МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ РАДІОЕЛЕКТРОНІКИ
КАФЕДРА ПРОГРАМНОЇ ІНЖЕНЕРІЇ

ЗВІТ ДО ЛАБОРАТОРНОЇ РОБОТИ №4
З ДИСЦИПЛІНИ «Математичне моделювання та чисельні методи»
НА ТЕМУ «Розв'язання задач пошуку екстремуму»

Виконав:
студент гр. ПЗПІ-17-1
Кириченко О. В.
Перевірив:
Матвєєв Д. І.

Харків – 2020

Мета роботи: Навчитись використовувати математичне середовище MATLAB для пошуку екстремумів функцій. Відтворити досліджену поведінку математичного середовища за допомогою мови програмування.

Задля локанічності нижче наведено загальні методи на мові C#, які будуть використані у завданнях:

```
public (double[] mins, double[] maxs) FindExtremumsUniform(
	double[] poly, double start, double end, double delta
){
	List<double> mins = new List<double>();
	List<double> maxs = new List<double>();

	double lastX = start;
	while(lastX <= end)
	{
		Extremum extremum = FindExtremumUniformForUnimodal(
			poly, lastX, end, delta
		);
		if (extremum == null)
			break;
		if (extremum.IsMin)
			mins.Add(extremum.X);
		else
			maxs.Add(extremum.X);

		lastX = extremum.X;
	}

	return (mins.ToArray(), maxs.ToArray());
}
```

```
public Extremum FindExtremumUniformForUnimodal(
	double[] poly, double start, double end, double delta
){
	double[] xs = Range.Make(start, end, delta);
	Extremum extremum = new Extremum() { IsMin = true, X = start };

	Predicate<int> breakCondition = i =>
		Poly.Val(poly, xs[i + 1])[0] >= Poly.Val(poly, xs[i])[0];

	if (breakCondition(0))
	{
		breakCondition = i =>
			Poly.Val(poly, xs[i + 1])[0] <= Poly.Val(poly, xs[i])[0];

		extremum.IsMin = false;
	}

	for (int i = 0; i < xs.Length - 1; ++i)
		if (breakCondition(i))
		{
			extremum.X = xs[i] + delta / 2;
			return extremum;
		}

	return null;
}

public class Extremum
{
	public double X { get; set; }
	public bool IsMin { get; set; }
}
```

```
public static class Poly
{
      public static double[] PolySolvePrimitive(double[] poly, double eq)
      {
            if (poly.Length == 2)
                  return new double[] { (eq - poly[0]) / poly[1] };

            throw new ArgumentException();
      }

      public static double[] GetPolyDerivative(double[] poly)
      {
            double[] diffPoly = new double[poly.Length - 1];

            for (int i = 0; i < diffPoly.Length; ++i)
                  diffPoly[i] = poly[i + 1] * (i + 1);

            return diffPoly;
      }

      public static Func<double, double> Get(double[] p)
      {
            return x => p.Select((a, i) => a * Math.Pow(x, i)).Sum();
      }

      public static double[] Val(double[] p, params double[] x)
      {
            Func<double, double> poly = Get(p);
            return x.Select(xi => poly(xi)).ToArray();
      }
}
```

```
public static class Range
{
    public static double[] Make(double start, double end, double delta)
    {
        int count = (int)Math.Ceiling((end - start) / delta);
        return Enumerable.Range(0, count).Select(
            i => i * delta + start
        ).ToArray();
    }
}
```

1. Визначити проміжки унімодальності функції $f(x) = x^3 - x^2 - 4.4 * 0.9x + 4 * 0.9$. Обчислити координати усіх точок екстремумів за допомогою метода рівномірного пошуку.

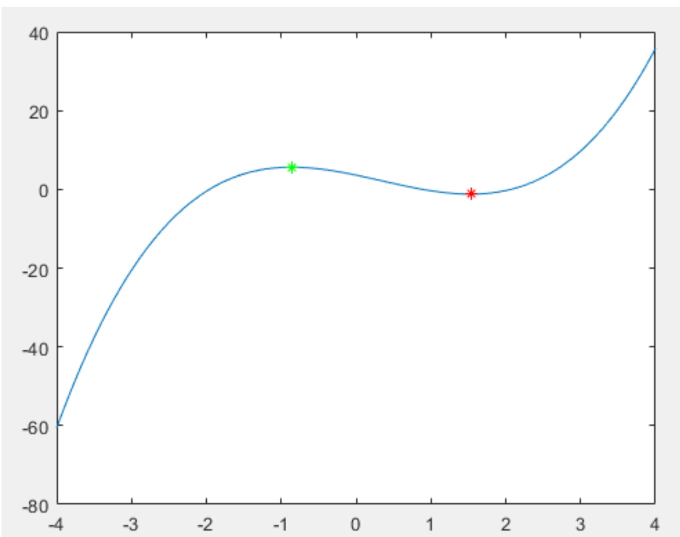
MATLAB

```matlab
function [minxs, maxxs] = getExtremumsUniform(f, a, b, delta)
    minxs = [];
    maxxs = [];

    last = a;
    while(last <= b)
        [extrx, ismin, success] = getExtremumUniform(f, last, b, delta);

        if(~success)
            break;
        end

        if(ismin == 1)
            minxs = [minxs extrx];
            last = extrx;
        else
            maxxs = [maxxs extrx];
            last = extrx;
        end
    end
end

function [extrx, ismin, success] = getExtremumUniform(f, a, b, delta)
    ismin = 1;
    extrx = b;
    success = 0;

    xs = a:delta:b;
    breakCondition = @(i)(f(xs(i + 1)) >= f(xs(i)));

    if(f(a) < f(a + delta))
        ismin = 0;
        breakCondition = @(i)(f(xs(i + 1)) <= f(xs(i)));
    end

    for i = 1:(length(xs) - 1)
        if(breakCondition(i))
            extrx = xs(i) + delta / 2;
            success = 1;
            break;
        end
    end
end
```

C#

```csharp
public Point Task1(
    Form form, Point location
) {
double a = 0.9;
double[] f = new double[] {
  4 * a, -4.4 * a, -1, 1
};
double[] firstDerivative = Poly.GetPolyDerivative(f);
double[] secondDerivative = Poly.GetPolyDerivative(firstDerivative);
double[] xs = Range.Make(-5, 5, 0.1);
double[] breakXs = Poly.PolySolvePrimitive(
    secondDerivative, 0
);
(double[] mins, double[] maxs) = FindExtremumsUniform(f, -4, 4, 0.01);
```

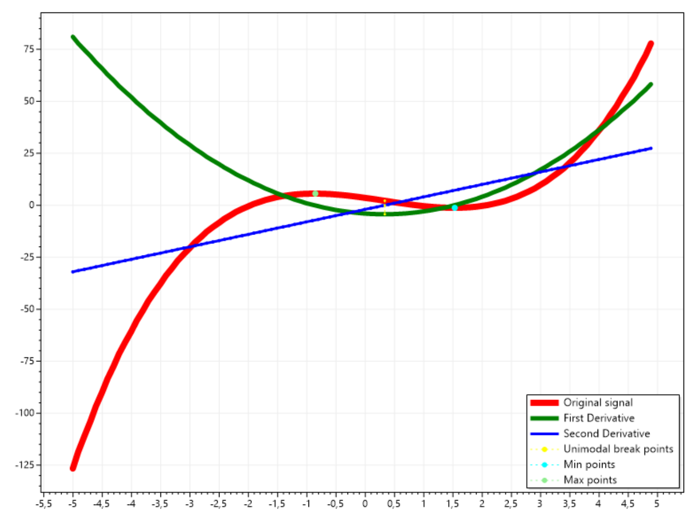
```
>> syms f(x)
>> a = 0.9

a =

    0.9000

>> f(x) = x^3 - x^2 - 4.4 * a * x + 4 * a

f(x) =

x^3 - x^2 - (99*x)/25 + 18/5

>> [minxs, maxxs] = getExtremumsUniform(f, -4, 4, 0.01)

minxs =

    1.5300


maxxs =

   -0.8550

>> xs = -4:0.1:4

>> plot(xs, f(xs), minxs, f(minxs), '*r', maxxs, f(maxxs), '*g')
```




```
Console.WriteLine(
    "MINS: " + string.Join(" ", mins)
);
Console.WriteLine(

    "MAXS: " + string.Join(" ", maxs)

);

Plot plot = new Plot();

plot.PlotSignalXY(

    xs, Poly.Val(f, xs), Color.Red, 7,

    label: "Original signal"

);

plot.PlotSignalXY(

    xs, Poly.Val(firstDerivative, xs),

    Color.Green, 5,

    label: "First Derivative"

);

plot.PlotSignalXY(

    xs, Poly.Val(secondDerivative, xs),

    Color.Blue, 3,

    label: "Second Derivative"

);

plot.PlotScatter(

  breakXs.Concat(breakXs).Concat(

    breakXs

  ).ToArray(),

  Poly.Val(f, breakXs).Concat(

    Poly.Val(firstDerivative, breakXs)

  ).Concat(

    Poly.Val(secondDerivative, breakXs)

  ).ToArray(),

  Color.Yellow,

  label: "Unimodal break points",

  markerSize: 3,

  lineStyle: LineStyle.Dot

);

plot.PlotScatter(

    mins, Poly.Val(f, mins),
```

```
    Color.Cyan, markerSize: 7,
    label: "Min points",
    lineStyle: LineStyle.Dot
);

plot.PlotScatter(
   maxs, Poly.Val(f, maxs),
   Color.LightGreen, markerSize: 7,
   label: "Max points",
   lineStyle: LineStyle.Dot
);
plot.Legend();
return plot.AddToForm(form, location);
}
```

```
MINS: 1,53
MAXS: -0,855
```



Також задля перевірки у MATLAB було створено скрипт, що знаходить екстремуми за допомогою вбудованої функції пошуку мінімуму fmincon:

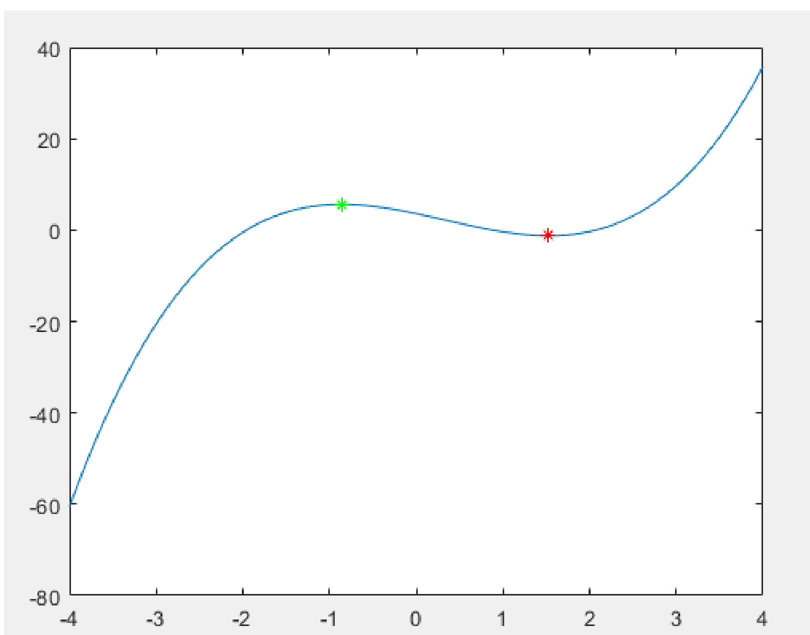
```
function [minxs, maxxs] = getExtremums(f, a, b)
    fwrapmin = @(x)(double(f(x)));
    fwrapmax = @(x)(double(-f(x)));

    minxs = []; maxxs = [];

    midpointsEq = diff(f, 2) == 0;
    midpoints = solve(midpointsEq);

    if(f(midpoints(1) - eps) >= f(midpoints(1)))
        maxxs = [maxxs appendFoundUnimodalExtremum(fwrapmax, a, midpoints(1))];
    else
        minxs = [minxs appendFoundUnimodalExtremum(fwrapmin, a, midpoints(1))];
    end

    for i = 1:(length(midpoints) - 1)
        if(f(midpoints(i) + eps) >= f(midpoints(i)))
            maxxs = [maxxs appendFoundUnimodalExtremum(fwrapmax, midpoints(i), midpoints(i + 1))];
        else
            minxs = [minxs appendFoundUnimodalExtremum(fwrapmin, midpoints(i), midpoints(i + 1))];
        end
    end

    if(f(midpoints(length(midpoints)) + eps) >= f(midpoints(length(midpoints))))
        maxxs = [maxxs appendFoundUnimodalExtremum(fwrapmax, midpoints(length(midpoints)), b)];
    else
        minxs = [minxs appendFoundUnimodalExtremum(fwrapmin, midpoints(length(midpoints)), b)];
    end
end

function [extrx] = appendFoundUnimodalExtremum(fwrap, a, b)
    extrx = [];

    if(b >= a)
        xs = fmincon(fwrap, double(a), [], [], [], [], double(a), double(b));
        if(xs >= a && xs <= b)
            extrx = xs;
        end
    end
end
```

Результат виконання функції, що використовує метод рівномірного пошуку, наближається до результату, отриманого з використанням вбудованої функції fmincon.

```
>> a = 0.9

a =

    0.9000

>> syms f(x)
>> f(x) = x^3 - x^2 - 4.4 * a * x + 4 * a

f(x) =

x^3 - x^2 - (99*x)/25 + 18/5

>> [ minxs, maxxs ] = getExtremums(f, -4, 4)
```

```
minxs =

    1.5296


maxxs =

   -0.8630

>> xs = -4:0.1:4
```

```
>> plot(xs, f(xs), minxs, f(minxs), '*r', maxxs, f(maxxs), '*g')
```



2. Розв'язати задачу лінійного програмування:

$$Z(x1, x2) = 2 * 0.9 * x1 + (1.5 + 0.9) * x2 => max$$

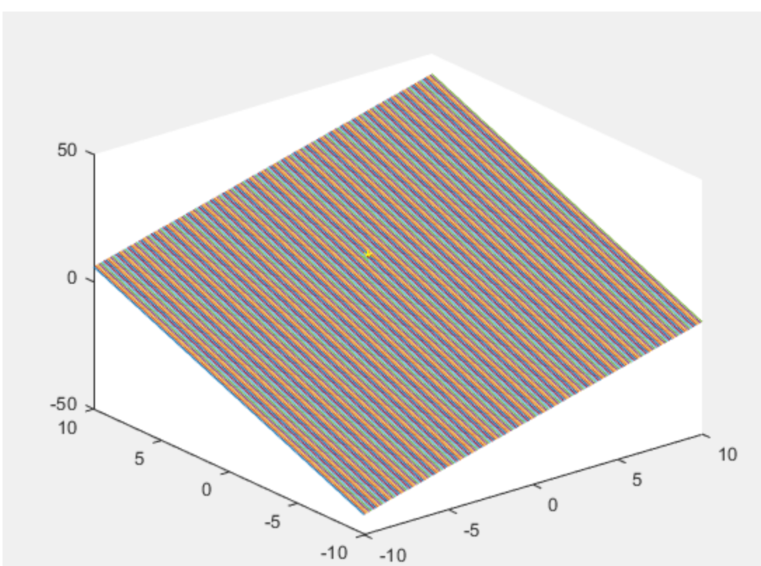
за обмежень:

$$\begin{cases} -(0.9 + 2)x1 + (5 - 0.9)x2 \leq 10 \\ x1 + (3.5 - 0.9)x2 \geq 3 \\ (0.9 + 1.8)x1 + (4.1 - 1.8)x2 \leq 10 \\ x1 \geq 0 \\ x2 \geq 0 \end{cases}$$

Було записано наступні MATLAB-вирази:

```
>> a = 0.9;
>> Z = @(x)(2 * a * x(1) + (1.5 + a) * x(2))

Z =

  function_handle with value:

    @(x)(2*a*x(1)+(1.5+a)*x(2))

>> syms Z1(x1, x2)
>> Z1(x1, x2) = 2 * a * x1 + (1.5 + a) * x2

Z1(x1, x2) =

(9*x1)/5 + (12*x2)/5

>> xr = fmincon( @(x)(-Z(x)), [0, 0], [ -2.9, 4.1; -1, -2.6; 2.7, 3.2; -1, 0; 0, -1 ], [ 10; -3; 10; 0; 0])
```

```
xr =

    0.4423    2.7518

>> [x1s, x2s] = meshgrid(-10:0.1:10);
>> zs = Z1(x1s, x2s);
>> plot3(x1s, x2s, zs, xr(1), xr(2), Z1(xr(1), xr(2)), '*')
```



3. Розв'язати наступну транспортну задачу:

ПОСТА-ЧАЛЬНИКИ	СПОЖИВАЧІ				ЗАПАСИ
	Сімферополь	**Ужгород**	**Вінниця**	**Луганськ**	
Київ	$1 + a$	$7 - a$	$9 - a$	4	**120 + 20*a***
Львів	1	2	6	5	**280**
Одеса	3	8	$1 + a$	2	**160**
ПОТРЕБИ	**130 + 10*a***	**220**	**60 + 10*a***	**150**	

Для розв’язання типової транспортної задачі було створено наступна MATLAB-функція:

```
function [transfers] = transport(deliveryPrices, storage, demand)
    strgc = height(storage);

    dp = [ deliveryPrices, zeros(strgc, 1) ];
    dmnd = [ demand, (sum(storage) - sum(demand)) ];
    dpv = reshape(dp', 1, []);

    argc = width(dpv);
    dmndc = width(dmnd);

    f = @(x)(dpv * x);

    x0 = ones(argc, 1);
    Aeq = [];
    beq = [];

    for i = 1:strgc
        aeq = [ zeros(1, (i - 1) * dmndc), ones(1, dmndc), zeros(1, argc - i * dmndc) ];

        Aeq = [ Aeq; aeq ];
        beq = [ beq; storage(i) ];
    end

    for i = 1:dmndc
        aeq = [ zeros(strgc, i - 1), ones(strgc, 1), zeros(strgc, dmndc - i) ];

        Aeq = [ Aeq; reshape(aeq', 1, []) ];
        beq = [ beq; dmnd(i) ];
    end

    temp = fmincon(f, x0, [], [], Aeq, beq, zeros(argc, 1), Inf(argc, 1));
    transfers = reshape(temp, size(dp'))';
end
```

Вхідні дані задачі було передано у створену функцію та отримано наступний результат – таблицю перевезень товарів від постачальників до споживачів (останній стовпець результату позначає кількість продукту, що має лишитись на складі):

```
>> a = 0.9

a =

    0.9000

>> DP = [
1 + a, 7 - a, 9 - a, 4, 0;
1, 2, 6, 5, 0;
3, 8, 1 + a, 2, 0
]

DP =

    1.9000    6.1000    8.1000    4.0000         0
    1.0000    2.0000    6.0000    5.0000         0
    3.0000    8.0000    1.9000    2.0000         0

>> STRG = [120 + 20 * a; 280; 160]

STRG =

   138
   280
   160

>> DEMAND = [ 130 + 10 * a, 220, 60 + 10 * a, 150 ]

DEMAND =

   139   220    69   150

>> DEMAND = [ DEMAND (sum(STRG) - sum(DEMAND)) ]

DEMAND =

   139   220    69   150     0

>> TRANSFERS = transport(DP, STRG, DEMAND)

TRANSFERS =

   79.0000     0.0000     0.0000    59.0000     0.0000
   60.0000   220.0000     0.0000     0.0000     0.0000
    0.0000     0.0000    69.0000    91.0000     0.0000
```

4. Розв’язати наступну задачу оптимізації:

Технологічна операція	**Норми витрат часу на обробку 1 км кабелю (годин)**				**Загальний фонд робочого часу (годин)**
	Марка 1	**Марка 2**	**Марка 3**	**Марка 4**	
Волочіння	$1.2 + a$	5	1.6	2.4	7 200
Накладення ізоляції	1	4	8	3	5 600
Скручування елементів у кабель	6.4	5.6	$6 + a$	$8 - a$	11 170
Оливування	$30 - a$	$1.2 + a$	1.8	2.4	3 600
Випробування і контроль	2.1	1.5	4	$3 + a$	4 200
Прибуток від реалізації 1 км кабелю	1.2	0.8	$1 + a$	1.3	

Для розв’язання типової задачі оптимізації було створено наступну MATLAB-функцію:

```
function [benefits] = mostBenefit(timeSpend, timeLeft, profit)
    argc = width(profit);
    f = @(x)(-profit * x);
    x0 = zeros(argc, 1);
    benefits = fmincon(f, x0, timeSpend, timeLeft, [], [], zeros(argc, 1), Inf(argc, 1));
end
```

Вхідні дані задачі було передано до створеної функції та отримано наступний результат – кількість кілометрів кожної марки кабелю, яку треба виробити, аби мати найбільший прибуток:

```
>> a = 0.9;
>> BENEFIT = [ 1.2, 0.8, 1 + a, 1.3 ]

BENEFIT =

    1.2000    0.8000    1.9000    1.3000

>> HOURS_LEFT = [ 7200; 5600; 11170; 3600; 4200 ]

HOURS_LEFT =

        7200
        5600
       11170
        3600
        4200
```

```
>> TIME_SPEND = [
1.2 + a, 5, 1.6, 2.4;
1, 4, 8, 3;
6.4, 5.6, 6 + a, 8 - a;
30 - a, 1.2 + a, 1.8, 2.4;
2.1, 1.5, 4, 3 + a
]

TIME_SPEND =

    2.1000    5.0000    1.6000    2.4000
    1.0000    4.0000    8.0000    3.0000
    6.4000    5.6000    6.9000    7.1000
   29.1000    2.1000    1.8000    2.4000
    2.1000    1.5000    4.0000    3.9000
```

```
>> mostBenefit(TIME_SPEND, HOURS_LEFT, BENEFIT)
```

```
ans =

   48.1064
    0.0000
  487.2633
  551.2624
```

Висновки: в ході роботи було здобуто навички використання математичного середовища MATLAB для пошуку екстремумів функцій, розв'язання транспортних задач та задач. Досліджену поведінку математичного середовища для задач пошуку екстремумів функцій було відтворено за допомогою мови програмування C#.