粒子群：

#include"stdio.h"

#include"stdlib.h"

#include"time.h"

#include"math.h"

const int NUM=40;//粒子数

const int DIM=30;//维数

const double c1=1.8;//参数

const double c2=1.8;//参数

double xmin=-100.0;//位置下限

double xmax=100.0;//位置上限

double gbestx[DIM];//全局最优位置

double gbestf;//全局最优适应度

struct particle {//定义一个粒子

    double x[DIM];//当前位置矢量

    double bestx[DIM];//历史最优位置

    double f;//当前适应度

    double bestf;//历史最优适应度

}swarm[NUM];//定义粒子群

#define randf ((rand()%10000+rand()%10000\*10000)/100000000.0) //产生-1随机浮点数

double f1(double x[]) {//测试函数：超球函数

    float z=0;

    for(int i=0;i<DIM;i++)

        z+=(x[i])\*(x[i]);

    return z;

}

void main() {

    for(int i=0; i<DIM; i++)//初始化全局最优

        gbestx[i]=randf\*(xmax-xmin)+xmin;

    gbestf=100000000000000.0;

    for(int i=0; i<NUM; i++) {//初始化粒子群

        particle\* p1=&swarm[i];

        for(int j=0; j<DIM; j++)

            p1->x[j]=randf\*(xmax-xmin)+xmin;

        p1->f=f1(p1->x);

        p1->bestf=100000000000000.0;

    }

    for(int t=0; t<5000; t++) {

        for(int i=0; i<NUM; i++) {

            particle\* p1=&swarm[i];

            for(int j=0; j<DIM; j++)//进化方程

                p1->x[j]+=c1\*randf\*(p1->bestx[j]-p1->x[j])

                +c2\*randf\*(gbestx[j]-p1->x[j]);

            p1->f=f1(p1->x);

            if(p1->f<p1->bestf) {//改变历史最优

                for(int j=0;j<DIM;j++)

                    p1->bestx[j]=p1->x[j];

                p1->bestf=p1->f;

            }

            if(p1->f<gbestf) {//改变全局最优

                for(int j=0;j<DIM;j++)

                    gbestx[j]=p1->x[j];

                for(int j=0; j<DIM; j++)//把当前全局最优的粒子随机放到另一位置

                    p1->x[j]=randf\*(xmax-xmin)+xmin;

                gbestf=p1->f;

            }

        }

    }

    printf("%g\n", gbestf);

}

模拟退火

#include <cstdio>

#include <algorithm>

#include <cstring>

#include <cmath>

using namespace std;

const double eps = 1e-8;

const double r = 0.99; //降温速度

const int dx[] = { 0, 0, 1, -1, 1, -1, 1, -1 };

const int dy[] = { 1, -1, 0, 0, -1, 1, 1, -1 };

double a, b, c, d, e, f;

double dis(double x, double y, double z) {

return sqrt(x \* x + y \* y + z \* z);

}

//已知x,y,求z

double getz(double x, double y) {

double A = c, B = e \* x + d \* y,

C = a \* x \* x + b \* y \* y + f \* x \* y - 1;

double delta = B \* B - 4 \* A \* C;

if (delta < 0) return 1e60;

double z1 = (-B + sqrt(delta)) / 2 / A,

z2 = (-B - sqrt(delta)) / 2 / A;

if (z1 \* z1 < z2 \* z2) return z1;

else return z2;

}

double solve() {

//模拟退火

double step = 1; //步长

double x = 0, y = 0, z;

while (step > eps) {

z = getz(x, y);

for (int i = 0; i < 8; i++) {

double nx = x + dx[i] \* step,

ny = y + dy[i] \* step,

nz = getz(nx, ny);

if (nz > 1e30) continue;

if (dis(nx, ny, nz) < dis(x, y, z)) {

x = nx; y = ny; z = nz;

}

}

step \*= r;

}

return dis(x, y, z);

}

int main() {

while (scanf("%lf%lf%lf%lf%lf%lf", &a, &b, &c, &d, &e, &f) != EOF) {

printf("%.8f\n", solve());

}

return 0;

}