МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение
высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»
Невинномысский технологический институт (филиал

МЕТОДЫ РЕШЕНИЯ НЕЛИНЕЙНЫХ УРАВНЕНИЙ И СИСТЕМ

Методические указания к выполнению лабораторных работ

15.03.04 Автоматизация технологических процессов и производств

Невинномысск 2021

В методических указаниях изложены основные сведения о методах решения нелинейных математических задач с помощью современных программных средств.

Составитель *канд. техн. наук, доцент Д.В. Болдырев*

Отв. редактор *канд. техн. наук А.А. Евдокимов*

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Целью работы является усвоение студентом основных методов решения нелинейных уравнений и их систем и получение навыков выполнения этих операций с помощью современных программных средств.

1 ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

1.1 Решение нелинейных уравнений

Методы поиска корней применяются, если необходимо решить задачу вида:

$$f(x) = 0, \tag{1.1}$$

где $f(x)$ — некоторая нелинейная функция.

Если уравнение (1.1) содержит только алгебраические функции (степенные, рациональные и т. п.), оно считается **алгебраическим**. Если в его состав входят другие функции (тригонометрические, показательные, логарифмические и т. п.), оно считается **трансцендентным**.

Методы поиска корней уравнений бывают двух видов.

Прямые (аналитические) методы целесообразно использовать, когда исходное уравнение можно явно разрешить относительно величины x, или можно получить некоторые конечные соотношения для расчета корней (пример — формулы Кардано вычисления корней полиномов до 4-го порядка включительно). Эти методы всегда обеспечивают получение *точного решения*.

Итерационные (численные) методы работают по принципу многократного применения одного и того же алгоритма, который всегда включает в себя следующие этапы:

- отыскание приближенного значения корня или содержащего его интервала значений x;

- последовательное уточнение приближенного значения корней до достижения некоторой заданной степени точности.

Каждый шаг подобного уточнения называется *итерацией*.

Итерационные методы бывают одношаговые и многошаговые. *Одношаговые методы* для нахождения последующего значения $x^{(k)}$ на k-м шаге используют информацию только об одной предыдущей итерации:

$$x^{(k)} = \phi\left[x^{(k-1)}\right], \tag{1.2}$$

где $\phi\left[x^{(k-1)}\right]$ — оператор преобразования, определяемый выбранным алгоритмом.

Многошаговые методы при решении используют информацию о двух и более предыдущих итерациях:

$$x^{(k)} = \phi\left[x^{(k-1)}, x^{(k-2)}, \ldots x^{(k-m)}\right]. \tag{1.3}$$

В ходе итераций формируется последовательность значений корня $\left\{x^{(k)}, k = 0,1,..\infty\right\}$. Если существует предел $\lim\limits_{k\to\infty} x^{(k)}$, то говорят, что итерационный процесс *сходится*. Если такого предела не существует, вычислительный процесс *расходится*. Обычно за конечное число итераций этот предел не достигается, поэтому работа метода продолжается до выполнения некоторого условия. Рациональным считается использование комбинации критериев:

$$\left| f\left[x^{(k)}\right] \right| < \varepsilon \cdot \left\{1 + \left| f\left[x^{(k)}\right] \right|\right\}, \tag{1.4}$$

$$\left| x^{(k)} - x^{(k-1)} \right| < \sqrt{\varepsilon} \cdot \left\{1 + \left| x^{(k)} \right| \right\}, \tag{1.5}$$

где ε — точность решения. Выполнение условия (1.4) позволит избежать преждевременной остановки метода на участке функции $f(x)$ с большой крутизной и малым изменением аргумента. Соблюдение критерия (1.5) гарантирует, что работа метода будет продол-

жаться на пологих участках $f(x)$.

Решение, полученное любым численным методом, всегда является *приближенным*, хотя оно может быть сколь угодно близко к точному решению.

Итерационные методы делятся на две группы.

Интервальные алгоритмы ведут поиск на отрезке значений аргумента $x \in [a, b]$, который включает в себя корень.

> ***Теорема***. *Если* $f(x)$ *— действительная функция, непрерывная в интервале от* $x = a$ *до* $x = b$, *где* a *и* b *— действительные числа, и если значения* $f(a)$ *и* $f(b)$ *имеют противоположные знаки, то между* a *и* b *имеется по крайней мере один действительный корень.*

Приближенное значение корня на k-й итерации выбирается из условия $a \le x^{(k)} \le b$. Алгоритмы методов всегда содержат шаг, на котором исходная функция $f(x)$ исследуется на границах отрезков $[a, x^{(k)}]$ и $[x^{(k)}, b]$. В качестве нового отрезка поиска выбирается тот, на границах которого $f(x)$ *имеет значения разных знаков* (произведение $f(a) \cdot f(x)$ или $f(b) \cdot f(x)$ отрицательно).

Достоинство этих методов — абсолютная надежность. Если функция $f(x)$ на отрезке $x \in [a, b]$ непрерывна и имеет там корень, он обязательно будет обнаружен.

Недостатки методов — необходимость предварительного знания отрезка локализации корня и большое число расчетов функции $f(x)$. Если эти расчеты достаточно сложны, быстродействие алгоритма значительно снижается.

Поисковые алгоритмы для своей работы требуют задания единственного начального приближения корня, которое корректируется по некоторым правилам. Их *достоинство* — высокая скорость сходимости (обычно на порядок выше, чем у методов предыдущей группы). Их *недостаток* — высокая чувствительность к выбору начального приближения. Его некорректное задание приводит к тому, что итерационный процесс *расходится или зацикливается*.

1.2 Интервальные методы решения нелинейных уравнений

Такие методы используют следующий обобщенный алгоритм:

```
ОПРЕДЕЛЕНИЕ ИСХОДНЫХ ДАННЫХ:
  a, b — границы отрезка поиска
  ε — точность поиска
ПОКА НЕ НАЙДЕН КОРЕНЬ, ВЫПОЛНИТЬ
 // Определение нового приближения x
 x ← ϕ(a, b)
 // Анализ функции в точках a, x, b
 ЕСЛИ f(a)·f(x) > 0 a ← x
 ИНАЧЕ b ← x
```

В **методе половинного деления** (или методе бисекции, или методе Больцано) в качестве приближения корня используется координата середины отрезка **[*a*, *b*]**:

$$\phi(a,b) = \frac{a+b}{2}. \tag{1.6}$$

В **методе хорд** (или методе ложного положения) приближением корня считается координата точки пересечения хорды ***AB*** с горизонтальной осью. Ее уравнение имеет вид:

$$\frac{f(x)-f(a)}{f(b)-f(a)} = \frac{x-a}{b-a}. \tag{1.7}$$

Выражение для расчета координаты точки пересечения ***AB*** с осью ***OX*** (при $f(x) = 0$) определяет форму функции ϕ:

$$\phi(a,b) = a - \frac{f(a)}{f(b)-f(a)} \cdot (b-a). \tag{1.8}$$

Схемы работы этих методов показаны на рисунке 1.1.

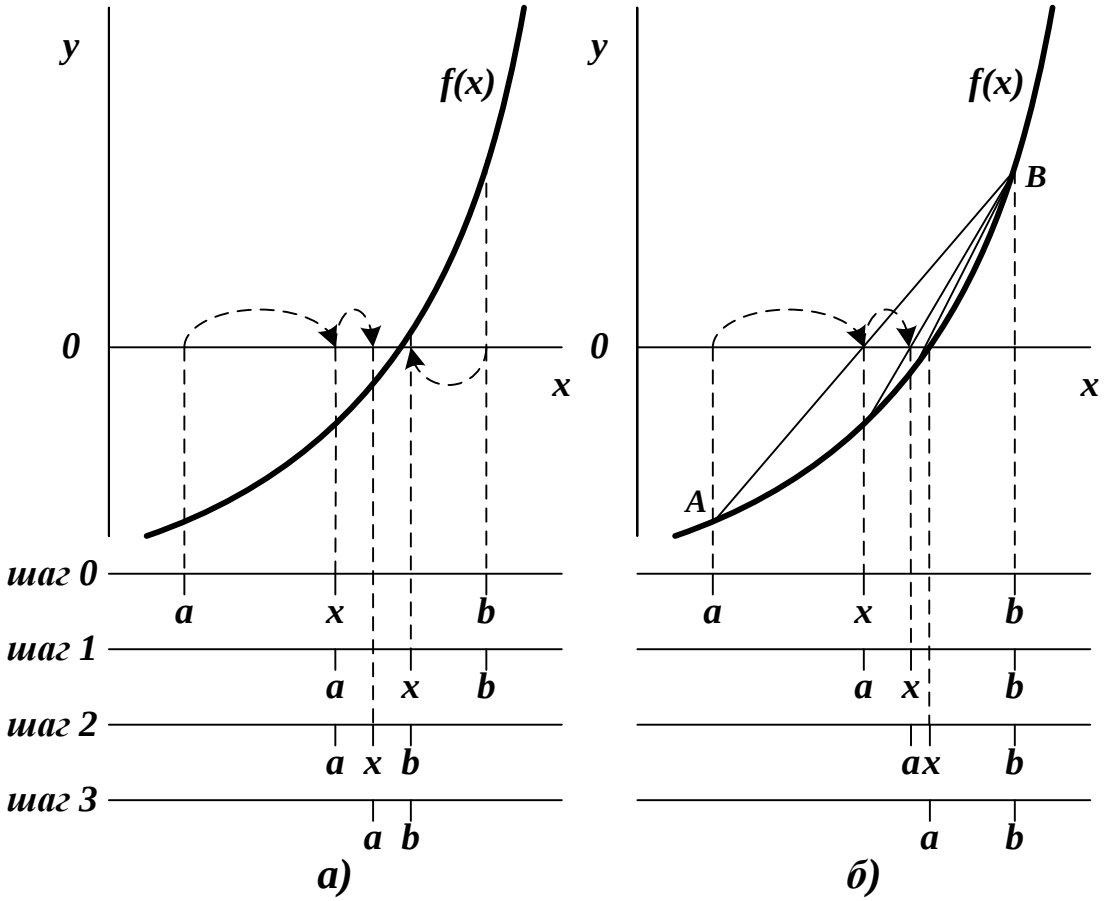


Рисунок 1.1 — Схемы работы интервальных методов: а) метод половинного деления; б) метод хорд

В общем случае для не пологих функций метод хорд обеспечивает более быструю сходимость.

В методе **обратной параболической интерполяции** используется функция $\varphi(x)$, обратная $f(x)$. Если x — корень уравнения (1.1), то $\varphi(0) = x$. Функция $\varphi(x)$ приближенно заменяется параболой, уравнение которой находится с помощью обратного интерполяционного полинома Ньютона [1] (для этого на отрезке $[a, b]$ необходимо любым способом получить одну внутреннюю точку c):

$$\begin{gathered}\varphi(x) = k_0 + k_1 \cdot [f(x) - f(a)] + \\ + k_2 \cdot [f(x) - f(a)] \cdot [f(x) - f(c)] \ , \\ \varphi(0) = x = k_0 - k_1 \cdot f(a) + k_2 \cdot f(a) \cdot f(c),\end{gathered} \qquad (1.9)$$

где $f(x) = 0$ — желаемое значение функции в точке x. Коэффициенты k_0, k_1 и k_2 находятся из системы уравнений:

$$\begin{cases} k_0 = a \\ k_0 + k_1 \cdot [f(c) - f(a)] = c \\ k_0 + k_1 \cdot [f(b) - f(a)] + k_2 \cdot [f(b) - f(a)] \cdot [f(b) - f(c)] = b \end{cases} \quad (1.10)$$

Данный метод использует следующий алгоритм.

```
ОПРЕДЕЛЕНИЕ ИСХОДНЫХ ДАННЫХ:
  a, b границы отрезка поиска
  ε — точность поиска
  c ← (a + b) / 2 — внутренняя точка
ПОКА НЕ НАЙДЕН КОРЕНЬ, ВЫПОЛНИТЬ
 // Определение нового приближения x
 k₀ ← a
 k₁ ← [c - k₀] / [f(c) - f(a)]
 k₂ ← [b - k₀ - k₁·[f(b) - f(a)]] /
      [f(c) - f(a)] / [f(b) - f(c)]
 x ← k₀ - k₁·f(a) + k₂·f(a)·f(c)
 // Половинное деление при неудачной
 // интерполяции, если x ∉ [a, b]
 ЕСЛИ (x - a)·(b - x) < 0 x ← (a + b) / 2
 // Сужение отрезка поиска.
 ЕСЛИ f(a)·f(c) > 0 a ← c
 ИНАЧЕ b ← c
 c ← x
```

Схема одного этапа работы метода показана на рисунке 1.2. Метод обратной параболической интерполяции обеспечивает высокую скорость сходимости (для гладких функций решение находится за ***6-8*** шагов).

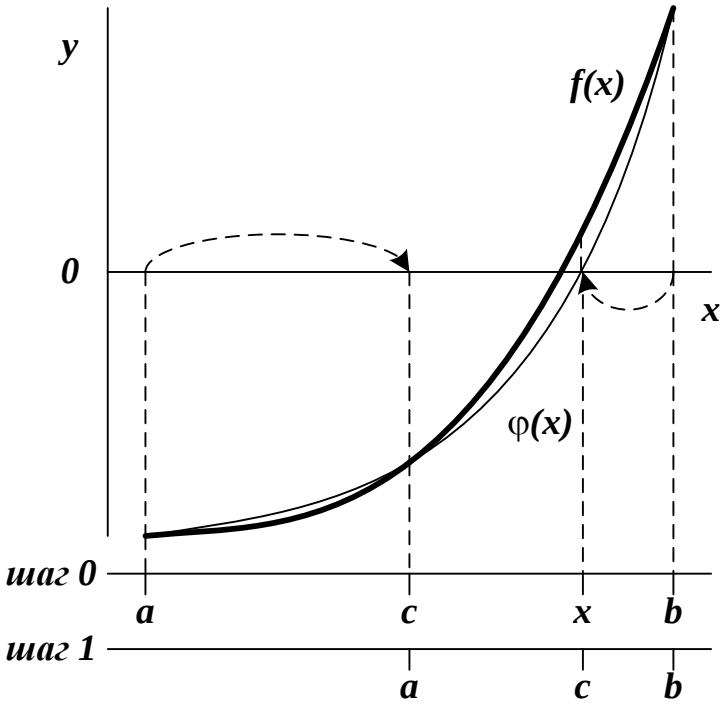


Рисунок 1.2 — Схема работы метода обратной параболической интерполяции

1.3 Поисковые методы решения нелинейных уравнений

Такие методы используют следующий обобщенный алгоритм:

```
ОПРЕДЕЛЕНИЕ ИСХОДНЫХ ДАННЫХ:
  x(0) — начальное приближение корня
  ε — точность поиска
  κ ← 0 // Начальный номер итерации
ПОКА НЕ НАЙДЕН КОРЕНЬ, ВЫПОЛНИТЬ
  // Определение нового приближения x
  x(k+1) ← ϕ(x(k))
  κ ← κ + 1
```

Метод простых итераций заключается в том, что из (1.1) явно выражается x и формируется эквивалентное уравнение:

$$x = \phi(x). \qquad (1.11)$$

Итерации строятся по правилам:

$$x^{(k+1)} = \phi\left[x^{(k)}\right] . \qquad (1.12)$$

Метод сходится только в том случае, если выполняется условие $|\phi'(x)| < 1$. Это существенно ограничивает область его применения.

Схемы работы метода простых итераций при различной сходимости показаны на рисунке 1.3.

В **методе релаксации** используется следующая итерационная формула:

$$x^{(k+1)} = \phi\left[x^{(k)}\right] = x^{(k)} - \tau \cdot f\left[x^{(k)}\right] , \qquad (1.13)$$

где значение константы τ выбирается из условия сходимости:

$$\begin{gathered} \phi(x) = x - \tau \cdot f(x), \\ |\phi'(x)| = |1 - \tau \cdot f'(x)| < 1, \\ 0 < \tau \cdot f'(x) < 2. \end{gathered} \qquad (1.14)$$

Если в некоторой области корня выполняются условия

$$0 < m < |f'(x)| < M), \qquad (1.15)$$

где m и M — положительные константы, то метод релаксации сходится при $\tau \in (0, 2 / M)$. Удачным выбором считается значение $\tau = 2 / (M + m)$.

В **методе Ньютона** (или методе касательных) функция $f(x)$ заменяется линейным разложением в ряд Тейлора в точке $x^{(k)}$:

$$f(x) \approx f\left[x^{(k)}\right] + f'\left[x^{(k)}\right] \cdot \left[x - x^{(k)}\right] . \qquad (1.16)$$

За следующее приближение $x^{(k+1)}$ принимается корень уравнения (1.16) (при $f(x) = 0$):

$$x^{(k+1)} = \phi\left[x^{(k)}\right] = x^{(k)} - \frac{f\left[x^{(k)}\right]}{f'\left[x^{(k)}\right]} . \qquad (1.17)$$

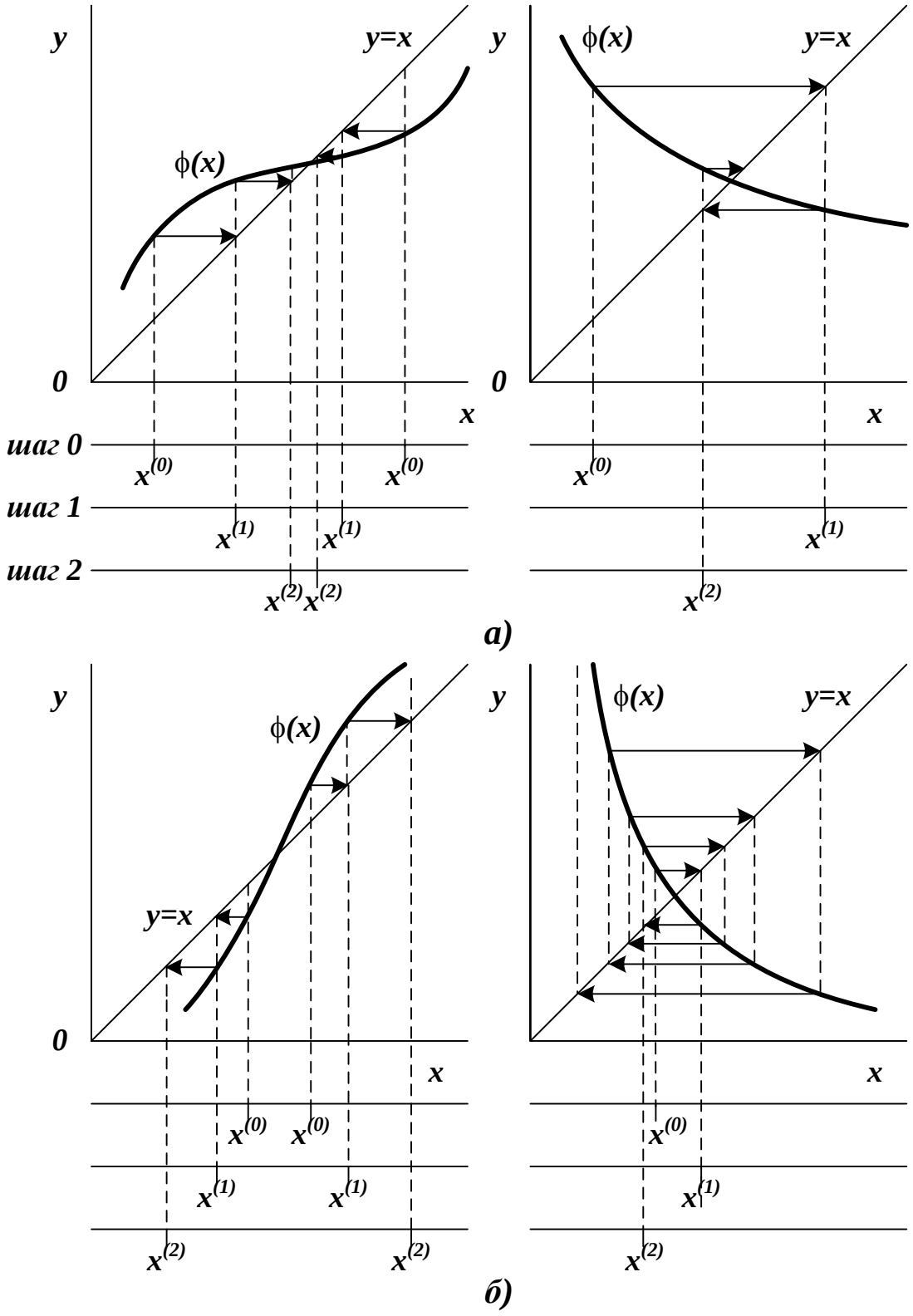


Рисунок 1.3 — Схемы работы метода простых итераций: а) при наличии сходимости; б) при отсутствии сходимости

Геометрическая интерпретация метода Ньютона заключается в замене функции $f(x)$ касательной в точке $x^{(k)}$ (отсюда второе название алгоритма) и определения координаты точки пересечения этой касательной с горизонтальной осью. Схема работы метода Ньютона показана на рисунке (1.4, а).

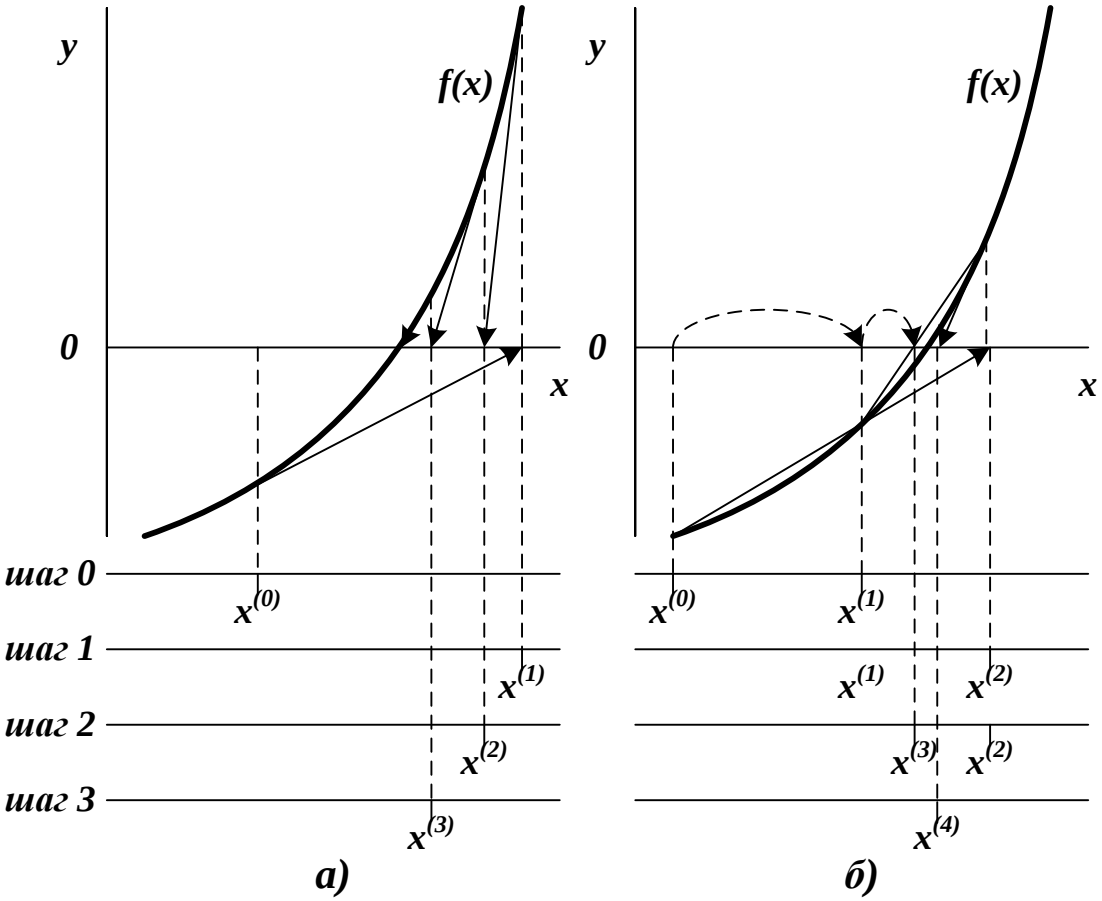


Рисунок 1.4 — Схемы работы поисковых методов: а) метода Ньютона; б) метода секущих

Метод Ньютона обеспечивает высокую скорость сходимости (для гладких функций — ***4-8*** итераций). Начальное приближение $x^{(0)}$ рекомендуется выбирать там, где выполняется условие:

$$f\left[x^{(0)}\right] \cdot f''\left[x^{(0)}\right] \geq 0. \qquad (1.18)$$

Ошибки при выборе $x^{(0)}$ могут являться причиной некорректной работы алгоритма (см. рис. 1.5).

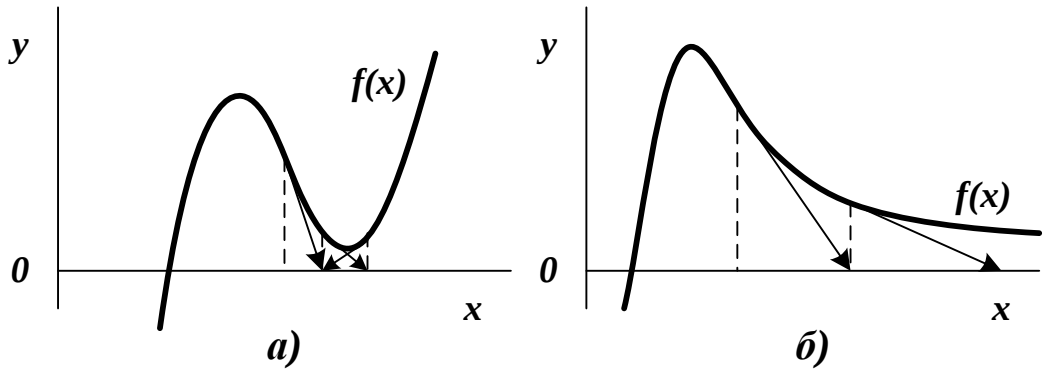


Рисунок 1.5 — Некорректная работа метода Ньютона: а) метод зацикливается; б) итерации расходятся

Если хотят избежать многократного вычисления производной $f'(x)$, то используют следующую модификацию метода Ньютона:

$$x^{(k+1)} = \phi\left[x^{(k)}\right] = x^{(k)} - \frac{f\left[x^{(k)}\right]}{f'\left[x^{(0)}\right]}. \qquad (1.19)$$

Модифицированный метод Ньютона предъявляет меньше требований к выбору начального приближения корня и гарантирует отсутствие деления на 0 при $f'[x^{(0)}] \neq 0$. Однако скорость его сходимости падает.

Если вычисление производной от функции $f(x)$ затруднено или нецелесообразно, можно использовать ее приближенное разностное представление:

$$f'\left[x^{(k)}\right] \approx \frac{f\left[x^{(k)}\right] - f\left[x^{(k-1)}\right]}{x^{(k)} - x^{(k-1)}}. \qquad (1.20)$$

Так формируется вычислительная схема **метода секущих**:

$$x^{(k+1)} = \phi\left[x^{(k)}\right] = x^{(k)} - \frac{f\left[x^{(k)}\right]}{f\left[x^{(k)}\right] - f\left[x^{(k-1)}\right]} \cdot \left[x^{(k)} - x^{(k-1)}\right]. \qquad (1.21)$$

Геометрическая интерпретация метода состоит в следующем. Через точки с координатами $\{x^{(k)}, f[x^{(k)}]\}$ и $\{x^{(k-1)}, f[x^{(k-1)}]\}$ проводится прямая линия. Новым приближением корня $x^{(k+1)}$ является точка

ее пересечения с горизонтальной осью. Если $x^{(k+1)} \in [x^{(k-1)}, x^{(k)}]$, то метод секущих становится эквивалентным методу хорд. Схема работы алгоритма показана на рисунке (1.4, б).

Метод является многошаговым, поэтому для начала его работы необходимо задание двух исходных значений — $x^{(0)}$ и $x^{(1)}$.

1.4 Решение полиномиальных уравнений

Полиномиальные уравнения с действительными коэффициентами имеют вид:

$$f(x) = a_n \cdot x^n + \ldots + a_2 \cdot x^2 + a_1 \cdot x + a_0 = A_n(x) = 0. \qquad (1.22)$$

Они обладают следующими свойствами.

1. Уравнение степени n всегда имеет n корней (возможно кратных). Корни могут быть как действительными, так и комплексными (см. рис. 1.5). Комплексные корни образуют сопряженные пары:

$$x^* = \alpha + j \cdot \omega, \quad \bar{x}^* = \alpha - j \cdot \omega,$$

$$\left[x - x^*\right] \cdot \left[x - \bar{x}^*\right] = \left[x - (\alpha + j \cdot \omega)\right] \cdot \left[x - (\alpha - j \cdot \omega)\right] = \qquad (1.23)$$

$$= x^2 - 2 \cdot \alpha \cdot x + \left(\alpha^2 + \omega^2\right) = x^2 + p \cdot x + q.$$

Уравнение нечетной степени всегда имеет хотя бы один действительный корень.

2. Если x_R — действительный корень уравнения (1.22), то справедливо соотношение:

$$A_n(x) = (x - x_R) \cdot B_{n-1}(x), \qquad (1.24)$$

где $B_{n-1}(x)$ — полином степени $n-1$, образованный путем сокращения уравнения (1.22) на $(x - x_R)$. Степень исходного уравнения при этом понижается на 1.

3. Если x_C и $\bar{x}_C$ — пара комплексно-сопряженных корней уравнения (1.22), то справедливо соотношение:

$$A_n(x) = (x - x_C) \cdot (x - \bar{x}_C) \cdot B_{n-2}(x), \qquad (1.25)$$

где $B_{n-2}(x)$ — полином степени $n-2$, образованный путем сокращения исходного уравнения на $(x - x_C) \cdot (x - \bar{x}_C)$. Степень исходного уравнения при этом понижается на 2.

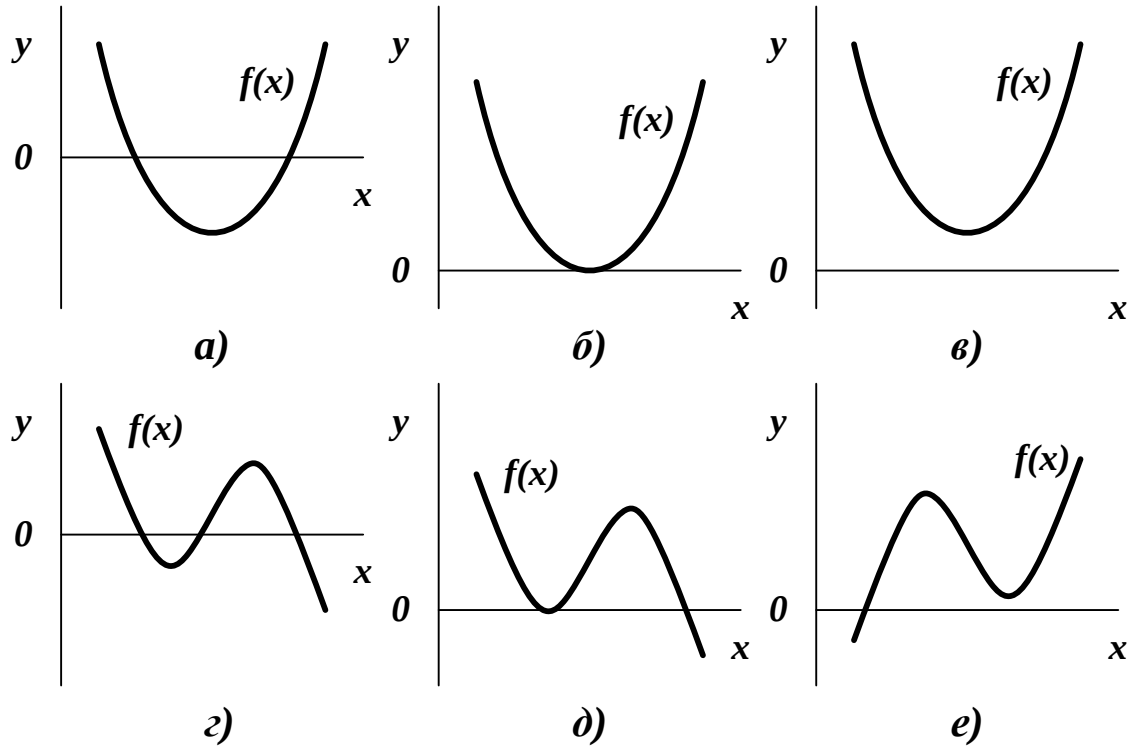


Рисунок 1.5 — Различные варианты расположения корней полиномиальных уравнений: а) $f(x) = A_2(x)$ — два некратных действительных корня; б) $f(x) = A_2(x)$ — два кратных действительных корня; в) $f(x) = A_2(x)$ — два комплексно-сопряженных корня; г) $f(x) = A_3(x)$ — три некратных действительных корня; д) $f(x) = A_3(x)$ — один некратный и два кратных действительных корня; е) $f(x) = A_3(x)$ — один действительный и два комплексно-сопряженных корня

Одним из методов определения действительных и комплексных корней полинома порядка $n > 1$ — **метод Лина** — заключается в его последовательном сокращении на $x^2 + p \cdot x + q$, определении на каждом шаге коэффициентов p и q и аналитическом решении урав-

нения $x^2 + p \cdot x + q = 0$. Уравнение (1.22) представляется в виде:

$$\begin{aligned} &a_n \cdot x^n + a_{n-1} \cdot x^{n-1} + \ldots + a_2 \cdot x^2 + a_1 \cdot x + a_0 = \\ &= (x^2 + p \cdot x + q) \cdot (b_n \cdot x^{n-2} + \ldots + b_3 \cdot x + b_2) + b_1 \cdot x + b_0, \end{aligned} \tag{1.26}$$

где $b_1 x + b_0$ — остаток от деления на квадратный двучлен. Приравнивая коэффициенты в левой и правой частях (1.26) при равных степенях x, можно записать соотношения:

$$\begin{aligned} a_n &= b_n, \\ a_{n-1} &= b_{n-1} + b_n \cdot p, \\ a_{n-2} &= b_{n-2} + b_{n-1} \cdot p + b_n \cdot q, \\ &\ldots \\ a_k &= b_k + b_{k+1} \cdot p + b_{k+2} \cdot q, \\ &\ldots \\ a_1 &= b_1 + b_2 \cdot p + b_3 \cdot q, \\ a_0 &= b_0 + b_2 \cdot q. \end{aligned} \tag{1.27}$$

Если в выражение $x^2 + p \cdot x + q$ в качестве x подставить корни (1.22), то остаток от деления будет равен нулю ($b_1 = b_0 = 0$). Значения b_k можно последовательно рассчитать по формулам:

$$\begin{aligned} b_n &= a_n, \\ b_{n-1} &= a_{n-1} - b_n \cdot p, \\ b_{n-2} &= a_{n-2} - b_{n-1} \cdot p - b_n \cdot q, \\ &\ldots \\ b_k &= a_k - b_{k+1} \cdot p - b_{k+2} \cdot q, \\ &\ldots \\ b_1 &= a_1 - b_2 \cdot p - b_3 \cdot q = 0, \\ b_0 &= a_0 - b_2 \cdot q = 0. \end{aligned} \tag{1.28}$$

Алгоритм метода следующий.

Шаг 1. Задание коэффициентов a_k, $k = 0,\ldots n$ и точности ε.

Шаг 2. Задание начальных приближений p и q.

Шаг 3. Расчет коэффициентов b_k, $k = n,\ldots 2$ по формулам (1.28).

Шаг 4. Расчет по двум последним уравнениям (1.28) новых значений p и q:

$$q = \frac{a_0}{b_2}, \quad p = \frac{a_1 - b_3 \cdot q}{b_2}. \tag{1.29}$$

Пересчет p и q (шаги 3 и 4) повторяется до тех пор, пока их значения не определятся с заданной точностью.

Шаг 5. Определение пары действительных или комплексно-сопряженных корней полинома аналитическим решением уравнения $x^2 + p \cdot x + q = 0$.

Шаг 6. Сокращение полинома на $x^2 + p \cdot x + q$:

$$a_k = b_{k+2}, \quad k = 0...n-2, \quad n = n-2. \tag{1.30}$$

Шаги ***2…6*** повторяются до тех пор, пока не будут определены все корни уравнения (1.22). Если исходный полином имеет нечетную степень, то последний действительный корень определяется аналитически.

В процессе понижения степени полинома коэффициенты b_k вычисляются с погрешностью, определяемой разрядностью аппаратных или программных средств, что существенно влияет на значения корней.

Пример Уилкинсона. *Пусть дано полиномиальное уравнение:*

$$(x-1)\cdot(x-2)\cdot\ldots\cdot(x-20) = x^{20} - 210\cdot x^{19} + \ldots = 0.$$

Его корни очевидны — ***1, 2, 3, 4,… 20****. Решение этой задачи программными средствами (например, с помощью* ***MathCAD****), в которых числовые константы задаются с погрешностью* $\approx 10^{-16}$*, содержит только четыре действительных значения* ***1, 2, 3, 3.91****. Остальные компоненты решения будут сопряженными комплексными числами.*

Вычислительный процесс рекомендуется начинать с определения меньших по модулю корней (выбирать начальные приближения $0, \pm 1$ и т. п.), чтобы сразу удалить их из уравнения до того, как накопится существенная ошибка.

1.5 Решение систем нелинейных уравнений

Системы нелинейных уравнений имеют вид:

$$F(x) = 0, \tag{1.31}$$

где

$X = \begin{bmatrix} x_1 \\ \dots \\ x_n \end{bmatrix}$ — вектор-столбец неизвестных,

$F(X) = \begin{bmatrix} f_1(X) \\ \dots \\ f_m(X) \end{bmatrix}$ — векторная функция уравнений системы.

Если $m = n$, система (1.31) считается **определенной**, если $m > n$ — **переопределенной**, если $m < n$ — **недоопределенной**. У недоопределенных систем однозначно находятся только m значений X, а остальные считаются произвольными (обычно, чтобы получить вектор решений минимальной длины, их считают нулевыми). В дальнейшем рассматриваются только методы решения определенных систем.

Для нахождения корней систем вида (1.31) обычно используют *поисковые методы*, которые можно записать в обобщенной форме:

$$\frac{B^{(k+1)}}{\tau^{(k+1)}} \cdot \left[X^{(k+1)} - X^{(k)}\right] + F\left[X^{(k)}\right] = 0, \tag{1.32}$$

где k — номер итерации, $B^{(k+1)}$ — некоторая матрица полного ранга, форма которой определяет алгоритм метода, $\tau^{(k+1)}$ — итерационный параметр, повышающий скорость сходимости. Если B и τ не зависят от k, метод считают *стационарным*. Если $B = I$ (I — единичная матрица), метод считают *явным*. Явные методы обычно обеспечивают более низкую скорость сходимости.

Алгоритмы решения систем нелинейных уравнений являются

естественным расширением одномерных вычислительных схем. Из (1.32) можно получить их обобщенную итерационную формулу:

$$X^{(k+1)} = X^{(k)} - \tau^{(k+1)} \cdot \left[B^{(k+1)}\right]^{-1} \cdot F\left[X^{(k)}\right]. \qquad (1.33)$$

В зависимости от вида матрицы B и значений τ различают следующие методы.

Итерационные формулы **метода простых итераций** получаются путем явного выражения X из (1.31):

$$X^{(k+1)} = \Phi\left[X^{(k)}\right]. \qquad (1.34)$$

Итерации сходятся только в том случае, если для евклидовой длины матрицы

$$\Phi'(X) = \begin{bmatrix} \frac{\partial \phi_1(X)}{\partial x_1} & \cdots & \frac{\partial \phi_1(X)}{\partial x_n} \\ \cdots & \cdots & \cdots \\ \frac{\partial \phi_n(X)}{\partial x_1} & \cdots & \frac{\partial \phi_n(X)}{\partial x_n(X)} \end{bmatrix}$$

выполняется условие $\|\Phi'(X)\| < 1$. Вариантом данного алгоритма можно считать **метод Гаусса-Зейделя**:

$$\begin{cases} x_1^{(k+1)} = \phi_1\left[x_1^{(k)}, x_2^{(k)}, \ldots x_n^{(k)}\right], \\ x_2^{(k+1)} = \phi_2\left[x_1^{(k+1)}, x_2^{(k)}, \ldots x_n^{(k)}\right], \\ \cdots \\ x_i^{(k+1)} = \phi_i\left[x_1^{(k+1)}, \ldots x_{i-1}^{(k+1)}, x_i^{(k)}, \ldots x_n^{(k)}\right], \\ \cdots \\ x_n^{(k+1)} = \phi_n\left[x_1^{(k+1)}, \ldots x_{n-1}^{(k+1)}, x_n^{(k)}\right]. \end{cases} \qquad (1.35)$$

В **методе релаксации** считается, что $B^{(k+1)} = I$, $\tau^{(k+1)} = \tau$:

$$X^{(k+1)} = X^{(k)} - \tau \cdot F\left[X^{(k)}\right]. \qquad (1.36)$$

Значение константы τ выбирается из условия сходимости:

$$\Phi(X) = X - \tau \cdot F(X),$$
$$\|\Phi'(X)\| = \|I - \tau \cdot J(X)\| < 1, \qquad (1.37)$$

где

$$J(X) = \begin{bmatrix} \frac{\partial f_1(X)}{\partial x_1} & \dots & \frac{\partial f_1(X)}{\partial x_n} \\ \dots & \dots & \dots \\ \frac{\partial f_n(X)}{\partial x_1} & \dots & \frac{\partial f_n(X)}{\partial x_n(X)} \end{bmatrix}$$ — матрица Якоби.

Метод Ньютона для уточнения значения корней использует линейное разложение функций $F(X)$ по формуле Тейлора в окрестности точки $X^{(k)}$:

$$\begin{cases} f_1\left[X^{(k+1)}\right] = f_1\left[X^{(k)}\right] + \sum_{i=1}^{n}\left\{\frac{\partial f_1}{\partial x_i}\right\}^{(k)} \cdot \left[x_i^{(k+1)} - x_i^{(k)}\right], \\ \dots \\ f_n\left[X^{(k+1)}\right] = f_n\left[X^{(k)}\right] + \sum_{i=1}^{n}\left\{\frac{\partial f_n}{\partial x_i}\right\}^{(k)} \cdot \left[x_i^{(k+1)} - x_i^{(k)}\right]. \end{cases} \qquad (1.38)$$

В соответствии с (1.31) левые части этих уравнений должны обращаться в 0. Тогда система уравнений (1.38) в матричной форме примет вид:

$$J\left[X^{(k)}\right] \cdot \left[X^{(k+1)} - X^{(k)}\right] + F\left[X^{(k)}\right] = 0, \qquad (1.39)$$

что соответствует $B^{(k+1)} = J\left[X^{(k)}\right]$ и $\tau^{(k+1)} = 1$. Итерационную схему определяет формула:

$$X^{(k+1)} = X^{(k)} - J^{-1}\left[X^{(k)}\right] \cdot F\left[X^{(k)}\right]. \qquad (1.40)$$

Если вычисление и обращение матрицы Якоби затруднено, то метод Ньютона может быть модифицирован (в ущерб сходимости):

$$X^{(k+1)} = X^{(k)} - J^{-1}\left[X^{(0)}\right] \cdot F\left[X^{(k)}\right] . \qquad (1.41)$$

Для повышения устойчивости и скорости работы алгоритма формулу (1.40) можно дополнить параметром:

$$X^{(k+1)} = X^{(k)} - \tau^{(k+1)} \cdot J^{-1}\left[X^{(k)}\right] \cdot F\left[X^{(k)}\right] . \qquad (1.42)$$

Методы решения систем нелинейных уравнений обладают всеми достоинствами и недостатками одномерных вычислительных схем, рассмотренных в п. 1.3. Успех их работы во многом определяется удачным выбором начального приближения $X^{(0)}$, которое должно быть достаточно близким к точному решению.

Работа поисковых методов продолжается до выполнения одного из двух (или обоих сразу) условий, аналогичных (1.4) и (1.5):

$$\left\| F\left[X^{(k)}\right] \right\| < \varepsilon \cdot \left\{ 1 + \left\| F\left[X^{(k)}\right] \right\| \right\}, \qquad (1.43)$$

$$\left\| X^{(k)} - X^{(k-1)} \right\| < \sqrt{\varepsilon} \cdot \left\{ 1 + \left\| X^{(k)} \right\| \right\}. \qquad (1.44)$$

2 РЕШЕНИЕ НЕЛИНЕЙНЫХ ЗАДАЧ В MathCAD

Для решения одного уравнения с одним неизвестным используется функция ***root***, основанная на методе секущих.

Имя функции	Возвращается...
root(f(x), x)	скалярное значение, при котором ***f(x)*** обращается в ***0***. Первый аргумент — скалярная функция или выражение. Второй аргумент — скалярное начальное приближение корня.

Если ***f(x)*** имеет несколько корней, то, задавая различные начальные приближения, можно определить их все. Если корни располагаются близко друг к другу, и один из них $x = a$ уже опре-

делен, то для нахождения остальных корней целесообразнее решать эквивалентную задачу:

$$h(x) = \frac{f(x)}{x - a} = 0.$$

Если ***f(x)*** имеет малый наклон в окрестности корня, функция ***root*** может дать очень грубое решение. В таких случаях рекомендуется произвести замену ***f(x)*** на функцию:

$$g(x) = \frac{f(x)}{\frac{d}{dx} f(x)}.$$

Если функция ***root*** не может найти решение, возвращается начальное приближение корня.

Для вычисления корней полинома рекомендуется функция ***polyroots***, не требующая задания начального приближения.

Имя функции	Возвращается...
polyroots(V)	вектор длины ***n***, содержащий все действительные и комплексные корни полинома степени ***n***. ***V*** — вектор длины ***n+1***, содержащий действительные и комплексные коэффициенты полинома.

Примеры использования функций ***root*** и ***polyroots*** показаны на рисунке 2.1.

MathCAD позволяет решать системы с максимальным числом уравнений и неизвестных, равным ***50***. Для этого используются функции ***Find*** и ***Minerr*** и решающий блок ***Given***.

Имя функции	Возвращается...
Find(x_1, x_2, …) ***Minerr(x_1, x_2, …)***	решение системы уравнений. Число уравнений должно быть равно числу неизвестных.

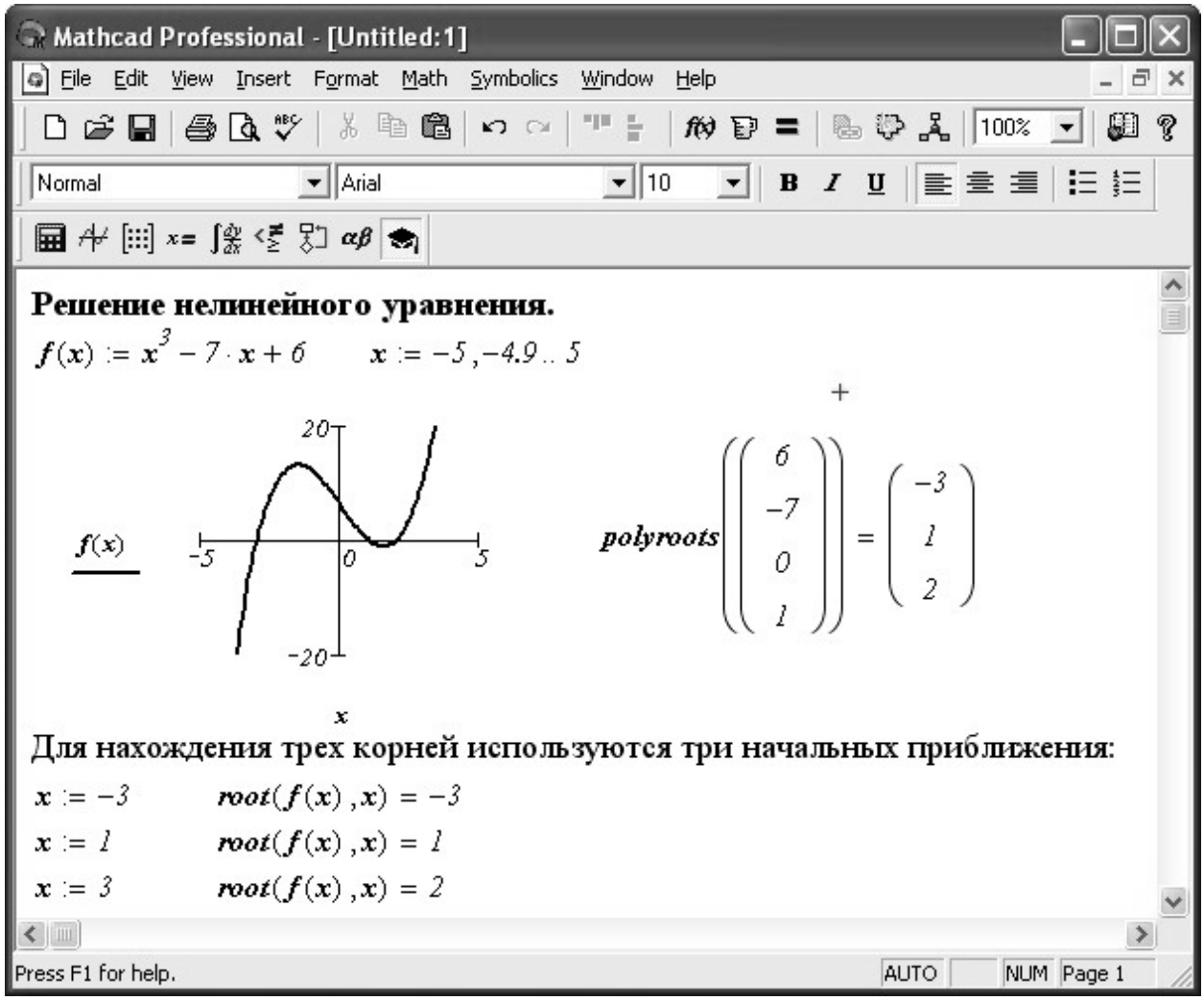


Рисунок 2.1 — Использование функций ***root*** и ***polyroots***

Решение системы производится в следующем порядке:

- задаются начальные приближения для всех неизвестных, входящих в систему;
- открывается блок ***Given*** (для набора ключевого слова можно использовать любой шрифт, прописные и строчные буквы);
- ниже ключевого слова ***Given*** вводятся все уравнения и неравенства системы (порядок произвольный, для набора знаков логических операций рекомендуется использовать соответствующую палитру инструментов);
- вводится любое выражение, содержащее функцию ***Find*** (для набора ключевого слова можно использовать любой шрифт,

прописные и строчные буквы).

Внутри решающего блока ***Given*** не допускается использовать:

- операции присваивания;
- дискретный аргумент и выражения, содержащие дискретный аргумент в любой форме;
- ограничения вида $a \neq b$ и $a < b < c$;
- внутренние решающие блоки.

Если функция ***Find*** имеет один аргумент, возвращается корень уравнения, включенного в блок ***Given***. Если эта функция имеет несколько аргументов, возвращается вектор корней системы уравнений, включенных в решающий блок.

Полученное решение можно вывести на экран, используя выражение вида ***Find(…) = …*** или присвоить скалярной переменной или вектору с помощью конструкции ***var := Find(…)***. Можно также определить с помощью ***Find*** другую функцию (это удобно для многократного решения параметризированных задач).

Примеры решения систем нелинейных уравнений показаны на рисунках 2.2 и 2.3.

Если задача сформулирована некорректно, или на какой либо итерации ***MathCAD*** не может найти более приемлемое решение, то вычисления прекращаются. Основные причины возникновения ошибок следующие:

- поставленная задача не имеет решения — рекомендуется изменить ее формулировку;
- число уравнений меньше числа переменных — в ряде случаев рекомендуется добавить фиктивные уравнения вида ***1 = 1*** и т. п.;
- достигнута точка, из которой не может быть получено более точное приближение к решению — рекомендуется изменить начальные приближения;
- достигнут предел точности вычислений, на которые начинают влиять ошибки округления, — изменить значение встро-

енной переменной ***TOL*** (не менее 10^{-15}).

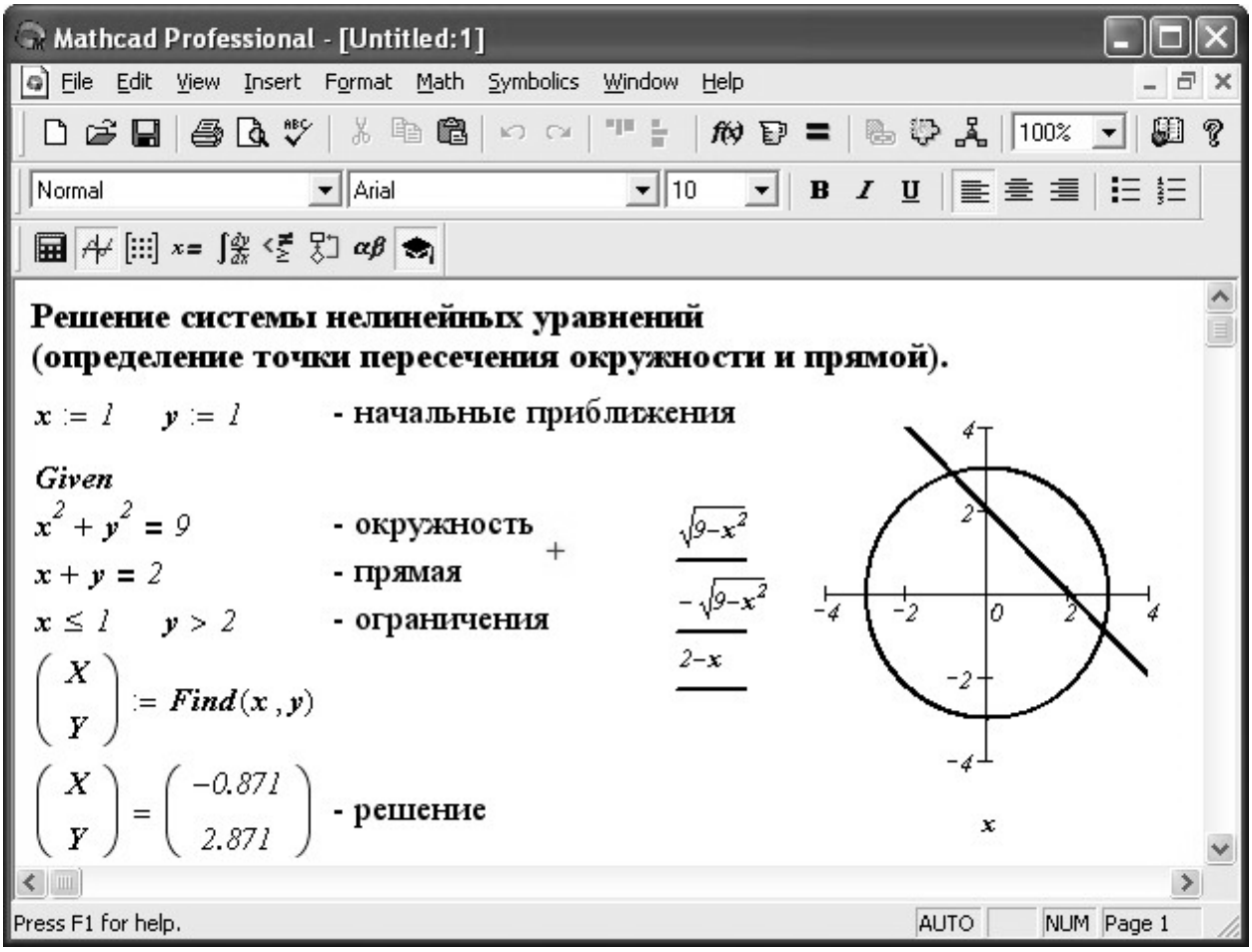


Рисунок 2.2 — Использование функции ***Find***

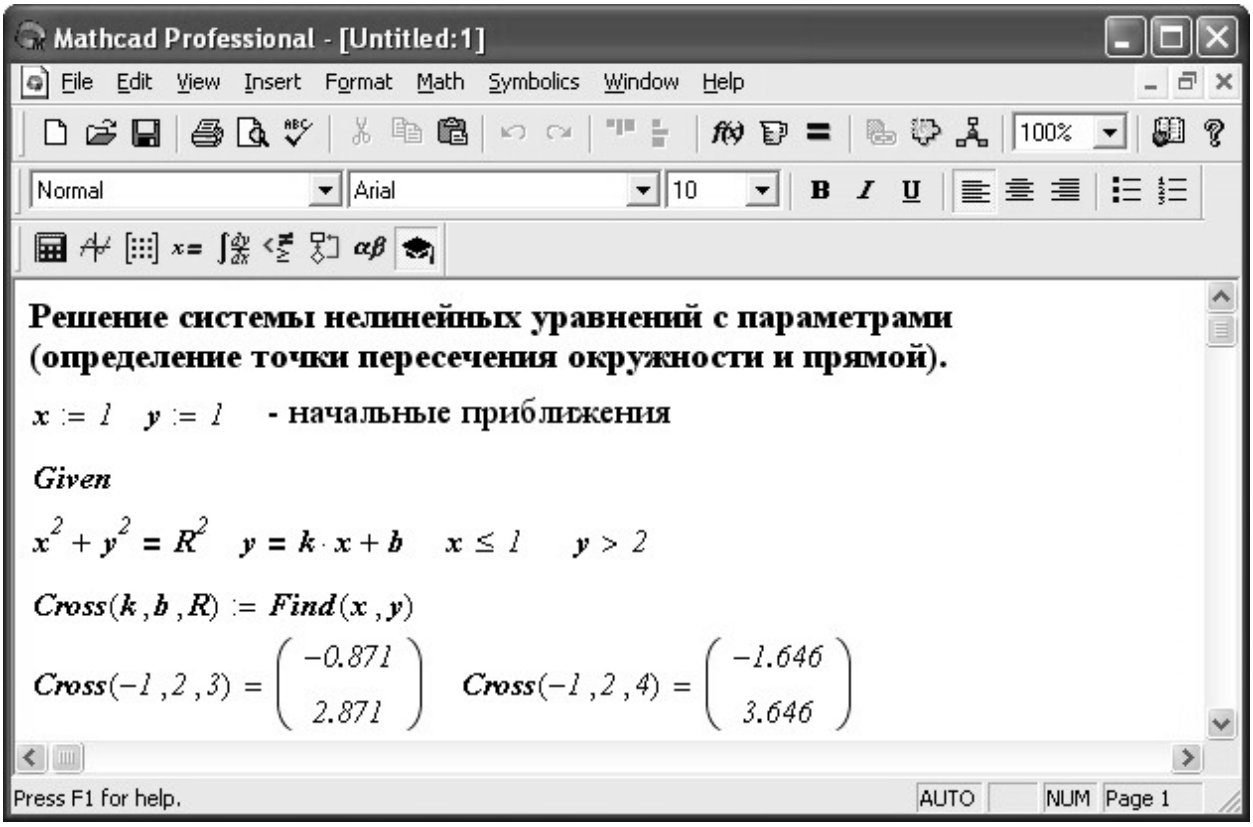


Рисунок 2.3 — Определение функции с помощью ***Find***

Функция ***Minerr*** работает аналогично ***Find***. Основное различие следующее. Если в процессе поиска решения уточнить его текущее приближение невозможно, то сообщения об ошибке не выдается, и возвращается это приближение.

Функцию ***Minerr*** удобно использовать для минимизации т. н. «функции невязки» при решении задач аппроксимации (приближения) наборов данных аналитическими зависимостями. Пример ее применения показан на рисунке 2.4.

3 ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

Работа заключается в получении решения нелинейных задач средствами системы ***MathCAD*** и указанными итерационными методом. Работа выполняется в следующем порядке.

1. Для заданного нелинейного уравнения самостоятельно выбирается начальное приближение корня и оцениваются границы интервала локализации корня (рекомендуется построить график

функции). Находится решение уравнения средствами ***MathCAD***, которое в дальнейшем будет использоваться в качестве точного.

2. Заданными интервальным и поисковым методами рассчитываются первые ***10-15*** приближенных значений корня. Оценивается сходимость методов к точному решению.

3. Для заданной системы нелинейных уравнений самостоятельно выбирается начальные приближения корней. Находится решение системы уравнений средствами ***MathCAD***, которое в дальнейшем будет использоваться в качестве точного. Систему уравнений дополнить условиями, исключающими возникновение неопределенности при расчете функций.

4. Заданным методом рассчитываются первые ***10-15*** приближенных значений решения системы. Оценивается сходимость методов к точному решению.

Правила реализации алгоритмов приведены в [5]. Примеры выполнения заданий представлены в приложениях Б и В.

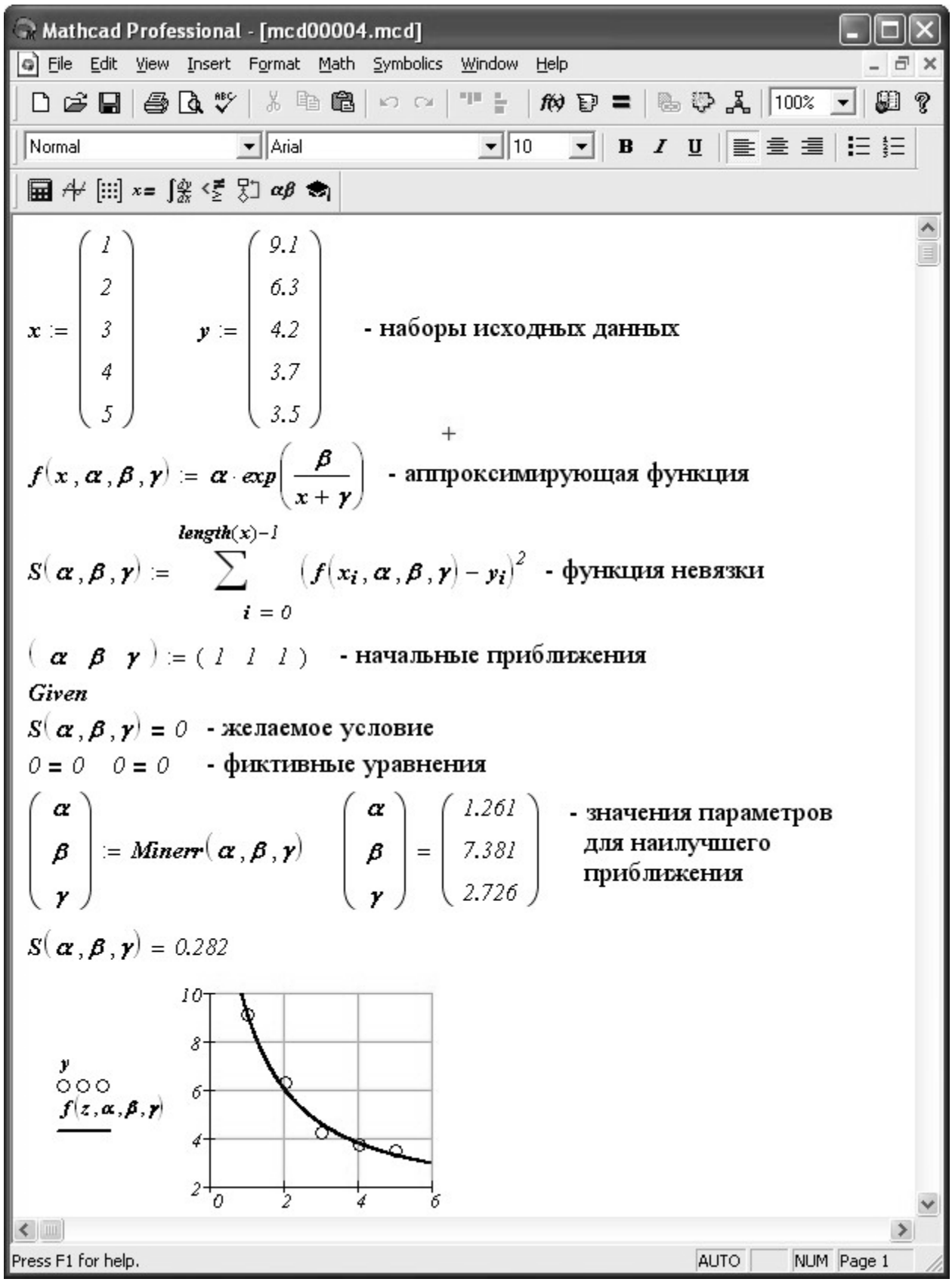


Рисунок 2.4 — Использование функции ***Minerr*** для решения регрессионной задачи

КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Какие существуют методы решения нелинейных уравнений?

2. На чем основывается работа интервальных алгоритмов решения нелинейных уравнений? В чем сущность метода половинного деления, метода хорд, метода обратной параболической интерполяции? Что определяет скорость сходимости методов?

3. На чем основывается работа поисковых алгоритмов решения нелинейных уравнений? В чем сущность метода простых итераций, метода релаксации, метода Ньютона, метода секущих? Что определяет скорость сходимости методов?

4. Как находятся действительные и комплексные корни полиномиальных уравнений?

5. На чем основывается работа поисковых алгоритмов решения систем нелинейных уравнений? В чем сущность метода простых итераций, метода релаксации, метода Ньютона? Что определяет скорость сходимости методов?

6. Как решаются нелинейные задачи средствами ***MathCAD***?

ЛИТЕРАТУРА

1. Самарский А.А., Гулин А.В. Численные методы. – М.: Наука, 1989.
2. Турчак Л.И. Основы численных методов. – М.: Наука, 2003.
3. Канахен Д., Моулер К., Нэш С. Численные методы и программное обеспечение. – М.: Мир, 1998.
4. Плис А.И., Сливина Н.А. MathCAD 2000. Математический практикум. – М.: Финансы и статистика, 2003.
5. Болдырев Д.В. Решение алгоритмических задач средствами системы MathCAD. — Невинномысск: изд-во НТИ (филиала) ГОУ ВПО «СевКавГТУ», 2005.

ПРИЛОЖЕНИЕ А

ВАРИАНТЫ ЗАДАНИЙ ДЛЯ ВЫПОЛНЕНИЯ ЛАБОРАТОРНЫХ РАБОТ

№	Уравнение	Методы решения
1	$tg(x)-\frac{1}{3}\cdot tg^3(x)+\frac{1}{5}\cdot tg^5(x)-\frac{1}{3}=0$	половинного деления, простой итерации
2	$0{,}6\cdot 3^x-2{,}3\cdot x-3=0$	хорд, релаксации
3	$3\cdot sin\left(\sqrt{x}\right)+0{,}35\cdot x-3{,}8=0$	интерполяции, Ньютона
4	$arccos(x)-\sqrt{1-0{,}3\cdot x^3}=0$	половинного деления, секущих
5	$exp(x)+ln(x)-10\cdot x=0=0$	хорд, простой итерации
6	$\sqrt{1-0{,}4\cdot x}-arcsin(x)=0$	интерполяции, релаксации
7	$1-x+sin(x)-ln(1+x)=0$	половинного деления, Ньютона
8	$x+\sqrt{x}+\sqrt[3]{x}-2{,}5=0$	хорд, секущих
9	$cos\left(\frac{2}{x}\right)-2\cdot sin\left(\frac{1}{x}\right)+\frac{1}{x}=0$	интерполяции, простой итерации
10	$x-2+sin\left(\frac{1}{x}\right)=0$	половинного деления, релаксации
11	$sin\left(ln(x)\right)-cos\left(ln(x)\right)+2\cdot ln(x)=0$	хорд, Ньютона
12	$x+cos(x^{0,6}+2)=0$	интерполяции, секущих
13	$3\cdot x-14+exp(x)-exp(-x)=0$	половинного деления, простой итерации
14	$sin\left(x^2\right)+cos\left(x^2\right)-10\cdot x=0$	хорд, релаксации
15	$cos(x)-exp\left(-\frac{x^2}{2}\right)+x-1=0$	интерполяции, Ньютона

№	Уравнение	Методы решения
16	$2 \cdot x \cdot \sin(x) - \cos(x) = 0$	половинного деления, секущих
17	$x^2 - \ln(1 + x) - 3 = 0$	хорд, простой итерации
18	$x \cdot tg(x) - \frac{1}{3} = 0$	интерполяции, релаксации
19	$\exp(x) + \sqrt{1 + \exp(2 \cdot x)} - 2 = 0$	половинного деления, Ньютона
20	$tg\left(\frac{x}{2}\right) - ctg\left(\frac{x}{2}\right) + x = 0$	хорд, секущих

ПРОДОЛЖЕНИЕ ПРИЛОЖЕНИЯ А

№	Система уравнений	Методы решения
1, 11	$\begin{cases} 0{,}25 \cdot x_1^3 + x_2 = 1{,}25 \\ \sqrt{1 - x_1} - tg(x_2) = 0 \end{cases}$	простой итерации
2, 12	$\begin{cases} 0{,}1 \cdot x_1^2 - x_2 \cdot \ln(x_1) = 0 \\ \ln(x_1) - x_2 = -1{,}8 \end{cases}$	Гаусса-Зейделя
3, 13	$\begin{cases} \exp(x_1) - \exp(-x_2) = -2 \\ 3 \cdot \sin\left(\sqrt{x_1}\right) + 0{,}35 \cdot x_2 = 3{,}8 \end{cases}$	релаксации
4, 14	$\begin{cases} x_1 + \sin\left(x_2^{-1}\right) = 2 \\ x_1 - \sin(x_1) + \ln(1 + x_2) = 1 \end{cases}$	Ньютона
5, 15	$\begin{cases} x_1^2 - \ln(1 + x_2) = 3 \\ 3 \cdot x - \sin(0{,}4 \cdot x_2) = 1 \end{cases}$	простой итерации
6, 16	$\begin{cases} \sqrt{1 - 0{,}4 \cdot x_1} - \arcsin(x_2) = 0 \\ x_1 + \cos(x_2^{0{,}6} + 2) = 0 \end{cases}$	Гаусса-Зейделя
7, 17	$\begin{cases} x_1 - 4 \cdot \ln(x_2) = 5 \\ x_1^2 + 10 \cdot x_2 = 10 \end{cases}$	релаксации
8, 18	$\begin{cases} \operatorname{arctg}(x_1) - x_2 = -0{,}4 \\ \arccos(x_1) - \sqrt{1 - 0{,}3 \cdot x_2^3} = 0 \end{cases}$	Ньютона
9, 19	$\begin{cases} \sqrt{1 - x_1} - \cos\left(\sqrt{1 - x_2}\right) = 0 \\ 2 \cdot x_1 \cdot \sin(x_1) - \cos(x_2) = 0 \end{cases}$	релаксации
10, 20	$\begin{cases} \exp(x_1) + \sqrt{1 + \exp(2 \cdot x_2)} = 5 \\ x_1 + \exp(x_1) - \exp(-x_2) = 0 \end{cases}$	Ньютона

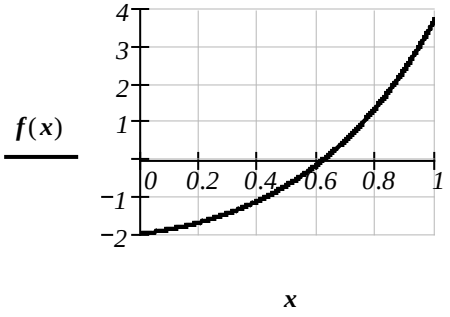
ПРИЛОЖЕНИЕ Б

ПРИМЕРЫ РЕШЕНИЯ НЕЛИНЕЙНЫХ УРАВНЕНИЙ (документ MathCAD)

Заданное уравнение:

$$f(x) := exp(-x) + exp(2 \cdot x) - 4$$

$$g(x) := \frac{d}{dx} f(x)$$



Граничные значения производной *f(x)* (для метода релаксации)

$g(0) = 1.0000$

$g(1) = 14.4102$

Интервал локализации корня (определяется по перемене знака функции):

$(\ a \quad b\ ) := (\ 0 \quad 1\ ) \qquad (\ f(a) \quad f(b)\ ) = (\ -2.0000 \quad 3.7569\ )$

Начальное приближение корня:

$z := 1$

Решение средствами MathCAD (используется как точное решение):

$root(f(z), z) = 0.6210$

Решение методом половинного деления:

$$Bisection(a, b) := \left| \begin{array}{l} \textbf{for} \quad k \in 0 .. 10 \\ \quad \left| \begin{array}{l} x_k \leftarrow \dfrac{a + b}{2} \\ a \leftarrow x_k \quad \textbf{if} \quad f(a) \cdot f\left(x_k\right) > 0 \\ b \leftarrow x_k \quad \textbf{otherwise} \end{array} \right. \\ x \end{array} \right.$$

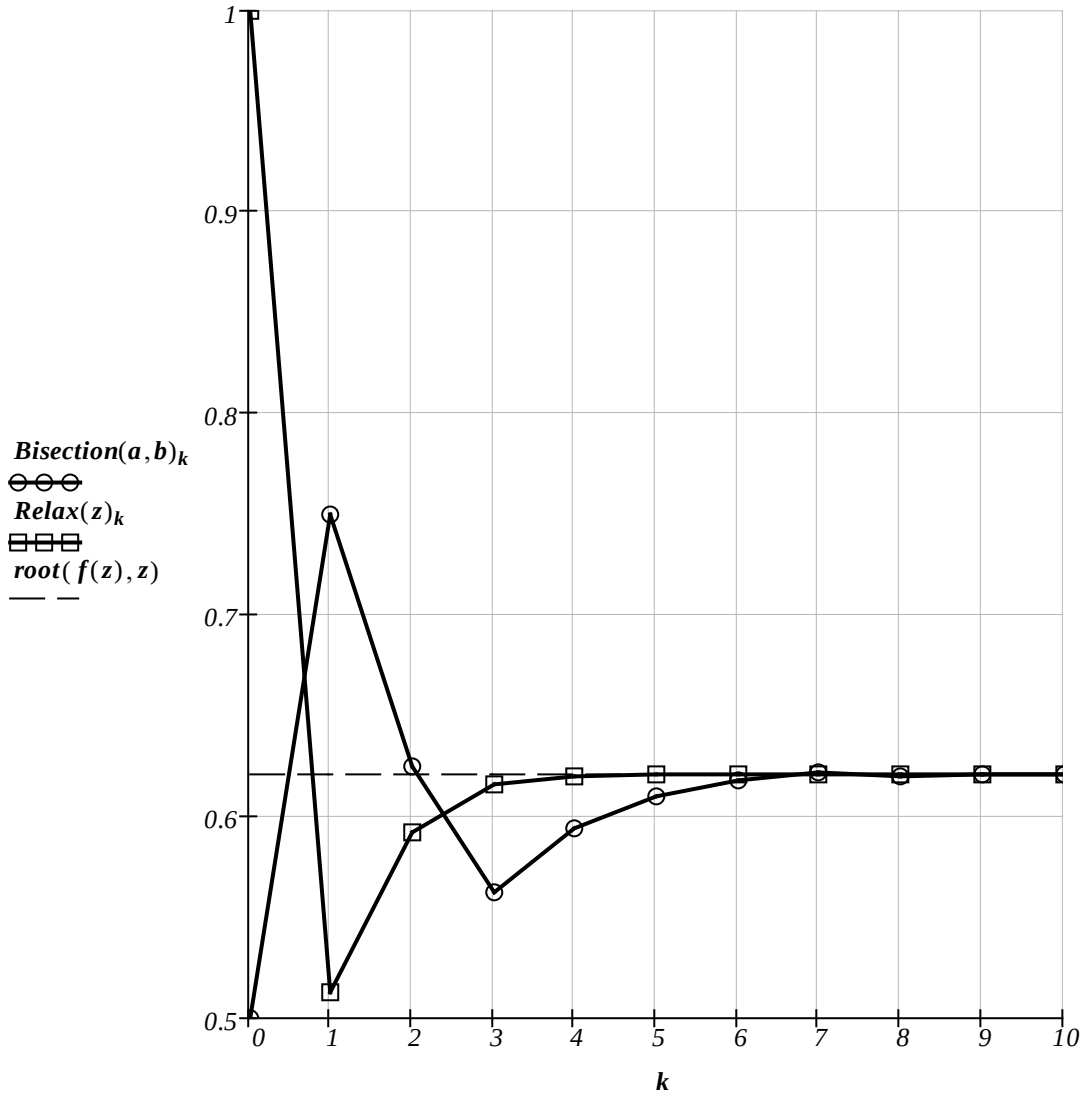
Решение методом релаксации:

$$\tau := \frac{2}{|g(0)| + |g(1)|}$$

$$Relax(z) := \left| \begin{array}{l} x_0 \leftarrow z \\ \textbf{for} \quad k \in 1 .. 10 \\ \quad x_k \leftarrow x_{k-1} - \tau \cdot f\left(x_{k-1}\right) \\ x \end{array} \right.$$

Сходимость к точному решению:

$k := 0 .. 10$



ПРИЛОЖЕНИЕ В

ПРИМЕРЫ РЕШЕНИЯ СИСТЕМ НЕЛИНЕЙНЫХ УРАВНЕНИЙ (документ MathCAD)

$ORIGIN \equiv 1$

Заданная система:

$$F(x) := \begin{bmatrix} (x_1)^2 + (x_2)^2 - 9 \\ x_1 + x_2 - 2 \end{bmatrix}$$

Начальное приближение корней:

$$x := \begin{pmatrix} 1 \\ 2 \end{pmatrix}$$

Решение средствами MathCAD (используется как точное решение):

$Given$

$F(x) = 0$

$z := Find(x) \qquad z^T = (\ -0.871 \quad 2.871\ )$

Решение методом Ньютона:

$$J(x) := \begin{pmatrix} 2 \cdot x_1 & 2 \cdot x_2 \\ 1 & 1 \end{pmatrix}$$

$X^{\langle 1 \rangle} := x$

$k := 1 .. 10$

$$X^{\langle k+1 \rangle} := X^{\langle k \rangle} - J\left(X^{\langle k \rangle}\right)^{-1} \cdot F\left(X^{\langle k \rangle}\right)$$

Сходимость к точному решению: