For 2a the codes has 1 function:”fun\_2a”

The function is listed as follow:

function fun = fun\_2a(x,L1,L2,R1,R2)

fun = [(x(1)-L1(1))^2+(x(2)-L1(2))^2-R1^2;

(x(1)-L2(1))^2+(x(2)-L2(2))^2-R2^2];

fun = fun(1)^2+fun(2)^2;

end

The code is listed as follow:

clc;

clear all;

L1 = [0,0];

L2 = [5,5];

R1=2.5;

R2=5;

fun = @(x) fun\_2a(x,L1,L2,R1,R2);

x0 = [10,0];

options = optimoptions('fminunc','OptimalityTolerance',10e-6);

[x1,fval] = fminunc(fun,x0,options);

x0\_2=[0,10];

[x2,fval2]=fminunc(fun,x0\_2,options);

theta = linspace(-pi,pi,1000);

circle1\_x = L1(1)+R1\*cos(theta);

circle1\_y = L1(2)+R1\*sin(theta);

circle2\_x = L2(1)+R2\*cos(theta);

circle2\_y = L2(2)+R2\*sin(theta);

figure(1)

hold on

plot(circle1\_x,circle1\_y);

plot(circle2\_x,circle2\_y);

plot(x1(1),x1(2),'Marker','\*');

plot(x2(1),x2(2),'Marker','\*');

xlabel('x-axis')

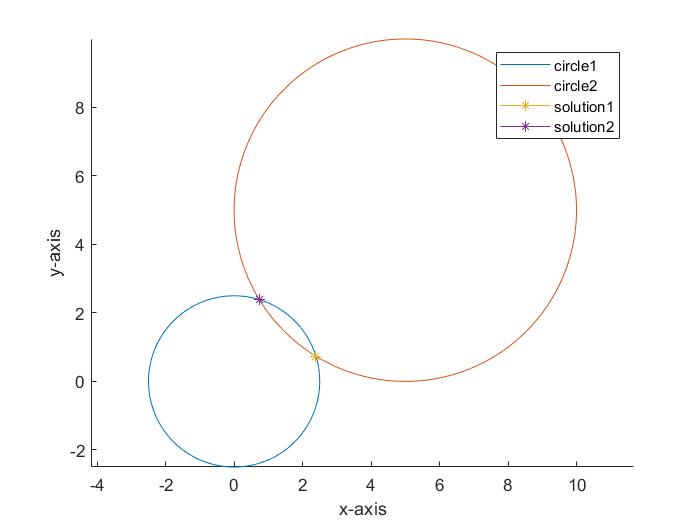
ylabel('y-axis')

axis equal

legend('circle1','circle2','solution1','solution2');

hold off

The picture looks as following:



Picture1: circles and solutions

For 2b the codes are listed as follows:

clc;

clear all;

global h;

h.x=[];

h.fval=[];

L1 = [0,0];

L2 = [5,5];

R1=2.5;

R2=5;

fun = @(x) fun\_2a(x,L1,L2,R1,R2);

x0 = [10,0];

options = optimoptions('fminunc','OptimalityTolerance',10e-16,'OutputFcn',@outfun);

[x1,fval] = fminunc(fun,x0,options);

x0\_2=[0,10];

%[x2,fval2]=fminunc(fun,x0\_2,options);

figure(1)

iter = 1:length(h.x);

xxx = h.x-x1;

semilogy(iter,abs(xxx(:,1)),'b','linewidth',1);

hold all

semilogy(iter,abs(xxx(:,2)),'g','linewidth',1);

xlim([min(iter),max(iter)]);

xticks(1:2:max(iter));

xlabel('iternation');

ylabel('logarithmic error');

legend('error of x','error of y');

grid on;

function stop = outfun(x,optimValues,state)

global h;

stop = false;

switch state

case 'iter'

h.fval = [h.fval; optimValues.fval];

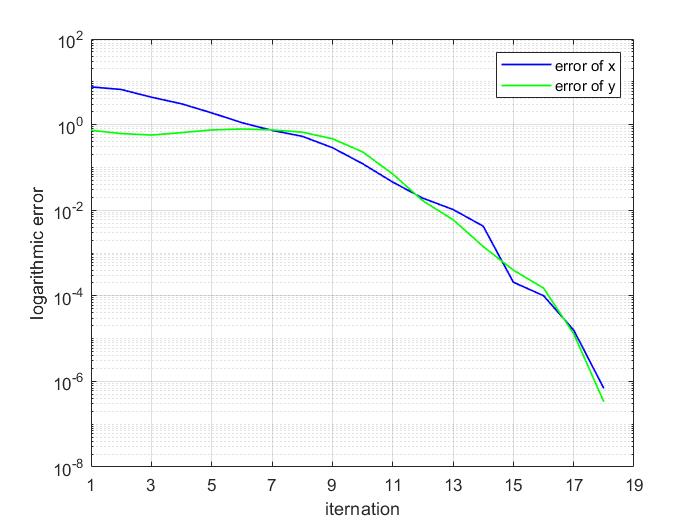
h.x=[h.x;x];

otherwise

end

end

The picture looks as following:



Picture2: error versus iteration plot

For 2c the codes are listed as follows:

clc;

clear all;

L1 = [0,0];

L2 = [5,5];

R1=2.5;

R2=5;

fun = @(x) fun\_2a(x,L1,L2,R1,R2);

x0 = [10,0];

options = optimoptions('fminunc','OptimalityTolerance',10e-11);

[x1,fval] = fminunc(fun,x0,options);

x0\_2=[0,10];

[x2,fval2]=fminunc(fun,x0\_2,options);

p = -5:0.5:10;

q=10:-0.5:-5;

up\_one = [];

down\_one = [];

local\_opt= [];

cc = [];

for i=1:length(p)

for j=1:length(q)

dot = [p(i),q(j)];

[x,fval] = fminunc(fun,dot,options);

if(norm(x2-x)<10e-6)

up\_one = [up\_one;dot];

else if(norm(x1-x)<10e-6)

down\_one = [down\_one;dot];

else

local\_opt = [local\_opt;dot];

end

end

end

end

hold all

scatter(up\_one(:,1),up\_one(:,2),[],'r');

scatter(down\_one(:,1),down\_one(:,2),[],'b');

scatter(local\_opt(:,1),local\_opt(:,2),[],'g');

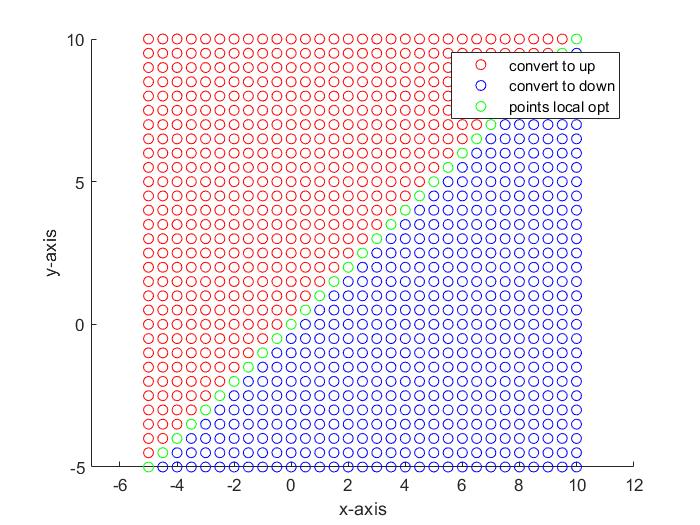
xlabel('x-axis')

ylabel('y-axis')

axis equal

legend('convert to up','convert to down','points local opt');

The picture looks as following:



Picture3: converge positions

For 3a we use a function:”fun\_3a”, codes are listed as following:

function fun = fun\_3a(x,L1,L2,L3,R1,R2,R3)

fun = [(x(1)-L1(1))^2+(x(2)-L1(2))^2-R1^2;

(x(1)-L2(1))^2+(x(2)-L2(2))^2-R2^2;

(x(1)-L3(1))^2+(x(2)-L3(2))^2-R3^2];

fun = fun(1)^2+fun(2)^2+fun(3)^2;

end

the main codes are listed as follows:

clc;

clear all;

L1 = [0,0];

L2 = [5,5];

L3=[2.5,0];

R1=2.5;

R2=5;

R3=3;

fun = @(x) fun\_3a(x,L1,L2,L3,R1,R2,R3);

x0 = [10,0];

options = optimoptions('fminunc','OptimalityTolerance',10e-16);

[x1,fval] = fminunc(fun,x0,options);

x0\_2=[0,10];

p = -5:1:10;

q=10:-1:-5;

solutions = [];

for i=1:length(p)

for j=1:length(q)

dot = [p(i),q(j)];

[sol,fval] = fminunc(fun,dot,options);

solutions=[solutions;sol];

end

end

theta = linspace(-pi,pi,1000);

circle1\_x = L1(1)+R1\*cos(theta);

circle1\_y = L1(2)+R1\*sin(theta);

circle2\_x = L2(1)+R2\*cos(theta);

circle2\_y = L2(2)+R2\*sin(theta);

c3\_x = L3(1)+R3\*cos(theta);

c3\_y = L3(2)+R3\*sin(theta);

figure(1)

hold all

plot(circle1\_x,circle1\_y);

plot(circle2\_x,circle2\_y);

plot(c3\_x,c3\_y);

scatter(solutions(:,1),solutions(:,2));

xlabel('x-axis')

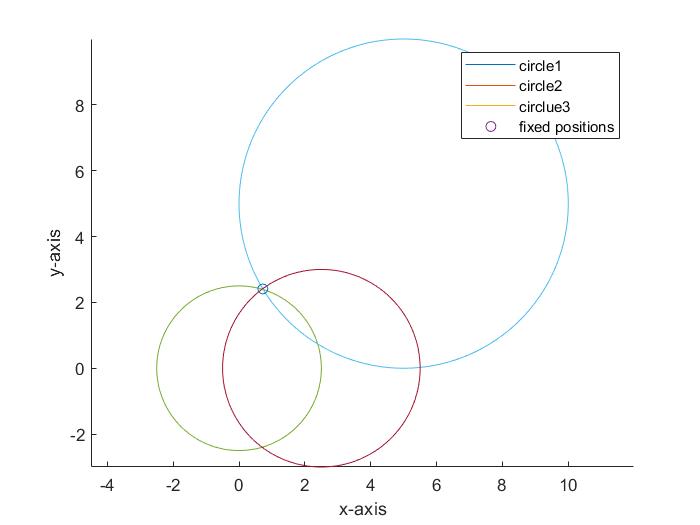
ylabel('y-axis')

axis equal

legend('circle1','circle2','circlue3','fixed positions');

hold off

The picture looks as follows:



Picture4: circles and fixed points

For 3b the codes are listed as follows:

clc;

clear all;

L1 = [0,0];

L2 = [5,5];

L3=[2.5,0];

R1=2.5;

R2=6;

R3=2;

fun = @(x) fun\_3a(x,L1,L2,L3,R1,R2,R3);

x0 = [10,0];

options = optimoptions('fminunc','OptimalityTolerance',10e-16);

[x1,fval] = fminunc(fun,x0,options);

x0\_2=[0,10];

[x2,fval] = fminunc(fun,x0\_2,options);

p = -5:5:10;

q=10:-5:-5;

solutions = [];

for i=1:length(p)

for j=1:length(q)

dot = [p(i),q(j)];

[sol,fval] = fminunc(fun,dot,options);

solutions=[solutions;sol];

end

end

theta = linspace(-pi,pi,1000);

circle1\_x = L1(1)+R1\*cos(theta);

circle1\_y = L1(2)+R1\*sin(theta);

circle2\_x = L2(1)+R2\*cos(theta);

circle2\_y = L2(2)+R2\*sin(theta);

c3\_x = L3(1)+R3\*cos(theta);

c3\_y = L3(2)+R3\*sin(theta);

figure(1)

hold all

plot(circle1\_x,circle1\_y);

plot(circle2\_x,circle2\_y);

plot(c3\_x,c3\_y);

scatter(solutions(:,1),solutions(:,2));

xlabel('x-axis')

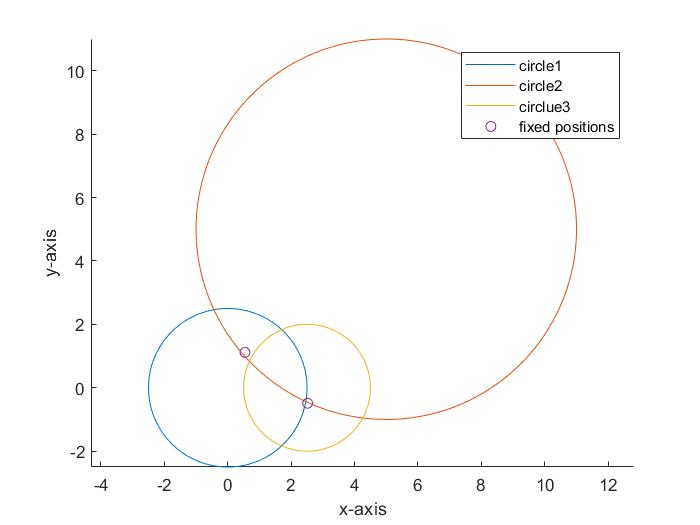
ylabel('y-axis')

axis equal

legend('circle1','circle2','circlue3','fixed positions');

hold off

The picture looks as follows



Picture5: circles and fixed positions

For 3c the codes are listed as follows:

clc;

clear all;

L1 = [0,0];

L2 = [5,5];

L3=[2.5,0];

R1=2.5;

R2=6;

R3=2;

fun = @(x) fun\_3a(x,L1,L2,L3,R1,R2,R3);

x0 = [10,0];

options = optimoptions('fminunc','OptimalityTolerance',10e-12);

[x1,fval\_1] = fminunc(fun,x0,options);

x0\_2=[0,10];

[x2,fval\_2] = fminunc(fun,x0\_2,options);

x0\_2=[0,10];

p = -5:0.5:10;

q=10:-0.5:-5;

sols1 = [];

sols2 = [];

for i=1:length(p)

for j=1:length(q)

dot = [p(i),q(j)];

[sol,fval] = fminunc(fun,dot,options);

if(norm(sol-x1)<10e-6)

sols1 = [sols1;dot];

else

sols2 = [sols2;dot];

end

end

end

hold all

scatter(sols1(:,1),sols1(:,2),[],'r');

scatter(sols2(:,1),sols2(:,2),[],'b');

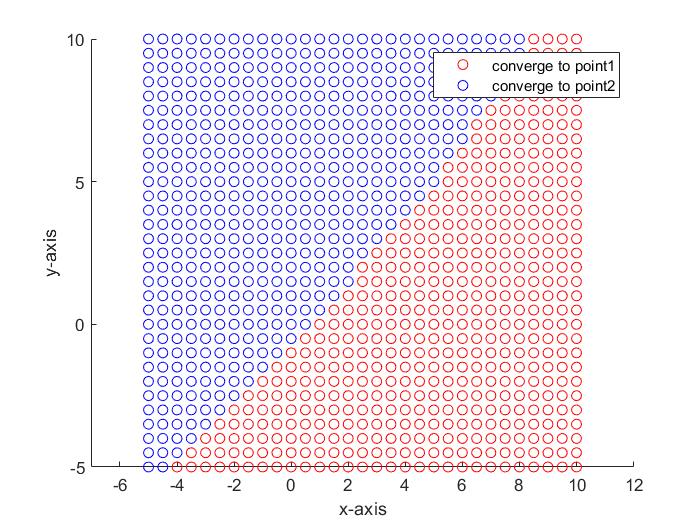
xlabel('x-axis')

ylabel('y-axis')

axis equal

legend('converge to point1','converge to point2');

The picture looks as following:



Picture6: converge positions

For 4a the codes are listed as follows:

clc;

clear all;

L1 = [0,0];

L2 = [5,5];

L3=[2.5,0];

R1=2.5;

R2=5;

R3=3;

fun = @(x) fun\_3a(x,L1,L2,L3,R1,R2,R3);

x0 = [0,4];

options = optimoptions('fminunc','OptimalityTolerance',10e-6);

[x1,fval] = fminunc(fun,x0,options);

theta = linspace(-pi,pi,1000);

circle1\_x = L1(1)+R1\*cos(theta);

circle1\_y = L1(2)+R1\*sin(theta);

circle2\_x = L2(1)+R2\*cos(theta);

circle2\_y = L2(2)+R2\*sin(theta);

c3\_x = L3(1)+R3\*cos(theta);

c3\_y = L3(2)+R3\*sin(theta);

figure(1)

hold all

plot(circle1\_x,circle1\_y,'b');

plot(circle2\_x,circle2\_y,'b');

plot(c3\_x,c3\_y,'b');

xlabel('x-axis')

ylabel('y-axis')

axis equal

nosie\_x = zeros(100,2);

a=2;

for i=1:1:100

R1\_new = R1+a\*(rand(1)-0.5);

R2\_new = R2+a\*(rand(1)-0.5);

R3\_new = R3+a\*(rand(1)-0.5);

fun = @(x) fun\_3a(x,L1,L2,L3,R1\_new,R2\_new,R3\_new);

options = optimoptions('fminunc','OptimalityTolerance',10e-6);

[x\_new,fval] = fminunc(fun,x0,options);

noise\_x(i,:) = x\_new;

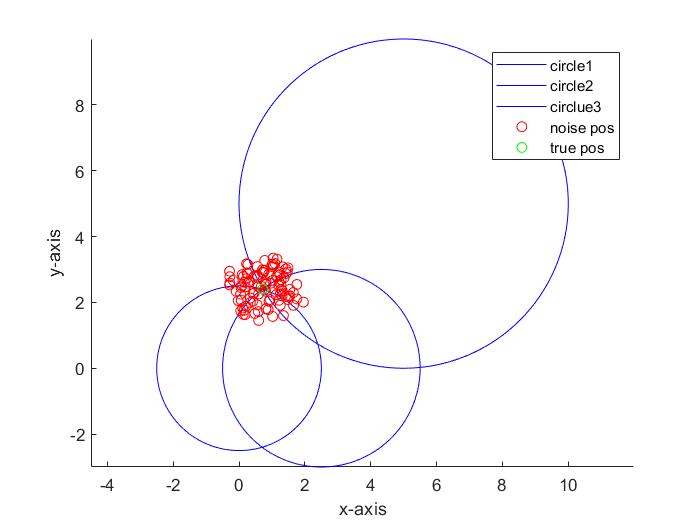
end

scatter(noise\_x(:,1),noise\_x(:,2),[],'r');

scatter(x1(1),x1(2),'g');

legend('circle1','circle2','circlue3','noise pos','true pos');

The picture looks as follows:



Picture7: true and noised pos

For 4b the codes are listed as follows:

clc;

clear all;

L1 = [0,0];

L2 = [5,5];

L3=[2.5,0];

R1=2.5;

R2=5;

R3=3;

fun = @(x) fun\_3a(x,L1,L2,L3,R1,R2,R3);

x0 = [0,4];

options = optimoptions('fminunc','OptimalityTolerance',10e-6);

[x1,fval] = fminunc(fun,x0,options);

%{

theta = linspace(-pi,pi,1000);

circle1\_x = L1(1)+R1\*cos(theta);

circle1\_y = L1(2)+R1\*sin(theta);

circle2\_x = L2(1)+R2\*cos(theta);

circle2\_y = L2(2)+R2\*sin(theta);

c3\_x = L3(1)+R3\*cos(theta);

c3\_y = L3(2)+R3\*sin(theta);

figure(1)

hold all

plot(circle1\_x,circle1\_y,'b');

plot(circle2\_x,circle2\_y,'b');

plot(c3\_x,c3\_y,'b');

xlabel('x-axis')

ylabel('y-axis')

axis equal

%}

x\_a=[];

far = [];

for a = 0.05:0.05:10

most\_dis = 0;

x\_a = [x\_a,a];

for i=1:1:100

R1\_new = R1+a\*(rand(1)-0.5);

R2\_new = R2+a\*(rand(1)-0.5);

R3\_new = R3+a\*(rand(1)-0.5);

fun = @(x) fun\_3a(x,L1,L2,L3,R1\_new,R2\_new,R3\_new);

options = optimoptions('fminunc','OptimalityTolerance',10e-6);

[x\_new,fval] = fminunc(fun,x0,options);

if(norm(x\_new-x1)>most\_dis)

most\_dis = norm(x\_new-x1);

end

end

far = [far;most\_dis];

end

figure(1)

semilogy(x\_a,far,'g','linewidth',1);

xlim([min(x\_a),max(x\_a)]);

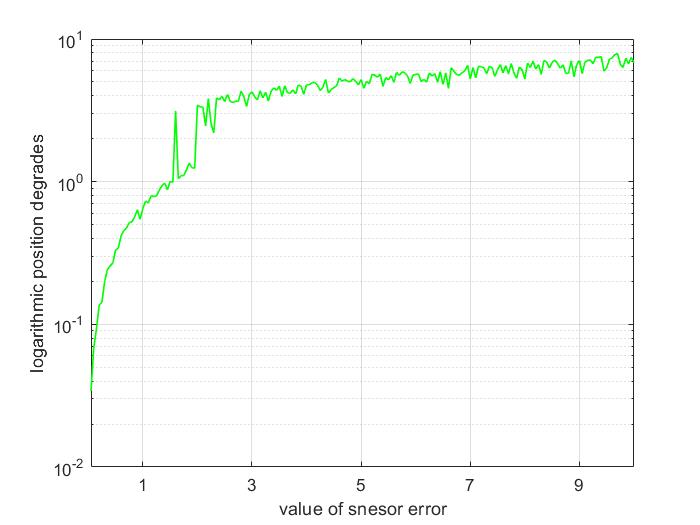
xticks(1:2:max(x\_a));

xlabel('value of snesor error');

ylabel('logarithmic position degrades');

grid on;

The picture looks as follows:



Picture8: degrades versus error