

Практическая работа №2: Изучение понятия обусловленности вычислительной задачи

Цель работы

Исследование обусловленности задачи нахождения корня уравнения на примере линейной функции.

Основные теоретические положения

Под обусловленностью вычислительной задачи понимают чувствительность ее решения к малым погрешностям входных данных.

Задачу называют хорошо обусловленной, если малым погрешностям входных данных отвечают малые погрешности решения, и плохо обусловленной, если возможны сильные изменения решения. Количественной мерой степени обусловленности вычислительной задачи является число обусловленности, которое можно интерпретировать как коэффициент возможного возрастания погрешностей в решении по отношению к вызвавшим их погрешностям входных данных. Пусть между абсолютными погрешностями входных данных x^* и решения y установлено неравенство: $|\Delta(y^*)| \leq \nu_{\Delta} |\Delta(x^*)|$, где x^* и y^* – приближённые входные данные и приближённое решение соответственно. Тогда величина ν_{Δ} называется абсолютным числом обусловленности.

Если же установлено неравенство $|\delta(y^*)| \leq \nu_{\delta} |\delta(x^*)|$ между относительными ошибками данных и решения, то величину ν_{δ} называют относительным числом обусловленности. Для плохо обусловленной задачи $\nu_{\delta} \gg 1$. Грубо говоря, если $\nu_{\delta} = 10^N$, где ν_{δ} – относительное число обусловленности, то порядок N показывает число верных цифр, которое может быть утеряно в результате по сравнению с числом верных цифр входных данных.

Ответ на вопрос о том, при каком значении ν задачу следует признать плохо обусловленной, зависит, с одной стороны, от предъявляемых требований к точности решения и, с другой, – от уровня обеспечиваемой точности исходных данных. Например, если требуется найти решение с точностью 0.1%, а входная информация задается с точностью 0.02%, то уже значение $\nu = 10$ сигнализирует о плохой обусловленности. Однако, при тех же требованиях к точности результата, гарантия, что исходные данные задаются с точностью не ниже 0.0001%, означает, что при $\nu = 10^3$ задача хорошо обусловлена.

Если рассматривать задачу вычисления корня уравнения $y = f(x)$, то роль числа обусловленности будет играть величина $\nu_{\Delta} = \frac{1}{|f'(\bar{x})|}$, где $\bar{x}$ – корень уравнения.

Постановка задачи

Используя программы-функции `bisect` и `Round` из файла `condition.m`, исследовать обусловленность задачи нахождения корня уравнения $f(x) = 0$ для линейной функции $f(x) = c(x - d)$. Значения функции $f(x)$ следует вычислить приближенно с точностью δ , варьируемой в пределах от 0.1 до 0.000001.

Порядок выполнения работы

1. Графически или аналитически отделить корень уравнения $f(x) = 0$, т.е. найти отрезки $[a, b]$, на которых функция удовлетворяет условиям применимости метода бисекции.
2. Запустить m-скрипт `condition.m`. Ввести значения, необходимые для запуска программы.
3. Провести вычисления по программе, варьируя значения параметров c (тангенс угла наклона прямой), ϵ (точность вычисления корня) и δ (точность задания исходных данных).
4. Проанализировать полученные результаты и обосновать выбор точности ϵ вычисления корня. Сопоставить полученные теоретические результаты с экспериментальными данными.

Значение параметра d выбирается каждым студентом самостоятельно и согласовывается с преподавателем.

Содержание отчёта

- Цель работы.
- Краткое изложение основных теоретических понятий.
- Постановка задачи с кратким описанием порядка выполнения работы.
- Таблицы с вычислениями при различных вариациях параметров.
- Краткие выводы по полученным результатам.
- Общий вывод по проделанной работе.
- Код программы.

Тексты программ

Исследование обусловленности задачи нахождения корня линейного уравнения

[condition.m](#)

```
## TASK2
## Studying the concept of conditionality
## of a computational problem
```

```

format long g

function [x, k] = bisect (a, b, c, d, epsilon, delta)
    Eps = abs (epsilon) * 2;
    fa = f (a, c, d, delta);
    fb = f (b, c, d, delta);
    k = ;

    if (fa * fb > )
        error ("\aInvalid interval setting!!!\n")
    endif

    if (epsilon <= )
        error ("\aInvalid accuracy setting!!!\n")
    endif

    if (fa == )
        x = fa;
        return
    endif

    if (fb == )
        x = fb;
        return
    endif

    while (b - a >= Eps)
        x = 0.5 * (a + b);
        y = f (x, c, d, delta);

        if (y == )
            return
        elseif (y * fa < )
            b = x;
        else
            a = x;
            fa = y;
        endif

        k++;
    endwhile
endfunction

function y = f (x, c, d, delta)
    s = c * (x - d);
    S = s / delta + merge (s / delta < , -0.5, 0.5);

    s = S * delta;
    y = Round (s, delta);
endfunction

```

```
function r = Round (x, delta)
    if (delta <= 1e-9)
        error ("\aInvalid setting of rounding precision!!!\n")
    endif

    r = delta * (x / delta + merge (x > , 0.5, -0.5));
endfunction

## Begin script
epsilon = input ("Enter 'eps': ");
c = input ("Enter 'c': ");
d = input ("Enter 'd': ");
a = input ("Enter 'a': ");
b = input ("Enter 'b': ");
delta = input ("Enter 'delta': ");
[x, k] = bisection (a, b, c, d, epsilon, delta);
printf ("\nx = %.14f, \tk = %u\n", x, k), format short
```

From:
<http://se.moevm.info/> - **se.moevm.info**

Permanent link:
http://se.moevm.info/doku.php/courses:computational_mathematics:prac2?rev=1619125348

Last update: **2022/12/10 09:08**

