Код Xaml

<Window

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:local="clr-namespace:NelderMid"

xmlns:chartingToolkit="clr-namespace:System.Windows.Controls.DataVisualization.Charting;assembly=System.Windows.Controls.DataVisualization.Toolkit" x:Class="NelderMid.MainWindow"

mc:Ignorable="d"

Title="МетодНелдера-Мида" Height="441.3" Width="1040.047">

<Grid>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="177\*"/>

<ColumnDefinition Width="856\*"/>

</Grid.ColumnDefinitions>

<GroupBox x:Name="groupBox" Header="Решение"HorizontalAlignment="Left" Margin="10,10,0,0"VerticalAlignment="Top" Height="382" Width="465"Grid.ColumnSpan="2">

<Grid Margin="-8,0,8,0">

<Label x:Name="label" Content="Функция"HorizontalAlignment="Left" Margin="173,0,0,0"VerticalAlignment="Top"FontFamily="Georgia"FontSize="16"FontStyle="Italic"/>

<Label x:Name="label\_Copy" Content="(x1-10)^2+(x2-12)^2+(x1-10)(x2-12)"HorizontalAlignment="Left" Margin="17,28,0,0"VerticalAlignment="Top"FontFamily="Georgia"FontSize="24"RenderTransformOrigin="0.474,1.893" Width="412" Height="47"/>

<Label x:Name="label\_Copy1" Content="alpha=1 beta=0.5 gamma=2 "HorizontalAlignment="Left" Margin="95,69,0,0"VerticalAlignment="Top"FontFamily="Georgia"FontSize="16"RenderTransformOrigin="0.474,1.893"/>

<TextBox x:Name="t2"HorizontalAlignment="Left" Height="228" Margin="10,124,0,-20"TextWrapping="Wrap"VerticalAlignment="Top" Width="120"VerticalScrollBarVisibility="Auto"/>

<Label x:Name="label1" Content="Точки"HorizontalAlignment="Left" Margin="43,97,0,0"VerticalAlignment="Top"FontFamily="Georgia"FontSize="14.667"/>

<Label x:Name="label1\_Copy" Content="Ходрешения"HorizontalAlignment="Left" Margin="248,97,0,0"VerticalAlignment="Top"FontFamily="Georgia"FontSize="14.667"/>

<TextBox x:Name="t"HorizontalAlignment="Left" Height="228" Margin="147,124,0,0"TextWrapping="Wrap"VerticalAlignment="Top" Width="296"VerticalScrollBarVisibility="Auto"/>

<TextBox x:Name="n1"HorizontalAlignment="Left" Height="21" Margin="62,37,0,0"TextWrapping="Wrap"VerticalAlignment="Top" Width="25"/>

<TextBox x:Name="n2"HorizontalAlignment="Left" Height="21" Margin="183,37,0,0"TextWrapping="Wrap"VerticalAlignment="Top" Width="26"/>

<TextBox x:Name="n3"HorizontalAlignment="Left" Height="22" Margin="303,37,0,0"TextWrapping="Wrap"VerticalAlignment="Top" Width="25"/>

<TextBox x:Name="n4"HorizontalAlignment="Left" Height="22" Margin="381,37,0,0"TextWrapping="Wrap"VerticalAlignment="Top" Width="23"/>

</Grid>

</GroupBox>

<GroupBox x:Name="groupBox1" Header="График" Margin="303,10,18,0"VerticalAlignment="Top" Height="382"Grid.Column="1">

<Grid>

<Button x:Name="button" Content="Найтирешение"HorizontalAlignment="Left" Margin="194,14,0,0"VerticalAlignment="Top" Width="133" Click="button\_Click"/>

<chartingToolkit:Chart x:Name="chart1"HorizontalAlignment="Left" Margin="10,51,0,0" Title="МетодНелдера-Мида"VerticalAlignment="Top" Height="299" Width="501">

<chartingToolkit:Chart.Axes>

<chartingToolkit:LinearAxis Name="axX"SeriesHost="{x:Null}" Orientation="X"/>

<chartingToolkit:LinearAxis Name="axY"SeriesHost="{x:Null}" Orientation="Y"/>

</chartingToolkit:Chart.Axes>

<chartingToolkit:Chart.DataContext>

<PointCollection>1,10 2,20 3,30 4,40</PointCollection>

</chartingToolkit:Chart.DataContext>

</chartingToolkit:Chart>

</Grid>

</GroupBox>

</Grid>

</Window>

Кодпрограммы

using System;

usingSystem.Collections.Generic;

usingSystem.Linq;

usingSystem.Text;

usingSystem.Threading.Tasks;

usingSystem.Windows;

usingSystem.Windows.Controls;

usingSystem.Windows.Controls.DataVisualization.Charting;

usingSystem.Windows.Data;

usingSystem.Windows.Documents;

usingSystem.Windows.Input;

usingSystem.Windows.Media;

usingSystem.Windows.Media.Imaging;

usingSystem.Windows.Navigation;

usingSystem.Windows.Shapes;

namespaceNelderMid

{

/// <summary>

/// ЛогикавзаимодействиядляMainWindow.xaml

/// </summary>

public partial class MainWindow : Window

{

publicMainWindow()

{

InitializeComponent();

}

double al = 1, b = 0.5, g = 2;

const double eps=1,T=6;

constint n = 2,m=n+1;

double[,] a = new double[m, n];

double[] F = new double[m];

double center;

double[,] C = new double[1, n];

double[,] R = new double[1, n];

double[,] E = new double[1, n];

double[,] S = new double[1, n];

voidinit()

{

for (int i = 0; i < m; i++)

{

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

{

if (i == 0) a[i, j] = 0;

else if (i == j + 1)

a[i, j] = (T / (n \* Math.Sqrt(2))) \* (Math.Sqrt(m) - 1 + n);

else

a[i, j] = (T / (n \* Math.Sqrt(2))) \* (Math.Sqrt(m) - 1);

}

}

}

Random randomGen = new Random();

voidgetGraph(double[,] ma, string nazv)

{

t2.Text += "Шаг "+num+" : \n";

Color gColor = gColor = System.Windows.Media.Color.FromRgb(Convert.ToByte(randomGen.Next(255)), Convert.ToByte(randomGen.Next(255)), Convert.ToByte(randomGen.Next(255)));

Brush b = new SolidColorBrush(gColor);

int j = 0;

while (j < m)

{

if (j > 0) nazv = "";

List<KeyValuePair<double, double>>MyValue = new List<KeyValuePair<double, double>>();

MyValue.Add(new KeyValuePair<double, double>(ma[j, 0], ma[j, 1]));

j++;

if (j == m)

MyValue.Add(new KeyValuePair<double, double>(ma[0, 0], ma[0, 1]));

else

MyValue.Add(new KeyValuePair<double, double>(ma[j, 0], ma[j, 1]));

showChart(MyValue, nazv, b);

}

for (int i = 0; i < m; i++)

{

t2.Text += Math.Round(a[i, 0], 2) + " \t" + Math.Round(a[i, 1], 2)+"\n";

}

num++;

t2.Text += "\n";

}

doublefunc(double[,] mas, int i)

{

double res = Math.Pow(mas[i, 0] - Convert.ToInt32(n1.Text), 2) + Math.Pow(mas[i, 1] - Convert.ToInt32(n2.Text), 2) + (mas[i, 0] - Convert.ToInt32(n3.Text)) \* (mas[i, 1] - Convert.ToInt32(n4.Text));

return res;

}

public void replace(int H)

{

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

{

a[H, j] = R[0, j];

}

t.Text += "отражение\n ";

}

private void showChart(List<KeyValuePair<double, double>>valueList, string title,BrushgColor)

{

System.Windows.Controls.DataVisualization.Charting.LineSeries lineSeries1 = new System.Windows.Controls.DataVisualization.Charting.LineSeries();

lineSeries1.Title = title;

lineSeries1.DependentValuePath = "Value";

lineSeries1.IndependentValuePath = "Key";

Style style = new Style(typeof(LineDataPoint));

style.Setters.Add(new Setter(LineDataPoint.BackgroundProperty, gColor));

if (title == "")

style.Setters.Add(new Setter(LineDataPoint.VisibilityProperty, Visibility.Hidden));

lineSeries1.DataPointStyle = style;

lineSeries1.ItemsSource = valueList;

chart1.Series.Add(lineSeries1);

}

intnum = 1;

private void button\_Click(object sender, RoutedEventArgs e)

{

num = 1;

init();

t.Clear();

t2.Clear();

double z = 1000;

chart1.Series.Clear();

getGraph(a, "Начальный многогранник");

while (z >eps)

{

//значения функции в точках

t.Text += "Значения функции в точках : \n";

for (int i = 0; i < m; i++)

{

F[i] = func(a, i);

t.Text += "F[" + (i + 1) + "]=" + Math.Round(F[i], 2) + " ";

}

t.Text += "\n";

//поиск макс и мин

t.Text += "Поиск макс и мин : \n";

int H = 0, L = 0, G = 0;

for (int i = 1; i < m; i++)

{

if (F[i] > F[H])

{

G = H;

H = i;

}

else if (F[i] < F[L]) L = i;

}

t.Text += "max = " + Math.Round(F[H], 2) + " H=" + (H + 1) + " min = " + Math.Round(F[L], 2) + " L=" + (L + 1) + "\n";

t.Text += "Центр тяжести : \n";

//центртяжести

double sum;

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

{

sum = 0;

for (int i = 0; i < m; i++) sum += a[i, j];

C[0, j] = (1.0 / n) \* (sum - a[H, j]);

t.Text += "C[" + (j + 1) + "]=" + Math.Round(C[0, j], 2) + " ";

}

center = func(C, 0);

t.Text += "Центр " + Math.Round(center, 2) + "\n";

//проверканасходимость

sum = 0;

for (int i = 0; i < m; i++)

{

sum += Math.Pow(F[i] - center, 2);

}

z = Math.Sqrt((1.0 / m) \* sum);

t.Text += "Cходимость : " + Math.Round(z, 2) + "\n";

if (z >eps)

{

//отражение вершины через центр тяжести

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

{

R[0, j] = C[0, j] + al \* (C[0, j] - a[H, j]);

t.Text += "R[" + (j + 1) + "]=" + Math.Round(R[0, j], 2) + " ";

}

doublefr = func(R, 0);

t.Text += "Отражениеfr = " + Math.Round(fr, 2) + "\n";

//проверка отражения

if (fr< F[L])

{

//растяжение

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

{

E[0, j] = C[0, j] + g \* (R[0, j] - C[0, j]);

t.Text += "E[" + (j + 1) + "]=" + Math.Round(E[0, j], 2) + " ";

}

doublefe = func(E, 0);

t.Text += "Растяжениеfe = " + Math.Round(fe, 2) + "\n";

//проверка растяжения

if (fe<fr)

{

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

{

a[H, j] = E[0, j];

}

t.Text += "Растяжениеэффективно\n";

getGraph(a, "Растяжение");

}

else

{

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

{

a[H, j] = R[0, j];

}

t.Text += "Отражениеэффективно\n";

getGraph(a, "Отражение");

}

}

else

{

bool flag = true;

if (fr< F[G])

{

//замена на отражение

replace(H);

flag = false;

}

else if (fr< F[H])

{

replace(H);

flag = true;

}

//сжатие

if (flag)

{

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

{

S[0, j] = C[0, j] + b \* (a[H, j] - C[0, j]);

t.Text += "S[" + (j + 1) + "]=" + Math.Round(S[0, j], 2) + " ";

}

doublefs = func(S, 0);

t.Text += "Сжатиеfs = " + Math.Round(fs, 2) + " \n";

//проверка сжатия

if (fs< F[H])

{

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

{

a[H, j] = S[0, j];

}

t.Text += "Сжатиеэффективно\n";

getGraph(a, "Сжатие");

}

else

{

//сжатие симплекса для всех точек

for (int i = 0; i < m; i++)

{

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

if (j != L) a[i, j] = a[L, j] + 0.5 \* (a[i, j] - a[L, j]);

}

t.Text += "Сжатие симплекса\n";

getGraph(a, "Сжатие симплекса");

}

}

}

}

t.Text += "\n";

}

}

}

}