**Задание курсовой работы по дисциплине "Численные методы"**

Решить заданное уравнение

* методом релаксации (при оптимальном значении параметра релаксации и отличном от него)
* методом Ньютона
* методом секущих.

В качестве начального приближения принять 2 точки, в одной из которых условие сходимости метода выполняется, а в другой нет.

Критерий останова итерационного процесса выбрать самостоятельно.

В отчете привести проверку условий сходимости в точках начального приближения, обоснование критерия останова, расчет параметра релаксации.

Сравнить эффективность методов по критерию потребовавшегося числа вычислений

значения функций при одинаковой точности нахождения корня. (Считать, что трудоемкости вычисления функции, задающей уравнение и ее производной примерно равны).

В приложении привести текст (или алгоритм) программ и поведение итерационной последовательности {k, xk и f(xk)} для всех вариантов численного эксперимента, где k – номер итерации, xk -найденное на данной итерации решение, f(xk) - значение функции в данной точке.

Вычислительный инструмент выбирает студент.

Уравнение, решаемое в данной курсовой работе:

1. **Метод релаксации.**
   1. **Теория**

Уравнение представляется в эквивалентном виде

где – параметр релаксации. Из этого следует, что

*при или т.к.*

Для того, что бы произвести минимальное число итераций, нужно указать оптимальное значение параметра , которое вычисляется по формуле

где

* 1. **Выбор начального приближения, проверка условия сходимости.**

Рассмотрим поведение функции . Функция существует только при , т.к. логарифм не существует при отрицательных значениях x. При значениях преобладает , а значит функция на этом участке - отрицательна, но при этом она возрастает и при , функция уже положительная.

При , функции стремятся к +∞. Из этого можно сделать вывод, что функция сходится, т.к. на всем участке она возрастает, а значит знак производной - постоянен.

Исходя из вышеперечисленного возьмем за начальную точку среднее значение между (При ) и (При этом значении х функция уже положительна) - 5 .

* 1. **Расчет параметров релаксации.**

Для нахождения оптимального значения параметра релаксации , необходимо найти максимальное и минимальное значение производной , для этого, возьмем вторую производную от , приравняем к нулю и найдем корни:

Вторая производная не имеет решения, а значит, первая производная не имеет экстремумов. Тогда возьмем за максимальное и минимальное значение первой производной граничные точки 0.5 и 1.

Возьмем так же .

* 1. Результаты вычислений

Результаты показали, что при решении данного уравнения методом релаксации, присутствует ошибка равная, машинной эпсилон.

Результат при посчитанном оптимальном параметре был получен за 14 итераций, когда при не оптимальном параметре за 68 итерации.

Так же методом подбора, был найден параметр , который дал 9 итераций, вместо 14. Его не удалось найти в пункте 1.3 из-за не оптимально взятых границ участка приближения.

1. **Метод Ньютона.**
   1. Теория

Если функция определена в достаточно малой окрестности начального приближения, то эту функцию можно разложить в ряд Тейлора:

Если S(x­1)=0, то x­1- корень уравнения. Вычислительная схема, для метода Ньютона:

* 1. Выбор начального приближения.

Возьмем приближение, найденное в п. 1.2, равное 0,75.

* 1. Результаты вычислений

Результат был посчитан за 4 итерации и при этом, ошибка уменьшилась по модулю в 2 раза, по сравнению с методом релаксации. Минус данного метода заключается в том, что для нахождения результата нужна производная от функции. Это ограничивает спектр применения данного метода.

1. **Метод Секущих.**
   1. Теория

Метод секущих получается из [метода касательных](http://www.machinelearning.ru/wiki/index.php?title=%D0%9C%D0%B5%D1%82%D0%BE%D0%B4_%D0%BA%D0%B0%D1%81%D0%B0%D1%82%D0%B5%D0%BB%D1%8C%D0%BD%D1%8B%D1%85_%28%D0%9D%D1%8C%D1%8E%D1%82%D0%BE%D0%BD%D0%B0-%D0%A0%D0%B0%D1%84%D1%81%D0%BE%D0%BD%D0%B0%29) заменой f'(x^k) разностным приближением:

f'(x^k) \approx \frac{f(x^k)-f(x^{k-1})}{x^k-x^{k-1}}.

В результате получим формулу итерационного процесса:

(1)

x^{k+1}=x^k-\frac{x^k-x^{k-1}}{f(x^k)-f(x^{k-1})}f(x^k), \;k=1,2,\ldots

Метод секущих является *двухшаговым*, то есть новое приближение x^{k+1} определяется двумя предыдущими итерациями x^k и x^{k-1}. В методе [(1)](http://www.machinelearning.ru/wiki/index.php?title=%D0%9C%D0%B5%D1%82%D0%BE%D0%B4_%D1%81%D0%B5%D0%BA%D1%83%D1%89%D0%B8%D1%85#eq:1) необходимо задавать *два*начальных приближения x^0 и x^1.

Скорость сходимости метода будет *линейной*: |x^{k+1}-x*|=O(k^k-x*).

* 1. Выбор начальных приближений.

Для метода Секущих необходимо 2 начального приближения, возьмем за первое приближение, приближение найденое в п 1.2 и равное 0,75, за второе значения приближения возьмем 0,8.

* 1. Результат вычислений

Результат был посчитан за 7 итераций, когда метод Ньютона посчитал за 5 итераций. Так же минусом данного метода является то, что итерация происходит в 2 действия.

1. Сравнение методов и выводы

Среди трех методов, наилучшим для данной задачи будет метод Ньютона, который решил уравнение за минимальное число итераций. При это у него есть минус- в каждой точке функция должна существовать и иметь производную, что не позволяет его использовать в некоторых задачах.

Метод Секущих решил задачу за 7 итераций, что так же является не плохим результатом, но при этом, он решает задачу за 2 вычисления.

Метод релаксации показал наихудший показатели числа итерации и решил задачу с большой погрешностью. Но при этом он не востребован к производной, что является плюсом.

Приложение

Код программ на языке Matlab

1. Метод релаксации
2. t\_opt=0.7; % "тау" оптимальное
3. i=1; % итерация-первая
4. x(i)=0.75; % значение "х", близкое к корню
5. func(i)=x(i)+log(x(i))-0.5; % Функция
6. while abs(func(i))>eps
7. i=i+1; % подсчет итераций
8. x(i)=x(i-1)-t\_opt\*func(i-1);
9. func(i)=x(i)+log(x(i))-0.5;
10. end
11. count=i
12. result\_x=x(i)
13. result\_f=func(i)
14. Метод Ньютона
15. i=1;
16. x(i)=0.75;
17. func(i)=x(i)+log(x(i))-0.5;
18. dfunc(i)=1+1/x(i);
19. while abs(func(i))>eps
20. i=i+1;
21. x(i)=x(i-1)-func(i-1)/dfunc(i-1);
22. func(i)=x(i)+log(x(i))-0.5;
23. dfunc(i)=1+1/x(i);
24. end
25. result\_x=x(i)
26. count\_of\_iterations=i
27. result\_func=func(i)
28. Метод Секущих
29. i=1;
30. x(i)=0.75;
31. func(i)=x(i)+log(x(i))-0.5;
32. i=i+1;
33. x(i)=0.8;
34. func(i)=x(i)+log(x(i))-0.5;
35. while abs(func(i))>eps
36. x(i+1)=x(i)-(x(i)-x(i-1))\*func(i)/(func(i)-func(i-1));
37. i=i+1;
38. func(i)=x(i)+log(x(i))-0.5;
39. end
40. result\_func=func(i)
41. result\_x=x(i);
42. count\_of\_iterations=i;

|  |  |  |
| --- | --- | --- |
| i | X­i | Function(xi) |
| 1 | 7.500000000000000e-01 | -3.768207245178090e-02 |
| 2 | 7.650728289807124e-01 | -2.711419424266337e-03 |
| 3 | 7.661573967504189e-01 | -2.102548125535453e-04 |
| 4 | 7.662414986754403e-01 | -1.638783176644010e-05 |
| 5 | 7.662480538081469e-01 | -1.277818283129673e-06 |
| 6 | 7.662485649354601e-01 | -9.963917091404539e-08 |
| 7 | 7.662486047911284e-01 | -7.769483723052417e-09 |
| 8 | 7.662486078989219e-01 | -6.058349932835938e-10 |
| 9 | 7.662486081412560e-01 | -4.724060032046396e-11 |
| 10 | 7.662486081601522e-01 | -3.683719995706269e-12 |
| 11 | 7.662486081616257e-01 | -2.872146964705280e-13 |
| 12 | 7.662486081617406e-01 | -2.237099394619690e-14 |
| 13 | 7.662486081617496e-01 | -1.609823385706477e-15 |
| 14 | 7.662486081617502e-01 | -1.110223024625157e-16 |

Метод Релаксации

Метод Ньютона

|  |  |  |
| --- | --- | --- |
| i | X­i | Function(xi) |
| 1 | 7.500000000000000e-01 | -3.768207245178090e-02 |
| 2 | 7.650728289807124e-01 | -2.711419424266337e-03 |
| 3 | 7.661573967504189e-01 | -2.102548125535453e-04 |
| 4 | 7.662414986754403e-01 | -1.638783176644010e-05 |
| 5 | 7.662480538081469e-01 | -1.277818283129673e-06 |
| 6 | 7.662485649354601e-01 | -9.963917091404539e-08 |
| 7 | 7.662486047911284e-01 | -7.769483723052417e-09 |
| 8 | 7.662486078989219e-01 | -6.058349932835938e-10 |
| 9 | 7.662486081412560e-01 | -4.724060032046396e-11 |
| 10 | 7.662486081601522e-01 | -3.683719995706269e-12 |
| 11 | 7.662486081616257e-01 | -2.872146964705280e-13 |
| 12 | 7.662486081617406e-01 | -2.237099394619690e-14 |
| 13 | 7.662486081617496e-01 | -1.609823385706477e-15 |
| 14 | 7.662486081617502e-01 | -1.110223024625157e-16 |

|  |  |  |
| --- | --- | --- |
| i | X­i | Function(xi) |
| 1 | 7.500000000000000e-01 | -3.768207245178090e-02 |
| 2 | 7.650728289807124e-01 | -2.711419424266337e-03 |
| 3 | 7.661573967504189e-01 | -2.102548125535453e-04 |
| 4 | 7.662414986754403e-01 | -1.638783176644010e-05 |
| 5 | 7.662480538081469e-01 | -1.277818283129673e-06 |
| 6 | 7.662485649354601e-01 | -9.963917091404539e-08 |
| 7 | 7.662486047911284e-01 | -7.769483723052417e-09 |
| 8 | 7.662486078989219e-01 | -6.058349932835938e-10 |
| 9 | 7.662486081412560e-01 | -4.724060032046396e-11 |
| 10 | 7.662486081601522e-01 | -3.683719995706269e-12 |
| 11 | 7.662486081616257e-01 | -2.872146964705280e-13 |
| 12 | 7.662486081617406e-01 | -2.237099394619690e-14 |
| 13 | 7.662486081617496e-01 | -1.609823385706477e-15 |
| 14 | 7.662486081617502e-01 | -1.110223024625157e-16 |

|  |  |  |
| --- | --- | --- |
| i | X­i | Function(xi) |
| 1 | 7.500000000000000e-01 | -3.768207245178090e-02 |
| 2 | 7.661494596221918e-01 | -2.285516519496822e-04 |
| 3 | 7.662486045295964e-01 | -8.372330606576384e-09 |
| 4 | 7.662486081617502e-01 | -1.110223024625157e-16 |

Метод Секущих

|  |  |  |
| --- | --- | --- |
| i | X­i | Function(xi) |
| 1 | 7.500000000000000e-01 | -3.768207245178090e-02 |
| 2 | 8.000000000000000e-01 | 7.685644868579034e-02 |
| 3 | 7.664495193745873e-01 | 4.630779244394478e-04 |
| 4 | 7.662461445689317e-01 | -5.678733117298052e-06 |
| 5 | 7.662486083445933e-01 | 4.214640858535290e-10 |
| 6 | 7.662486081617504e-01 | 4.440892098500626e-16 |
| 7 | 7.662486081617502e-01 | -1.110223024625157e-16 |