Федеральное государственное бюджетное образовательное учреждение высшего образования
«Новосибирский государственный технический университет»

Факультет Автоматики и вычислительной техники
Кафедра Вычислительной техники

ВЫЧИСЛИТЕЛЬНАЯ МАТЕМАТИКА
Численные методы решение систем нелинейных уравнений

Методические указания к лабораторной работе
для студентов II курса дневного отделения АВТФ
Образовательные программы:
09.03.01 «Информатика и ВТ»;
09.03.04 «Программная инженерия»;
12.03.01 «Приборостроение»

НОВОСИБИРСК
2017

Содержание

1 Описание лабораторной работы

Цель работы: сформировать у студентов представление о методах решения систем нелинейных уравнений, привить умения составлять и применять алгоритмы для решения таких систем уравнений, выработать навыки в использовании программных средств для решения систем уравнений.

Задание

1. В соответствии с вариантом контрольного задания исследуйте существование и найдите решение системы нелинейных уравнений с точностью ε не ниже 0,00001 тремя методами из таблицы1, например:
- методом итераций;
- методом Ньютона;
- по выбору студента или с модификацией одного из обязательных 1-3.

2. Написать программы, реализующие алгоритмы решения систем нелинейных уравнений методами согласно варианту Табл. 1.
3. Для дублирования решений заданной системы нелинейных уравнений применить существующие стандартные функции МСАД.
4. Для каждого метода и способа решения исследовать ресурсоемкость программ вычислений, а также факторы, влияющие на точность результатов и устойчивость вычислений.
5. Сравнить способы решения систем уравнения по быстродействию, точности и зависимости от начальных условий; выбрать наиболее эффективный вычислительный процесс поставленной задачи.

6. Проанализировать результаты работы, сделать выводы и дать рекомендации.

Результаты численных расчетов должны быть оформлены по всем правилам записи приближенных чисел, т.е. запись приближенного решения только с верными значащими цифрами и допускаемой погрешностью. Анализ численных результатов должен дать ответ на вопрос, соответствуют ли полученные результаты искомому решению поставленной задачи и почему. В отчете обязательно сформулируйте необходимые и достаточные условия существования решений и сходимости методов; проверьте точность решений. Проанализируйте особенности применения рассмотренных методов, дайте рекомендации по решению соответствующих задач на ЭВМ.

2 Варианты контрольных заданий

Варианты методов решения систем нелинейных уравнений Таблица 1

№ вар.	Методы	Система уравнений	№ вар.	Методы	Система уравнений
1	1, 2, по выбору	$\begin{cases} tg(xy+0{,}4)=x^2, \\ 0{,}6x^2+2y^2=1, \quad x>0, y>0 \end{cases}$	16	1, 2, по выбору	$\begin{cases} \sin(x+y)-1{,}4x=0 \\ x^2+y^2=1 \end{cases}$
2	1, 3, по выбору	$\begin{cases} \sin(x+y)-1{,}6x=0 \\ x^2+y^2=1, \quad x>0, y>0 \end{cases}$	17	1, 3, по выбору	$\begin{cases} tg(xy+0{,}1)=x^2, \\ 0{,}5x^2+2y^2=1 \end{cases}$
3	2, 3, по выбору	$\begin{cases} tg(xy+0{,}1)=x^2, \\ x^2+2y^2=1 \end{cases}$	18	2, 3, по выбору	$\begin{cases} \sin(x+y)=1{,}1x+0{,}1 \\ x^2+y^2=1 \end{cases}$
4	1, 2, по выбору	$\begin{cases} \sin(x+y)-1{,}2x=0{,}2 \\ x^2+y^2=1 \end{cases}$	19	1, 2, по выбору	$\begin{cases} tg(x-y)-xy=0, \\ x^2+2y^2=1 \end{cases}$
5	1, 3, по выбору	$\begin{cases} tg(xy+0{,}3)=x^2, \\ 0{,}9x^2+2y^2=1 \end{cases}$	20	1, 3, по выбору	$\begin{cases} \sin(x-y)-xy=-1 \\ x^2-y^2=0{,}75 \end{cases}$
6	2, 3, по выбору	$\begin{cases} \sin(x+y)-1{,}3x=0 \\ x^2+y^2=1 \end{cases}$	21	2, 3, по выбору	$\begin{cases} tg(xy+0{,}2)=x^2, \\ x^2+2y^2=1 \end{cases}$
7	1, 2, по выбору	$\begin{cases} tg(xy)=x^2, \\ 0{,}8x^2+2y^2=1 \end{cases}$	22	1, 2, по выбору	$\begin{cases} \sin(x+y)-1{,}5x=0 \\ x^2+y^2=1 \end{cases}$
8	1, 3, по выбору	$\begin{cases} \sin(x+y)-1{,}5x=0{,}1 \\ x^2+y^2=1 \end{cases}$	23	1, 3, по выбору	$\begin{cases} tg(xy)=x^2, \\ 0{,}5x^2+2y^2=1 \end{cases}$
9	2, 3, по выбору	$\begin{cases} tg(xy)=x^2, \\ 0{,}7x^2+2y^2=1 \end{cases}$	24	2, 3, по выбору	$\begin{cases} \sin(x+y)=1{,}2x-0{,}2 \\ x^2+y^2=1 \end{cases}$
10	1, 2, по выбору	$\begin{cases} \sin(x+y)-1{,}2x=0{,}1 \\ x^2+y^2=1 \end{cases}$	25	1, 2, по выбору	$\begin{cases} tg(xy+0{,}1)=x^2, \\ 0{,}7x^2+2y^2=1 \end{cases}$
11	1, 3, по выбору	$\begin{cases} tg(xy+0{,}2)=x^2, \\ 0{,}6x^2+2y^2=1 \end{cases}$	26	1, 3, по выбору	$\begin{cases} \sin(x+y)=1{,}5x+0{,}2 \\ x^2+y^2=1 \end{cases}$
12	2, 3, по выбору	$\begin{cases} \sin(x+y)=1{,}2x-0{,}1 \\ x^2+y^2=1 \end{cases}$	27	2, 3, по выбору	$\begin{cases} tg(xy)=x^2, \\ 0{,}6x^2+2y^2=1 \end{cases}$

13	1, 2, по выбору	$\begin{cases} tg(xy+0{,}4)=x^2, \\ 0{,}8x^2+2y^2=1 \end{cases}$	28	1, 2, по выбору	$\begin{cases} \sin(x+y)-1{,}2x=0 \\ x^2+y^2=1 \end{cases}$
14	1, 3, по выбору	$\begin{cases} \sin(x+y)=1{,}5x-0{,}1 \\ x^2+y^2=1 \end{cases}$	29	1, 3, по выбору	$\begin{cases} tg(xy+0{,}3)=x^2, \\ 0{,}5x^2+2y^2=1 \end{cases}$
15	2, 3, по выбору	$\begin{cases} tg(xy+0{,}1)=x^2, \\ 0{,}9x^2+2y^2=1 \end{cases}$	30	2, 3, по выбору	$\begin{cases} \sin(x+y)-1{,}1x=0{,}1 \\ x^2+y^2=1 \end{cases}$

3 Краткая теория к лабораторной работе

Система нелинейных уравнений имеет вид:

$$\begin{cases} f_1(x_1,\dots x_n)=0 \\ \dots \\ f_n(x_1,\dots x_n)=0 \end{cases} \qquad (1)$$

Здесь $x_1, x_2, \dots, x_n$ - неизвестные переменные, а система (1) называется нормальной системой порядка n, если хотя бы одна из функций $f_i(x_1, x_2, \dots, x_n)$ нелинейна. В более краткой векторной форме система (1) имеет вид:

$$\vec{\mathbf{f}}(\vec{\mathbf{x}})=0, \ где \ \vec{\mathbf{x}}=X=\left(x_1,x_2,...,x_n\right)^T, \ \ \vec{\mathbf{f}}=\left(f_1,f_2,...,f_n\right)^T. \quad (2)$$

Пусть функции f_i определены в областях действительных значений $\Omega_i, i=\overline{1,n}$. Тогда область $\Omega=\bigcap_{i=1}^{n}\Omega_i$ и будет той областью, где можно найти решение.

Необходимость решения системы (1) возникает как непосредственно при моделировании объектов системами уравнений, так и опосредовано при решении многих прикладных задач, например, поиска безусловного экстремума функций многих переменных с помощью необходимых условий, при использовании неявных методов интегрирования обыкновенных дифференциальных уравнений и т.д. Решение систем нелинейных уравнений – одна из трудных задач вычислительной математики. Трудность состоит в том, чтобы определить: имеет ли система решение, и, если – да, то сколько. Уточнение решений в заданной области – более простая задача. При этом вся сложность их решения заключается в обеспечении устойчивости вычислительного процесса в широком диапазоне заданных областей, изолирующих корни систем уравнений.

Существует большое число разнообразных подходов к решению систем уравнений. В определенном смысле, некоторые численные методы, пригодные для одного (скалярного) уравнения, могут служить примерами обобщения на системы не линейных уравнений. В частности, это относится к графическому методу, методу поиска экстремумов положительно или отрицательно определенных функций левой части системы (1) и так далее. Однако их применение по разным объективным причинам мало пригодны для практического

применения. Тем не менее, ряд итерационных методов решения нелинейных уравнений может быть распространен и на системы нелинейных уравнений. Наиболее распространенными методами уточнения решения являются метод простых итераций и метод Ньютона. Они имеют ряд модификаций, улучшающих их достоинства и компенсирующих их недостатки.

3.1. Метод простых итераций для решения систем нелинейных уравнений

Из исходной системы (1) путем эквивалентных преобразований переходим к системе вида:

$$\begin{cases} x_1 = \Phi_1(x_1, \ldots, x_n) \\ \ldots \\ x_n = \Phi_n(x_1, \ldots, x_n) \end{cases} \quad (3)$$

Итерационный процесс, определяемый формулами

$$\begin{cases} x_1^{k+1} = \Phi_1\left(x_1^{(k)}, \ldots, x_n^{(k)}\right) \\ \ldots \qquad\qquad\qquad , \; k = 0,1,\ldots \\ x_n^{k+1} = \Phi_n\left(x_1^{(k)}, \ldots, x_n^{(k)}\right) \end{cases} \quad (4)$$

можно начать, задав начальное приближение $x^{(0)} = \left(x_1^{(0)}, \ldots, x_n^{(0)}\right) \in \overline{\Omega} \subset \Omega.$

Итерационный процесс (4) начинается с некоторого начального приближения $X^{(0)}$ и продолжается до тех пор, пока модули приращений всех аргументов после одной итерации не станут меньше заданного ε:

$$\left\| X^{(k+1)} - X^{(k)} \right\|_\infty = \max_{1 \le i \le n} \left\{ \left| x_i^{k+1} - x_i^k \right| \right\} \le \varepsilon.$$

С равным успехом можно пользоваться условиями (нормами разности векторов), например,

$$\left\| X^{(k+1)} - X^{(k)} \right\|_e = \sqrt{\sum_{i=1}^{n} (x_i^{k+1} - x_i^k)^2} \le \varepsilon.$$

Достаточным условием сходимости итерационного процесса является одно из двух условий:

$$\sum_{j=1}^{n} \left| \frac{\partial \Phi_i}{\partial x_j} \right| < 1, \forall i = \overline{1,n} \text{ или } \sum_{i=1}^{n} \left| \frac{\partial \Phi_i}{\partial x_j} \right| < 1, \forall j = \overline{1,n}.$$

Распишем первое условие:

$$\left| \frac{\partial \Phi_1}{\partial x_1} \right| + \left| \frac{\partial \Phi_1}{\partial x_2} \right| + \ldots + \left| \frac{\partial \Phi_1}{\partial x_n} \right| < 1 \text{ при } i = 1$$

$$\left| \frac{\partial \Phi_n}{\partial x_1} \right| + \left| \frac{\partial \Phi_n}{\partial x_2} \right| + \ldots + \left| \frac{\partial \Phi_n}{\partial x_n} \right| < 1 \text{ при } i = n.$$

Распишем второе условие:

$$\left|\frac{\partial \Phi_1}{\partial x_1}\right| + \left|\frac{\partial \Phi_2}{\partial x_1}\right| + \ldots + \left|\frac{\partial \Phi_n}{\partial x_1}\right| < 1 \text{ при } j = 1$$

$$\left|\frac{\partial \Phi_1}{\partial x_n}\right| + \left|\frac{\partial \Phi_2}{\partial x_n}\right| + \ldots + \left|\frac{\partial \Phi_n}{\partial x_n}\right| < 1 \text{ при } j = n.$$

На практике для проверки сходимости рассматривают матрицу:

$$M_{ij} = \max\left|\left(\frac{\partial \Phi_i}{\partial x_j}\right)\right|.$$

Норма этой матрицы мажорирует соответствующие нормы матрицы производных. Поэтому достаточным условием сходимости является условие $\|M_{ij}\| < 1$. Для различных норм матрицы это условие принимает разные формы:

$$\sum_{i=1}^{n} M_{ij} < 1, \quad \sum_{j=1}^{n} M_{ij} < 1 \quad \textit{или} \quad \sum_{i,j=1}^{n} M_{ij}^2 < 1.$$

Поскольку в конечномерном пространстве все нормы матриц эквивалентны, из сходимости итераций в одной норме следует сходимость во всех остальных.

Нулевое приближение в случае n = 2 можно выбрать графически, изобразив в плоскости (x_1, x_2) кривые $f_1(x_1, x_2) = 0$ и $f_2(x_1, x_2) = 0$ и определив приближённо точки их пересечения.

Рассмотрим один из способов приведения системы (1) к виду (3), допускающему сходящиеся итерации.

Пусть задана система второго порядка вида:

$$\begin{cases} f_1(x, y) = 0 \\ f_2(x, y) = 0 \end{cases}.$$

Требуется привести ее к виду:

$$\begin{cases} x = \Phi_1(x, y) \\ y = \Phi_2(x, y) \end{cases}.$$

Умножим первое уравнение системы на неизвестную постоянную α, второе - на β, затем сложим их и добавим в обе части уравнения x. Получим первое уравнение преобразованной системы

$$x + \alpha f_1(x, y) + \beta f_2(x, y) = x;$$

где $\Phi_1(x, y) = x + \alpha f_1(x, y) + \beta f_2(x, y).$

Далее, умножим первое уравнение системы на неизвестную постоянную γ, второе - на δ, затем сложим их и добавим в обе части уравнения y. Тогда второе уравнение преобразованной системы будет иметь вид

$$y+\gamma f_1(x,y)+\delta f_2(x,y)=y;$$

где $\Phi_2(x,y)=y+\gamma f_1(x,y)+\delta f_2(x,y).$

Неизвестные постоянные $\alpha,\beta,\gamma,\delta$ определим из достаточных условий сходимости

$$\left|\frac{\partial\Phi_1}{\partial x}\right|+\left|\frac{\partial\Phi_2}{\partial x}\right|<1 \text{ и } \left|\frac{\partial\Phi_1}{\partial y}\right|+\left|\frac{\partial\Phi_2}{\partial y}\right|<1.$$

Запишем эти условия более подробно:

$$\left|1+\alpha\frac{\partial f_1}{\partial x}+\beta\frac{\partial f_2}{\partial x}\right|+\left|\gamma\frac{\partial f_1}{\partial x}+\delta\frac{\partial f_2}{\partial x}\right|<1$$

$$\left|\alpha\frac{\partial f_1}{\partial y}+\beta\frac{\partial f_2}{\partial y}\right|+\left|1+\gamma\frac{\partial f_1}{\partial y}+\delta\frac{\partial f_2}{\partial y}\right|<1$$

Полагая равными нулю выражения под знаком модуля, получим систему из четырех уравнений с четырьмя неизвестными для определения постоянных $\alpha,\beta,\gamma,\delta$:

$$\begin{cases}1+\alpha\frac{\partial f_1}{\partial x}+\beta\frac{\partial f_2}{\partial x}=0\\ \gamma\frac{\partial f_1}{\partial x}+\delta\frac{\partial f_2}{\partial x}=0\\ \alpha\frac{\partial f_1}{\partial y}+\beta\frac{\partial f_2}{\partial y}=0\\ 1+\gamma\frac{\partial f_1}{\partial y}+\delta\frac{\partial f_2}{\partial y}=0\end{cases}.$$

При таком выборе параметров условия сходимости будут соблюдены, если частные производные функций $f_1(x,y)$ и $f_2(x,y)$ будут изменяться не очень быстро в окрестности точки $\left(x^{(0)},y^{(0)}\right)$.

Чтобы решить систему, нужно задать начальное приближение $\left(x^{(0)},y^{(0)}\right)\in\overline{\Omega}\subset\Omega$ и вычислить значения производных $\left.\frac{\partial f_i}{\partial x}\right|_{\left(x^{(0)},y^{(0)}\right)}$ и $\left.\frac{\partial f_i}{\partial y}\right|_{\left(x^{(0)},y^{(0)}\right)}$, $i=\overline{1,2}$ в этой точке.

Вычисление $\alpha,\beta,\gamma,\delta$ осуществляется на каждом k шаге итераций, при этом

$$\left.\frac{\partial f_i}{\partial x}\right|_{\left(x^{(k)},y^{(k)}\right)},\ \left.\frac{\partial f_i}{\partial y}\right|_{\left(x^{(k)},y^{(k)}\right)},i=\overline{1,2}.$$

Метод простых итераций является самоисправляющимся, универсальным, прямо ведет к решению и легко реализуется на ЭВМ. Он имеет два существенных недостатка. Один из них - довольно жесткое условие сходимости, состоящее в том. что для всех значений $X^{(k)}$ из некоторой окрестности решения X, должно выполняться условие сходимости : $\left\|M_{ij}\right\|<1$, другой - слабая сходимость, особенно при большом числе уравнений.

3.2. Метод Зейделя для решения систем нелинейных уравнений

Одной из модификаций метода простой итерации, направленной на ускорение сходимости, является модификация, носящая название метода Зейделя. В этом методе итерационный процесс описывается формулами:

$$\begin{cases} x_1^{k+1} = \phi_1\left(x_1^{(k)},\ldots,x_n^{(k)}\right) \\ x_2^{k+1} = \phi_2\left(x_1^{(k+1)},\ldots,x_n^{(k)}\right) \\ \ldots \\ x_n^{k+1} = \phi_n\left(x_1^{(k+1)},\ldots,x_{n-1}^{(k+1)},x_n^{(k)}\right) \end{cases} \qquad (5)$$

Здесь следует обратить внимание на то, что уточненное значение x_1 сразу же используется для уточнения x_2. Затем, по новым значениям x_1 и x_2 вычисляется x_3, и т.д. Скорость сходимости у итерационного процесса (5) несколько выше, чем у простой итерации (3). но проблема выбора начального приближения остается по-прежнему острой. Иногда эту проблему удается решить с помощью *метода возмущения параметров*, рассматриваемого в работе [3].

Наряду с итерационным процессом (5) можно также рассмотреть процесс, в котором компоненты вектора приближений определяются из уравнений:

$$\begin{cases} f_1\left(\zeta, x_2^{(k)},\ldots,x_n^{(k)}\right) = 0, \\ f_2\left(x_1^{(k+1)},\zeta,\ldots,x_n^{(k)}\right) = 0 \\ \ldots \\ f_n\left(x_1^{(k+1)},\ldots,x_{n-1}^{(k+1)},\xi\right) = 0 \end{cases} \qquad (6)$$

Каждое из них представляет собой уравнение с одним неизвестным ξ. Значение ξ_1, являющееся корнем первого из уравнений совокупности (6), рассматривается в качестве нового приближения для компоненты $\mathbf{x_1}$: $x_1^{(k+1)} = \zeta_1$. Затем корень ξ_2, второго равнения считается новым приближением для $\mathbf{x_2}$: $x_2^{(k+1)} = \zeta_2$ и т.д. Привлекательность такого подхода состоит в возможности использования сравнительно простых методов решения одного уравнения. Но на практике это может привести к очень большому объему вычислений.

2.3. Метод Ньютона для решения систем нелинейных уравнений

Пусть требуется решить систему нелинейных уравнений вида (1). Предположим, что решение существует в некоторой области $\overline{\Omega}$, в которой все функции $f_i(x_1,\ldots,x_n),\ i=\overline{1,n}$ непрерывны и имеют, по крайней мере, первую производную.

Метод Ньютона представляет собой итерационный процесс, который осуществляется по определенной формуле следующего вида:

$$x^{(k+1)} = x^{(k)} - W^{-1}\left(x^{(k)}\right) f\left(x^{(k)}\right) \qquad (3)$$

где $W\left(x^{(k)}\right)=\left.\begin{pmatrix} (f_1)'_{x_1},\ldots,(f_1)'_{x_n} \\ \ldots \\ (f_n)'_{x_1},\ldots,(f_n)'_{x_n} \end{pmatrix}\right|_{\left(x_1^{(k)},\ldots,x_n^{(k)}\right)}$ - матрица Якоби.

Трудности при использовании метода Ньютона определяются вопросами:

- существует ли обратная матрица Якоби?
- не выходит ли $x^{(k+1)}$ за пределы области $\overline{\Omega}$?

Модифицированный метод Ньютона облегчает первую задачу. Модификация состоит в том, что матрица вычисляется не в каждой точке, а лишь в начальной. Таким образом, модифицированный метод Ньютона имеет следующую формулу:

$$x^{(k+1)} = x^{(k)} - W^{-1}\left(x^{(0)}\right) f\left(x^{(k)}\right) \qquad (4)$$

Но ответа на второй вопрос, модифицированный метод Ньютона не дает.

Итерационный процесс по формулам (2) или (4) заканчивается, если выполняется следующее условие

$$\left\| x^{(k+1)} - x^{(k)} \right\| \le \varepsilon .$$

Достоинством метода Ньютона является его быстрая сходимость по сравнению с методом простых итераций. Он характеризуется квадратичной сходимостью из хорошего начального приближения при условии невырожденности матрицы Якоби. К недостаткам метода Ньютона следует отнести: необходимость задавать достаточно хорошее начальное приближение; отсутствие глобальной сходимости для многих задач; необходимость вычисления матрицы Якоби на каждой итерации; необходимость решения на каждой итерации системы линейных уравнений, которая может быть плохо обусловленной.

3.4 Решение систем нелинейных уравнений на MathCad

В среде Mathcad существуют аналитический и численный способ решения системы нелинейных уравнений. Они хорошо описаны в работе [2].

С помощью символьного процессора Mathcad можно получить аналитическое решение системы уравнений. Сделать это можно двумя способами. Во-первых, можно воспользоваться оператором **solve** (решить). В этом случае система должна быть внесена в его левый маркер в виде вектора. Переменные, значение которых должно быть найдено, следует ввести через запятую в правый маркер оператора **solve**. Ответ будет возвращен в виде матрицы, в строках которой будут записаны корни найденных решений. Их порядок будет таким же, каким был порядок соответствующих переменных в правом маркере оператора **solve.** Во-вторых, можно использовать так называемый вычислительный блок. Вычислительным блоком в Mathcad мы будем называть систему из вводного слова (**Given** (Дано)) и функции той или иной математической операции (например, **Find** — решение систем уравнений, **Minerr** — поиск точки минимальной невязки системы). Чтобы решить систему уравнений с помощью вычислительного блока, выполните следующую последовательность действий.

1. Наберите вводное слово **Given.**
2. Строго под вводным словом задайте систему уравнений. Делается это, в отличие от случая использования оператора **solve**, точно так же, как при ее решении на бумаге. В качестве знаков равенства следует использовать логическое равенство (Bold Equal —

<Ctrl>+ < =>). Если система приведена к стандартному виду, то, аналогично поиску корней скалярных уравнений, можно определить лишь левые части ее уравнений.

3. Введите функцию решения систем уравнений **find(xl,x2,...).** В скобках через запятую задайте переменные в том порядке, в котором должны быть расположены в ответе соответствующие им корни.

4. В качестве оператора вывода результата работы функции **find(xl,x2,...)** используйте оператор символьного вывода << → > >. Если же вы примените оператор численного вывода ≪=≫, то для решения системы, при условии добавления начальных приближений, будет запущен один из численных алгоритмов.

Существенных различий между решением системы уравнений с помощью оператора **solve** и вычислительного блока **Given-F**ind нет. Однако есть небольшая, но совсем неочевидная разница в форме представления результата. Она заключается в том, что значения корней, относящиеся к одному решению, в случае использования вычислительного блока располагаются в столбцах, а при применении оператора **solve** — в строках матрицы ответа. Если этого не знать, то можно очень легко запутаться и сделать ошибку (особенно если количество решений и переменных совпадает).

Как и в случае скалярных уравнений, к численным методам решения системы уравнений следует обращаться тогда, когда с ней не справится аналитический процессор Mathcad. Реально это приходится делать очень часто, так как лишь редкие системы нелинейных уравнений могут быть решены символьно. В Mathcad для численного решения систем уравнений служит блок **Given-Find**. Правила его задания схожи с правилами, использующимися при символьном решении систем уравнений, однако есть и особенности. Отметим их.

• Аналогично численным методам решения уравнений с одним неизвестным, сначала следует определить начальные приближения. В случае систем уравнений приближение должно быть определено для каждой переменной.

• Для вывода результата после функции **find** следует ввести оператор численного вывода ≪=≫.

Редкое нелинейное уравнение имеет только один корень. Многие из них (периодические функции) могут иметь бесконечное множество решений. В случае же систем нелинейных уравнений ситуация еще более усложняется тем, что для большинства из них весьма проблематично определить начальные приближения и даже количество корней. С этими проблемами можно было очень просто справиться в случае уравнения с одним неизвестным, определяя нужные характеристики чисто визуально на графике. Аналогично можно поступить, в принципе, и для систем из двух уравнений, но лишь в тех редких случаях, когда одна переменная однозначно выражается через другую.

Между численными методами, используемыми для решения одного уравнения и систем уравнений, нет принципиальных различий. Следовательно, основным параметром, определяющим точность решения, является порог, при достижении которого функция считается равной нулю. Как вы помните (ЛабРаб 1), при использовании функции **root** этому порогу соответствовала системная переменная TOL. Ту же роль исполняет данная переменная и при решении систем уравнений. Уменьшая TOL, вы увеличиваете точность, однако при этом возрастает время расчета и чувствительность к виду функции. Минимальное значение TOL — 10^{-17}, стандартное — 10^{-3}. Стоит воздержаться от ≪профилактического≫ назначения TOL минимального значения. Дело в том, что количество итераций, которые может совершить численный метод, ограничено ввиду накопления ошибки. Поэтому при очень малом значении TOL имеется вероятность, что алгоритм просто не доберется до корня, исчерпав лимит на количество итераций. Величина TOL должна задаваться исходя из того, сколько знаков в ответе действительно значимо. В большинстве случаев $T0L=10^{-3}$ обеспечивает вполне приемлемую точность. При решении линейных уравнений в блоке **Given-Find** TOL не оказывает влияния на точность результата.

В Mathcad реализовано несколько алгоритмов численного решения систем уравнений.

Если алгоритм, используемый по умолчанию, не справится с задачей, можно попытаться сменить его на другой. Для этого щелкните правой кнопкой мыши на тексте функции **find**. При этом откроется контекстное меню (Рис. 1), в котором среди стандартных команд имеются три пункта, определяющие тип используемого метода.

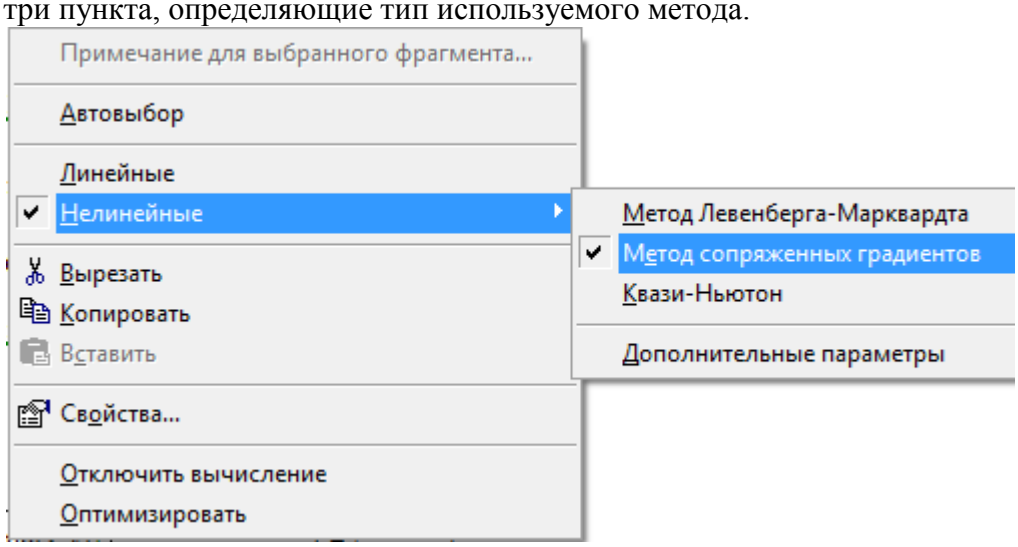

Рис. 1 Палитра настройки функции решения системы уравнений

• Пункт AutoSelect (Автоматический выбор). Численный алгоритм и его настройки будут определены автоматически. Если система линейная, то будет использован соответствующий метод. Если она нелинейная, то сначала будет осуществлена попытка найти решения методом сопряженных градиентов. Если она окажется неудачной, то будет задействован метод Левенберга. Если и он окажется неэффективным, то система применит квазиньютоновский метод. Чаще всего нет никакого смысла менять пункт AutoSelect на какой-то другой, так как при его активности Mathcad самостоятельно будет пытаться найти решения всеми доступными способами. Сменить его стоит лишь, если вы хотите применить тонкие настройки или же получить решение каким-то конкретным способом.

Пункт Linear (Линейный) отвечает за реализацию итеративного метода (симплекс-метод) решения системы линейных уравнений. Обычно Mathcad самостоятельно относит систему уравнений к той или иной группе, поэтому специально настраивать этот параметр нет необходимости. При решении линейных уравнений в блоке **Given-Find** TOL не оказывает влияния на точность результата.

При подведении курсора к строке NonLinear (Нелинейный) откроется всплывающее меню, содержащее ссылки на три различных численных метода решения систем уравнений: сопряженных градиентов (Conjugate gradient), Левенберга (Levenberg-Marquardt), квази-ньютоновский (Quasi-Newton). Вдаваться в подробности каждого из них нет особого смысла, так как в обычной практике очень трудно предсказать, какой из методов окажется эффективнее для данной системы уравнений. Чаще всего их эффективность оказывается близка, поэтому гораздо более принципиальное значение имеет правильный выбор начальных приближений и точности. Проверьте это утверждение экспериментально на примере своего варианта системы, совершив для этого её решение различными методами в одних и тех же условиях.

Помимо выбора самого численного метода решения систем уравнений, можно настроить некоторые параметры его работы. Сделать это можно в специальном окне Advanced Options (Дополнительные параметры). Настройки окна Advanced Options работают только для метода сопряженных градиентов и квазиньютоновского метода. На алгоритм метода Левенберга они никак не влияют.

В окне Advanced Options (Дополнительные параметры) имеется пять строк, в которых находятся по два переключателя, ответственных за ту или иную настройку (рис.2).

В первой строке (Derivative Estimation) можно настроить тип аппроксимации производной: либо центральной (Central), либо правой (Forward) конечной разностью. По умолчанию производная вычисляется по трехточечной симметричной схеме: это хотя и несколько более медленно, зато дает лучшие результаты. Так, при замене в методе сопряженных градиентов центральной схемы на правую двухточечную рассмотренная выше система решена не была, хотя при использовании параметра **Central** корни могут быть найдены быстрее и с большой точностью. Поэтому не стоит менять схему вычисления производной, даже если корни не были найдены: это почти наверняка приведет лишь к худшим результатам.

Во второй строке можно определить тип аппроксимации функции: либо касательной (Tangent), либо параболой (Quadratic). Квадратичная аппроксимация более точная, но и более чувствительная к виду функции. Поэтому, дабы избежать лишних ошибок, по умолчанию определено использование касательных. Но если решение найдено не было, можно попробовать использовать квадратичное приближение.

Строка Linear Variable Check (линейная проверка) отвечает за проверку линейности вашей функции относительно переменных. Это может помочь сэкономить время за счет того, что частные производные будут приняты на первом шаге за константы и не будут вычисляться при каждой итерации. Впрочем, задачи, для которых этот параметр может быть полезен, крайне специфичны и редки, поэтому менять установки в этой строке не стоит.

Строка Multistart (мультистарт) интересна только в том случае, если в блоке **Given** используется не функция **find**, а функции **Maximize** и **Minimize** (они служат для определения минимума и максимума функции и могут быть применены для решения соответствующих систем уравнений, у которых экстремальные значения левых частей совпадает со значениями равными или достаточно близкими к нулю). При активации данного параметра система будет пытаться найти глобальный экстремум, а не ограничится поиском ближайшего локального экстремума.

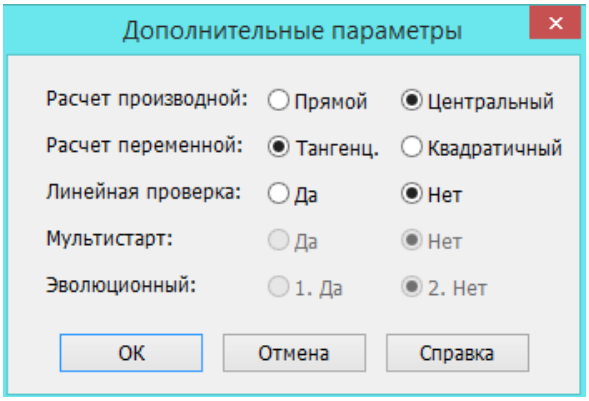

Рис. 2 Окно дополнительных параметров

Настройка Evolutionary (эволюционный) может помочь найти решения в случае функций с многочисленными экстремумами, точками и областями недифференцируемости. Используемые блоком **Given-Find** градиентные методы очень чувствительны к виду функции, поэтому столь важно правильно выбирать начальное приближение. Модификация же, активируемая настройкой Evolutionary, позволяет значительно снизить данную чувствительность. Также она помогает найти из нескольких решений оптимальное, даже если оно расположено далеко от точки начального приближения.

Нельзя не упомянуть о еще одной очень полезной возможности блока Given-Find: в состав системы могут входить не только уравнения, но и неравенства (в которых можно использовать любые логические операторы). В некоторых случаях это может помочь ограничить количество решений, и тем самым облегчить поиск нужного.

4 Вопросы самоконтроля

1. Основные этапы решения систем нелинейных уравнений и краткая их характеристика.
2. Способы и возможности графических средств для изоляции корней систем уравнений.
3. Влияние начальных условий на точность и длительность решения систем нелинейных уравнений.
4. Какие виды норм матриц вам известны, как их вычислять, для чего они применяются?
5. Определите необходимые и достаточные условия применения метода простых итераций для решения систем нелинейных уравнений.
6. Назовите характерные особенности метода Зейделя.
7. В чем сущность метода Ньютона для решения систем нелинейных уравнений?
8. Каким требованиям должна удовлетворять матрица Якоби для успешного решения системы уравнений итерационным методом?
9. Перечислите основные характеристики вычислительных процессов решения систем уравнений и способы их оценки.
10. Как можно проверить правильность, точность и длительность вычислений.
11. Какие функции вы знаете для решения систем нелинейных уравнений в Mathcad?
12. В каких случаях Mathcad не может найти корень систему уравнений?
13. Дайте сравнительную характеристику числовым методам решения систем нелинейных уравнений и функциям Mathcad.
14. Как изменить метод и точность, с которой ищутся решения систем нелинейных уравнений функциям Mathcad?
15. Как символьно (аналитически) решить систему уравнений в Mathcad?
16. Назовите особенности использования символьного решения уравнений.

Литература

1. Алексеев Е. Р., Чеснокова О. В. Решение задач вычислительной математики в пакетах Mathcad 12, MATLAB 7, Maple 9/Алексеев Е. Р., Чеснокова О. В. - М. : НТ Пресс, 2006. - 496 с.: ил. - {Самоучитель).
2. Гурский Д. А., Турбина Е. С**.** Вычисления в Mathcad 12. — СПб.: Питер, 2006. — 544 с : ил.
3. Зайцев, В.В. Численные методы для физиков. Нелинейные уравнения н оптимизация: учебное пособие / В.В.Зайцев. В.М Трещев. – Самара, 2005**.** -86 с.: ил.
4. Ханова А. А. Численное решение уравнений и систем уравнений: Методическое пособие для студентов института Информационно- технологий телекоммуникаций, Астрахань -2001, -43 стр.
5. Поршнев С. В., Беленкова И. В. Численные методы на базе Mathcad. СПб.; БХВ-Петербург, 2005. 464 с: ил.
6. Методы решения систем нелинейных уравнений http://mathhelpplanet.com/static.php?p=metody-resheniya-sistem-nelineynykh-uravneniy

Приложение.

Примеры для выполнения задания лабораторной работы и применения средств MathCad для решения уравнений содержатся в файлах:

П1. Примеры к выполн ЛР РешСистНеУравн 1.xmcd

П2. Примеры Метода Зейделя.xmcd

П3. Примеры к вып ЛР РешСистНеУравн МетодНьютона.xmcd