#### Тема 5.2. Технология решения нелинейных уравнений средствами MatLab

Для решения систем алгебраических уравнений и одиночных урав­нений служит функция solve:

* **solve(expr1, expr2,... exprN, var1, var2,... varN)**- возвращает значения переменных **var1**, при которых соблюдаются равенства, заданные выражениями **exprI**. Если в выражениях не используются знаки равенства, то полагается **ехргI = 0**;
* **solve(expr1, expr2, ..... exprN)***-*аналогична предшествующей функ­ции, но переменные, по которым ищется решение, определяются функцией **findsym**.

**Пример 4.2-1. Решить несколько нелинейных уравнений.**

|  |
| --- |
| **Пример4.2-1** |
| **» syms x у;**  **» so1ve(x^3 -1, x)**  **ans =**  **[ 1]**  **[ -1/2+1/2\*i\*3^(1/2)]**  **[ -1/2-1/2\*i\*3^(1/2)]**  **» syms a b с**  **» solve(a\*x^2+b\*x+c)**  **ans =**  **[ 1/2/a\*(-b+(b^2-4\*a\*c)^(1/2))]**  **[ 1/2/a\*(-b-(b^2-4\*a\*c)^(1/2))]**  **»S = solve(**'**x+y=3**'**,** '**x\*y^2=4**'**, x, y)**  **S =**  **x : [3x1 sym]**  **y : [3x1 sym]**  **» S.x**  **ans =**  **[4]**  **[1]**  **[1]**  **» S.y**  **ans =**  **[-1]**  **[2]**  **[2]**  **» solve(**'**sin(x)=0.5**'**, x)**  **ans =**  **0.52359877559829887307710723054658**  **>>** |

**Пример4.2-2.  Отделить корни уравнения  с непрерывной правой частью.**



Найдем отрезок, на концах которого имеет разные знаки. Т.е. решим уравнение   .



|  |
| --- |
| **Пример 4.2-2.** |
| syms x  >> f=sym('2^x-4\*x');  >> ezplot(f,-6,6)  >> grid on  >> |



В математическом пакете MatLab имеется также ряд встроенных функций для численного вычисления корней уравнений.

Рассмотрим программные средства MatLabна примерах**.**

**Пример 4.2-3. Локализовать корни уравнения f(x)=x3–cos(x)+1.**

|  |
| --- |
| **Пример 3.4-20** |
| **>>f = inline('x.^3 - cos(x) + 1');**  **>>x = linspace(-10,10,100); >>figure('Name', '[-10,10]'); >>axes('NextPlot', 'Add'); >>grid on >>plot(x, f(x));** |
| **>>figure('Name', '[-1,1]'); >>axes('NextPlot', 'Add'); >>grid on >>x = linspace(-1,1,100);**  **>>plot(x, f(x));**  **>>figure('Name', '[-0.6,-0.4]'); >>axes('NextPlot', 'Add'); >>grid on >>x = linspace(-0.6,-0.4,100); >>plot(x, f(x));**  **>>figure('Name', '[-0.2,0.2]'); >>axes('NextPlot', 'Add'); >>grid on >>x = linspace(-0.2,0.2,100); >>plot(x, f(x));**  **>>** |

Решение алгебраических и трансцендентных уравнений в среде MatLAB проще реализовать с помощью встроенных функций: solve(), fzero(), roors().

Для нахождения вещественных корней уравнений вида f(х)=0 используется функция fzero(). Алго­ритм, реализованный этой функцией, представляет собой **комбинацию метода дихотомии (деления пополам), метода секущих и метода обрат­ной квадратичной интерполяции**. В простейшем варианте обращения кроме указателя на функцию, корень которой ищется, задается окрестность х0, с которой начинается поиск: х = fzero(f, x0).

Аргумент fможет быть задан одним из способов:

* формойс неизвестным х, заключенная в апострофы;
* именем m-файла (в апострофах и без расширения m);
* указателем на функцию (например, @f\_name);
* указателем на анонимную функцию (например, f\_handie).

При этом формула, заключенная в апострофы, в качестве независимой переменной мо­жет содержать только х. Использование независимой переменной с другим именем вызовет сообщение об ошибке.

Аргумент х0 может быть задан одним из двух способов:

* вектором [a;b], представляющим интервал (а<b),на концах которого функция f()меняет знак, что гарантирует нахождение, по крайней мере, одного корня на этом интервале;
* скалярным значением, в окрестности которого предполагается нахождение корня. В этом случае функция fzero() сама пытается найти отрезок с центром в заданной точке х0, на концах которого функцияf ()меняет знак.

Чтобы облегчить работу по выбору начального приближения, разумнее всего построить график функции y=f (x).

**Пример 4.2-4. Построить график функции f(x)=x∙e-x+sin(x) для локализации корня.**

|  |
| --- |
| **Пример 4.2-4.** |
| **x = 0 : 0.1 : 2\* pi;**  **y = x.\*exp(-x)+sin(x);**  **plot(x, y)**  **grid on**  **title('y=x\*exp(-x)+sin(x) ')** |

Из графика видно, что один из корней находится на интервале [3;4]. Используем полученную информациюи обратимся к функ­ции fzero( ):

|  |
| --- |
| **Пример 4.2-4** |
| >> x=fzero('x.\*exp(-x)+sin(x)',[3,4])  x=  3.2665  >> |

Вместо явного задания формулы для функции f мы могли бы объявить соответствующую функцию, запомнив ее в виде автономного m-файла или включив ее в качестве подфункции в файл нашей программы.

|  |
| --- |
| **Пример 4.2-4** |
| **function fzerol**  **x=fzero(@f1, [3;4])**  **function y=f1(z)**  **y= z\*exp(-z)+sin(z);** |

Если мы хотим получить не только значение корня, но и узнать значение функции в найденной точке, то к функции fzero( ) можно обратиться с двумя выходными параметрами.

В ряде задач такая точность может оказаться излишней. MatLab предо­ставляет пользователю возможность формировать различные условия прекращения итерационного процесса - по точности вычисления координаты х, по модулю значения функции f(), по количеству обращений к функции f() и т. д.

**Пример 4.2-5. Найти решения tg(x)=0 на интервале[1;2].**

|  |
| --- |
| **Пример 4.2-5** |
| **>> [x,f]=fzero('tan(x) ', [1,2])**  **х =**  **1.5708**  **f =**  **-1.2093e+015**  **>>** |

Якобы «корень», соответствующий приближенному значению /2, на самом деле является точкой разрыва, при переходе через которую функция меняет знак. Выведенное значение функции в найденной точке убеждает нас в том, что найден не корень.

Функция fzero() может возвратить еще два выходных параметра.

|  |
| --- |
| **Пример 4.2-5** |
| **>> [x,f,e\_flag,inform] = fzero{f,x0)**  **>>** |

Положительное значение e\_fiag (обычно, это 1) означает, что удалось най­ти интервал, на концах которого функция f( ) меняет знак (пример с tg(x)не должен притупить вашу бдительность). Если такой интервал не обнару­жен, то e\_fiag=-1. Структура inform содержит три поля с именами iterations, funcCountи algorithm. В первом из них находится количество итераций, выполненных при поиске корня, во втором – количество обращений к функции f( ), в третьем – наименование алгоритма, использован­ного для нахождения корня.

**Пример 4.2-6. Найти корень уравнения с помощью функции fzero().**

|  |
| --- |
| **Пример 4.2-6** |
| **» [x,f,e\_flag,inform]=fzero('x.\*exp(-)+sin(x)',[3,4])**  **x = 3.2665**  **f= 2.0817e-016**  **e\_flag =**  **inform =**  **iterations: 8**  **funcCount: 8**  **algorithm: 'bisection, interpolation'**  **>>** |

В данном случае достижение высокой точности по­требовалось 8 итераций. Простое деление отрезка пополам для дости­жения такой же точности потребовало бы больше итераций.

Отделение отрезка, на концах которого функция принимает значения разных знаков, является принципиальным для алгоритма, исполь­зованного в функции **fzero().** Даже в таких тривиальных уравнениях, как х2=0, обычно не удается найти решение.

Для символьного (аналитического) решения уравнений в MatLab используется функция solve(),которая представляется в следующем виде:solve('f(x)',x),solve('f(x)'),где: 'f(x)'– решаемое уравнение, записанное в одиночных кавычках и представленное в произвольной форме; x– искомая символьная неизвестная (symsx).

Рассмотрим технологию определения корня с помощью функции solve() на примерах.

**Пример 4.2-7.Найти решение уравнения2x–3(a–b)=0 в символьном виде.**

|  |
| --- |
| **Пример 4.2-7** |
| **>>syms x**  **>> y=solve(‘2^x-3\*(a-b)=0’)**  **y =**  **log((3\*a-3\*b)/log(2))**  **>>** |

**Пример 4.2-8.Решить уравнение 2x–4∙x+3=0 аналитически.**

|  |
| --- |
| **Пример 4.2-8** |
| **>>syms x**  **>> y=solve(‘2^x-4\*x+3=0’)**  **y =**  **1.418**  **3.413**  **>>** |

Функция в ряде случаев позволяет определить все корни уравнения f(x)=0 без указания начальных значений **x** или областей изоляции корней.

Функция solve() имеет следующий недостаток. Она не требует информации о начальном значении корня или области его изоляции. Поэтому в случае трансцендентных уравнений и в ряде других случаев, она не находит всех корней уравнения.

**Пример 4.2-9.Решить уравнение 2x–3(a–b)=0, где a – независимая переменная.**

|  |
| --- |
| **Пример 4.2-9** |
| **>>syms a**  **>> y=solve(‘2^x-3\*(a-b)=0’, a)**  **y =**  **1/3\*2^x+**  **>>** |