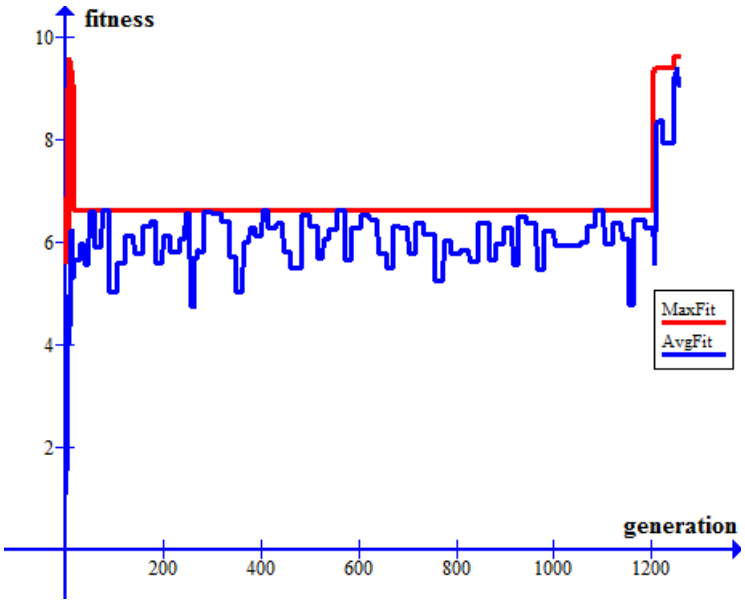
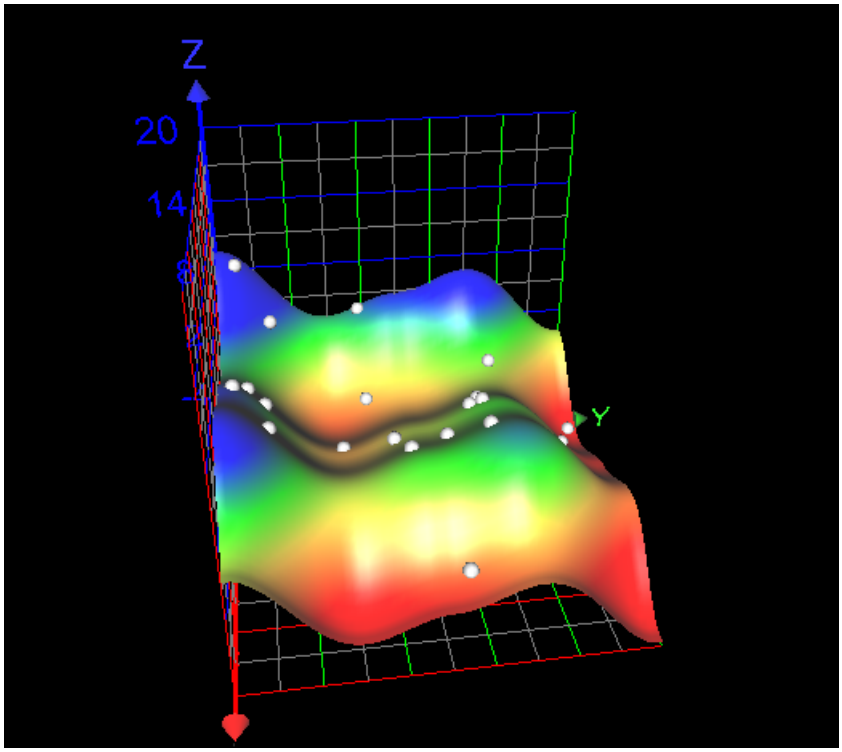
3-D version of Schwefel's function Exercise

1. Maximize y = x1 sin(|x1|) + x2 sin(|x2|) in the following way!
    (1) Represent value of x by a 10-bit binary chromosome.
    (2) Create a population of 20 chromosomes at random, with fitness being y.
    (3) Evolve this population till fitness dosen't change.
2. Show
    (1) the graph of fitness vs generation.
    (2) all 20 points (x, y) in the 1st, an intermediate, and final generation.

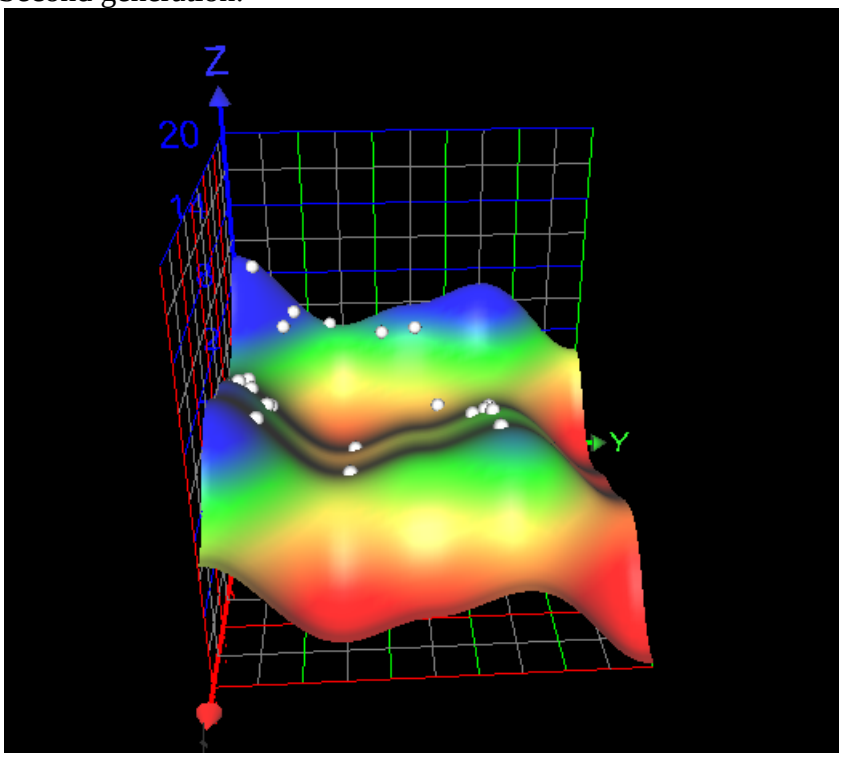
Fitness vs generation:



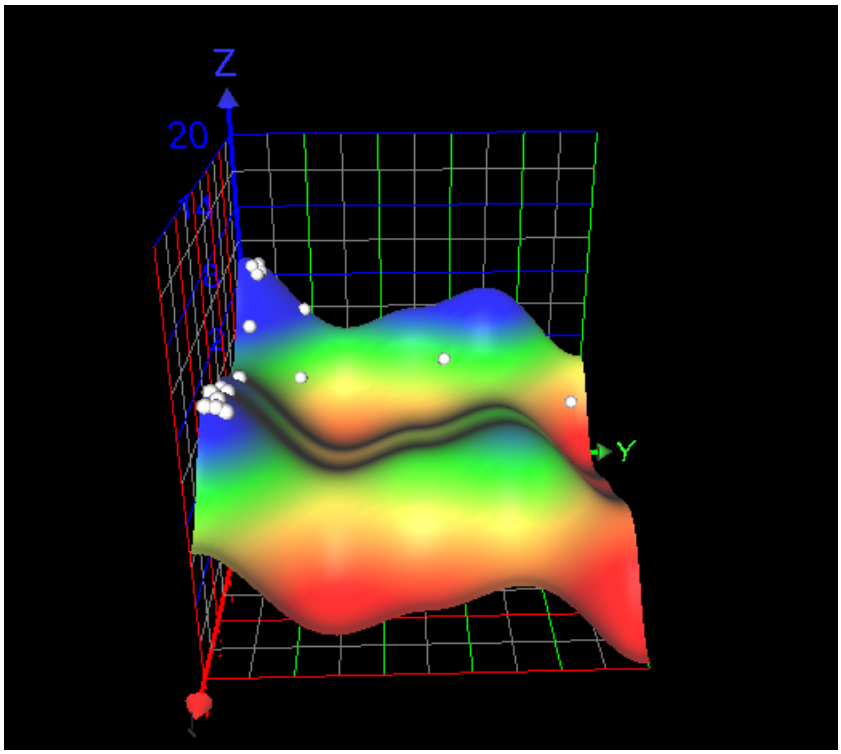
First generation:



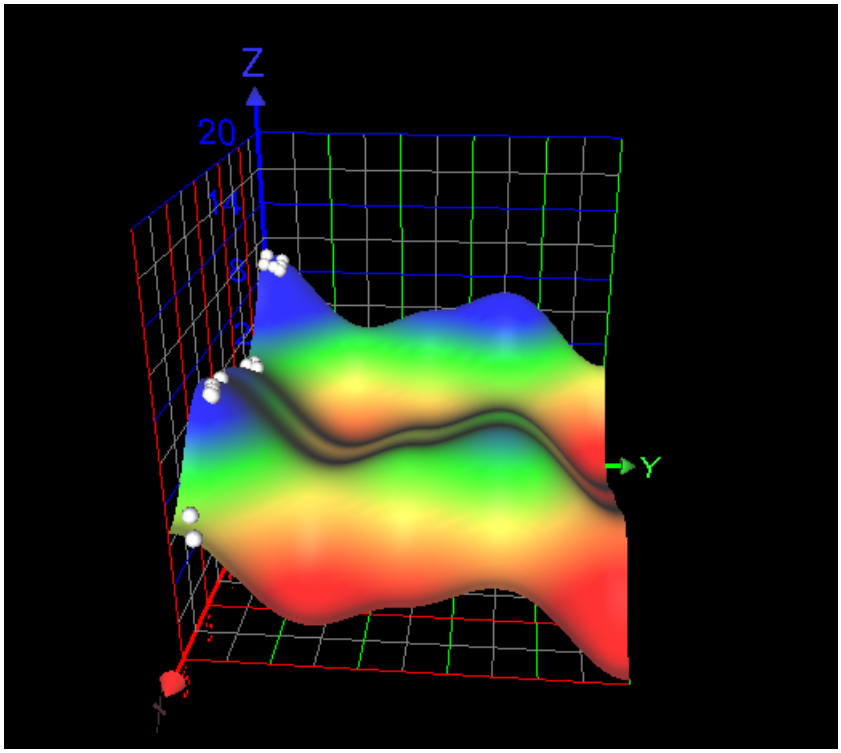
Second generation:



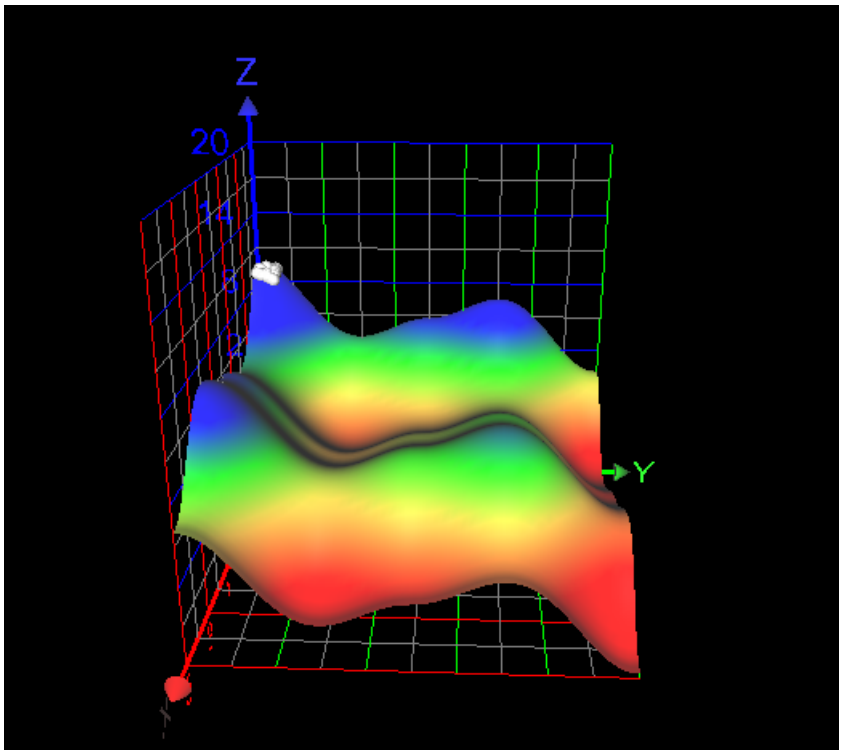
Generation #5:



Generation #7:



Last generation:



Source code

generation.cs:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
```

```
using System.Threading.Tasks;
using System.IO;

namespace siit_2
{
    class generation
    {
        public static int numOfGens = 22;
        public static int numOfChromo = 20;
        List<int[]> gens;
        List<double> fitness { get; }
        List<float> probability { get; }
        List<double> chromSelect;


        public double averagefitness = 0f;
        Random mutat = new Random();
        int rando = 0;

        public generation()
        {
            gens = new List<int[]>();
            fitness = new List<double>();
            probability = new List<float>();
            chromSelect = new List<double>();



            for (int j = 0; j < numOfChromo; j++)
            {
              int[] gen = new int[numOfGens];

                gens.Add(gen);
                fitness.Add(0);
                probability.Add(0f);
                chromSelect.Add(0);
            }
        }

        public generation(List<int[]> new_gens)
        {
            gens = new List<int[]>();
            fitness = new List<double>();
            probability = new List<float>();
            chromSelect = new List<double>();

            gens = new_gens;
            for (int j = 0; j < numOfChromo; j++)
            {

                fitness.Add(0);
                probability.Add(0f);
```

```
            chromSelect.Add(0);
        }
    }

    public void randomize()
    {
        int tmp = -1;
        Random rand = new Random();
        for (int i = 0; i < numOfChromo; i++)
        {
            for (int j = 0; j < numOfGens; j++)
            {

                gens[i][j] = rand.Next() % 2;
            }
        }
    }
    public void setFitness(StreamWriter X1,StreamWriter X2,StreamWriter fit)
    {
        for (int i = 0; i < numOfChromo; i++)
        {
            double sum = 0;
            List<double> x = new List<double>();
            for (int z = 0; z < 2; z++)
            {
                bool minus = false;
                string x1 = "";
                for (int j = z*11; j < numOfGens/2+z*11; j++)
                {
                    if (j == 0 || j == 11)
                    {
                        if (gens[i][j] == 0) minus = true;
                    }
                    else x1 = x1 + gens[i][j].ToString();
                }
                x.Add(((double)Convert.ToInt32(x1, 2)*5)/1023);
                if (minus) x[z] *= -1;
                if (z == 0)
                    X1.WriteLine(x[z]);
                else X2.WriteLine(x[z]);
            }
            for (int j = 0; j < 2; j++)
            {
                sum += x[j] * Math.Sin(Math.Abs(x[j]));


            }
            fitness[i] = sum;
            fit.WriteLine(fitness[i]);
        }
        fit.WriteLine();
        X1.WriteLine();
```

```
    X2.WriteLine();


}
public void setProbability()
{
    double mass = 0; ;
    for (int i = 0; i < numOfChromo; i++)
    {
        mass += fitness[i];
    }
    averagefitness = mass / numOfChromo;
    for (int i = 0; i < numOfChromo; i++)
    {
        probability[i] = (float)fitness[i] / (float)mass;
    }
}
public int[] newChild()
{

    Random rand = new Random(DateTime.Now.TimeOfDay.Milliseconds + rando);
    rando++;
    if (rando == 10000000) rando = 0;
    int rand_num = rand.Next(numOfChromo/2);
    float sum = 0f;
    int[] chrom_1 = new int[numOfGens], chrom_2 = new int[numOfGens];

    //for (int i = 0; i < 20; i++)
    //{
    //    sum += probability[i] * 1000000000;
    //    if (rand_num <= sum)
    //    {
    //        chromSelect[i]++;
    //        chrom_1 = gens[i];
    //        break;
    //    }

    //}
    chrom_1 = gens[rand_num];                    // for truncate
    sum = 0f;
    rand_num = rand.Next(numOfChromo/2);
    //for (int i = 0; i < 20; i++)
    //{
    //    sum += probability[i] * 1000000000;
    //    if (rand_num <= sum)
    //    {
    //        chromSelect[i]++;
    //        chrom_2 = gens[i];
    //        break;
    //    }
    //}
    chrom_2 = gens[rand_num];                     // for truncate
```

```
    int[] new_chrom = new int[numOfGens];

    //unified crossover
    for (int i = 0; i < numOfGens; i++)
    {
        if (rand.Next() % 2 == 1) new_chrom[i] = chrom_1[i];
        else new_chrom[i] = chrom_2[i];
    }

    //one point crossover
    //int point = rand.Next() % 1000;
    //for (int i = 0; i < 1000; i++)
    //{
    //    if (i < point) new_chrom[i] = chrom_1[i];
    //    else new_chrom[i] = chrom_2[i];
    //}
    Mutation(new_chrom);
    return new_chrom;
}
public double bestFitness()
{
    return fitness.Max();
}

public void Sort()
{
    for (int i = 0; i < numOfChromo - 1; i++)
    {
        bool swapped = false;
        for (int j = 0; j < numOfChromo - i - 1; j++)
        {
            if (fitness[j] < fitness[j + 1])
            {
                int[] tmp_gen = gens[j];
                gens[j] = gens[j + 1];
                gens[j + 1] = tmp_gen;

                double tmp_fit = fitness[j];
                fitness[j] = fitness[j + 1];
                fitness[j + 1] = tmp_fit;
                swapped = true;
            }

        }
        if (!swapped) break;
    }
}
public double getAverageFit()
{
```

```
            return averagefitness;
        }
        public void WriteTable(StreamWriter file1, StreamWriter file2)
        {
            for (int i = 0; i < numOfChromo; i++)
            {
                file1.WriteLine(chromSelect[i].ToString());
                file2.WriteLine(i.ToString());
            }
            file1.WriteLine();
            file1.WriteLine();
        }
        public int[] GetMaxChromo()
        {
            return gens[0];
        }

        private void Mutation(int[] chromo)
        {
            for (int i = 0; i < numOfGens; i++)
            {
                if (mutat.Next() % 50 == 5)
                {
                    int tmp = -1;
                    if (chromo[i] == 1) chromo[i] = 0;
                    else chromo[i] = 1;
                }
            }
        }
    }
}
```

main.cs:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.IO;

namespace siit_2
{
    class Program
    {
        static void Main(string[] args)
        {
            StreamWriter avgFitFile = new StreamWriter("averageFit.txt");
            StreamWriter maxFitFile = new StreamWriter("maxFit.txt");
            StreamWriter numGenFile = new StreamWriter("numGen.txt");
            StreamWriter tableFile = new StreamWriter("Table.txt");
```

```
        StreamWriter tablenum = new StreamWriter("Num.txt");
        StreamWriter X1 = new StreamWriter("X1.txt");
        StreamWriter X2 = new StreamWriter("X2.txt");
        StreamWriter fit = new StreamWriter("fit.txt");
        generation old_gens = new generation();
        old_gens.randomize();
        old_gens.setFitness(X1,X2,fit);
        old_gens.setProbability();
        double maxFit = 0;
        int numGeneration = 0;
        for (int j = 0; j < 100; numGeneration++)
        {
           numGenFile.WriteLine(numGeneration.ToString());
           Console.WriteLine(old_gens.bestFitness() + " " + old_gens.getAverageFit());
           //if (old_gens.bestFitness() == 0) break;
           List<int[]> new_tmp = new List<int[]>();
           old_gens.Sort();                                    //for truncate
           for (int i = 0; i < generation.numOfChromo; i++)
           {
              new_tmp.Add(old_gens.newChild());
           }
           old_gens.WriteTable(tableFile, tablenum);
           generation new_gens = new generation(new_tmp);
           old_gens = new_gens;
           old_gens.setFitness(X1,X2,fit);
           old_gens.setProbability();
           avgFitFile.WriteLine(old_gens.getAverageFit().ToString());
           maxFitFile.WriteLine(old_gens.bestFitness().ToString());
           if (old_gens.bestFitness() > maxFit)
           {
              maxFit = old_gens.bestFitness();
              j = 0;
           }
           if (old_gens.bestFitness() == maxFit) j++;

        }

        tablenum.Close();
        tableFile.Close();
        numGenFile.Close();
        avgFitFile.Close();
        maxFitFile.Close();
        fit.Close();
        X1.Close();
        X2.Close();
     }
  }
}
```