Лабораторная работа №4

Численное решение нелинейных уравнений

Михалькевич Д.Н.
гр. 221701

Вариант 8

Задание 1.

Отделите графически корни алгебраического уравнения f (x) = 0 с помощью функции Plot. Найдите один из них (нецелый) с точностью ξ = 10^-3 методом хорд. Укажите потребовавшееся число итераций. Проиллюстрируйте графически нахождение первых двух приближений (постройте график функции и хорды).

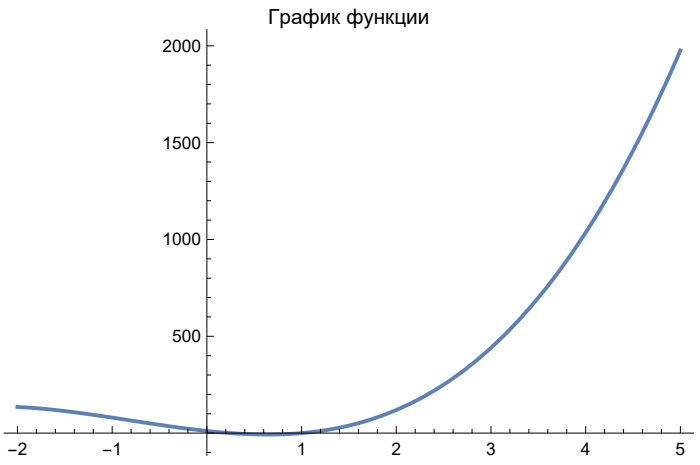
In[3546]:=

```
f[x_] := 12 x^3 + 29 x^2 - 52 x + 11;
ϵ = 10^-3;
a = 0.7;
b = -0.5;
```

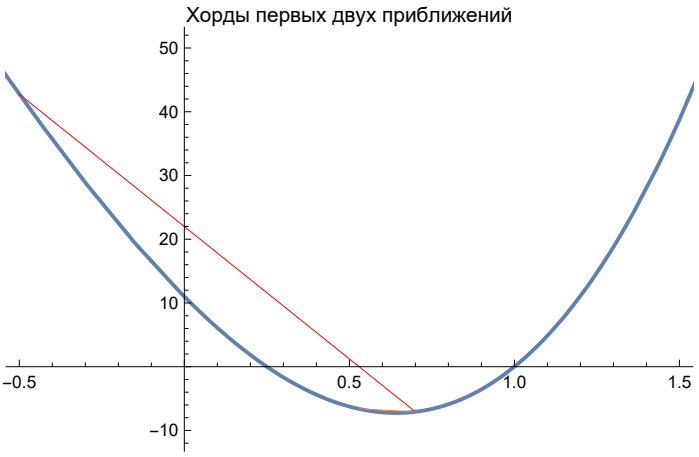
In[3550]:=

```
initialPlot = Plot[f[x], {x, -2, 5}, PlotLabel → "График функции"];
              график функции          пометка графика
ChordMethod[f_, a_, b_, eps_] := Module[{x0 = a, x1 = b, x2, n = 0},
                                       программный модуль
   While[Abs[f[x1]] > eps && Abs[x1 - x0] > eps, x2 = x1 - f[x1] (x1 - x0) / (f[x1] - f[x0]);
   цик··· абсолютное значение абсолютное значение
    x0 = x1; x1 = x2; n++;];
   {x1, n}];

{root, iterations} = ChordMethod[f, a, b, ϵ];

firstApproximation = Line[{{a, f[a]}, {b, f[b]}}];
                     (ломаная) линия
secondApproximationX = b - f[b] (b - a) / (f[b] - f[a]);
secondApproximation =
  Line[{{a, f[a]}, {secondApproximationX, f[secondApproximationX]}}];
  (ломаная) линия
Show[initialPlot]
показать
Show[initialPlot, Graphics[{Red, firstApproximation, Orange, secondApproximation}],
показать          графика    красный                 оранжевый
 PlotRange → {{-0.5, 1.5}, {-10, 50}}, PlotLabel → "Хорды первых двух приближений"]
 отображаемый диапазон графика          пометка графика

Print["Найденный корень: ", N[root, 4], "; Число итераций: ", iterations];
печатать                   численное приближение
```

Out[3556]=



Out[3557]=




```
Найденный корень: 0.249974; Число итераций: 6
```

Задание 2.

Отделите графически и найдите с помощью функций Solve, NSolve, Roots, FindRoot корни алгебраического уравнения f (x) = 0. Разложите многочлен f (x) на множители,используя функцию Factor.

In[3559]:=

```
Clear[f];
```

очистить

In[3560]:=

```
f := x^6 + x^5 - 31 x^4 - 13 x^3 + 306 x^2 - 864;
Plot[f, {x, -5, 4}]
```
график функции

```
Print["Корни Solve: ", N[Solve[f == 0, x], 1]]
```
печатать решить у··· решить уравнения

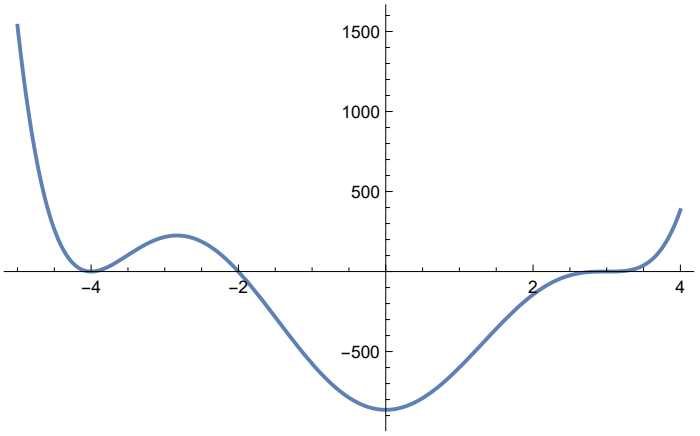
```
Print["Корни NSolve: ", N[NSolve[f == 0, x], 1]]
```
печатать численное··· численное решение уравнений

```
Print["Корни Roots: ", N[Roots[f == 0, x], 1]]
```
печатать корни мн··· корни многочлена

```
Print["Корень FindRoot: ", N[FindRoot[f == 0, {x, -5}], 1]]
```
печатать найти корень найти корень

```
Print["Разложение: ", Factor[f]]
```
печатать факторизовать

Out[3561]=



Корни Solve: {{x → -4.}, {x → -4.}, {x → -2.}, {x → 3.}, {x → 3.}, {x → 3.}}

Корни NSolve: {{x → -4.}, {x → -4.}, {x → -2.}, {x → 3.}, {x → 3.}, {x → 3.}}

Корни Roots: x == -4. || x == -4. || x == -2. || x == 3. || x == 3. || x == 3.

Корень FindRoot: {x → -4.}

Разложение: $(-3 + x)^3 (2 + x) (4 + x)^2$

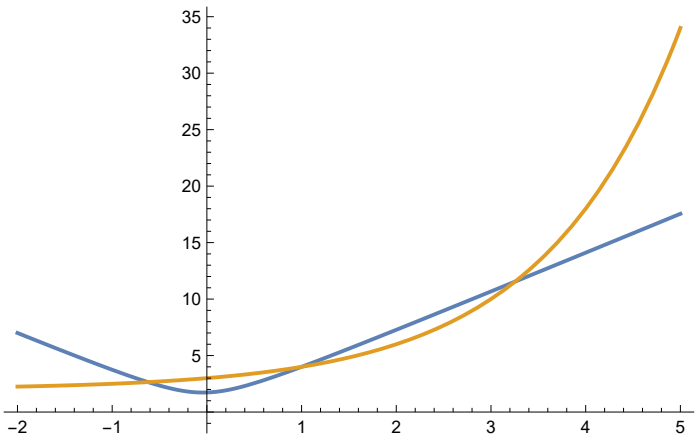
Задание 3.

Отделите графически корни трансцендентного уравнения с помощью функции Plot. Найдите один из них с точностью ξ=10^-3: а) методом Ньютона; б) методом секущих. Укажите потребовавшееся число итераций.

In[3567]:=

```
Clear[f];
очистить
f[x_] := √(12 x^2 + x + 3);
g[x_] := 2^x + 2
func[x_] := √(12 x^2 + x + 3) - 2^x - 2
initialPlot = Plot[{f[x], g[x]}, {x, -2, 5}];
                график функции
Show[initialPlot]
показать
```

Out[3572]=



Метод Ньютона

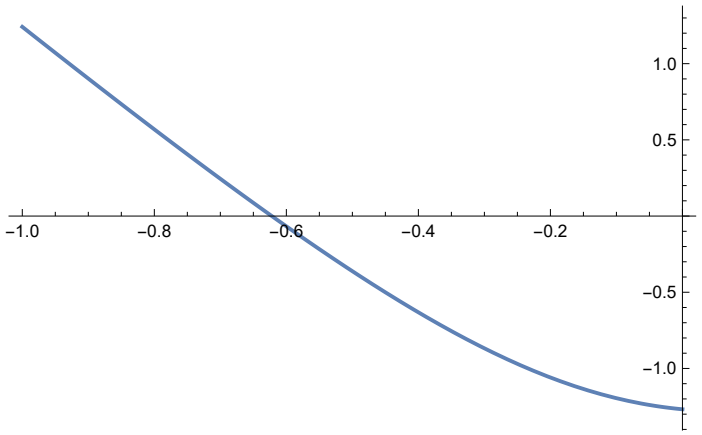
In[3573]:=

```
NewtF[f_, g_, func_, a_, b_, eps_] :=
    Module[{NewtonMethod},
    программный модуль
   NewtonMethod[x0_] :=
            Module[{root, iter = 0},
            программный модуль
            root = x0;
            While[Abs[func[root]] > eps && iter < 100,
            цик··· абсолютное значение
                root = root - func[root] / func'[root];
                iter++;
            ];
            Print[Plot[func[x], {x, a, b}]];
            печа··· график функции
            Print["Корень = ", root];
            печатать
            Print["Количество итераций = ", iter];
            печатать
        ];
   Print["Метод Ньютона:"];
   печатать
        NewtonMethod[a];
  ];
```

In[3574]:=

```
NewtF[f, g, func, -1.0, 0.0 , 10^(-3)]
NewtF[f, g, func, 0.5, 1.5 , 10^(-3)]
NewtF[f, g, func, 3.0, 4.0 , 10^(-3)]
```

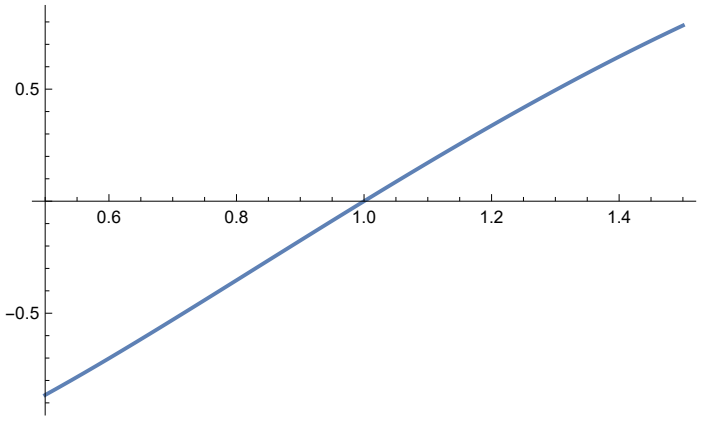
Метод Ньютона:



Корень = −0.622097

Количество итераций = 2

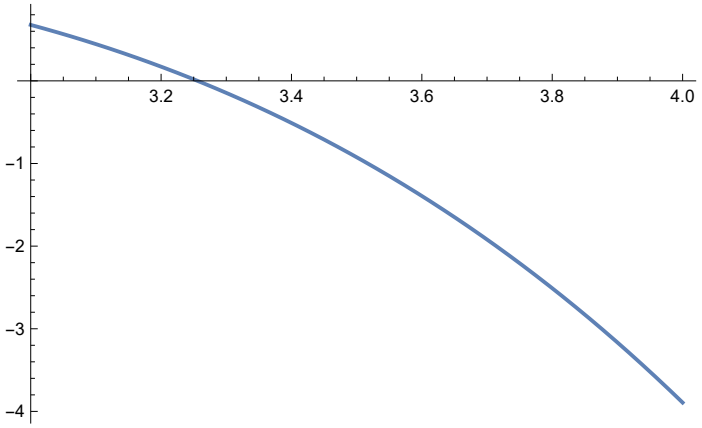
Метод Ньютона:



Корень = 0.999645

Количество итераций = 2

Метод Ньютона:



Корень = 3.25594

Количество итераций = 3

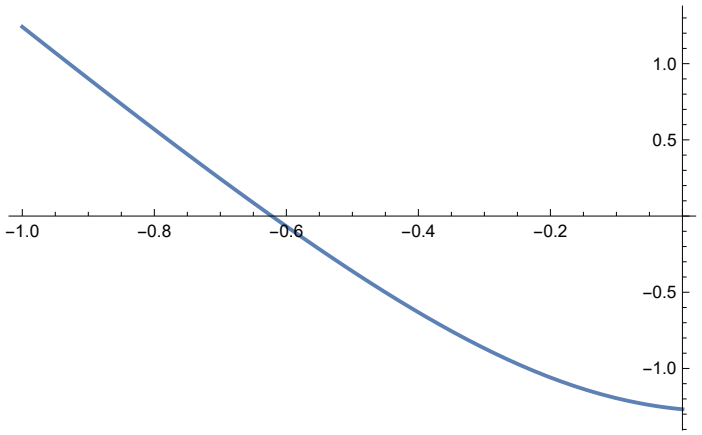
In[3577]:=

Метод Секущих

In[3578]:=

```
SecF[f_, g_, func_, a_, b_, eps_] :=
    Module[{SecantMethod},
    программный модуль
        SecantMethod[X0_, X1_] :=
            Module[{x0, x1, root, iter = 1},
            программный модуль
            x0 = X0; x1 = X1;
            root = x1 - func[x1] * (x1 - x0) / (func[x1] - func[x0]);
            While[Abs[func[root]] > eps && iter < 100,
            цик··· абсолютное значение
                x0 = x1;
                x1 = root;
                root = x1 - func[x1] * (x1 - x0) / (func[x1] - func[x0]);
                iter++;
            ];
            Print[Plot[func[x], {x, a, b}]];
            печа··· график функции
            Print["Корень = ", root];
            печатать
            Print["Количество итераций = ", iter];
            печатать
        ];
        Print["Метод секущих:"];
        печатать
        SecantMethod[a, b];

    ];
SecF[f, g, func, -1.0, 0.0 , 10^(-3)]
SecF[f, g, func, 0.5, 1.5 , 10^(-3)]
SecF[f, g, func, 3.0, 4.0 , 10^(-3)]
```

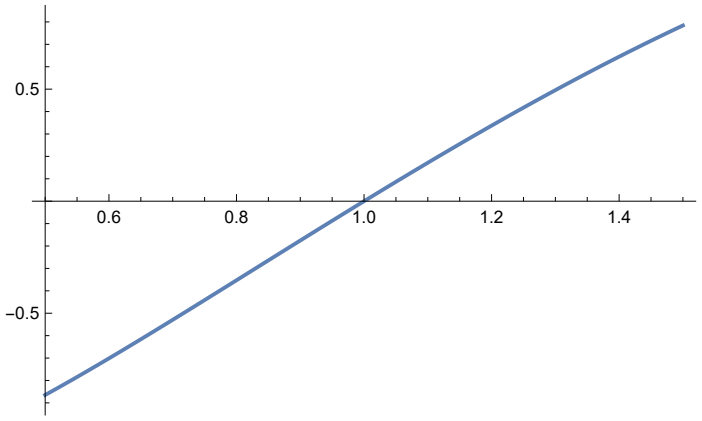
Метод секущих:



Корень = −0.621999

Количество итераций = 4

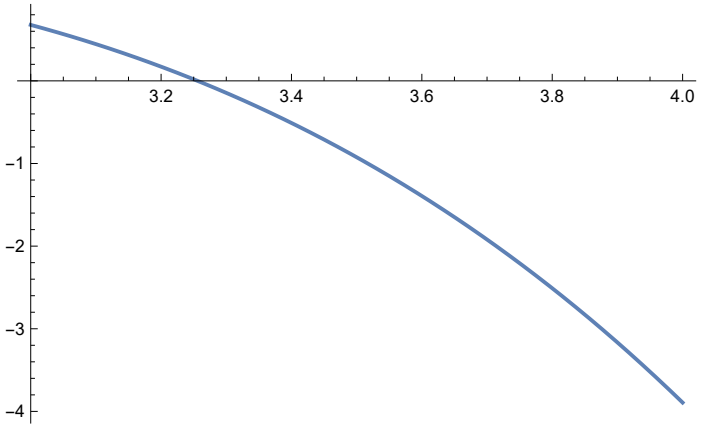
Метод секущих:



Корень = 1.00001

Количество итераций = 3

Метод секущих:



Корень = 3.25583

Количество итераций = 4

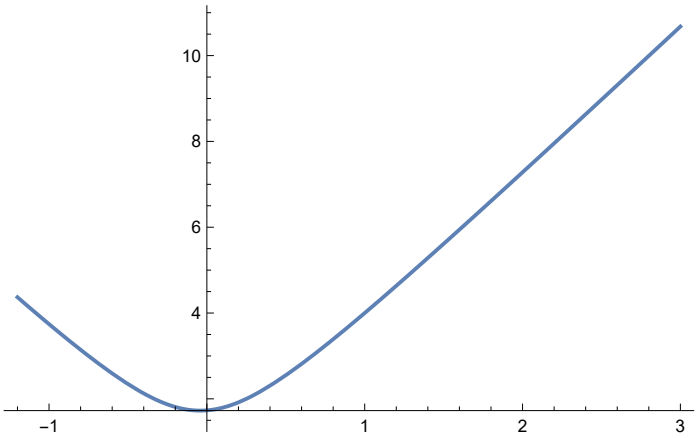
In[3582]:=

Задание 4.

Приведите уравнение к виду,пригодному для итераций. Найдите его корни методом простых итераций с точностью ξ=10^-3. Укажите потребовавшееся число итераций.

In[3583]:=
```
ClearAll
```
очистить всё
```
func[x_] = -12 x^2 - 3 + (2^x + 2)^2;
```
Out[3583]=
```
ClearAll
```

In[3585]:=
```
F4[func_, a_, b_, eps_] :=
    Module[{SimpleIterMethod},
    программный модуль
        Print[Plot[func[x], {x, a, b}]];
        печа··· график функции

        SimpleIterMethod[x0_] :=
            Module[{root, iter = 0},
            программный модуль
                root = x0;
                While[Abs[func[root]] > eps && iter < 100,
                цик··· абсолютное значение
                    root = func[root];
                    iter++
                    ];
                Print["Корень= ", root];
                печатать
            ];
        Print["Метод простых итераций:"];
        печатать
        SimpleIterMethod[-1.1];
        SimpleIterMethod[2.8];
    ];
F4[f, -1.2, 3.0, 10^(-3)];
```



Метод простых итераций:

Корень= 1.08857×10^{54}

Корень= 2.64442×10^{54}

Задание 5.

Решите уравнение с помощью функций Solve, NSolve, FindRoot.

In[3587]:=

```
solveRoots = Solve[f[x] == 0, x];
             решить уравнения
nSolveRoots = NSolve[f[x] == 0, x];
              численное решение уравнений
findRootRoots = FindRoot[f[x] == 0, {x, 0}];
                найти корень

Print["Уравнение решается только с помощью функции FindRoot: ", N[findRootRoots, 4]]
печатать                                                      найти корень  численное приближение
```

FindRoot: The line search decreased the step size to within tolerance specified by AccuracyGoal and PrecisionGoal but was unable to find a sufficient decrease in the merit function. You may need more than MachinePrecision digits of working precision to meet these tolerances.

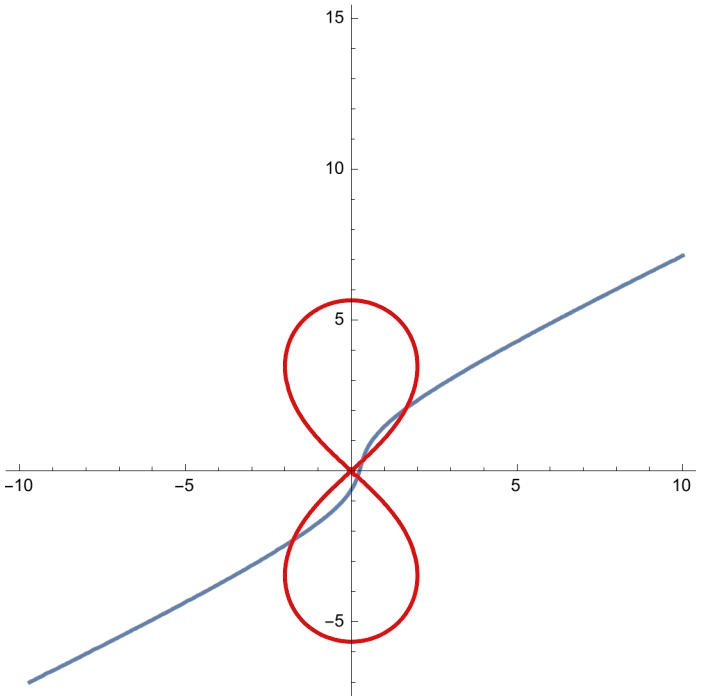
Уравнение решается только с помощью функции FindRoot: {x → −0.041668}

Задание 6

In[3591]:=

```
ClearAll;
очистить всё
f[x_, y_] = Sinh[2 y - x] - 3 x + 1 - y;
            гиперболический синус
g[x_, y_] = (x^2 + y^2)^2 - 32 (y^2 - x^2);

graph1 =
  ContourPlot[f[x, y] == 0, {x, -10, 10}, {y, -7, 15}, Axes → True, Frame → False];
  контурный график                                    оси  ист···  рамка  ложь
graph2 = ContourPlot[g[x, y] == 0, {x, -10, 10},
         контурный график
   {y, -7, 15}, Axes → True, Frame → False, ColorFunction → Hue];
                оси  ист···  рамка  ложь  функция оцвечи···  тон
Show[graph1, graph2]
показать

FindRoot[{f[x, y] == 0, g[x, y] == 0}, {x, -2}, {y, -3}]
найти корень
FindRoot[{f[x, y] == 0, g[x, y] == 0}, {x, 0}, {y, 0}]
найти корень
FindRoot[{f[x, y] == 0, g[x, y] == 0}, {x, 0.5}, {y, 0.5}]
найти корень
FindRoot[{f[x, y] == 0, g[x, y] == 0}, {x, 2}, {y, 2}]
найти корень
```

Out[3596]=



Out[3597]=

```
{x → -1.76022, y → -2.30332}
```

Out[3598]=

$$\{x \to 0.193271,\ y \to -0.193723\}$$

Out[3599]=

$$\{x \to 0.336354,\ y \to 0.338757\}$$

Out[3600]=

$$\{x \to 1.66188,\ y \to 2.08239\}$$