№1 Можно ли в методе Ньютона для решения системы нелинейных уравнений построить вычислительный процесс так, чтобы не пересчитывать матрицу Якоби на каждой итерации? Какова будет цена такой модификации алгоритма?

Ответ для Мальцевой: нет, нельзя. Потому что это будет уже другой алгоритм!!!!11!

Да, в методе Ньютона для решения системы нелинейных уравнений можно построить вычислительный процесс так, чтобы не пересчитывать матрицу Якоби на каждой итерации. Такая модификация называется методом Ньютона с фиксированной матрицей Якоби или модифицированным методом Ньютона.

Принцип метода

В этом подходе матрица Якоби вычисляется только на начальной итерации и затем используется неизменной на всех последующих итерациях. Это может значительно сократить вычислительные затраты, так как вычисление и обращение матрицы Якоби обычно является самой ресурсоемкой частью метода Ньютона.

Цена модификации

Скорость сходимости:

Классический метод Ньютона обладает квадратичной скоростью сходимости, что означает, что количество правильных цифр в приближении удваивается на каждой итерации.

Модифицированный метод Ньютона с фиксированной матрицей Якоби теряет эту квадратичную сходимость и может сходиться медленнее. В худшем случае скорость сходимости может быть линейной.

Число итераций:

Поскольку матрица Якоби не обновляется, потребуется больше итераций для достижения заданной точности решения.

Робастность:

Модифицированный метод Ньютона может оказаться менее устойчивым и более чувствительным к выбору начального приближения. Если исходная матрица Якоби не является хорошим представлением для всего процесса, алгоритм может не сойтись.

Пример применения

Модифицированный метод Ньютона может быть полезен в ситуациях, когда:

- Вычисление матрицы Якоби слишком дорогостоящее.
- Функция или система уравнений относительно слабо нелинейна.
- Приближение начальной матрицы Якоби достаточно хорошее.

Вывод

Использование фиксированной матрицы Якоби может снизить вычислительные затраты на итерацию, но при этом увеличивает общее число итераций и может снизить скорость и устойчивость сходимости. В зависимости от конкретной задачи и требований к точности решения этот компромисс может быть оправданным или нет.

Пример использования с одним уравнением: просто берем уравнение и считаем

xn+1 = xn – f(xn)/f'(xn) или xn+1 = xn – f(xn)/f'(x0).

№2.

Аппроксимируя производную конечно-разностным аналогом, построить на основе метода Ньютона метод секущих для решения уравнения f (x) = 0.

$$f'(x_n) = \frac{f(x_n) - f(x_{n-1})}{x_n - x_{n-1}}$$

Метод секущих: $x_{n+1} = x_n - f(x_n) * \frac{x_n - x_{n-1}}{f(x_n) - f(x_{n-1})}$

№3. Учитывая, что метод Ньютона является одношаговым, метод секущих – двухшаговым, построить трехшаговый метод парабол для решения уравнения f (x) = 0 . Какие новые возможности открывает метод парабол при решении рассматриваемой задачи.

$$x_{n+1} = x_n - f(x_n) \cdot \frac{(x_n - x_{n-1})(x_n - x_{n-2})}{(f(x_n) - f(x_{n-1}))(f(x_n) - f(x_{n-2}))}$$

1. **Более высокая скорость сходимости**:

- Метод парабол обычно сходится быстрее, чем метод секущих, так как использует аппроксимацию функции квадратичной параболой, что обеспечивает более точное приближение.

2. **Отсутствие необходимости в производной**:

- Метод парабол, как и метод секущих, не требует вычисления производной функции, что делает его полезным для функций, где вычисление производной затруднено или невозможно.

3. **Лучшее использование информации**:

- Использование трех точек для построения параболы позволяет лучше аппроксимировать функцию и улучшает устойчивость метода.

№4. Разработать алгоритм нахождения корней функции, заданной таблично, применяя метод Ньютона (секущих, парабол).

Метод Ньютона применять проблематично, т.к. там нужна производная.

Пример метод секущих:

Задана таблица значений x и f(x). Заданы начальные приближения x0 и x1. На каждом шаге интерполируем функцию в точке x_n и x_n-1. По формуле метода секущих ищем x_n+1.

№5. Разработать алгоритм нахождения решения системы нелинейных уравнений методом Ньютона

$$f_1(x,y)=0,$$
$$f_2(x,y)=0,$$

если функции $f_1(x,y)$, $f_2(x,y)$ заданы таблично.

Для нахождения вектора значений функции применяем двумерную интерполяцию.

Матрица Якоби – заменяем частные производные конечно-разностным аналогом.

Частная производная по x

Если (x_i, y_j) и (x_{i+1}, y_j) - две соседние точки по x, то частная производная по x может быть аппроксимирована как:

$\frac{\partial f(x_i,y_j)}{\partial x} \approx \frac{f(x_{i+1},y_j)-f(x_i,y_j)}{x_{i+1}-x_i}$

Частная производная по y

Если (x_i, y_j) и (x_i, y_{j+1}) - две соседние точки по y, то частная производная по y может быть аппроксимирована как:

$\frac{\partial f(x_i,y_j)}{\partial y} \approx \frac{f(x_i,y_{j+1})-f(x_i,y_j)}{y_{j+1}-y_j}$

№6.

6. Составить разностную схему для уравнения из п.3 с краевыми условиями общего вида

$$\begin{cases} y'' - y^3 = x^2, \\ \quad 0 \le x \le 1, \\ x = 0,\ y' = \alpha\, y, \\ x = 1,\ y' = \beta\, y - \gamma. \end{cases}$$

где α, β, γ - заданные числа.

Линеаризовать систему уравнений и разработать алгоритм метода прогонки для её решения.

Замечание.

При построении разностной схемы аппроксимацию краевых условий выполнить с помощью простейших односторонних разностных производных, т.е.

$$x = 0,\ \frac{y_1 - y_0}{h} = \alpha\, y_0,$$

$$x = 1,\ \frac{y_N - y_{N-1}}{h} = \beta\, y_N - \gamma$$

Решение.

Линеаризация уравнения

Рассмотрим центральные разности для второго порядка производной и одномерные разности для первого порядка производной на границах.

Внутренние узлы

Для центральных разностей на внутреннем узле i:

$$\frac{y_{i+1} - 2y_i + y_{i-1}}{h^2} - y_i^3 = x_i^2,$$

где $h = \frac{1}{N}, x_i = i \cdot h, i = 1, 2, \ldots, N-1.$

Граничные условия

Для левой границы $x = 0$:

$$\frac{y_1 - y_0}{h} = \alpha y_0,$$

откуда:

$$y_1 = h\alpha y_0 + y_0 = y_0(1 + h\alpha).$$

Для правой границы $x = 1$:

$$\frac{y_N - y_{N-1}}{h} = \beta y_N - \gamma,$$

откуда:

$$y_{N-1} = y_N(1 - h\beta) + h\gamma.$$

Линеаризация нелинейного члена

Для решения методом прогонки необходимо линеаризовать нелинейный член y_i^3. Используем метод Ньютона, который предполагает разложение функции в ряд Тейлора и линеаризацию:

$$y_i^3 \approx y_{i,\text{пред}}^3 + 3y_{i,\text{пред}}^2(y_i - y_{i,\text{пред}}).$$

Здесь $y_{i,\text{пред}}$ - значение y_i на предыдущей итерации.

(просто разложили y^3 в ряд Тейлора в точке y_i-1 и взяли два первых члена. F(x) = f(x0) + f'(x0)(x-x0)…

Разностная схема

Разностная схема для уравнения на внутреннем узле i после линеаризации:

$$\frac{y_{i+1} - 2y_i + y_{i-1}}{h^2} - (y_{i,\text{пред}}^3 + 3y_{i,\text{пред}}^2(y_i - y_{i,\text{пред}})) = x_i^2.$$

Приведем к стандартной форме:

$$y_{i+1} - \left(2 + 3h^2 y_{i,\text{пред}}^2\right) y_i + y_{i-1} = h^2 x_i^2 + y_{i,\text{пред}}^3 h^2 - 3y_{i,\text{пред}}^3 h^2.$$

Метод прогонки

Метод прогонки применяется для решения систем линейных уравнений вида $Ay = b$.

1. **Коэффициенты уравнений:**

   Для внутреннего узла i:

$$\begin{cases} a_i = 1, \\ b_i = -(2 + 3h^2 y_{i,\text{пред}}^2), \\ c_i = 1, \\ d_i = h^2 x_i^2 + y_{i,\text{пред}}^3 h^2 - 3y_{i,\text{пред}}^3 h^2. \end{cases}$$

2. **Прогонка:**

- Прямой ход:

$$\begin{cases} \alpha_1 = \frac{c_1}{b_1}, \\ \beta_1 = \frac{d_1}{b_1}, \\ \alpha_i = \frac{c_i}{b_i - a_i \alpha_{i-1}}, \\ \beta_i = \frac{d_i + a_i \beta_{i-1}}{b_i - a_i \alpha_{i-1}}. \end{cases}$$

- Обратный ход:

$$y_N = \beta_N,$$
$$y_i = \alpha_i y_{i+1} + \beta_i.$$

Алгоритм

1. Инициализация начальных условий и параметров.
2. Линеаризация и составление системы линейных уравнений.
3. Решение системы методом прогонки.
4. Обновление значений и итерация до сходимости.