### *Тема 2.4. Использование математического пакета MathCad для аналитических и численных решений*

#### 2.4.1. Погрешность

#### 2.4.2. Технология решения нелинейных уравнений средствами MathCad

В математическом пакете MathCad имеются как программные средства для реализации алгоритмов уточнения корней уравнений, так и встроенные функции для численного и аналитического вычисления корней уравнений.

Рассмотрим примеры , иллюстрирующие средства **MathCad.**

**Пример 2.4-1.Отделить корни уравненияx3-cos(x)+1=0 графическим методом.**

|  |
| --- |
| **Проведем анализ функции**  **1) Область допустимыхзначениий 2) Сократим интервал**  **достаточно большая**  **3) Получим два отрезка локализации:**  **Простой корень на отрезке [-0.6;-0.4] и кратный корень на отрезке [-0.2;0.2]** |

**Пример 2.4-2.Отделить корень уравненияf(x)=1–3x+cos(x)=0 аналитически**.

|  |
| --- |
| **Первая и вторая производные на [0;1] непрерывны и знакопостоянны**  **a=0 b=1**  **Уравнение 1-3x+cos(x)=0 имеет на отрезке [0;1] один корень** |

**Пример 2.4-3**.**Выполнить «ручным расчетом» три итерации нахождения корня уравнения f(x)= 1 – 3х + cos(x) = 0 методом половинного деления.**

|  |
| --- |
| >0следовательно  <0следовательно  <0следовательно |

**Пример 2.4-4.Уточнить корень уравнения f(x)=1 – 3x + cos(x)=0 методом итерации на отрезке [0;1].**

Приведем уравнение1 – 3х + cos(x) = 0к видуx = (cos(x)+1)/3и проведем исследование:

|  |
| --- |
| **для всех значений аргумента х на отрезке [0;1]** |

**Пример 2.4-5.Привести уравнение x2–3∙x+3.25–5∙cos(x)=0 к виду, удобному для итерации.**

|  |
| --- |
| Будем искать простой корень уравнения, находящийся на отрезке локализации [-0.4;0]  Найдем корень с помощью встроенной функции root  **1 способ**.Приведем уравнение к виду x=ϕ(x) , где  Проверим условие сходимости:  График призводной  Максимальное по модулю значение производной итерационной функции достигается в левом конце отрезка  ϕ(x)=x-λf(x), где λ - итерационный параметр  Выполним 3 итерации по расчетной формуле x=ϕ(x)  1-я итерация:  2-я итерация:  3-яитерация:  Погрешность найденного значения корня:  **2 способ.** Приведем уравнение к виду x=x-λf(x), где итерирующая функция ϕ(x)=x- λf(x), а  λ - итерационный параметр. λ выбирем из условия λ=2/(m+M), где m - минимальное, а  М - максисальное значения f'(x) на отрезке [-0.4,0]  1-я итерация:  2-я итерация:  3-я итерация:  Погрешность найденного значения корня: |

**Пример 2.4-6.Выполнить «ручным расчетом» три итерации, решая уравнение f(x)=1 – 3x + cos(x)=0 методом Ньютона.**

В нашем случае



|  |
| --- |
|  |

ВMathcad имеется ряд встроенных средств для поиска корней нелинейных уравнений. Функция

root(f(var1, var2, ...),var1, [a, b])

имеет два необязательных аргумента **a** и **b**, которые определяют границы интервала, на котором следует искать корень. На концах интервала [a;b]функция f должна менять знак (f(a)f(b)<0)**.** Задавать начальное приближение для корня не нужно. Функция rootиспользует алгоритм Риддера (в основу которого положен метод хорд) и Брента. Метод Брента соединяет быстроту метода Риддера и гарантированную сходимость метода деления отрезка пополам.

**Пример 2.4-7. Определить корни уравнения , используя расширенный поиск.**



Для оценки местоположения корней построим график этой функции

|  |
| --- |
|  |

**Пример 2.4-8.Отделить корень уравнения 1–3x+Co(x)=0, а затем с помощью встроенной функции root( ) найти его значение с точностью TOL = 0.001.**

Значение переменная TOL принимает по умолчанию. Если требуется изменить точность вычислений, то переменную TOL следует переопределить, например, следующим образом TOL:=0.00001. В данном примере, поскольку параметры a и b не заданы, то функция **root** возвращает первый вычисленный корень.

|  |
| --- |
|  |

Если уравнение имеет несколько корней, то для их нахождения можно использовать разложение функции **f(x)** на простые множители f(x)=(x-x1)(x-x2) …(x-xn), где x1, x2, …, xn - корни уравнения. Начальное приближение можно задать только для первого корня, а в качестве функции взять, например,



Если уравнение не имеет действительных корней, то есть на графике функция f(x) нигде не равна нулю, то для вывода комплексных корней надо ввести начальное значение приближения к корню в комплексной форме, где для вывода мнимой части использовать символы 1iи 1j**.**

**Пример 2.4-9. Найти решения нелинейного уравнения , имеющего несколько корней, часть из которых мнимые.**



|  |
| --- |
|  |

#### 2.4.3. Технология интерполяции функций в среде системы MathCad

Для решения задач интерполяции в **Mathcad**имеются встроенные функции двух видов: позволяющие увидеть аналитическую зависимость, то есть возвращающие набор аппроксимирующих коэффициентов, и не позволяющие увидеть аналитическую зависимость, а позволяющие только получить значения функции в промежуточных точках. Кроме того, в**Mathcad** имеется несколько функций интерполяции, различающихся способом «соединения» точек (прямой линией или различными кривыми).

Рассмотрим средства интерполяции в системе **Mathcad**на примерах.

**Пример 2.4-10. Пусть значения функции, полученные в ходе эксперимента, представлены в виде таблицы:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **X** | 1.2 | 1.4 | 1.6 | 1.8 | 2.0 |
| **y(x)** | -0.085 | -0.462 | 0.128 | 3.546 | 2.654 |

**Выполнить линейную интерполяцию данных (экспериментальные точки соединяются отрезками прямой) с использованием функции linterp(x, y, t), где x – вектор значений аргументов, y – вектор значений функции и t – текущее значение аргумента, при котором вычисляется функция.**

|  |  |
| --- | --- |
| |  | | --- | |  | |

**Пример 2.4-11. Выполнить интерполяцию таблично заданной функции по методу Лагранжа.**

Преимущество этой формулы в том, что число и расположение узловых точек могут быть любыми (в том числе неравномерными).

|  |
| --- |
|  |

Помимо вычисления значений функций в пределах интервала данных все рассмотренные ранее функции могут осуществлять **экстраполяцию** (прогнозирование поведения функции за пределами интервала заданных точек) с помощью зависимости, основанной на анализе расположения нескольких исходных точек на границе интервала данных. В **Mathcad** имеется и специальная **функция предсказания**predict(Y, m, n), где Y – вектор заданных значений функции, обязательно взятых через равные интервалы аргумента, а **m** – число последовательных значений Y, на основании которых функция predict возвращает nзначений Y**.**

Значений аргумента для данных не требуется, поскольку по определению функция действует на данных, идущих друг за другом с одинаковым шагом. Функция использует линейный алгоритм предсказания, который точен, когда экстраполируемая функция гладкая. Функция может быть полезна, когда требуется экстраполировать данные на небольшие расстояния. Вдали от исходных данных результат чаще всего оказывается неудовлетворительным.

**Пример 2.4-12. Задан массив из 60 точек .**



**На основании первых m точек провести экстраполяцию (предсказание) значений n точек.**

|  |
| --- |
|  |

**MathCad**имеется трисплайн-функции:

* cspline( )
* pspline( )
* lspline( )

Эти функции возвращают вектор коэффициентов вторых производных, который мы будем называть S. Этот вектор обычно используется в функции interp( )**,** описанной ниже. Аргументы должны быть вещественными векторами одинаковой длины. Значения вектора должны быть расположены в порядке возрастания.

Эти три функции отличаются только граничными условиями:

* функция lspline( ) генерирует кривую сплайна, которая приближается к прямой линии в граничных точках;
* функция pspline( ) генерирует кривую сплайна, которая приближается к параболе в граничных точках.
* функция cspline( ) генерирует кривую сплайна, которая может быть кубическим полиномом в граничных точках.
* interpвозвращает интерполируемое значение, соответствующее аргументу.

Вектор вычисляется на основе векторов данных и одной из функций pspline( ), lspline( )илиcspline( )*.*

**Пример 2.4-13. Пусть значения функции, полученные в ходе эксперимента, представлены в виде таблицы:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **X** | 1.2 | 1.4 | 1.6 | 1.8 | 2.0 |
| **y(x)** | -0.085 | -0.462 | 0.128 | 3.546 | 2.654 |

**Применить кубическую сплайн-интерполяцию, при которой экспериментальные точки соединяются отрезками кубических полиномов.**

Для этого одновременно используются две функции: interp(s,x,y,t**)**и cspline(x,y), где x– вектор значений аргументов, y – вектор значений функции, s – вектор вторых производных, создаваемый функцией cspline, t – значение аргумента, при котором вычисляется функция.

|  |
| --- |
|  |

#### 2.4.4. Технология вычисления интегралов в среде системы MathCad

Способ вычисления определенного интеграла с использованием системы **Mathcad** впрямую зависит от способа задания подынтегральной функции. Если подынтегральная функция задана в аналитическом виде, то интеграл, также как и производные, может быть вычислен с использованием встроенного в систему оператора, шаблон которого также расположен на палитре **Исчисления**. При этом выражение для подынтегральной функции может быть или предварительно описано в виде функции, или непосредственно введено в формат интеграла.

**Пример 2.4-14. Найтиопределенный интеграл в символьном виде и вычислить его значение.**

|  |
| --- |
|  |

Однако нередко возникает необходимость вычисления определенного интеграла для таблично заданной функции. Тогда прямое применение встроенного в систему оператора вычисления интеграла оказывается невозможным.

**Пример 2.4-15.Вычислить определенные интегралы методами трапеций (It) и парабол (Симпсона) (Ic).**

|  |
| --- |
| Метод трапеций  Метод Симпсона |

**Пример 2.4-16.Вычислить значения определенного интеграла методом средних прямоугольников при условии, что подынтегральная функция задана аналитически.**

|  |
| --- |
| Метод средних прямоугольников |

С помощью средств **Mathcad** могут быть найдены символьные выражения для производных и интегралов. Символьный знак равенства (**стрелка**) расположен на палитре **Символика** панели **Математика**. Для получения производных и интегралов в символьных выражениях в шаблон производной или интеграла нужно ввести выражение, щелкнуть по изображению символьного знака равенства, а затем щелкнуть по свободному пространству рабочего поля экрана.

**Пример 2.4-17. Найти символьные выражениядля производных и интегралов.**

|  |
| --- |
|  |

**Пример 2.4-18. Вычислить определенный интеграл от заданной функции  с различными значениями точностей.**

|  |
| --- |
|  |

**Пример 2.4-19. Вычислить значения определенного интеграла с шагом и ( и ) по формуле средних прямоугольникови оценить погрешность.**



|  |
| --- |
| i:=0…n-1 |

**Пример 2.4-20. Вычислить значения определенного интеграла с шагом и ( и ) по формуле трапеций.**



|  |
| --- |
|  |

**Пример 2.4-21. Вычислить значения определенного интеграла с шагом и ( и ) по формуле Симпсона.**



|  |
| --- |
|  |

#### 2.4.5. Технология решения дифференциальных уравнений средствами MathCad

При решении **ОДУ** его следует привести к нормальной форме (к виду разрешенному относительно производнойисходного **ОДУ**)



Для**ОДУ** с разделяющимися переменными исходное уравнение можно привести к виду , тогда выражение задает решение задачи Коши с начальными условиями как функцию y от переменной х.



**Пример 2.4-22. Решить ОДУ вида .**



Найдем частное решение данного **ОДУ** с использованием средств **Mathcad**, сначала методом разделения переменных, а затем с использованием функции odesolve(x, xk, n), где х – имя переменной, относительно которой решается уравнение, xk – конец интервала интегрирования, n – количество шагов, на которых вычисляется решениеОДУ. Результаты подтверждают правильность преобразований.

|  |
| --- |
| произвольная постоянная  Аналитическое решение ОДУ  Численное решение ОДУ |

Аналитическое выражение для решений **ОДУ** удается получить достаточно редко, поэтому широкое распространение при решении **ОДУ** получили численные методы.

**Пример 2.4-23. Решить ОДУ на отрезке [0;3] методами Рунге-Кутты с постоянным шагом h=0,6.**



В приведенном ниже документе решение, полученное методом Эйлера, обозначено как y1, методом Рунге-Кутты 2-го порядка – y2, а 4-го порядка – y4.

|  |
| --- |
| **Метод Эйлера**  **Метод Рунге-Кутты 2-го порядка**  **Метод Рунге-Кутты 4-го порядка** |

В **Mathcad** нет средств символьного решения **ОДУ,** но достаточно широко представлены методы численного решения задачи Коши. Для этого предназначена, например, функция rkfixed(y, x0, xend, N, D), где y– первоначально равно y0, x0 и xend– начальное и конечное значения аргумента, N – количество проводимых вычислений решения, а переменной D(x,y) должно быть присвоено выражение для вычисления правой части уравнения. Результатом вычислений функции rkfixe( ) служит матрица, в первом столбце которой содержатся координаты узлов x0 … xend, а во втором – значения приближенного решения в соответствующих узлах. В функции rkfixed( ) вместо метода Рунге-Кутты используется метод Булирша-Штера. Ниже приведены решения и их графическая иллюстрация, полученные с шагом 0.6 и 0.15.

**Пример 2.4-24. Решить ОДУ у’= на отрезке [0;3] методами Рунге-Кутты с постоянным шагом h=0.6.**



|  |
| --- |
|  |

Решение ОДУ 2-го порядка вида у”=F(x, y, z), где z=y’ также может быть получено методом Рунге-Кутты 4-го порядка. Ниже приведены формулы для решения ОДУ:

|  |
| --- |
|  |

Система Mathcad имеет специальную встроенную функцию для решения дифференциальных уравнений. Она имеет вид:Odesolve( x , b [ , steps ] ).

Для решения задачи Коши необходимы так называемые начальные условия и указание конца интервала. Эти данные вместе с самим уравнением записываются в блок функции Given, и лишь затем применяется сама функция odesolve( ). Функция имеет ряд особенностей. Если указано число шагов step, то решение выполняется с фиксированным шагом, иначе - адаптивным методом.

**Пример 2.4-25. Решить дифференциальное уравнение.**

|  |
| --- |
| Задано дифференциальное уравнение  Заданы начальные условия  Задано решение дифференциального уравнения  Вычисление производной от b(a)  График решения заданного дифференциального уравнения b(a) и производной от функции решения -c(a) |

#### 2.4.6. Технология решения задач одномерной оптимизации средствами MathCad

Пакет Mathcad с помощью встроенных функций решает задачу нахождения только локального экстремума. Для нахождения глобального экстремума необходимо вычислить все локальные экстремумы и выбрать среди них наибольший (наименьший). Отметим несколько подходов в поиске экстремума.

Для непрерывной функции от одной переменной можно использовать равенство нулю её производной, и путем решения полученного уравнения получить точки экстремумов. Уравнение можно решить с использованием встроенной функции **root**. При этом следует принимать во внимание знак второй производной. Если на отрезке, содержащем точку экстремума,, то это локальный минимум, а если , то это локальный максимум.



**Пример 2.4-26. Найти глобальный минимум функции .**



|  |
| --- |
|  |

Дальнейшее исследование показало, что глобальным минимумом является точка х = -3.679.

Для непрерывных функций также удобно пользоваться такими встроенными функциями как **Maximize(y,x)** и **Minimize(y,x).** Здесь ключевое слово **Given** можно опускать, поскольку оно необходимо лишь при наличии ограничений.

**Пример 2.4-27. Найти минимум и максимум функции y(x)=2x3-16x+5.**

|  |
| --- |
|  |

Для ступенчатой функции или функции с переломами можно использовать встроенную функцию **Minеrr( )**. Предварительно по графику выбирается число, заведомо большее (или меньшее) экстремального значения функции, и записывается в качестве ограничения в блоке **Given**. Функция **Minеrr( )** возвращает значение аргумента, при котором расхождение между заданным числом и значением функции минимально.

**Пример 2.4-28. Найти минимум и максимум ступенчатой функции.**

|  |
| --- |
|  |

**Пример 2.4-29. Найти минимум функции одной переменной.**

|  |
| --- |
| Пример поиска минимума функции одной переменной |

#### 2.4.7. Технология решения задач аппроксимации функций средствами MathCad

В Mathcad имеется несколько функций аппроксимации, различающихся способом «соединения» точек (прямой линией или различными кривыми).

**Пример 2.4-30. Получить аппроксимирующие полиномы первой и второй степени методом наименьших квадратов для функции, заданной таблично.**

|  |
| --- |
|  |

**Пример 2.4-31. Осуществить аппроксимацию таблично заданной функции многочленом 1-й, 2-й и 3-й степени.**

В этом примере рассмотрено использование функции linfit(x,y,f), где x,y- соответственно векторы значений аргументов и функции, а f – символьный вектор базисных функций. Использование этой функции позволяет определить вектор коэффициентов аппроксимации методом наименьших квадратов и далее невязку - среднеквадратическую погрешность приближения исходных точек к аппроксимирующей функции (сkо). Степень аппроксимирующего многочлена задается при описании символьного вектора f. В примере представлена аппроксимация таблично заданной функции многочленом 1-й, 2-й и 3-й степени. Вектор s представляет собой набор аппроксимирующих коэффициентов, что позволяет получить аппроксимирующую функцию в явном виде.

|  |
| --- |
|  |

В **Mathcad** имеется также большое количество встроенных функций, предназначенных для получения аналитического выражения функции регрессии. Однако в этом случае необходимо знать форму аналитического выражения. Ниже приведены встроенные функции, различающиеся видом регрессии, позволяющие (при заданных начальных приближениях) определить аналитическую зависимость функции, то есть возвращающие набор аппроксимирующих коэффициентов:

expfit(X,Y,g)Решение ОДУ 2-го порядка вида у”=F(x, y, z), где z=y’ также может быть получено методом Рунге-Кутты 4-го порядка. Ниже приведены формулы для решения ОДУ:

* регрессия экспонентой



* sinfit(X,Y,g)– регрессия синусоидой



* pwfit(X,Y,g)– регрессия степенной зависимостью



* logfit(X,Y,g)– регрессия логарифмической функцией



В этих функциях: х – вектор аргументов, элементы которого расположены в порядке возрастания; y– вектор значений функции; g – вектор начальных приближений коэффициентов a, b и с; t - значение аргумента, при котором определяется функция.

В приведенных ниже примерах для оценки связи между массивами данных и значениями аппроксимирующей функции подсчитывается коэффициент корреляции corr(). Если табличные данные неплохо аппроксимируется каким-либо видом регрессии, то коэффициент корреляции близок к единице. Чем меньше коэффициент, тем хуже связь между значениями этих функций.

**Пример 2.4-32. Найти аппроксимирующие полиномы первой, второй, третьей и четвертой степени и вычислить коэффициенты корреляции.**

|  |
| --- |
|  |

Помимо вычисления значений функций в пределах интервала данных все рассмотренные ранее функции могут осуществлять **экстраполяцию** (прогнозирование поведения функции за пределами интервала заданных точек) с помощью зависимости, основанной на анализе расположения нескольких исходных точек на границе интервала данных. В **Mathcad** имеется и специальная **функция** предсказания predict(Y, m, n), где Y – вектор заданных значений функции, обязательно взятых через равные интервалы аргумента, а m – число последовательных значений Y, на основании которых функция predict возвращает n значений Y.

Значений аргумента для данных не требуется, поскольку по определению функция действует на данных, идущих друг за другом с одинаковым шагом. Функция использует линейный алгоритм предсказания, который точен, когда экстраполируемая функция гладкая. Функция может быть полезна, когда требуется экстраполировать данные на небольшие расстояния. Вдали от исходных данных результат чаще всего оказывается неудовлетворительным.

#### 2.4.8 Технология решения задач многомерной оптимизации средствами MathCad

Нахождение экстремумов **функции нескольких переменных** проводится аналогично функции одной переменной. Для этого используются функции Maximize(f, y, x) и Minimize(f, y, x), где f – имя функции, а y и x – имена переменных. При использовании функций Мinerr(x,y) или Find(x, y) находятся значения x и y, являющиеся решением системы уравнений, составленной из частных производных исходной функции по x и y. При этом следует помнить, что функция Find дает точное решение, а Мinerr - приближенное. Ниже приведены примеры поиска значения экстремума двумерной функции с использованием функции Minimize(f, y, x), график двумерной функции и график линий уровней.

**Пример 2.4-33. Решить задачу оптимизации аналитическим методом для функции .**

|  |
| --- |
|  |

**Пример 2.4-34. Решить задачу оптимизации для функции двух переменных градиентным методом.**

|  |
| --- |
| **Начальные значения переменных для поиска минимума**    **Решение: xmin=0 ymin=0 f(xmin,ymin)=0** |

**Пример 2.4-35. Решить задачу оптимизации с помощью встроенных функций miner( ) .**

|  |
| --- |
| **Построим трехмерный график функции f(x,y)** |

|  |
| --- |
| Построим график линий уровня функции f(x,y) |

**Пример 2.4-36. Решить задачу оптимизации с помощью встроенных функций**

**Maximize (Minimize).**

|  |
| --- |
|  |

**Пример 2.4-37. Определить минимум функции .**

Для заданной функции, как известно, координаты точки минимума равны (0;0). Для этой функции график линий уровня представляет собой концентрические окружности, а точка минимума находится строго по центру.

|  |
| --- |
|  |

#### 2.4.9. Технология решения систем линейных уравнений средствами MathCad

В пакете **Mathcad** системы линейных уравнений решаются с использованием вычислительного блока Given и функции Find. Однако в том случае, когда система линейных уравнений невырождена, то есть ее определитель отличен от нуля, более изящным (хотя и не самым эффективным с точки зрения вычислительной математики) является матричный способ решения.

**Пример 2.4-38. Решить линейную систему уравнений.**

|  |
| --- |
|  |

**Пример 2.4-39. Решить линейную систему уравнений с помощью встроенной функции lsolve( ).**

При условии, что заданы матрицы коэффициентов и свободных членов, решение может быть получено следующим образом:

|  |
| --- |
|  |

Если система линейных уравнений не имеет точного решения, то вместо функции Find(x,y) следует использовать функцию Minner(x,y), позволяющую получить приближенное решение системы. При этом если точное решение существует, то функции Find( ) и Minner( ) дают одинаковые результаты.

**Пример 2.4-40. Решить линейную систему уравнений с помощью встроенных функций Find( ) и Minner( ).**

|  |
| --- |
|  |

Наряду с использованием встроенных функций пользователь может самостоятельно реализовать численный метод решения системы линейных уравнений, например, метод итераций. Для этого исходную систему уравнений следует привести к виду: . В методе итераций последовательность приближений к решению определяется соотношением , где k=0,1, ..(n-1), а (n-1) – количество неизвестных. В качестве начального приближения может быть выбран любой вектор. Известно, что метод итераций сходится, если для нормы матрицы  выполнено условие .

**Пример 2.4-41. В данном примере справа от уравнений указаны преобразования, которые выполнены с целью обеспечения сходимости.**



Приведем полученную систему уравнений к виду, удобному для итераций:



Норма матрицы, состоящая из сумм модулей коэффициентов при неизвестных в правых частях уравнения, равна {0.53; 0.75; 0.57}=0.75<1.

Для вычисления используем следующую итеративную формулу метода итераций:

.

Выбрав в качестве начальных приближений значения свободных членов, выполним 3 итерации, обеспечив точность результатов 0.01.

|  |
| --- |
| **1-я итерация**    **2-я итерация**    **3-я итерация** |

#### 2.4.10. Технология решения систем нелинейных уравнений средствами MathCad

**Для решения СЛУ необходимо задать начальные приближения для всех переменных, входящих в систему. В** MathCAD **решение системы уравнений требует использования вычислительного блока, начало которого отмечается ключевым словом Given. Для записи системы уравнений вместо ввода традиционного знака равно** = **вставляется жирный знак равенства – оператор отношения, расположенный на палитре «**Логические**». Если по условию задачи существуют ограничения на поиск решения, то они задаются в виде неравенств (например: a<x>b).**

**Решение СЛУ находится с помощью Find(x,y,z), где x, y, z – список переменных.**

Пример 2.4-42. Решить систему нелинейных уравнений

при начальных условиях**:** x0=1; y0=1; z0=1.

|  |
| --- |
| **Матричный метод решения системы нелинейных уравнений:**    **Решение системы нелинейных уравнений с использованием функции Find( )** |

MathCAD **позволяет решать системы уравнений не только в скалярной, но и в матричной форме. При этом начальные значения и ограничения задаются в виде векторов. Ниже приведены примеры решения системы уравнений в матричной форме путем обращения матрицы коэффициентов и с использованием функции Find( )**.

Пример 10.4-2. Решить систему нелинейных уравнений



|  |
| --- |
|  |

**Если система уравнений не имеет точного решения, то вместо функции Find( ) следует использовать функцию Minner, поскольку в этом случае функция Find( ) указывает на ошибку, функция Minner( ) находит минимум невязки, то есть возвращает значение аргумента, соответствующее минимальному расхождению между значением y и функцией y(x).**

СПИСОК ЛИТЕРАТУРЫ

Программа дисциплины «ИНФОРМАТИКА» 2009 г.

Шакин В.Н. , Семенова Т.И., Кравченко О.М. ИНФОРМАТИКА: Лабораторный практикум для студентов МТУСИ: Раздел 6. Модели и алгоритмы решения задач численных методов с использованием математических пакетов. – М: МТУСИ, 2009.

Электронное учебное пособие и практикум «Информатика» для студентов МТУСИ, 2009.

Кравченко О.М., Семенова Т.И., Шакин В.Н. Учебное пособие: Модели решения вычислительных задач (численные методы и оптимизация) по дисциплине «Информатика» для студентов, обучающихся по направлению подготовки «Телекоммуникации»: М.,2003.- 2003.

Амосов А.А., Дубинский Ю.А., Копченова Н.В. Вычислительные методы для инженеров: М., Высшая школа,1994.

Бахвалов Н.С. Численные методы М., Наука, 1973.

Банди Б. Методы оптимизации. Вводный курс: М., Радио и связь, 1988.

Копченова Н.В., Марон И.А. Вычислительная математика в примерах и задачах: М., Наука, 1972.

Демидович Б.Л., Марон И.А. Основы вычислительной математики: М., Наука, 1970.

Васильев В.К., Семенова Т.И. Численные методы решения задач на ЭВМ. Уч. пособие: М., МТУСИ, 1993 г.

Семенова Т.И., Шакин В.Н. Практикум: Математический пакет MathCad в дисциплине «Информатика»: МТУСИ. М.,2006.

Дьяконов В.П. МаhtCad 11/12/13 в математике. Справочник. – М.: Горячая линия – Телеком, 2007. – 958 с.

Половко А.М., Бутусов П.Н. MatLab для студентов.- СПб-Петербург, 205.-320с.

Дьяконов В.П. Maple 10/11/12/13/14 в математических расчетах. – М.: ДМК Пресс, 2011. – 800 с.

Дьяконов В.П. MaTLab. Полный самоучитель. – М.: ДМК Пресс, 2012. – 768 с.

Васильев А.Н. MatLab. Самоучитель. Практический подход. СПб.: Наука и Техника, 2012.-448 с. Содержание

Раздел 4. Использование математического пакета

MathCad для аналитических и численных решений..3

Тема 4.1. Элементы теории погрешностей……………………… 4

Тема 2.4. Технология решения нелинейных уравнений средствами

математического пакета MathCad................................... 5

Тема 2.4. Технология интерполяции функций в среде

математического пакета MathCad................................. 11

Тема 2.4.. Технология вычисления интегралов в среде

математического пакета MathCad............................... 14

Тема 2.4. Технология решения обыкновенных дифференциальных

уравнений средствами математического пакета

MathCad......................................................................................17

Тема 2.4. Технология решения задач одномерной оптимизации

средствами математического пакета........................... 22

Тема 2.4. Технология решения задач аппроксимации функций

средствами математического пакета…...................……25

Тема 2.4. Технология решения задач многомерной оптимизации

средствами математического пакета MathCad....... …28

Тема 2.4 . Технология решения систем линейных уравнений

средствами математического пакета MathCad........ 31

Тема 6.10. Технология решения систем нелинейных уравнений

средствами математического пакета MathCad…. 33

СПИСОК ЛИТЕРАТУРЫ.............................................................................35