ИТМО

Отчет
По лабораторной работе №3
Решение нелинейных уравнений.
Метод Простых Итераций

Выполнил:
Рахматов Нематджон
P3233
Преподаватель:
Перл Ольга
Вячеславовна

Санкт-Петербург
2024

Оглавление

Задача……………………………………………………………………2
Описание Метода...2
Блок-Схема Алгоритма...5
Код Программы..6
Примеры работы программы...7
Вывод...9

Задача

Дана система нелинейных уравнений. По заданному начальному приближению необходимо найти решение системы с точностью до 5 верного знака после запятой при помощи метода простых итераций.

Формат входных данных:
k
n
x0
y0
...

где k - номер системы, n - количество уравнений и количество неизвестных, а остальные значения - начальные приближения для соответствующих неизвестных.

Формат выходных данных: список такого же типа данных, как списки входных данных, содержащие значения корня для каждой из неизвестных с точностью до 5 верного знака.

Описание Метода

Метод простых итераций (или метод последовательных приближений) предназначен для решения систем нелинейных уравнений. Пусть дана система уравнений:

$$[\begin{cases} f_1(x_1, x_2, \ldots, x_n) = 0 \\ f_2(x_1, x_2, \ldots, x_n) = 0 \\ \vdots \\ f_n(x_1, x_2, \ldots, x_n) = 0 \end{cases}]$$

где $(x_1, x_2, \ldots, x_n)$ - неизвестные переменные, $(f_1, f_2, \ldots, f_n)$ - функции, определяющие систему уравнений.

Шаги метода простых итераций:

1. Преобразование системы уравнений к виду, удобному для итераций:

$$[\begin{cases} x_1 = g_1(x_1, x_2, \ldots, x_n) \\ x_2 = g_2(x_1, x_2, \ldots, x_n) \\ \vdots \\ x_n = g_n(x_1, x_2, \ldots, x_n) \end{cases}]$$

2. Выбор начального приближения $((x_1^{(0)}, x_2^{(0)}, \ldots, x_n^{(0)}))$.
3. Последовательное вычисление новых приближенных значений переменных путем подстановки текущих значений в функции системы уравнений:

$$[\begin{cases} x_1^{(k+1)} = g_1(x_1^{(k)}, x_2^{(k)}, \ldots, x_n^{(k)}) \\ x_2^{(k+1)} = g_2(x_1^{(k)}, x_2^{(k)}, \ldots, x_n^{(k)}) \\ \vdots \\ x_n^{(k+1)} = g_n(x_1^{(k)}, x_2^{(k)}, \ldots, x_n^{(k)}) \end{cases}]$$

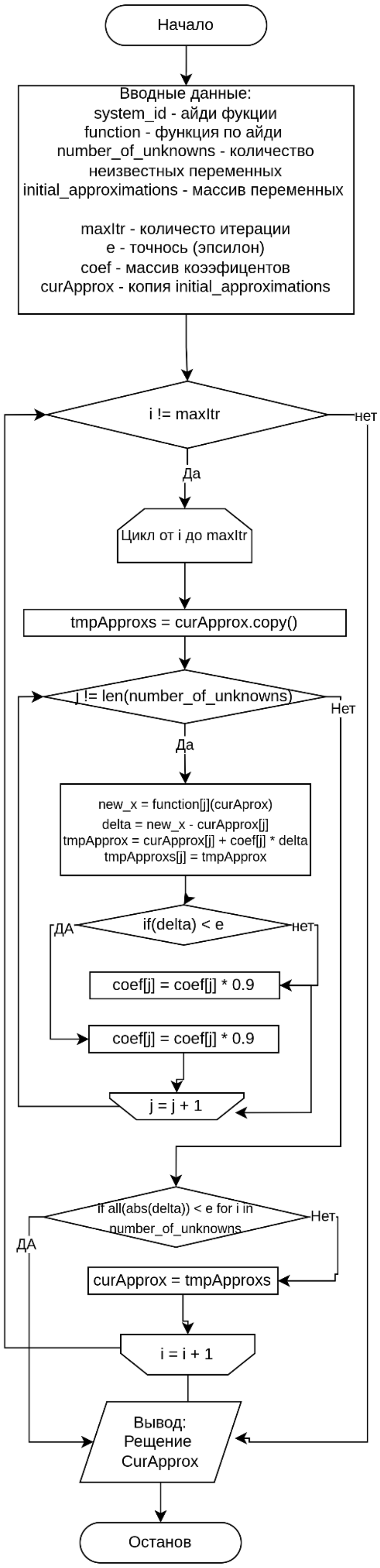
4. Проверка условия остановки. Обычно метод завершается, когда изменение приближенного решения становится достаточно малым:

$$[\sqrt{\sum_{i=1}^{n}(x_i^{(k+1)} - x_i^{(k)})^2} < \varepsilon]$$

где (ε) - заданная точность. Если условие не выполнено, то повторяем шаг 3.
5. Метод завершается, когда условие остановки выполнено, либо достигнуто максимальное количество итераций.

Метод простых итераций прост в реализации и понимании, но требует осторожного выбора начального приближения и функций системы уравнений для обеспечения сходимости. Также он может сходиться медленно или вообще не сходиться для некоторых видов систем уравнений, поэтому выбор метода решения зависит от конкретной задачи.

Блок-схема алгоритма



Код программы

```
def calc(funcs, number_of_unknowns, initial_approximations):
   itr = 0
   maxItr = 20000
   e = 1e-5
   coef = [0.01] * number_of_unknowns
   curApprox = initial_approximations[:]

   while itr < maxItr:
       tmpAppoxs = curApprox.copy()
       for i in range(number_of_unknowns):
           new_x = funcs[i](curApprox)
           tmpAppox = curApprox[i] + coef[i] * (new_x -
curApprox[i])

           tmpAppoxs[i] = tmpAppox

           if abs(new_x - curApprox[i]) < e:
               coef[i] *= 1.1
           else:
               coef[i] *= 0.9

       if all(abs(tmpAppoxs[i] - curApprox[i]) < e for i in
range(number_of_unknowns)):
           break

       curApprox = tmpAppoxs

   return [round(approximation, 5) for approximation in curApprox]

def solve_by_fixed_point_iterations(system_id, number_of_unknowns,
initial_approximations):
   funcs = get_functions(system_id)

   return calc(funcs, number_of_unknowns, initial_approximations)
```

Примеры Работы программы

1) Система уравнений с двумя неизвестными:

```
system_id = 1
number_of_unknowns = 2
initial_approximations = [1.0, 1.0]
result = solve_by_fixed_point_iterations(system_id,
number_of_unknowns, initial_approximations)
print(result)
```

```
[0.98451, 0.95091]
```

2) Система уравнений с тремя неизвестными и начальными приближениями, равными нулю:

```
system_id = 2
number_of_unknowns = 3
initial_approximations = [0.0, 0.0, 0.0]
result = solve_by_fixed_point_iterations(system_id,
number_of_unknowns, initial_approximations)
print(result)
```

```
[0.933, 0.208, 0.506]
```

3) Система уравнений с одним неизвестным:

```
system_id = 0
number_of_unknowns = 1
initial_approximations = [2.0]
result = solve_by_fixed_point_iterations(system_id,
number_of_unknowns, initial_approximations)
print(result)
```

```
[1.80929]
```

4) Система уравнений с двумя неизвестными, начальные приближения заданы так, что метод не применим:

```
system_id = 1
number_of_unknowns = 2
initial_approximations = [0.0, 0.0]
result = solve_by_fixed_point_iterations(system_id,
number_of_unknowns, initial_approximations)
print(result)
```

```
[0.0, 0.0]
```

5) Пример с линейной системой уравнений:

```
system_id = 1
number_of_unknowns = 2
```

```
initial_approximations = [1.0, 2.0]
result = solve_by_fixed_point_iterations(system_id,
number_of_unknowns, initial_approximations)
print(result)
```

```
[0.9845, 1.90172]
```

6) Пример с разными начальными приближениями:

```
system_id = 1
number_of_unknowns = 2
initial_approximations = [1.0, 1.0]
result = solve_by_fixed_point_iterations(system_id,
number_of_unknowns, initial_approximations)
print("Решение:", result)

initial_approximations = [0.5, 1.5]
result = solve_by_fixed_point_iterations(system_id,
number_of_unknowns, initial_approximations)
print("Решение:", result)
```

```
Решение: [0.98451, 0.95091]
Решение: [0.49796, 1.39144]
```

7) Пример с граничными значениями:

```
system_id = 1
number_of_unknowns = 2
initial_approximations = [0.0, 0.0]
result = solve_by_fixed_point_iterations(system_id,
number_of_unknowns, initial_approximations)
print("Решение:", result)

initial_approximations = [100.0, 100.0]
result = solve_by_fixed_point_iterations(system_id,
number_of_unknowns, initial_approximations)
print("Решение:", result)
```

```
Решение: [0.0, 0.0]
Решение: [90.44315, 6541.43809]
```

8) Пример с исключительной ситуацией (недостаточное количество уравнений):

```
system_id = 5  # Недостаточное количество уравнений
number_of_unknowns = 1
initial_approximations = [1.0]
result = solve_by_fixed_point_iterations(system_id,
number_of_unknowns, initial_approximations)
```

```
print(result)
```

```
[0.90469]
```

Вывод

Метод простых итераций хорошо справляется с решением систем нелинейных уравнений на различных данных, но требует тщательного выбора начальных приближений и параметров для сходимости. По сравнению с методом Ньютона, он менее эффективен, но не требует вычисления якобиана. Однако метод не гарантирует абсолютную точность из-за численных ошибок, и его алгоритмическая сложность может быть высокой при большом количестве итераций.