [], [], [], options);

fname = sprintf('filename\_PS\_Func2\_D%d\_Iter%d.fig', dim, i);

savefig(fname);

% Store results in arrays

ps\_main\_val(i) = ps\_val;

ps\_main\_exit\_flag(i) = ps\_exit\_flag;

if isfield(ps\_output, 'fval')

ps\_main\_op(i) = min(ps\_output.fval);

else

ps\_main\_op(i) = NaN;

end

end

% Pattern Search iteration measures

% ... similar to GA ...

end

**# Genetic Algorithm Optimization**

**for Function 3 with D = 2 and D = 10**

clc, clear, close all

rng default

global initial\_flag

options = optimoptions('ga', 'PlotFcn', {@gaplotbestf, @gaplotdistance});

dimensions = [2, 10];

func = 3;

for dim = dimensions

ga\_main\_val = zeros(1, 15);

ga\_main\_exit\_flag = zeros(1, 15);

ga\_main\_op = zeros(1, 15);

for i = 1:15

initial\_flag = 0;

[ga\_x, ga\_val, ga\_exit\_flag, ga\_output] = ga(@(x) benchmark\_func(x, func), dim, [], [], [], [], [], [], [], options);

fname = sprintf('filename\_GA\_Func3\_D%d\_Iter%d.fig', dim, i);

savefig(fname);

% Store results in arrays

ga\_main\_val(i) = ga\_val;

ga\_main\_exit\_flag(i) = ga\_exit\_flag;

if isfield(ga\_output, 'fval')

ga\_main\_op(i) = min(ga\_output.fval);

else

ga\_main\_op(i) = NaN;

end

end

% GA iteration measures for Dimension dim

ga\_val\_max = max(ga\_main\_val);

ga\_val\_min = min(ga\_main\_val);

ga\_val\_mean = mean(ga\_main\_val);

ga\_val\_std = std(ga\_main\_val);

% Store or display these statistics as needed

end

**% Particle Swarm Optimization**

**for Function 3 with D = 2 and D = 10**

clc, clear, close all

rng default

global initial\_flag

options = optimoptions('particleswarm', 'PlotFcn', {@pswplotbestf});

dimensions = [2, 10];

func = 3;

for dim = dimensions

pso\_main\_val = zeros(1, 15);

pso\_main\_exit\_flag = zeros(1, 15);

pso\_main\_op = zeros(1, 15);

for i = 1:15

initial\_flag = 0;

[pso\_x, pso\_val, pso\_exit\_flag, pso\_output] = particleswarm(@(x) benchmark\_func(x, func), dim, [], [], options);

fname = sprintf('filename\_PSO\_Func3\_D%d\_Iter%d.fig', dim, i);

savefig(fname);

% Store results in arrays

pso\_main\_val(i) = pso\_val;

pso\_main\_exit\_flag(i) = pso\_exit\_flag;

if isfield(pso\_output, 'fval')

pso\_main\_op(i) = min(pso\_output.fval);

else

pso\_main\_op(i) = NaN;

end

end

% PSO iteration measures

% ... similar to GA ...

end

**# Pattern Search Optimization**

**for Function 3 with D = 2 and D = 10**

clc, clear, close all

rng default

global initial\_flag

options = optimoptions('patternsearch', 'PlotFcn', {@psplotbestf, @psplotbestx});

dimensions = [2, 10];

func = 3;

for dim = dimensions

ps\_main\_val = zeros(1, 15);

ps\_main\_exit\_flag = zeros(1, 15);

ps\_main\_op = zeros(1, 15);

for i = 1:15

initial\_flag = 0;

[ps\_x, ps\_val, ps\_exit\_flag, ps\_output] = patternsearch(@(x) benchmark\_func(x, func), rand(1, dim), [], [], [], [], [], [], options);

fname = sprintf('filename\_PS\_Func3\_D%d\_Iter%d.fig', dim, i);

savefig(fname);

% Store results in arrays

ps\_main\_val(i) = ps\_val;

ps\_main\_exit\_flag(i) = ps\_exit\_flag;

if isfield(ps\_output, 'fval')

ps\_main\_op(i) = min(ps\_output.fval);

else

ps\_main\_op(i) = NaN;

end

end

% Pattern Search iteration measures

% ... similar to GA ...

end