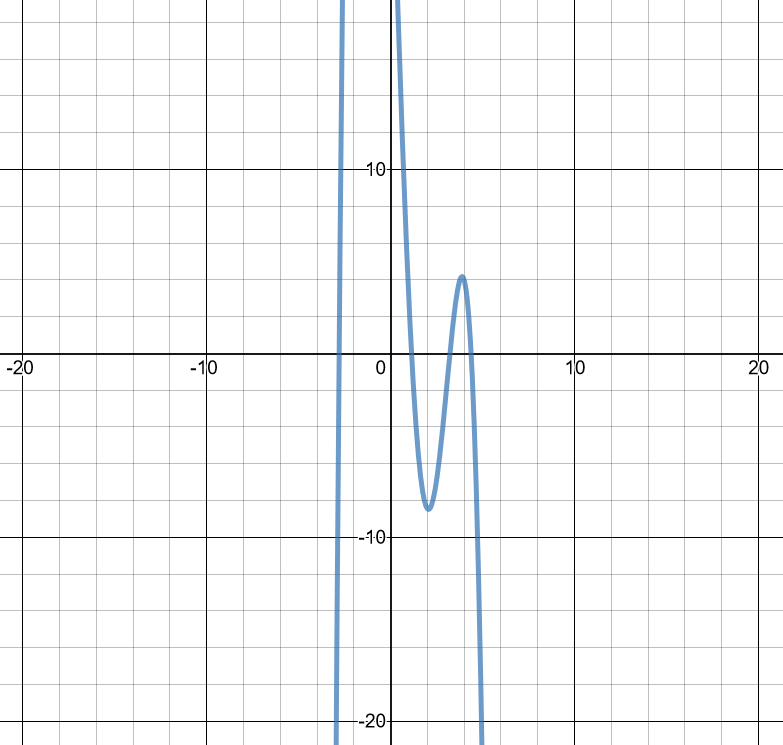
# Netiesinių lygčių sprendimas

Duotos dvi netiesinės lygtys: daugianaris f(x) = 0 ir transcendentinė funkcija g(x) = 0.

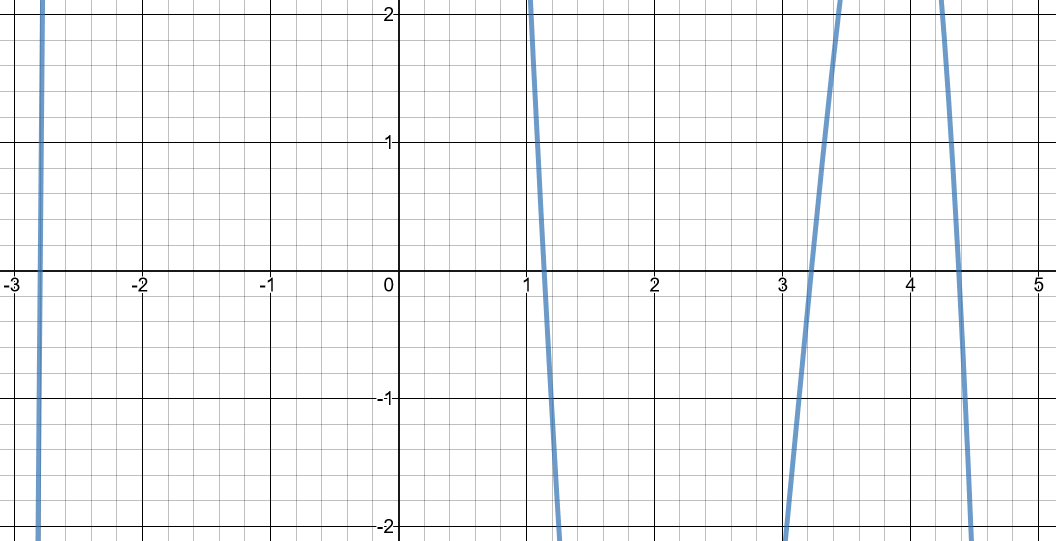
|  |  |
| --- | --- |
| Daugianaris f(x) | Funkcija g(x) |
|  |  |
| Sprendimo metodai: stygų, paprastųjų iteracijų, skenavimo su mažėjančiu žingsniu | |

## Lygties f(x) = 0 (f(x) – daugianaris) sprendimas

* **Daugianario šaknų intervalo įverčiai**



pav. 1 Daugianario grubaus šaknų intervalo iverčiai. Grafiko dydis yra grubaus šaknų intervalo dydžio



pav. 2 Daugianario tikslesnio šaknų intervalo įverčiai. Grafiko dydis yra tikslesnio šaknų intervalo dydžio

* **1 lentelė.** Šaknų intervalo įverčiai.

|  |  |
| --- | --- |
| Grubus lygties 𝑓(𝑥) = 0 šaknų intervalo įvertis | [-20; 20 ] |
| Tikslesnis lygties 𝑓(𝑥) = 0 šaknų intervalo įvertis | [-3; 5] |

* **Šaknų atskyrimas skenavimo metodu**

Skenavimas atliekamas intervale [-3; 5], skenavimo žingsnis lygus 0.1.

**2 lentelė.** Šaknies atskyrimo intervalai.

|  |  |
| --- | --- |
| Intervalo Nr. | Intervalas |
| 1 | [-2,9; 2,8] |
| 2 | [1,1; 1,2] |
| 3 | [3,2; 3.3] |
| 4 | [4.3; 4.4] |

* **Šaknų tikslinimas stygų,** **paprastųjų iteracijų, skenavimo su mažėjančiu žingsniu**

Tariama, kad 𝑥𝑔 yra šaknis (stabdomi skaičiavimai), jei |𝑓(𝑥𝑔)| < 1𝑒 −6 . Skaičiavimuose naudojamas šaknies tikslumo įvertis |𝑓(𝑥𝑔)|.

Čia ir toliau papildomos funkcijos reikalingos paprastųjų iteracijų metodui naudosiu Niutono Rapsono formuluotę, kuri padės rasti labiausi konverguojančią funkciją:

Taip gaunu funciją reikalingą šaknims nustatyti paprastųjų iteracijų metode:

**3 lentelė.** Rezultatų lentelė.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ***Stygų metodas*** | Pradinis intervalas | Šaknis | Tikslumas | Iteracijų skaičius |
| [-2,9; 2,8] | -2.8001120 | 0.0000001 | 4 |
| [1,1; 1,2] | 1.1379328 | -0.0000002 | 4 |
| [3,2; 3.3] | 3.2264350 | 0.0000002 | 3 |
| [4.3; 4.4] | 4.3786013 | 0.0000009 | 4 |
| ***Skenavimo metodas*** | Pradinis intervalas | Šaknis | Tikslumas | Iteracijų skaičius |
| [-2,9; 2,8] | -2.8001120 | 0.0000002 | 63 |
| [1,1; 1,2] | 1.1379328 | 0.0000007 | 37 |
| [3,2; 3.3] | 3.2264350 | 0.0000001 | 24 |
| [4.3; 4.4] | 4.3786013 | 0.0000008 | 30 |
| ***Paprastųjų iteracijų*** | Pradinis artinys | Šaknis | Tikslumas | Iteracijų skaičius |
| -2,9 | -2.8001120 | 0.0000000 | 3 |
| 1,1 | 1.1379328 | 0.0000000 | 3 |
| 3,2 | 3.2264350 | 0.0000000 | 2 |
| 4.3 | 4.3786014 | -0.0000004 | 3 |

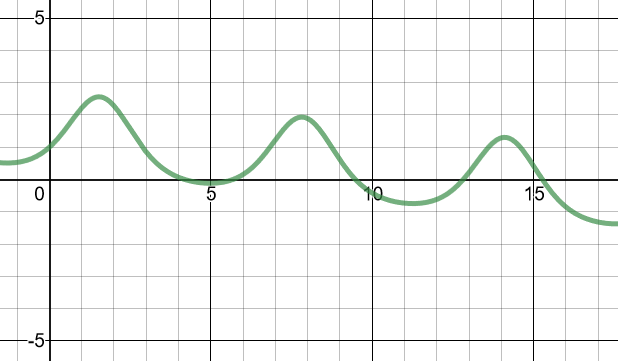
Pagal turimus rezultatus galima teigti kad f(x) = 0 šaknis parinkus tinkamą artinį randa paprastųjų iteracijų metodas. Taip pas šis metodas atrodo tiksliausia. Metodų rezultatai ir tikslumas yra labai panašūs.

## Lygties g(x)=0 (g(x) – transcendentine funkcija) sprendimas

Antra funkcija paprastųjų iteracijų metodui:

**4 lentelė.** Šaknies atskyrimo intervalai.

|  |  |
| --- | --- |
| Intervalo Nr. | Intervalas |
| 1 | [4.1; 4.2] |
| 2 | [5.6; 5.7] |
| 3 | [9.4; 9.5] |
| 4 | [12.8; 12.9] |



pav. 4 Funkcijos šaknų intervalų rėžiai, pavaizduoti grafiškai

**5 lentelė.** Rezultatų lentelė.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ***Stygų metodas*** | Pradinis intervalas | Šaknis | Tikslumas | Iteracijų skaičius |
| [4.1; 4.2] | 4.1944265 | -0.0000006 | 3 |
| [5.6; 5.7] | 5.6825214 | -0.0000001 | 3 |
| [9.4; 9.5] | 9.4783758 | 0.0000000 | 4 |
| [12.8; 12.9] | 12.8171944 | -0.0000005 | 3 |
| ***Skenavimo metodas*** | [4.1; 4.2] | 4.1944220 | 0.0000008 | 25 |
| [5.6; 5.7] | 5.6825200 | -0.0000007 | 20 |
| [9.4; 9.5] | 9.4783750 | 0.0000008 | 34 |
| [12.8; 12.9] | 12.8171950 | 0.0000001 | 27 |
| ***Paprastųjų iteracijų*** | Pradinis artinys | Šaknis | Tikslumas | Iteracijų skaičius |
| 4.1 | 4.1944247 | 0.0000000 | 3 |
| 5.6 | 5.6825218 | 0.0000000 | 3 |
| 9.4 | 9.4783757 | 0.0000000 | 3 |
| 12.8 | 12.8171949 | 0.0000000 | 2 |

Iš gautų rezultų galima teigti, kad g(x) = 0 šaknų radimui mažiausiai iteracijų atlieka paprastųjų iteracijų metodas. Parinkus tinkamus artinius jis rezultatus randa greičiau ir tiksliau negu kiti du metodai. Taip pat šiuo atveju stygų metodas pagal spartą yra antroje vietoja ir yra tikslesnis bei greitesnis už skenavimo metodą.

## Sąlyginio uždavinio sprendimas

**Sąlyga.**

Skysčio tūris 𝑉 sferos formos talpoje priklauso nuo skysčio lygio (aukščio) ℎ pagal dėsnį:

Koks yra skysčio aukštis ℎ talpoje, jeigu žinomas 𝑟, 𝑉?

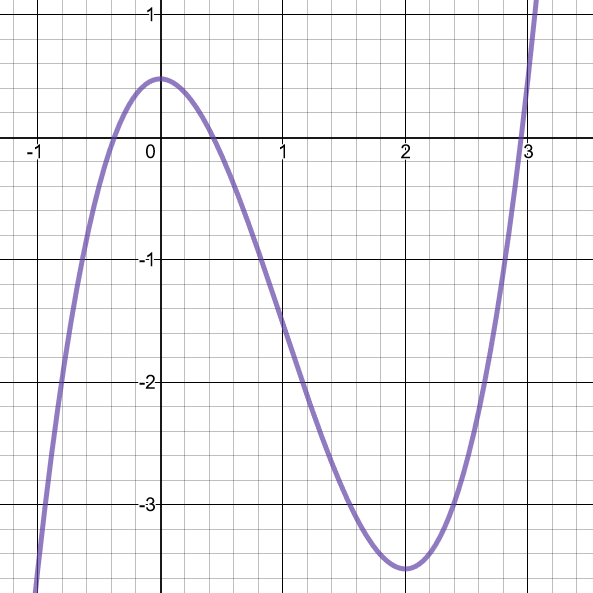
r = 1; V = 0,5;

**Sprendimas.**

Įsistatome duotas reikšmes, o *h* pakeičiame į *x*:

Pertvarkome lygybę ir išsireškiame funkciją:

Antra funkcija paprastųjų iteracijų metodui:



pav. 5 Sąlyginio uždavinio grafikas ir jo šaknys

**6 lentelė.** Šaknų atskyrimo intervalai.

|  |  |
| --- | --- |
| Intervalo Nr. | Intervalas |
| 1 | [-0.4; -0.3] |
| 2 | [0.4; 0.5] |
| 3 | [2.9; 3] |

**7 lentelė.** Rezultatų lentelė.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ***Stygų metodas*** | Pradinis intervalas | Šaknis | Tikslumas | Iteracijų skaičius |
| [-0.4; -0.3] | -0.3760668 | 0.0000004 | 4 |
| [0.4; 0.5] | 0.4311204 | 0.0000005 | 4 |
| [2.9; 3] | 2.9449462 | -0.0000008 | 4 |
| ***Skenavimo metodas*** | [-0.4; -0.3] | -0.3760670 | -0.0000001 | 24 |
| [0.4; 0.5] | 0.4311210 | -0.0000007 | 12 |
| [2.9; 3] | 2.9449462 | -0.0000007 | 34 |
| ***Paprastųjų iteracijų*** | Pradinis artinys | Šaknis | Tikslumas | Iteracijų skaičius |
| -0.4 | -0.3760670 | 0.0000000 | 3 |
| 0.4 | 0.4311207 | 0.0000000 | 3 |
| 2.9 | 2.9449463 | 0.0000000 | 3 |

Gautos trys šaknys. Įdomu, kad viena iš jų yra neigiama. Todėl tūrį galime traktuoti šiuo atveju gal net kaip vektorinį dydį. Paprastųjų iteracijų metodui surasti šaknis prireikė mažiau iteracijų, tačiau rezultatai panašūs visų trijų metodų.

# Išvados

Laboratorinio darbo metu buvo analizuojama algebrinės lygties su vienu nežinomuoju sprendimo etapai – šaknų atskyrimo ir jų tikslinimo uždaviniai. Buvo įgyvendinti stygų, skenavimo metodai ir paprastųjų iteracijų metodas pasitelkiant Niutono Rapsono formulę. Visų trijų uždavinių sprendimas parodė, kad parinkus tinkamą artinį sparčiausia ir tiksliausias yra paprastųjų iteracijų metodas. Jam surasti sprendinį reikia maždaug trijų iteracijų, o tikslumas dažniausia būna net didenis negu 0.0000001. Antroje vietoje – stygų metodas, kuris užduotį įvykto per maždaug 5 iteracijas. Ir galiausiai – skenavimo metodas, kuriam prireikia nuo 20 iki 60 iteracijų.

# Programos tekstai

**Stygų metodas:**

private void Button6\_Click(object sender, EventArgs e)

{

ClearForm(); // išvalomi programos duomenys

prepareForm();

x1 = -3; // izoliacijos intervalo pradžia

x2 = -2; // izoliacijos intervalo galas

iii = 0; // iteraciju skaičius

richTextBox1.AppendText("Iteracija x F(x) x1 x2 F(x1) F(x2) \n");

// Nubraižoma f-ja, kuriai ieskome saknies

Fx = chart1.Series.Add("F(x)");

Fx.ChartType = SeriesChartType.Line;

double x = -8;

for (int i = 0; i < 300; i++)

{

Fx.Points.AddXY(x, F(x));

x = x + 0.1;

}

Fx.BorderWidth = 3;

X1X2 = chart1.Series.Add("X1X2");

X1X2.MarkerStyle = MarkerStyle.Circle;

X1X2.MarkerSize = 8;

X1X2.ChartType = SeriesChartType.Point;

X1X2.ChartType = SeriesChartType.Line;

XMid = chart1.Series.Add("XMid");

XMid.MarkerStyle = MarkerStyle.Circle;

X1X2.ChartType = SeriesChartType.Point;

X1X2.ChartType = SeriesChartType.Line;

XMid.MarkerSize = 8;

var thing = intervals.Dequeue();

x1 = thing.Item1;

x2 = thing.Item2;

timer4.Enabled = true;

timer4.Interval = 500; // timer2 intervalas milisekundemis

timer4.Start();

}

private void Timer4\_Tick(object sender, EventArgs e)

{

k = (float)(Math.Abs(F(x1) / F(x2)));

xtemp = (x1 + k \* x2) / (1 + k);

if (Math.Abs(F(xtemp)) > 1e-6 & iii <= N)

// tikrinama salyga, ar funkcijos absoliuti reiksme daugiau uz nustatyta (norima)

// tiksluma ir nevirsytas maksimalus iteraciju skaicius

{

X1X2.Points.Clear();

XMid.Points.Clear();

X1X2.Points.AddXY(x1, F(x1));

X1X2.Points.AddXY(x2, F(x2));

XMid.Points.AddXY(xtemp, F(xtemp));

richTextBox1.AppendText(String.Format(" {0,6:d} {1,12:f7} {2,12:f7} {3,12:f7} {4,12:f7} {5,12:f7} {6,12:f7}\n",

++iii, xtemp, F(xtemp), x1, x2, F(x1), F(x2)));

if (Math.Sign((double)F(x1)) != Math.Sign((double)F(xtemp)))

{

x2 = xtemp;

}

else

{

x1 = xtemp;

}

}

else

{

richTextBox1.AppendText($" {++iii,6:d} {xtemp,12:f7} {F(xtemp),12:f7} {x1,12:f7} {x2,12:f7} {F(x1),12:f7} {F(x2),12:f7}\n");

richTextBox1.AppendText("Skaičiavimai baigti\n");

iii = 0;

if (intervals.Any())

{

var thing = intervals.Dequeue();

x1 = thing.Item1;

x2 = thing.Item2;

}

else

{

timer4.Stop();

}

}

}

**Skenavimo metodas:**

private void button8\_Click(object sender, EventArgs e)

{

ClearForm();

double x = -8;

int i = 0;

if (comboBox1.SelectedIndex == 2)

{

//x = 0.01;

i = 1;

}

prepareForm();

richTextBox1.AppendText("Iteracija x F(x) x1 x2 F(x1) F(x2) \n");

Fx = chart1.Series.Add("F(x)");

Fx.ChartType = SeriesChartType.Line;

for (; i < 300; i++)

{

Fx.Points.AddXY(x, F(x)); x = x + (2 \* Math.PI) / 50;

}

Fx.BorderWidth = 3;

X1X2 = chart1.Series.Add("X1X2");

X1X2.MarkerStyle = MarkerStyle.Circle;

X1X2.MarkerSize = 8;

X1X2.ChartType = SeriesChartType.Point;

X1X2.ChartType = SeriesChartType.Line;

XMid = chart1.Series.Add("XMid");

XMid.MarkerStyle = MarkerStyle.Circle;

X1X2.ChartType = SeriesChartType.Point;

X1X2.ChartType = SeriesChartType.Line;

XMid.MarkerSize = 8;

var thing = intervals.Dequeue();

x1 = thing.Item1;

x2 = thing.Item2;

timer5.Enabled = true;

timer5.Interval = 50;

timer5.Start();

}

private void timer5\_Tick(object sender, EventArgs e)

{

xtemp = x1 + stepSize;

if (Math.Abs(F(xtemp)) > 1e-6 & iii <= N)

{

X1X2.Points.Clear();

X1X2.Points.AddXY(x1, 0);

X1X2.Points.AddXY(xtemp, 0);

richTextBox1.AppendText($" {iii,6:d} {xtemp,12:f7} {F(xtemp),12:f7} {x1,12:f7} {x2,12:f7} {F(x1),12:f7} {F(x2),12:f7}\n");

if (Math.Sign(F(x1)) != Math.Sign(F(xtemp)))

{

stepSize /= 10;

}

else

{

x1 += stepSize;

}

iii = iii + 1;

}

else

{

richTextBox1.AppendText($" {iii,6:d} {xtemp,12:f7} {F(xtemp),12:f7} {x1,12:f7} {x2,12:f7} {F(x1),12:f7} {F(x2),12:f7}\n");

richTextBox1.AppendText("Skaičiavimai baigti");

iii = 0;

if (intervals.Any())

{

var thing = intervals.Dequeue();

x1 = thing.Item1;

x2 = thing.Item2;

stepSize = 0.1;

}

else

{

timer5.Stop();

}

}

}

**Paprastųjų iteracijų:**

private void button7\_Click(object sender, EventArgs e)

{

ClearForm();

double x = -8;

int i = 0;

prepareForm();

switch (comboBox1.SelectedIndex)

{

case 2:

i = 1;

break;

}

richTextBox1.AppendText("Iteracija x F(x) x1 x2 F(x1) F(x2) \n");

Fx = chart1.Series.Add("F(x)");

Fx.ChartType = SeriesChartType.Line;

Gx = chart1.Series.Add("G(x)");

Gx.ChartType = SeriesChartType.Line;

XY = chart1.Series.Add("y = x");

XY.ChartType = SeriesChartType.Line;

for (; i < 300; i++)

{

Fx.Points.AddXY(x, F(x));

Gx.Points.AddXY(x, G(x));

XY.Points.AddXY(x, x); x = x + (2 \* Math.PI) / 50;

}

Fx.BorderWidth = 3;

Gx.BorderWidth = 3;

XY.BorderWidth = 3;

X1X2 = chart1.Series.Add("X1X2");

X1X2.MarkerStyle = MarkerStyle.Circle;

X1X2.MarkerSize = 8;

X1X2.ChartType = SeriesChartType.Point;

X1X2.ChartType = SeriesChartType.Line;

XMid = chart1.Series.Add("XMid");

XMid.MarkerStyle = MarkerStyle.Circle;

X1X2.ChartType = SeriesChartType.Point;

X1X2.ChartType = SeriesChartType.Line;

XMid.MarkerSize = 8;

var thing = intervals.Dequeue();

x1 = thing.Item1;

timer6.Enabled = true;

timer6.Interval = 500;

timer6.Start();

}

private void timer6\_Tick(object sender, EventArgs e)

{

xtemp = G(x1);

x1 = xtemp;

if (Math.Abs(F(x1)) > 1e-6 & iii <= N)

{

X1X2.Points.Clear();

XMid.Points.Clear();

XMid.Points.AddXY(x1, 0);

richTextBox1.AppendText($" {iii,6:d} {xtemp,12:f7} {F(xtemp),12:f7} {x1,12:f7} {x2,12:f7} {F(x1),12:f7} {F(x2),12:f7}\n");

iii = iii + 1;

}

else

{

richTextBox1.AppendText($" {iii,6:d} {xtemp,12:f7} {F(xtemp),12:f7} {x1,12:f7} {x2,12:f7} {F(x1),12:f7} {F(x2),12:f7}\n");

richTextBox1.AppendText("Skaičiavimai baigti");

iii = 0;

if (intervals.Any())

{

var thing = intervals.Dequeue();

x1 = thing.Item1;

}

else

{

timer6.Stop();

}

}

}

**Papildomos funkcijos:**

private double F(double x)

{

switch (comboBox1.SelectedIndex)

{

case 0:

return (double)(-0.7 \* Math.Pow(x, 4) + 4.16 \* Math.Pow(x, 3) + 1.19 \* Math.Pow(x, 2) - 33.4 \* x + 31.51);

case 1:

return (double)((double)(Math.Pow(Math.E, Math.Sin(x))) - (double)(x/10));

case 2:

return (double)(Math.Pow(x, 3) - (3 \* Math.Pow(x, 2)) + (double)(1.5 / Math.PI));

}

return 0;

}

private double G(double x)

{

switch (comboBox1.SelectedIndex)

{

case 0:

return (double)(x - ((-0.7 \* Math.Pow(x, 4) + 4.16 \* Math.Pow(x, 3) + 1.19 \* Math.Pow(x, 2) - 33.4 \* x + 31.51) / (-2.8 \* Math.Pow(x, 3) + 12.48 \* Math.Pow(x, 2) + 2.38 \* x - 33.4)));

case 1:

return (double)(x - ((Math.Pow(Math.E, Math.Sin(x)) - x / 10) / (Math.Pow(Math.E, Math.Sin(x)) \* Math.Cos(x) - 0.1)));

case 2:

return (double)(x- ((Math.Pow(x, 3) - (3 \* Math.Pow(x, 2)) + (double)(1.5 / Math.PI)) / (3 \* Math.Pow(x, 2) - 6 \* x)));

}

return 0;

}

private void FillIntervals()

{

int1.Clear();

int2.Clear();

int3.Clear();

comboBox1.SelectedIndex = 0;

x1 = -3;

x2 = 5;

var xTemp = x1;

for (x1 += stepSize; x1 < x2; x1 += stepSize)

{

if (Math.Sign(F(x1)) != Math.Sign(F(xTemp)))

{

Console.WriteLine("{0} {1}",xTemp, x1);

int1.Enqueue(new Tuple<double, double>(xTemp, x1));

}

xTemp = x1;

}

comboBox1.SelectedIndex = 1;

x1 = 1;

x2 = 15;

xTemp = x1;

for (x1 += stepSize; x1 < x2; x1 += stepSize)

{

if (Math.Sign(F(x1)) != Math.Sign(F(xTemp)))

{

Console.WriteLine("{0} {1}", xTemp, x1);

int2.Enqueue(new Tuple<double, double>(xTemp, x1));

}

xTemp = x1;

}

comboBox1.SelectedIndex = 2;

x1 = -1;

x2 = 4;

xTemp = x1;

for (x1 += stepSize; x1 < x2; x1 += stepSize)

{

if (Math.Sign(F(x1)) != Math.Sign(F(xTemp)))

{

Console.WriteLine("{0} {1}", xTemp, x1);

int3.Enqueue(new Tuple<double, double>(xTemp, x1));

}

xTemp = x1;

}

}