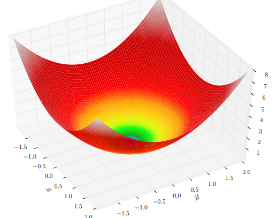
Overleaf Document Link: https://www.overleaf.com/3771542299ndmxbvjffbfh

Algorithms

* BFGS Quasi Newton with cubic line search
  + Implement with fminunc in MATLAB
* Subspace Trust Region Method
  + Implement with fminunc in MATLAB
* Simplex Search Method
  + Implement with fminsearch in MATLAB

Test Problems

* Sphere Function (https://en.wikipedia.org/wiki/Test\_functions\_for\_optimization)
* Rosenbrock Function (https://en.wikipedia.org/wiki/Rosenbrock\_function)
* Rastrigin Function (https://en.wikipedia.org/wiki/Rastrigin\_function)
* Himmelblau’s Function (<https://en.wikipedia.org/wiki/Himmelblau%27s_function>)

**--------------------------------------------------------------**

**MA 5630 Numerical Optimization Project 1**

**Aditya Koranne 02/10/2021**

**--------------------------------------------------------------**

Sphere Function [1]

**Solver**

n=3; % Number of dimensions

save n.mat;

fun=@sphere; % Select test problem (rastrigin, sphere, rosenbrock)

options=optimoptions('fminunc','Algorithm',"quasi-newton");

q=zeros(1,4);r=zeros(1,n);t=0;

for k=1:3 % Conducting k runs to obtain average results

x0=5-10\*rand(n,1); %random initialization [-5, 5]

tic;

[x,fval,exitflag,output]=fminunc(fun,x0,options);

t1=toc;

[q]=[q;fval,exitflag,output.funcCount,output.iterations];

[r]=[r;x'];

[t]=[t,t1];

end

q(1,:)=[];r(1,:)=[];t(1)=[];

q=mean(q);r=mean(r);t=mean(t);

fprintf(['AVERAGE RESULTS (',num2str(k),' samples):',...

'\n 1. Test Function: ',func2str(fun), ...

'\n 2. Algorithm: ',num2str(options.Algorithm),...

' \n 3. f(x\*):',num2str(q(1,1)),...

' \n 4. x\*:',num2str(r),...

' \n 5. fcount:',num2str(round(q(1,3))),...

' \n 6. iterations:',num2str(round(q(1,4))),...

' \n 7. computiontime:',num2str(t),'s']);

AVERAGE RESULTS (3 samples):

1. Test Function: sphere

2. Algorithm: quasi-newton

3. f(x\*):1.9287e-13

4. x\*:9.3495e-09 -8.9836e-08 -2.572e-07

5. fcount:12

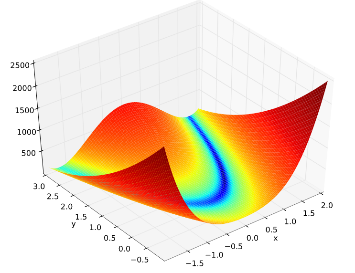
6. iterations:2

7. computiontime:0.036284s

**REMARKS: Sphere is the simplest convex optimization problem that makes it ideal for pilot run to any algorithm. The built in Quasi Newton algorithm in MATLAB is used to find global minimum of 3-dimensional sphere function. The results are accurate for the default optimal tolerance limit set at 10e-6. Observing computation time, it can be concluded that the algorithm works well for finding the global minimum of the function.**

**REFERNECES:**

[1] https://en.wikipedia.org/wiki/Test\_functions\_for\_optimization

**--------------------------------------------------------------**

**MA 5630 Numerical Optimization Project 1**

**Aditya Koranne 02/10/2021**

**--------------------------------------------------------------**

Rosenbrock Function [1]

**Solver**

n=3; % Number of dimensions

save n.mat;

fun=@rosenbrock; % Select test problem (rastrigin, sphere, rosenbrock)

options=optimoptions('fminunc','Algorithm',"quasi-newton");

q=zeros(1,4);r=zeros(1,n);t=0;

for k=1:3 % Conducting k runs to obtain average results

x0=5-10\*rand(n,1); %random initialization [-5, 5]

tic;

[x,fval,exitflag,output]=fminunc(fun,x0,options);

t1=toc;

[q]=[q;fval,exitflag,output.funcCount,output.iterations];

[r]=[r;x'];

[t]=[t,t1];

end

q(1,:)=[];r(1,:)=[];t(1)=[];

q=mean(q);r=mean(r);t=mean(t);

fprintf(['AVERAGE RESULTS (',num2str(k),' samples):',...

'\n 1. Test Function: ',func2str(fun), ...

'\n 2. Algorithm: ',num2str(options.Algorithm),...

' \n 3. f(x\*):',num2str(q(1,1)),...

' \n 4. x\*:',num2str(r),...

' \n 5. fcount:',num2str(round(q(1,3))),...

' \n 6. iterations:',num2str(round(q(1,4))),...

' \n 7. computiontime:',num2str(t),'s']);

AVERAGE RESULTS (3 samples):

1. Test Function: rosenbrock

2. Algorithm: quasi-newton

3. f(x\*):9.9493e-05

4. x\*:0.99784 0.99551 0.99087

5. fcount:215

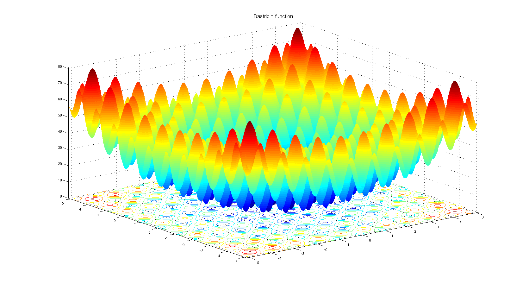
6. iterations:46

7. computiontime:0.36513s

**REMARKS: Rosenbrock is a non-convex optimization problem. The built in Quasi Newton algorithm in MATLAB is used to find global minimum of 3-dimensional rosenbrock function. The results are accurate for the default optimal tolerance limit set at 10e-6. Observing computation time, it can be concluded that the algorithm works well for finding the global minimum but may fetch better results by comparing other algorithms such as Trust Region Method.**

**REFERNECES:**

[1] https://en.wikipedia.org/wiki/Rosenbrock\_function

**--------------------------------------------------------------**

**MA 5630 Numerical Optimization Project 1**

**Aditya Koranne 02/10/2021**

**--------------------------------------------------------------**

Rastrigin Function[1]

**Solver**

n=3; % Number of dimensions

save n.mat;

fun=@rastrigin; % Select test problem (rastrigin, sphere, rosenbrock)

options=optimoptions('fminunc','Algorithm',"quasi-newton");

q=zeros(1,4);r=zeros(1,n);t=0;

for k=1:3 % Conducting k runs to obtain average results

x0=5-10\*rand(n,1); %random initialization [-5, 5]

tic;

[x,fval,exitflag,output]=fminunc(fun,x0,options);

t1=toc;

[q]=[q;fval,exitflag,output.funcCount,output.iterations];

[r]=[r;x'];

[t]=[t,t1];

end

q(1,:)=[];r(1,:)=[];t(1)=[];

q=mean(q);r=mean(r);t=mean(t);

fprintf(['AVERAGE RESULTS (',num2str(k),' samples):',...

'\n 1. Test Function: ',func2str(fun), ...

'\n 2. Algorithm: ',num2str(options.Algorithm),...

' \n 3. f(x\*):',num2str(q(1,1)),...

' \n 4. x\*:',num2str(r),...

' \n 5. fcount:',num2str(round(q(1,3))),...

' \n 6. iterations:',num2str(round(q(1,4))),...

' \n 7. computiontime:',num2str(t),'s']);

AVERAGE RESULTS (3 samples):

1. Test Function: rastrigin

2. Algorithm: quasi-newton

3. f(x\*):61.5068

4. x\*:-2.6532 -0.66329 0.99494

5. fcount:43

6. iterations:8

7. computiontime:0.089839s

**REMARKS: Rastrigin is a multimodal non-convex optimization problem that makes it extremely difficult to reach global optimum using derivative methods. The built in Quasi Newton algorithm in MATLAB is used to find global minimum of 3-dimensional rastrigin function. The results obtained are of local minimum and varies depending on the initialization of the function. Per my hypothesis, evolutionary algorithms might work well for this problem(s) which can be explored in future projects.**

**REFERNECES:**

[1] https://en.wikipedia.org/wiki/Rastrigin\_function