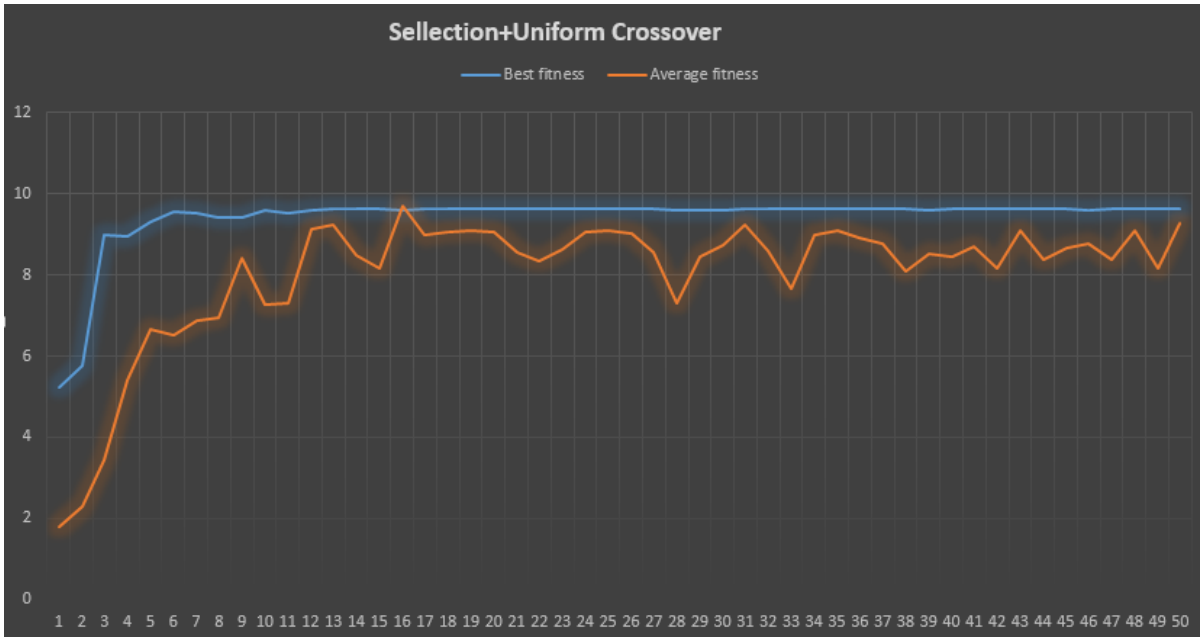
2. Maximization of 3-D Schefel function

Student: Uladzimier Shukailo

Brest 2016

Function: y=x1*sin(abs(x1))+x2*sin(abs(x2))

Sellection+Uniform Crossover:



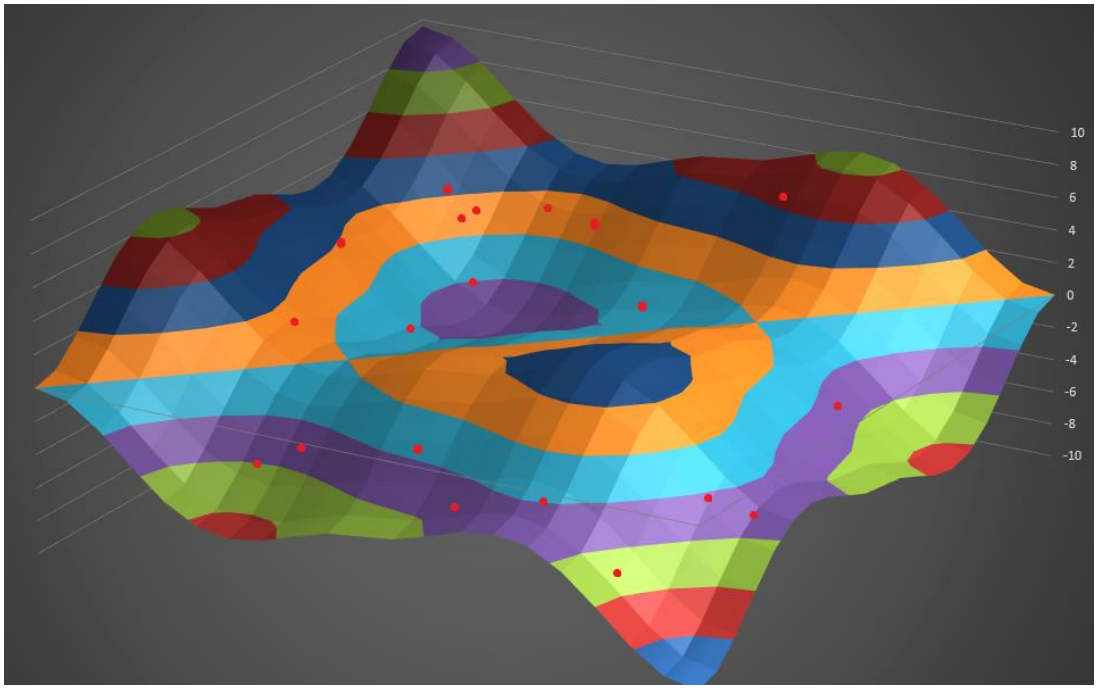
Listing:

```
#include "stdafx.h"
#include "time.h"
#include <iostream>
#include <fstream>

using namespace std;

double generation_x1[20][20];
double generation_x2[20][20];
double fitness[20];
double prev_max;
int gen=0;
int q=0;

ofstream file1;
ofstream file2;
ofstream file3;

void fit()
{
	for(int i=0;i<20;i++)
	{
		fitness[i]=20;
		for(int j=0;j<20;j++)

	fitness[i]+=(generation_x1[i][j]*sin(generation_x1[i][j])+generation_x2[i][j]*sin(
generation_x2[i][j]));
	}
```

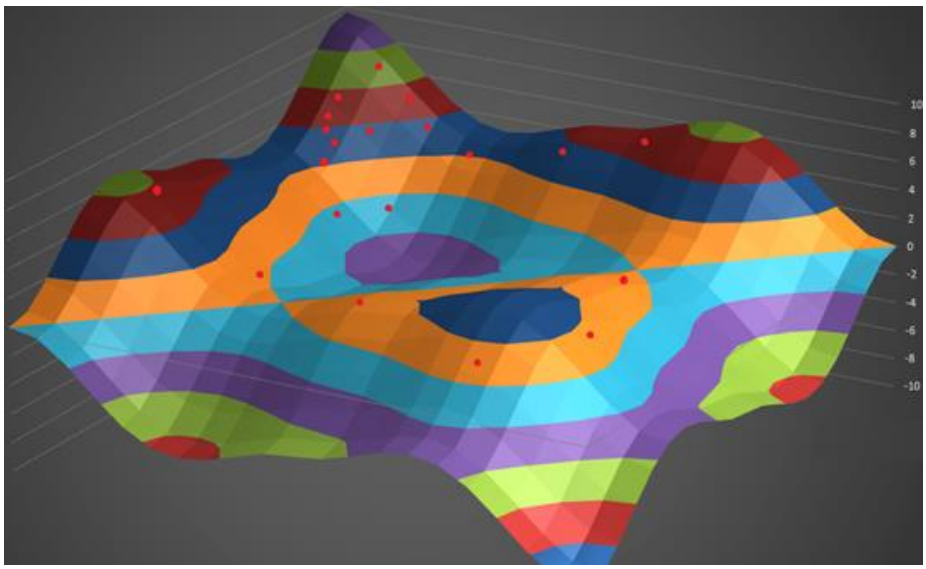
```
}
void cross()
{
	srand(time(NULL));
	double new_gener[20][20];
	double new_geners[20][20];
	for(int i=0;i<10;i++)
	{
		int m=rand()%10;
		int p=rand()%10;
		int cros_gen=rand()%18+1;
		for(int k=0;k<cros_gen;k++){
			new_gener[i*2][k]=generation_x1[m][k];
			new_geners[i*2][k]=generation_x2[m][k];
		}
		for(int j=cros_gen;j<20;j++){
			new_gener[i*2][j]=generation_x1[p][j];
			new_geners[i*2][j]=generation_x2[p][j];
		}
		for(int k=0;k<cros_gen;k++){
			new_gener[i*2+1][k]=generation_x1[p][k];
			new_geners[i*2+1][k]=generation_x2[p][k];
		}
		for(int j=cros_gen;j<20;j++){
			new_gener[i*2+1][j]=generation_x1[m][j];
			new_geners[i*2+1][j]=generation_x2[m][j];
		}
	}
	memcpy(generation_x1,new_gener,20*20*sizeof(double));
	memcpy(generation_x2,new_geners,20*20*sizeof(double));
}
void stats()
{
	cout<<"max: "<<fitness[0]<<endl;
	file1<<fitness[0]<<endl;
	double average=0;
	for(int i=0;i<20;i++)
		average+=fitness[i];
	average/=20;
	cout<<"aver: "<<average<<endl;
	file2<<gen<<endl;
	file3<<average<<endl;
	cout<<"generation: "<<++gen<<endl;
	cout<<"_________________"<<endl;
	if(prev_min==fitness[0])
		q++;
	else
	{
		prev_min=fitness[0];
		q=0;
	}
}
void mutation()
{
	srand(time(NULL));
	for(int i=0;i<20;i++)
		for(int j=0;j<20;j++)
			if(rand()%20==0)
			{
				generation_x1[i][j]=(double)(rand()%101)/100;
				generation_x2[i][j]=(double)(rand()%101)/100;
				if(rand()%2)
					generation_x1[i][j]=-generation_x1[i][j];
					generation_x2[i][j]=-generation_x2[i][j];
```

```
                    }
}
int _tmain(int argc, _TCHAR* argv[])
{
        srand(time(NULL));
        file1.open("max.txt",ios_base::trunc);
        file2.open("av.txt",ios_base::trunc);
        file3.open("avtrue.txt",ios_base::trunc);
        for(int i=0;i<20;i++)
        {
                for(int j=0;j<20;j++)
                {
                        generation_x1[i][j]=(double)(rand()%101)/100;
                        if(rand()%2)
                                generation_x1[i][j]=-generation_x1[i][j];
                        generation_x2[i][j]=(double)(rand()%101)/100;
                        if(rand()%2)
                                generation_x2[i][j]=-generation_x2[i][j];
                }
        }
        for(int i=0;i<20000;i++)
        {
                fit();
                sort();
                stats();
                cross();
                mutation();
        }
        file1.close();
        file2.close();
        file3.close();
        return 0;
}
```

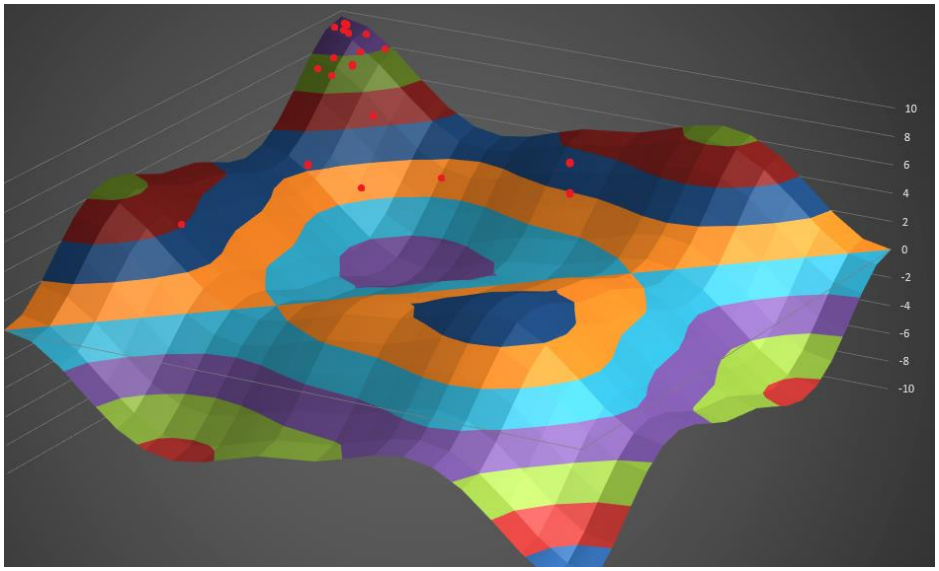
1 generetion:



100 generetion:



500 generetion:



5000 generetion: