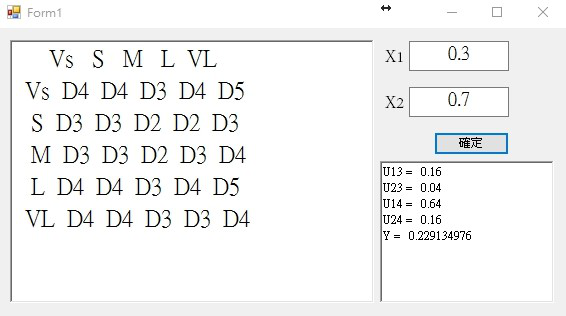
房志剛-1103105345-W03-20170408

結果



程式碼:

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

public MF[,] table = new MF[5, 5];

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

{

int N = 5;

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

{

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

{

table[i, j] = new MF();

}

}

string[] lines = System.IO.File.ReadAllLines("data.txt");

foreach (string line in lines)

{

string[] nums = line.Split(' ');

Node node = new Node(Convert.ToDouble(nums[1]), Convert.ToDouble(nums[3]), Convert.ToDouble(nums[5]));

table[node.i, node.j].y.Add(node.y);

}

richTextBox\_table\_out.AppendText(" ");

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

{

richTextBox\_table\_out.AppendText(String.Format("{0,4}", RankCovert(i)));

}

richTextBox\_table\_out.AppendText("\n");

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

{

richTextBox\_table\_out.AppendText(String.Format("{0,4}", RankCovert(i)));

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

{

String str = ("D" + Rank\_D(table[i, j].y.Average()));

richTextBox\_table\_out.AppendText(String.Format("{0,4}", str));

}

richTextBox\_table\_out.AppendText("\n");

}

}

private int Rank\_D(Double x)

{

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

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

for (int i = 0; i < 5; i++)

{

distanse[i] = Math.Abs(x - num[i]);

}

Double min = distanse[0];

int index = 0;

for (int i = 1; i < 5; i++)

{

if (distanse[i] < min)

{

min = distanse[i];

index = i;

}

}

return (index + 1);

}

private String RankCovert(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 "";

}

}

public class MF

{

public List<Double> y = new List<double>();

public MF()

{

}

}

public class Node

{

public int i;

public int j;

public Double x1;

public Double x2;

public Double y;

public Node(Double x1, Double x2, Double y)

{

this.x1 = x1;

this.x2 = x2;

this.y = y;

i = Zone(x1);

j = Zone(x2);

}

private int Zone(Double x)

{

if (x >= 0 && x <= 0.12)

{

return 0;

}

else if (x > 0.12 && x <= 0.36)

{

return 1;

}

else if (x > 0.36 && x <= 0.59)

{

return 2;

}

else if (x > 0.59 && x <= 0.83)

{

return 3;

}

else

{

return 4;

}

}

}

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()

{

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

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

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

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;

}

Double u13 = (u1 \* u3), u23 = (u2 \* u3), u14 = (u1 \* u4), u24 = (u2 \* u4);

richTextBox\_MF\_out.Text = "U13 = " + u13.ToString("0.00") + "\n";

richTextBox\_MF\_out.Text += "U23 = " + u23.ToString("0.00") + "\n";

richTextBox\_MF\_out.Text += "U14 = " + u14.ToString("0.00") + "\n";

richTextBox\_MF\_out.Text += "U24 = " + u24.ToString("0.00") + "\n";

Double Y = (u13 \* table[zone\_x1, zone\_x2].y.Average() + u23 \* table[zone\_x1, zone\_x2 + 1].y.Average() + u14 \* table[zone\_x1 + 1, zone\_x2].y.Average() + u24 \* table[zone\_x1 + 1, zone\_x2 + 1].y.Average()) / (u13 + u23 + u14 + u24);

richTextBox\_MF\_out.Text += "Y = " + Y;

}

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;

}

}