**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра математического обеспечения и применения ЭВМ**

отчет

**по практической работе №2**

**по дисциплине «Вычислительная математика»**

Тема: **Изучение понятия обусловленности вычислительной задачи**

|  |  |  |
| --- | --- | --- |
| Студент гр. 8382 |  | Ларин А. |
| Преподаватель |  | Сучков А.И. |

Санкт-Петербург

2019

**Цель работы.**

Исследовать обусловленность нахождения корня уравнения для 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 вводных данных *Х* и уравнения *Y* установлено неравенство:

, (1)

где *x*\* и *y*\* - приближенные входные данные и приближенное решение. Тогда величина  называется абсолютным числом обусловленности.

Если же установлено неравенство

, (2)

между относительными ошибками данных и решения, то величину  называют относительным числом обусловленности. Для плохо обусловленной задачи **>> 1. Грубо говоря, если **= 10*N*, где ** 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  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 **= 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 **= 103 задача хорошо обусловлена.

Если рассматривать задачу вычисления корня уравнения *Y* = *f*(*X*), то роль числа обусловленности будет играть величина

, (3)

где - корень уравнения.

**Постановка задачи.**

Исследовать обусловленность задачи нахождения корня уравнения *f*(*x*)=0 для линейной функции *f*(*x*) = *c*(*x*-*d*). Значения функции *f*(*x*) следует вычислять приближенно с точностью *Delta* (точность входных данных), варьируемой в пределах от 0.1 до 0.000001. Корень уравнения будем находить методом бисекции с задаваемой точностью *Eps*. Порядок выполнения работы следующий:

* 1. Графически или аналитически отделить корень уравнения f(x)=0, т.е. найти отрезки , на которых функция удовлетворяет условиям применимости метода бисекции.
  2. Составить подпрограмму вычисления функции для параметров c и d, вводимых с клавиатуры. Предусмотреть округление вычисленных значений функции с использованием программы-функции с точностью , также вводимой с клавиатуры.
  3. Составить головную программу, вычисляющую корень уравнения с заданной точностью и содержащую обращение к подпрограмме , программам-функциям , и представление результатов.
  4. Провести вычисления по программе, варьируя значения параметров c (тангенс угла наклона прямой), (точность вычисления корня) и (точность задания сходных данных).
  5. Проанализировать полученные результаты и обосновать выбор точности вычисления корня. Сопоставить полученные теоретические результаты с экспериментальными данными.

**Выполнение работы.**

Метод бисекции состоит в построении последовательности вложенных друг в друга отрезков, на концах которых функция имеет разные знаки. Каждый последующий отрезок получается делением пополам предыдущего. Этот процесс позволяет найти нуль функции *f*(*x*) с любой заданной точностью.

Если рассматривать задачу вычисления корня уравнения *Y* = *f*(*X*), то роль числа обусловленности будет играть величина (3).

Определим абсолютное число обусловленности задачи вычисления корня:

. (4)

Таким образом, с аналитической точки зрения абсолютное число обусловленности задачи вычисления корня обратно пропорционально величине тангенса угла наклона прямой. То есть, чем меньше модуль тангенса, тем хуже задача обусловлена. Можно определить условие максимального значения обусловленности при *Delta* для попадания в рамки погрешности *Eps* как

. . (5)

Обозначим правую часть неравенства (5) как

Пусть *d* = 11. Найдем отрезок , на котором функция будет удовлетворять условиям применимости метода бисекции, то есть на концах данного отрезка значение функции должны быть разными по знаку и в его пределах должен быть лишь один корень.

В табл. 1 рассмотрим случай для *с* = 5, то есть *f*(*x*) = 5(*x*-11).

Таблица 1 – Поиск отрезка

|  |  |
| --- | --- |
| Значение *x* | Значение *f*(*x*) |
| 5 | -30 |

Продолжение таблицы 1

|  |  |
| --- | --- |
| 10 | -5 |
| 15 | 20 |
| 20 | 45 |

Проанализировав результаты табл.1, был получен отрезок [10;15].

Для построения табл.2, табл.3, табл.4, табл.5 была использована программа (см. Приложение А).

В табл. 2 *Eps* и *Delta* постоянные, изменяется *c*. *d* = 11, *a* = 10, *b* = 15.

Таблица 2 – Изменение параметра *с*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Значение *c* | Значение  *Eps* | Значение  *Delta* | Значение  *X* | Значение  *νΔ\_*max | Значение  *νΔ* | Значение  *νΔ* ≤ *νΔ\_*max |
| 200 | 0.01 | 0.1 | 10.996094 | 0.1 | 0.005 | хор. |
| 20 | 0.01 | 0.1 | 10.996094 | 0.1 | 0.05 | хор. |
| 0.2 | 0.01 | 0.1 | 11.015625 | 0.1 | 5 | пл. |
| 0.02 | 0.01 | 0.1 | 11.015625 | 0.1 | 50 | пл. |

В табл. 3 *Eps* и *c* постоянные, меняется *Delta*. *d* = 11, *a* = 10, *b* = 15.

Таблица 3 – Изменение параметра *Delta*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Значение *c* | Значение  *Eps* | Значение  *Delta* | Значение  *X* | Значение  *νΔ\_*max | Значение  *νΔ* | Значение  *νΔ* ≤ *νΔ\_*max |
| 25 | 0.01 | 1 | 11.015625 | 0.01 | 0.04 | пл. |
| 25 | 0.01 | 0.1 | 10.996094 | 0.1 | 0.04 | хор. |
| 25 | 0.01 | 0.01 | 10.996094 | 1 | 0.04 | хор. |
| 25 | 0.01 | 0.001 | 10.996094 | 10 | 0.04 | хор. |

В табл. 4 изменяется только значение *Eps*. *d* = 11, *a* = 10, *b* = 15.

Таблица 4 – Изменение параметра *Eps*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Значение *c* | Значение  *Eps* | Значение  *Delta* | Значение  *X* | Значение  *νΔ\_*max | Значение  *νΔ* | Значение  *νΔ* ≤ *νΔ\_*max |
| 2 | 1 | 0.01 | 11.25 | 100 | 0.5 | хор. |
| 2 | 0.1 | 0.01 | 11.09375 | 10 | 0.5 | хор. |
| 2 | 0.01 | 0.01 | 10.996094 | 1 | 0.5 | хор. |
| 2 | 0.001 | 0.01 | 10.999756 | 0.1 | 0.5 | пл. |

В табл. 5 изменяются *c* и *Delta*, *Eps* постоянная. *d* = 11, *a* = 10, *b* = 15.

Таблица 5 – Изменение параметров *c* и Delta

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Значение *c* | Значение  *Eps* | Значение  *Delta* | Значение  *X* | Значение  *νΔ\_*max | Значение  *νΔ* | Значение  *νΔ* ≤ *νΔ\_*max |
| 200 | 0.01 | 1 | 10.996094 | 0.01 | 0.005 | хор. |
| 20 | 0.01 | 0.1 | 10.996094 | 0.1 | 0.05 | хор. |
| 0.2 | 0.01 | 0.01 | 11.015625 | 1 | 5 | пл. |
| 0.02 | 0.01 | 0.001 | 11.015625 | 10 | 50 | пл. |

Если значение *νΔ* ≤ *νΔ\_*max  - хор., то будем считать, что задача хорошо обусловлена, иначе пл. – плохо.

**Выводы.**

Проанализировав данные вычислительного эксперимента можно сделать вывод, что обусловленность задачи нахождения корня уравнения *f*(*x*) = 0 для линейной функции *f*(*x*) = *c*(*x*-*d*) прямо пропорциональна абсолютной величине тангенса угла наклона прямой и точности задания исходных данных, и также обратно пропорциональна точности вычисления корня.

Можно утверждать, что максимальное значение абсолютного числа обусловленности определяется требуемой точностью результата и точностью входных данных. Если число абсолютной обусловленности превышает максимальное значение, то результат не будет обладать требуемой точностью. В противном случае задача обусловлена хорошо, то есть точность результата будет соответствовать требуемой точности.

Приложение А

**ИСХОДНЫЙ КОД ПРОГРАММЫ**

#include <math.h>

#include <stdio.h>

#include <stdlib.h>

#include <conio.h>

double delta, c, d;

double BISECT(double, double, double, int&);

double F(double);

double Round(double, double);

int main() {

int k;

long int s;

float a1, b1, c1, d1, eps1, delta1;

double a, b, eps, x;

printf("Введите eps: ");

scanf\_s("%f", &eps1);

eps = eps1;

printf("Введите c: ");

scanf\_s("%f", &c1);

c = c1;

printf("Введите d: ");

scanf\_s("%f", &d1);

d = d1;

printf("Введите a: ");

scanf\_s("%f", &a1);

a = a1;

printf("Введите b: ");

scanf\_s("%f", &b1);

b = b1;

printf("Введите delta: ");

scanf\_s("%f", &delta1);

delta = delta1;

x = BISECT(a, b, eps, k);

printf("x = %lf k = %d\n", x, k);

printf("eps=%lf c=%lf d=%lf a=%lf b=%lf delta=%lf\n", eps, c, d, a, b, delta);

return 0;

}

double BISECT(double Left, double Right, double Eps, int &N) {

double E = fabs(Eps) \* 2.0;

double FLeft = F(Left);

double FRight = F(Right);

double X = 0.5 \* (Left + Right);

double Y;

if (FLeft \* FRight > 0.0) {

puts("Неверное задание интервала\n");

exit(1);

}

if (Eps <= 0.0) {

puts("Неверное задание точности\n");

exit(1);

}

if (FLeft == 0.0) {

return Left;

}

if (FRight == 0.0) {

return Right;

}

for (N = 0; Right - Left >= E; N++) {

X = 0.5 \* (Right + Left); // вычисление середины отрезка

Y = F(X);

if (Y == 0.0) {

return (X);

}

if (Y \* FLeft < 0.0) {

Right = X;

}

else {

Left = X;

FLeft = Y;

}

}

return X;

}

double F(double x) {

extern double c, d, delta;

double s;

long S;

s = c \* (x - d);

if (s / delta < 0) {

S = s / delta - 0.5;

}

else {

S = s / delta + 0.5;

}

s = S \* delta;

s = Round(s, delta);

return s;

}

double Round(double X, double Delta) {

if (Delta <= 1E-9) {

puts("Неверное задание точности округления\n");

exit(1);

}

if (X > 0.0) {

return Delta \* long(X / Delta + 0.5);

}

else {

return Delta \* long(X / Delta - 0.5);

}

}