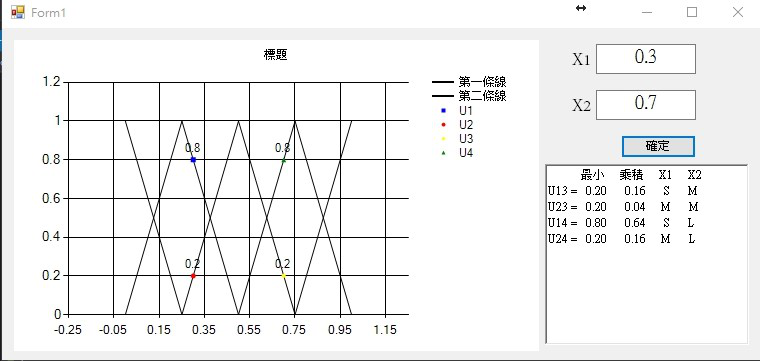
房志剛-1103105345-W02-20170330

結果



public partial class Form1 : Form

{

Double[] Y1 = { 1, 0, 1, 0, 1 };

Double[] Y2 = { 0, 1, 0, 1, 0 };

Double[] X = { 0, 0.25, 0.5, 0.75, 1 };

Series U1 = new Series("U1", 1);

Series U2 = new Series("U2", 1);

Series U3 = new Series("U3", 1);

Series U4 = new Series("U4", 1);

public Form1()

{

InitializeComponent();

}

private void Form1\_Load(object sender, EventArgs e)

{

//標題 最大數值

Series series1 = new Series("第一條線", 1);

Series series2 = new Series("第二條線", 1);

//設定線條顏色

series1.Color = Color.Black;

series2.Color = Color.Black;

U1.Color = Color.Blue;

U2.Color = Color.Red;

U3.Color = Color.Yellow;

U4.Color = Color.Green;

//折線圖

series1.ChartType = SeriesChartType.Line;

series2.ChartType = SeriesChartType.Line;

U1.ChartType = SeriesChartType.Point;

U2.ChartType = SeriesChartType.Point;

U3.ChartType = SeriesChartType.Point;

U4.ChartType = SeriesChartType.Point;

//將數值顯示在線上

U1.IsValueShownAsLabel = true;

U2.IsValueShownAsLabel = true;

U3.IsValueShownAsLabel = true;

U4.IsValueShownAsLabel = true;

Double x = 0;

//將數值新增至序列

for (int index = 0; index < Y1.Length; index++)

{

series1.Points.AddXY(X[index], Y1[index]);

series2.Points.AddXY(X[index], Y2[index]);

x += 0.25;

}

//將序列新增到圖上

this.chart1.Series.Add(series1);

this.chart1.Series.Add(series2);

this.chart1.Series.Add(U1);

this.chart1.Series.Add(U2);

this.chart1.Series.Add(U3);

this.chart1.Series.Add(U4);

//標題

this.chart1.Titles.Add("標題");

}

private void Button\_confirm\_Click(object sender, EventArgs e)

{

if (CheckRange())

{

CalculateMF();

}

else

{

MessageBox.Show("數值範圍錯誤", "錯誤");

}

}

private void CalculateMF()

{

U1.Points.Clear();

U2.Points.Clear();

U3.Points.Clear();

U4.Points.Clear();

Double X1 = Convert.ToDouble(textBox\_X1.Text);

Double X2 = Convert.ToDouble(textBox\_X2.Text);

int zone\_x1 = -1, zone\_x2 = -1;

Double u1, u2, u3, u4;

for (int i = 0; i < X.Length - 1; i++)

{

if (X1 >= X[i] && X1 <= X[i + 1]) zone\_x1 = i;

if (X2 >= X[i] && X2 <= X[i + 1]) zone\_x2 = i;

}

if (Y1[zone\_x1] > Y1[zone\_x1 + 1])

{

u1 = (X1 - X[zone\_x1]) / 0.25;

u2 = 1 - u1;

}

else

{

u1 = 1 - (X1 - X[zone\_x1]) / 0.25;

u2 = 1 - u1;

}

if (Y2[zone\_x2] > Y2[zone\_x2 + 1])

{

u3 = (X2 - X[zone\_x2]) / 0.25;

u4 = 1 - u3;

}

else

{

u3 = 1- (X2 - X[zone\_x2]) / 0.25;

u4 = 1 - u3;

}

U1.Points.AddXY(X1, u1);

U2.Points.AddXY(X1, u2);

U3.Points.AddXY(X2, u3);

U4.Points.AddXY(X2, u4);

richTextBox\_out.Text = " 最小 乘積 X1 X2" + Environment.NewLine;

richTextBox\_out.Text += "U13 = " + Math.Min(u1, u3).ToString("0.00") + " " + (u1 \* u3).ToString("0.00") + " " + RangCovert(zone\_x1 ) + " " + RangCovert(zone\_x2) + Environment.NewLine;

richTextBox\_out.Text += "U23 = " + Math.Min(u2, u3).ToString("0.00") + " " + (u2 \* u3).ToString("0.00") + " " + RangCovert(zone\_x1 + 1) + " " + RangCovert(zone\_x2 ) + Environment.NewLine;

richTextBox\_out.Text += "U14 = " + Math.Min(u1, u4).ToString("0.00") + " " + (u1 \* u4).ToString("0.00") + " " + RangCovert(zone\_x1 ) + " " + RangCovert(zone\_x2 + 1) + Environment.NewLine;

richTextBox\_out.Text += "U24 = " + Math.Min(u2, u4).ToString("0.00") + " " + (u2 \* u4).ToString("0.00") + " " + RangCovert(zone\_x1+1) + " " + RangCovert(zone\_x2 + 1) + Environment.NewLine;

//MessageBox.Show( "zone\_x1 : " + zone\_x1 + Environment.NewLine + "zone\_x2 : " + zone\_x2,"位置");

//MessageBox.Show("zone\_x1 : " + zone\_x1 + Environment.NewLine + "zone\_x2 : " + zone\_x2 + Environment.NewLine + "u1 : " + u1 + Environment.NewLine + "u2 : " + u2, "位置");

}

private bool CheckRange()

{

try

{

if (Convert.ToDouble(textBox\_X1.Text) < 0)

return false;

if (Convert.ToDouble(textBox\_X1.Text) > 1)

return false;

if (Convert.ToDouble(textBox\_X2.Text) < 0)

return false;

if (Convert.ToDouble(textBox\_X2.Text) > 1)

return false;

}

catch (Exception e)

{

return false;

}

return true;

}

private String RangCovert(int i)

{

switch (i)

{

case 0: return "Vs";

case 1: return "S";

case 2: return "M";

case 3: return "L";

case 4: return "VL";

default:return "";

}

}

}