public class MNK

{

private int size;

private double[] x;

private double[] y;

private double[][] matrix;

private double sumX;

private double sumXX;

private double sumXXX;

private double sumXXXX;

private double sumY;

private double sumXY;

private double sumXXY;

private double A0;

private double A1;

private double A2;

public MNK(int size)

{

this.size = size;

x = new double[size];

y = new double[size];

}

private void CalculateSumX()

{

double sum = 0;

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

{

sum += x[i];

}

sumX = sum;

}

private void CalculateSumXX()

{

double sum = 0;

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

{

sum += x[i] \* x[i];

}

sumXX = sum;

//Console.WriteLine("SumXX : "+sumXX);

}

private void CalculateSumXXX()

{

double sum = 0;

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

{

sum += x[i] \* x[i] \* x[i];

}

sumXXX = sum;

}

private void CalculateSumXXXX()

{

double sum = 0;

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

{

sum += x[i] \* x[i] \* x[i] \* x[i];

}

sumXXXX = sum;

}

private void CalculateSumY()

{

double sum = 0;

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

{

sum += y[i];

}

sumY += sum;

}

private void CalculateSumXY()

{

double sum = 0;

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

{

sum += y[i] \* x[i];

}

sumXY += sum;

}

private void CalculateSumXXY()

{

double sum = 0;

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

{

sum += y[i] \* x[i] \* x[i];

}

sumXXY += sum;

}

public final void setSize(int size)

{

this.size = size;

}

public final void setX(double[] inputX)

{

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

{

x[i] = inputX[i];

}

}

public final void setY(double[] inputY)

{

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

{

y[i] = inputY[i];

}

}

private double calculateDeterminant()

{

double determinant;

double firstPart = matrix[0][0] \* matrix[1][1] \* matrix[2][2] + matrix[0][1] \* matrix[1][2] \* matrix[2][0] + matrix[0][2] \* matrix[1][0] \* matrix[2][1];

double secondPart = matrix[0][2] \* matrix[1][1] \* matrix[2][0] + matrix[0][0] \* matrix[1][2] \* matrix[2][1] + matrix[0][1] \* matrix[1][0] \* matrix[2][2];

determinant = firstPart - secondPart;

return determinant;

}

private void changeCols(int firstNumberIndex, int secondNumberIndex)

{

double temp;

for (int i = 0; i < 3; i++)

{

temp = matrix[i][firstNumberIndex];

matrix[i][firstNumberIndex] = matrix[i][secondNumberIndex];

matrix[i][secondNumberIndex] = temp;

}

}

private void createMatrix()

{

matrix = new double[3][4];

matrix[0][0] = sumXXXX;

matrix[0][1] = sumXXX;

matrix[0][2] = sumXX;

matrix[0][3] = sumXXY;

matrix[1][0] = sumXXX;

matrix[1][1] = sumXX;

matrix[1][2] = sumX;

matrix[1][3] = sumXY;

matrix[2][0] = sumXX;

matrix[2][1] = sumX;

matrix[2][2] = size;

matrix[2][3] = sumY;

}

private void OutputMatrix()

{

for (int i = 0;i < 3;i++)

{

for (int j = 0; j < 4; j++)

{

System.out.print(matrix[i][j] + " ");

}

System.out.println();

}

public final double[] CalculateCoef()

{

CalculateSumX();

CalculateSumXX();

CalculateSumXXX();

CalculateSumXXXX();

CalculateSumXXY();

CalculateSumXY();

CalculateSumY();

createMatrix();

double determinant = calculateDeterminant();

changeCols(0, 3);

double firstDeterminant = calculateDeterminant();

changeCols(3, 0);

changeCols(1, 3);

double secondDeterminant = calculateDeterminant();

changeCols(3, 1);

changeCols(2, 3);

double thirdDeterminant = calculateDeterminant();

changeCols(3, 2);

A2 = firstDeterminant / determinant;

A1 = secondDeterminant / determinant;

A0 = thirdDeterminant / determinant;

return getCoef();

}

private double[] getCoef()

{

double[] coef = new double[3];

coef[0] = A0;

coef[1] = A1;

coef[2] = A2;

return coef;

}

}

public class Usefulness

{

public Usefulness()

{

}

public final double getCoefScale(double k11, double k21, double k12, double k22, double[] firstCoef, double[] secondCoef)

{

double Uk11 = getUseOdnomer(k11, firstCoef);

double Uk12 = getUseOdnomer(k12, firstCoef);

double Uk21 = getUseOdnomer(k21, secondCoef);

double Uk22 = getUseOdnomer(k22, secondCoef);

return (Uk22 - Uk21) / (Uk11 - Uk21 - Uk12 + Uk22);

}

public final double getUseOdnomer(double k, double[] coef)

{

return (coef[2] \* k \* k + coef[1] \* k + coef[0]);

}

public final double getUseMnogomer(double j, double firstUse, double secondUse)

{

return j \* firstUse + (1 - j) \* secondUse;

}

public final int MaxUse(double[] Use)

{

int j = 0;

for (int i = 0; i < Use.length; i++)

{

if (Use[i] > Use[j])

{

j = i;

}

}

return j;

}

}

public class Program

{

static void main(String[] args)

{

final int size = 4;

double[] y = new double[size];

y[0] = 1;

y[1] = 2;

y[2] = 3;

y[3] = 4;

double[] x1 = new double[size];

x1[0] = 0.01;

x1[1] = 0.013333;

x1[2] = 0.02;

x1[3] = 0.04;

double[] x2 = new double[size];

x2[0] = 0.0125;

x2[1] = 0.016667;

x2[2] = 0.025;

x2[3] = 0.05;

MNK firstMNK = new MNK(size);

firstMNK.setX(x1);

firstMNK.setY(y);

double[] firstCoefs = new double[3];

firstCoefs = firstMNK.CalculateCoef();

MNK secondMNK = new MNK(size);

secondMNK.setX(x2);

secondMNK.setY(y);

double[] secondCoefs = new double[3];

secondCoefs = secondMNK.CalculateCoef();

System.out.println("Коэффциенты для первой полезности: ");

for (int i = 0; i < firstCoefs.length; i++)

{

System.out.println(firstCoefs[i] + " ");

}

System.out.println("Коэффциентыдлявторойполезности: ");

for (int i = 0; i < secondCoefs.length; i++)

{

System.out.println(secondCoefs[i] + " ");

}

double[] k1 = new double[size];

k1[0] = 0.03;

k1[1] = 0.02;

k1[2] = 0.0125;

k1[3] = 0.04;

double[] k2 = new double[size];

k2[0] = 0.02;

k2[1] = 0.03;

k2[2] = 0.05;

k2[3] = 0.018;

Usefulness use = new Usefulness();

double j = use.getCoefScale(x1[1], x2[1], x1[0], x2[2], firstCoefs, secondCoefs);

System.out.println("\nКоэффициент j = " + j + "\n");

double[] firstUse = new double[size];

double[] secondUse = new double[size];

double[] UseMnog = new double[size];

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

{

System.out.println("----------------------------------------------------");

System.out.println("k1 = " + k1[i] + " , k2 = " + k2[i]);

firstUse[i] = use.getUseOdnomer(k1[i], firstCoefs);

secondUse[i] = use.getUseOdnomer(k2[i], secondCoefs);

System.out.println("Одномернаяполезность U1(k1) = " + firstUse[i]);

System.out.println("Одномернаяполезность U2(k2) = " + secondUse[i]);

UseMnog[i] = use.getUseMnogomer(j, firstUse[i], secondUse[i]);

System.out.println("Многомернаяполезность U(k1,k2) = " + UseMnog[i]);

}

int number = use.MaxUse(UseMnog);

System.out.println("\nЭффективным решением является " + (number + 1) + " решение.");

System.in.read();

}

}

public class MainWindowModel

{

private PlotModel plotModel;

public final PlotModel getplotModel()

{

return plotModel;

}

private void setplotModel(PlotModel value)

{

plotModel = value;

}

public MainWindowModel()

{

this.setplotModel(new PlotModel());

this.getplotModel().Title = "Кривыебезразличия";

LinearAxis OX = new LinearAxis();

OX.Position = AxisPosition.Bottom;

OX.AbsoluteMaximum = 0.5;

OX.AbsoluteMinimum = 0;

getplotModel().Axes.Add(OX);

LinearAxis OY = new LinearAxis();

OY.Position = AxisPosition.Left;

OY.AbsoluteMinimum = 0;

OY.AbsoluteMaximum = 0.5;

getplotModel().Axes.Add(OY);

addCurves();

}

private void addCurves()

{

double[] k1 = new double[]{0.01, 0.013, 0.02, 0.04};

double[] k2 = new double[] {0.0125, 0.016, 0.025, 0.05};

LineSeries u0 = new LineSeries();

u0.StrokeThickness = 2;

u0.MarkerSize = 3;

u0.CanTrackerInterpolatePoints = false;

u0.Smooth = true;

LineSeries u1 = new LineSeries();

u1.StrokeThickness = 2;

u1.MarkerSize = 3;

u1.CanTrackerInterpolatePoints = false;

u1.Smooth = true;

LineSeries u2 = new LineSeries();

u2.StrokeThickness = 2;

u2.MarkerSize = 3;

u2.CanTrackerInterpolatePoints = false;

u2.Smooth = true;

LineSeries u3 = new LineSeries();

u3.StrokeThickness = 2;

u3.MarkerSize = 3;

u3.CanTrackerInterpolatePoints = false;

u3.Smooth = true;

LineSeries u4 = new LineSeries();

u4.StrokeThickness = 2;

u4.MarkerSize = 3;

u4.CanTrackerInterpolatePoints = false;

u4.Smooth = true;

u0.Points.Add(new DataPoint(k1[0], k2[0]));

u1.Points.Add(new DataPoint(k1[1], k2[0]));

u1.Points.Add(new DataPoint(k1[0], k2[1]));

u2.Points.Add(new DataPoint(k1[2], k2[0]));

u2.Points.Add(new DataPoint(k1[1], k2[1]));

u2.Points.Add(new DataPoint(k1[0], k2[2]));

u3.Points.Add(new DataPoint(k1[3], k2[0]));

u3.Points.Add(new DataPoint(k1[2], k2[1]));

u3.Points.Add(new DataPoint(k1[1], k2[2]));

u3.Points.Add(new DataPoint(k1[0], k2[3]));

u4.Points.Add(new DataPoint(k1[3], k2[1]));

u4.Points.Add(new DataPoint(k1[2], k2[2]));

u4.Points.Add(new DataPoint(k1[1], k2[3]));

getplotModel().Series.Add(u0);

getplotModel().Series.Add(u1);

getplotModel().Series.Add(u2);

getplotModel().Series.Add(u3);

getplotModel().Series.Add(u4);

}

}