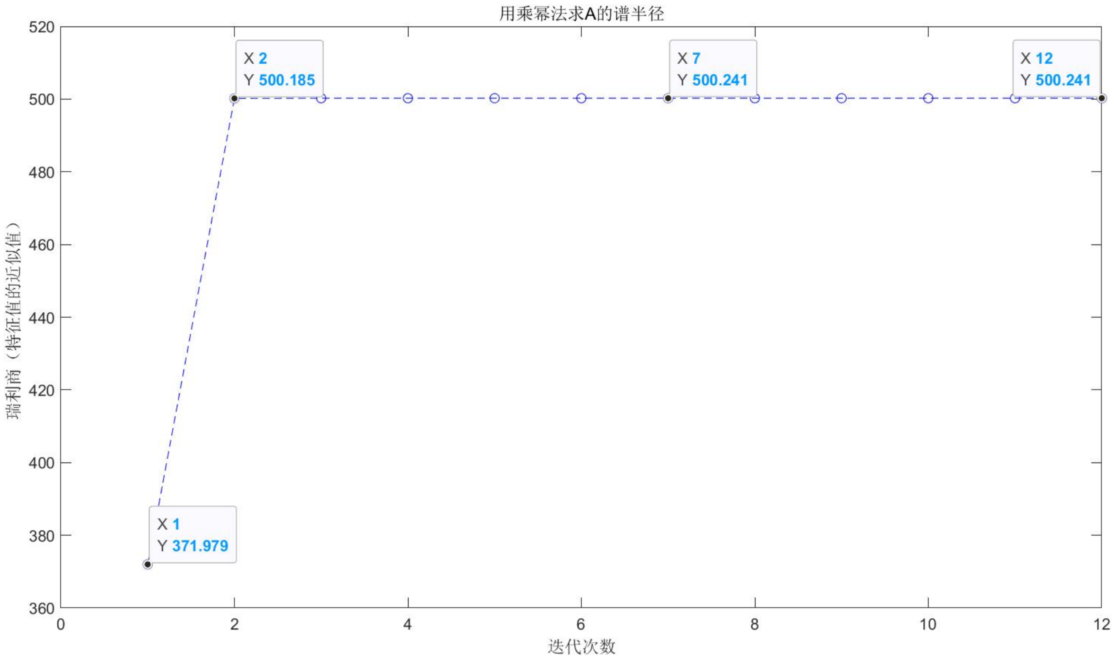
第二题

- 使用以下 matlab 代码：

```
format long;
A=rand(1000,1000);
x=rand(1000,1);
norm_of_A=norm(A);
x=x/norm(x);%输入的 x 进行归一化
flg=0;
count=0;
lambda_list=zeros(10^3,1);
for i=1:(10^3) %迭代次数保护
    count=count+1;
    y=A*x;
    L_square=x'*x;
    lambda=(x'*y)/L_square; %瑞利商
    lambda_list(count,1)=lambda;%记录一下本次获得的特征值的近似值
    r=y-lambda*x;%残差
    x=y/norm(y);%使用 2-范数进行归一化，得到单位化的 x 进入下一轮循环
    if norm(r)<=(norm_of_A+abs(lambda))*(10^-16)%
        % 10^-16 是机器精度，这个 tolerance 是相对 A 与 lambda 而变化的
        norm(r) %显示一下 r 的 2-范数
        disp("r 足够小，结束迭代")
        flg=1;
        break;
    end
end
if flg==0
    disp("触发迭代保护而退出")
end
plot([1:count],lambda_list(1:count,1),'b--o');
title("用乘幂法求 A 的谱半径")
xlabel("迭代次数")
ylabel("瑞利商（特征值的近似值）")
```

➢ 得到的收敛过程如下：



➢ 迭代次数为 12 次。

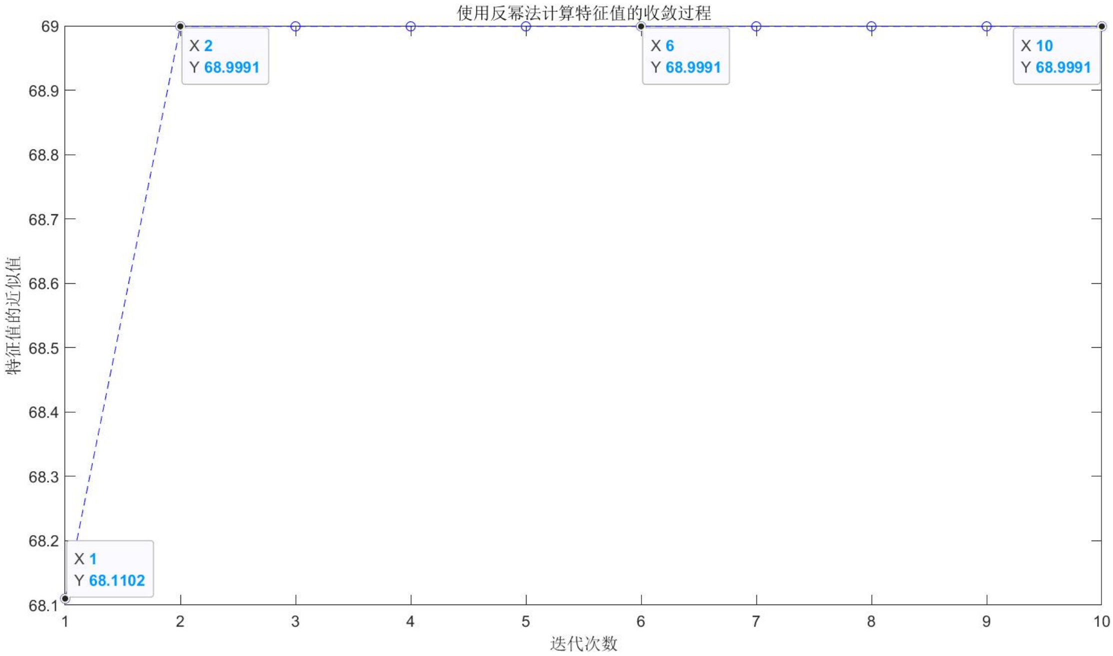
第三题

➢ 使用以下 matlab 代码，选择 69 作为本次实验的对象

```
%先随机产生一个实对称矩阵
Q = orth(rand(200,200));
D = diag([1:200]);
A = Q*D*Q';
%选择 lambda=69 作为试验对象
offset=0.0001;
[l,u,p] = lu(A-(69+offset)*eye(200));%距离 69+offset 最近的特征值是 69
N=norm(inv(A-(69+offset)*eye(200)));
x = rand(200,1);
x = x/norm(x);%归一化
flg=0;
count=0;
lambda_list=zeros(10^4,1);
for i = 1:(10^4)%迭代次数保护
    count=count+1;
    y = u\(l\(p'*x));
   %y=(A-(69+offset)*eye(200))\x;
```

```
    t=(y'*x)/(x'*x);%瑞利商
    lambda=1/t+(69+offset);
    lambda_list(count,1)=lambda;
    r=y-t*x;%计算残差
    x=y/norm(y);%归一化，以备进入下一轮循环
    if norm(r)<=(N+abs(t))*(10^-16)
        disp("残差 r 足够小，结束反幂法迭代")
        flg=1;
        break
    end
end
if flg==0
    disp("触发迭代保护")
end
if flg==1
    plot([1:count],lambda_list(1:count,1),'b--o');
    title("使用反幂法计算特征值的收敛过程")
    xlabel("迭代次数")
    ylabel("特征值的近似值")
end
```

- 收敛过程如下：



- 迭代次数为 10 次，可以观察到瑞利商迅速收敛到 69 附近。

➢ 运行的时间为 0.182，每一行代码运行时间的详细报告如下：

函数列表

时间	调用次数	行	
0.013	1	2	Q = orth(rand(200,200));
< 0.001	1	3	D = diag([1:200]);
< 0.001	1	4	A = Q*D*Q';
		5	%选择lambda=69作为试验对象
< 0.001	1	6	offset=0.0001;
0.002	1	7	[l,u,p] = lu(A-(69+offset)*eye(200));%距离69+offset最近的特征值
0.006	1	8	N=norm(inv(A-(69+offset)*eye(200)));
< 0.001	1	9	x = rand(200,1);
< 0.001	1	10	x = x/norm(x);%归一化
< 0.001	1	11	flg=0;
< 0.001	1	12	count=0;
< 0.001	1	13	lambda_list=zeros(10^4,1);
< 0.001	1	14	for i = 1:(10^4)%迭代保护
< 0.001	10	15	count=count+1;
0.002	10	16	y = u\(l\(p'*x));
		17	%y=(A-(69+offset)*eye(200))\x;
< 0.001	10	18	t=(y'*x)/(x'*x);%瑞利商
< 0.001	10	19	lambda=1/t+(69+offset);
< 0.001	10	20	lambda_list(count,1)=lambda;
< 0.001	10	21	r=y-t*x;%计算残差
< 0.001	10	22	x=y/norm(y);%归一化，以备进入下一轮循环
< 0.001	10	23	if norm(r)<=(N+abs(t))*(10^-16)
< 0.001	1	24	disp("残差r足够小，结束反幂法迭代")
< 0.001	1	25	flg=1;
< 0.001	1	26	break
< 0.001	9	27	end
< 0.001	9	28	end
< 0.001	1	29	if flg==0
		30	disp("触发迭代保护")
< 0.001	1	31	end
< 0.001	1	32	if flg==1
0.110	1	33	plot([1:count],lambda_list(1:count,1),'b--o');
0.026	1	34	title("使用反幂法计算特征值的收敛过程")
0.011	1	35	xlabel("迭代次数")
0.007	1	36	ylabel("特征值的近似值")

➢ 除去一开始随机生成正交基、对（A-λI）进行 LU 分解，以及最后绘图占用的大部分时间，在迭代过程中，占用最多的是解方程组（A-λI）*y=x 的步骤。

➢

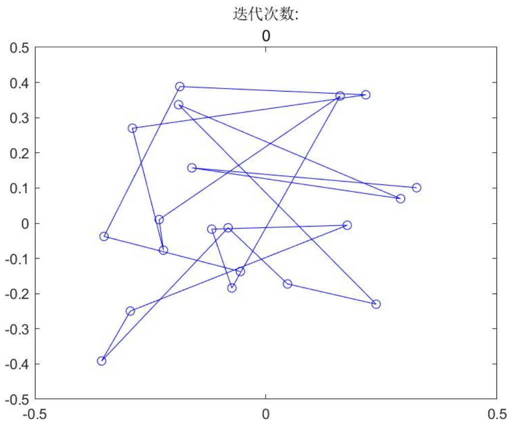
第四题

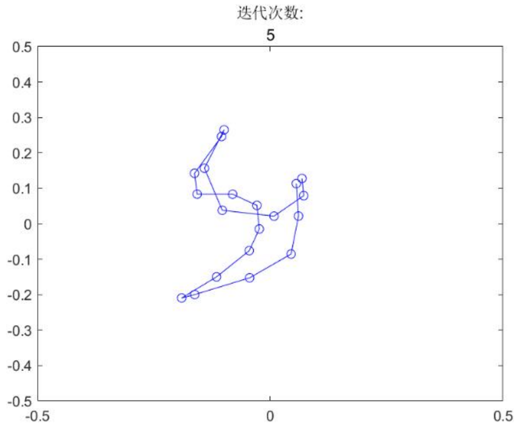
实验一（数据点趋向“平均点”）

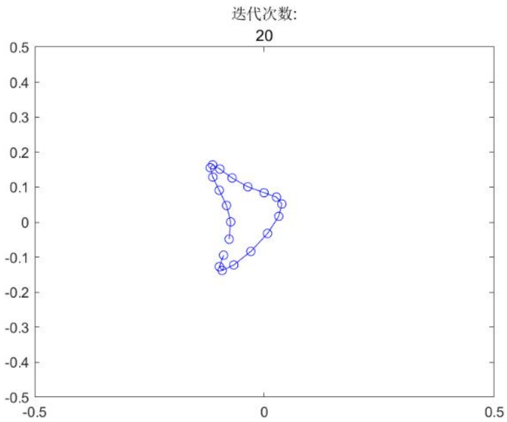
➢ Matlab 代码如下

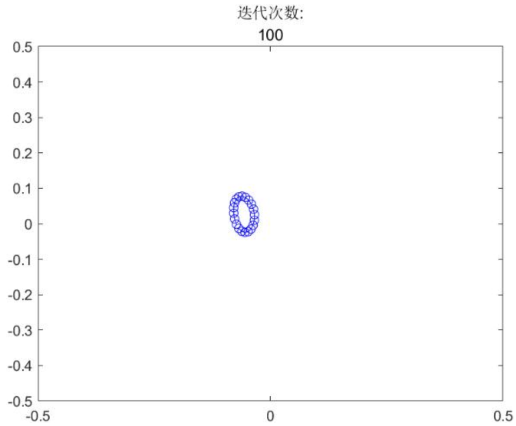
```
n = 20;
M=diag(ones(n,1))+diag(ones(n-1,1),1);
M(n,1)=1;
M=0.5*M;
x = -0.5+rand(n,1);
x = x/norm(x);
y = -0.5+rand(n,1);
y = y/norm(y);
figure();%初始状态
plot(x,y,'b-o');
lim=0.5;
xlim([-lim,lim]);
ylim([-lim,lim]);
title(["迭代次数:",0]);
for k = 1:200
    x=M*x;
    y=M*y;
    if k==5 || k==20 || k==100 ||k==200
        figure();
        plot(x,y,'b-o')
        xlim([-lim,lim]);
        ylim([-lim,lim]);
        title(["迭代次数:",k]);
    end
end
```

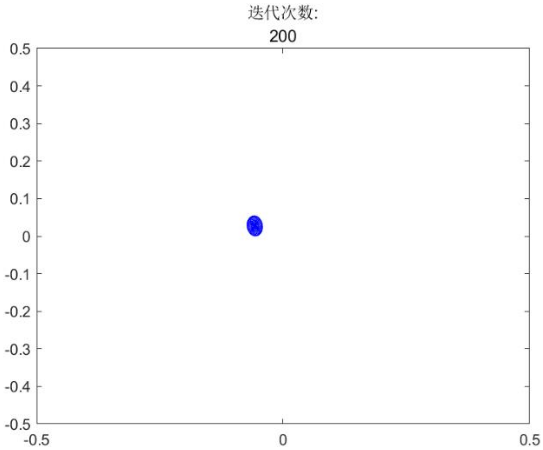
➢ 取 n=20，当迭代次数为 0，5，20，100，200 时，试验结果如下：











➢ 可以观察到数据点逐渐向"平均点"靠近

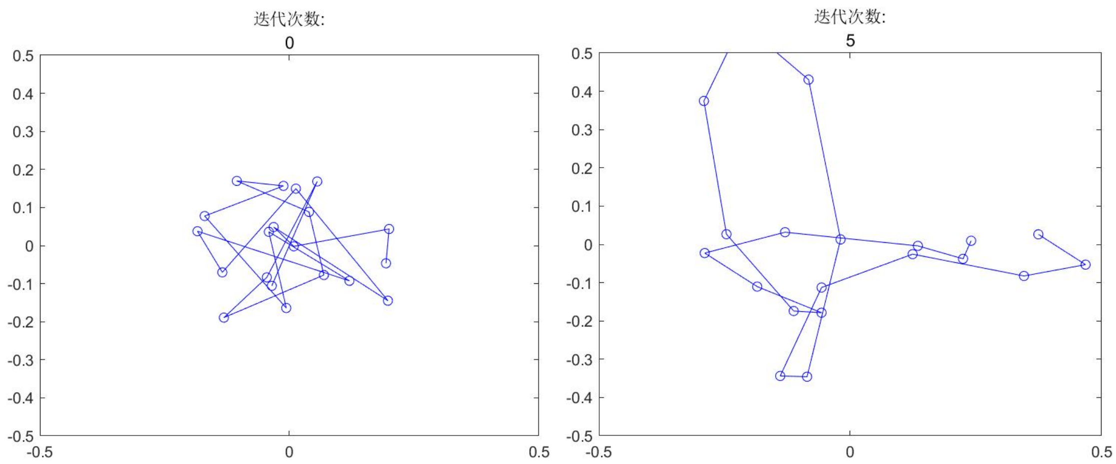
试验二（每轮迭代进行归一化，并且坐标平均值为 0）

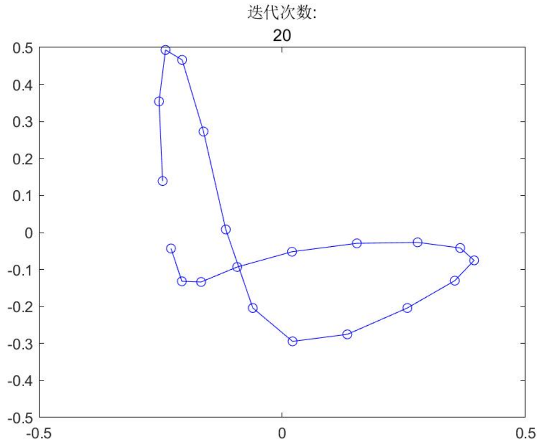
➢ Matlab 代码如下：

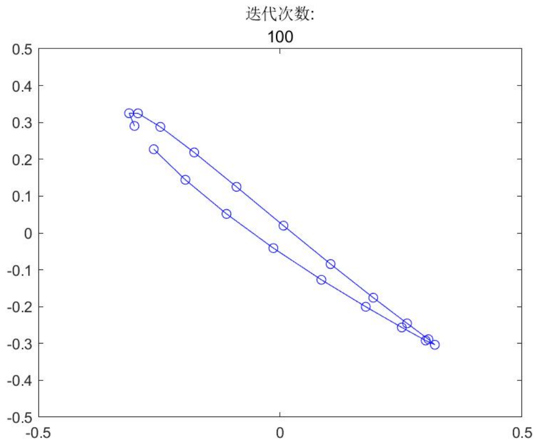
```
n = 20;
M=diag(ones(n,1))+diag(ones(n-1,1),1);
M(n,1)=1;
M=0.5*M;
```

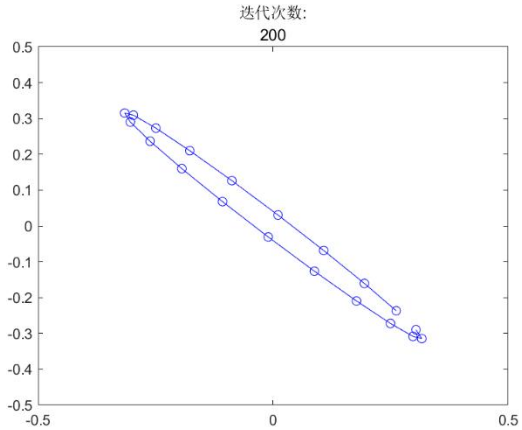
```
x = rand(n,1);
x = x/norm(x);
y = rand(n,1);
y = y/norm(y);
x=x-sum(x)/n;
y=y-sum(y)/n;
figure();%初始状态
plot(x,y,'b-o');
lim=0.5;
xlim([-lim,lim]);
ylim([-lim,lim]);
title(["迭代次数:",0]);
for k = 1:200
    x=M*x;
    x=x/norm(x);
    y=M*y;
    y=y/norm(y);
    if k==5 || k==20 || k==100 ||k==200
        figure();
        plot(x,y,'b-o')
        xlim([-lim,lim]);
        ylim([-lim,lim]);
        title(["迭代次数:",k]);
    end
end
```

➢ 试验结果如下









➢ 可以观察到，数据点逐渐趋向于分布在倾角为 45°的椭圆上

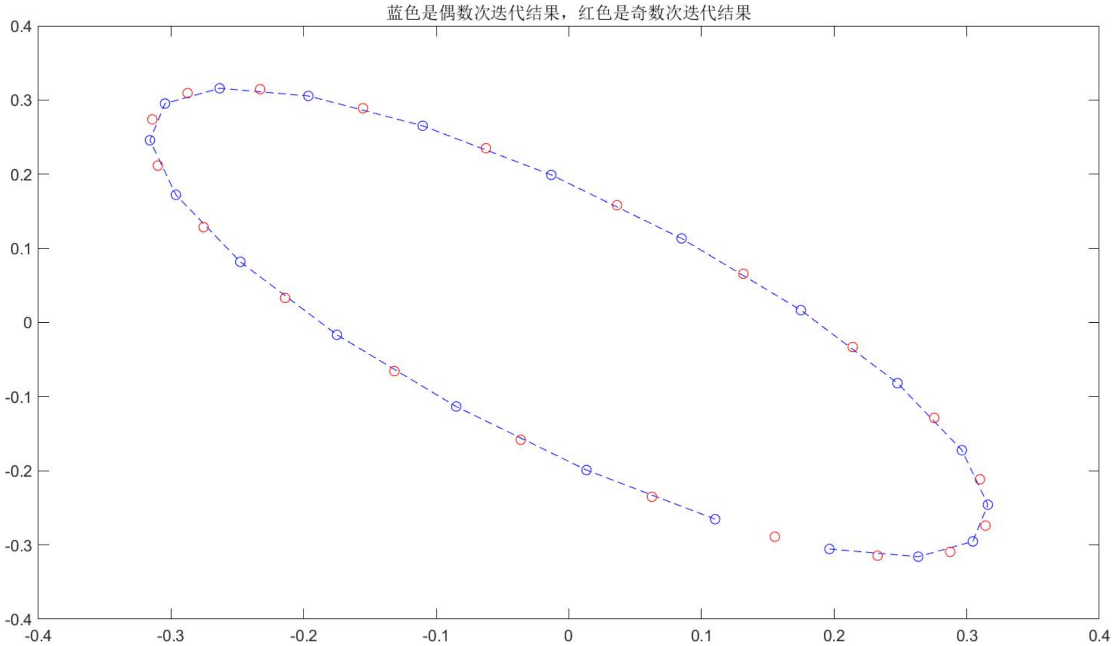
实验三（c 与 s 落在不变子空间中）

➢ Matlab 代码如下

```
n = 20;
M=diag(ones(n,1))+diag(ones(n-1,1),1);
M(n,1)=1;
M=0.5*M;
for j =1:n
    tao(j,1)=2*pi/n*(j-1);
end
c=sqrt(2/n)*cos(tao);
s=sqrt(2/n)*sin(tao);
theta1=10*rand();
theta2=10*rand();
x = cos(theta1)*c + sin(theta1)*s;
y = cos(theta2)*c + sin(theta2)*s;
figure();%初始状态
plot(x,y,'b--o');
lim=0.4;
xlim([-lim,lim]);
```

```
ylim([-lim,lim]);
hold on
for k = 1:10
    x=M*x;
    x=x/norm(x);
    y=M*y;
    y=y/norm(y);
    if mod(k,2)==0
        scatter(x,y,'b')
        hold on
    end
    if mod(k,2)==1
        scatter(x,y,'r')
        hold on
    end
end
title("蓝色是偶数次迭代结果，红色是奇数次迭代结果")
```

➢ 试验结果如下:



➢ 可以观察到，奇数次的迭代结果是相同的，偶数次的迭代结果也是相同的。这是因为 c 和 s 都落在了矩阵 M 的不变子空间中。

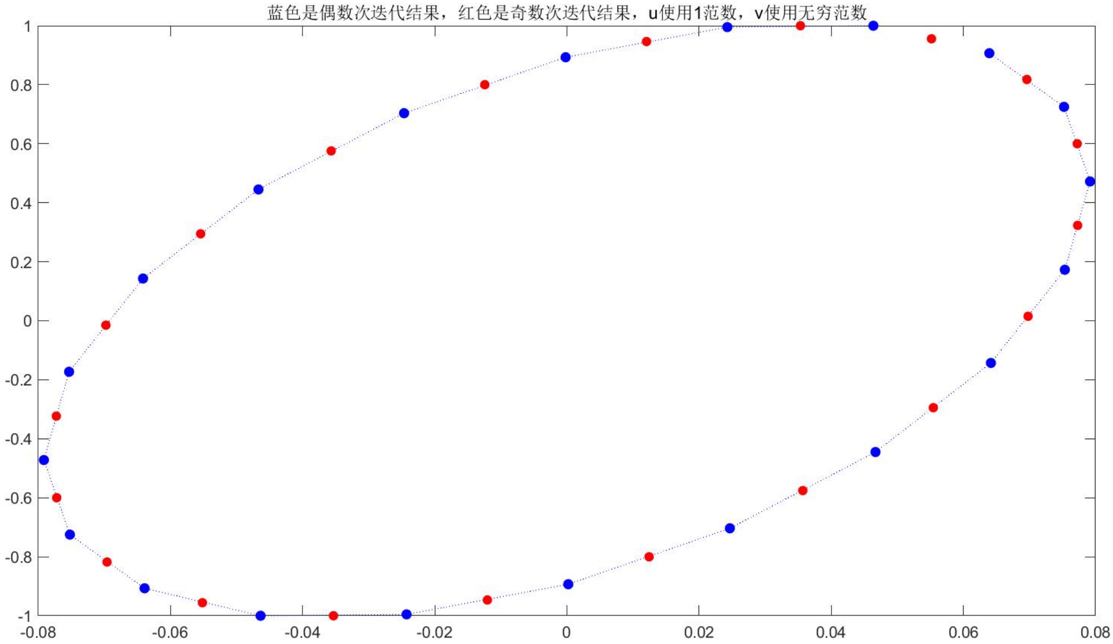
实验四（重复实验三，但使用其他范数）

- Matlab 代码如下

```
n = 20;
M=diag(ones(n,1))+diag(ones(n-1,1),1);
M(n,1)=1;
M=0.5*M;
for j =1:n
    tao(j,1)=2*pi/n*(j-1);
end
c=sqrt(2/n)*cos(tao);
s=sqrt(2/n)*sin(tao);
theta1=10*rand();
theta2=10*rand();
u = cos(theta1)*c + sin(theta1)*s;
v = cos(theta2)*c + sin(theta2)*s;
u = u/norm(u,1);
v = v/norm(v,"inf");

figure();%初始状态
plot(u,v,'b:o');
lim=0.4;
hold on
for k = 1:5
    u=M*u;
    v=M*v;
    u = u/norm(u,1);
    v = v/norm(v,"inf");
    if mod(k,2)==0
        scatter(u,v,'b','filled')
    end
    if mod(k,2)==1
        scatter(u,v,'r','filled')
    end
end
title("蓝色是偶数次迭代结果，红色是奇数次迭代结果，u 使用 1 范数，v 使用无穷范数")
```

- 试验结果如下



- 本实验中，向量 u 使用 1-norm 进行归一化，向量 v 使用∞-norm 进行归一化。
- 同样可以看到，数据点分布在椭圆上，并且奇数次迭代结果相同，偶数次迭代结果也相同。