Министерство Общего и Профессионального образования

# Таганрогский Государственный

# Радиотехнический университет

Кафедра микропроцессорных систем


### Курсовая работа

*по курсу:*

#### **Основы обработки данных**

*на тему:*

##### Вычисление элементарных функций

 Выполнил:

 студент группы Р-106

 Рябчевский К.Л.

 Проверил:

 к.т.н.д. Ледовской М.И.

**Таганрог 1998 г.**

**Содержание**

**1. Задание**

**2. Аннотация**

**3. Введение**

**4. Теоретические основы таблично - алгоритмического метода**

**5. Расчет параметров алгоритма**

 **5.1.Расчет величин S и h**

 **5.2.Выбор масштабных коэффициентов**

**6. Масштабирование алгоритма**

1. **Граф – схема программы**
2. **Листинг и описание подпрограммы-функции**
3. **Листинг программы**
4. **Результаты работы программы**
5. **Заключение**
6. **Список литературы**

### 1. Задание

1. Используя литературу и методические указания выполнить проектирование алгоритма вычисления элементарной функции с использованием таблично – алгоритмического метода в соответствии с заданным вариантом. Алгоритм должен ориентироваться на целочисленные вычисления в формате байт со знаком в дополнительном коде.
2. Выполнить масштабирование алгоритма.
3. Разработать Граф – схему алгоритма для реализации в целочисленном режиме.
4. Разработать подпрограмму-функцию с использованием целочисленных операторов языка Turbo Pascal.
5. Разработать программу для вычисления функции на заданном интервале аппроксимации и экспериментального анализа, полной, методической и вычислительной погрешности.
6. Построить графики функций и всех погрешностей на интервале аппроксимации.
7. Сопоставить экспериментальные значения погрешности с теоретическими оценками и сделать выводы.

 Функция

Интервал аппроксимации [0,0.5]

Разрядность 8

Метод нахождения поправки одночленный ряд

 Тейлора

### 3. Введение

Элементарными функциями называются функции одного аргумента, значения которых получаются с помощью конечного числа вычислительных операций над аргументом, зависимой переменной и постоянными числами.

Элементарные функции делятся на алгебраические и трансцендентные. Вычисление значений элементарных функций один из наиболее часто встречающихся типов вычислительных операций, выполняемых в микро ЭВМ при решении задач управления движением роботов-манипуляторов, навигации, стабилизации и т. д. В этой связи важное значение приобретает разработка алгоритмов вычисления элементарной функции для их программной и аппаратной реализации, обеспечение максимального быстродействия.

Алгоритм вычисления элементарной функции в микроЭВМ охватывает три следующих типа:

1. привидение аргумента к интервалу аппроксимации (уменьшение интервала изменения аргумента);
2. вычисление элементарной функции на интервале аппроксимации;
3. пост-обработка.

Задача проектирования алгоритма вычисления элементарной функции сводится к нахождению алгоритма приведения аргумента к интервалу аппроксимации и выбору численного метода для приближенного вычисления значений этой функции на интервале аппроксимации. Приведение аргумента к интервалу аппроксимации является обязательным этапом как при использовании итеративных методов вычисления элементарных функций, так и при многочленной и рациональной аппроксимации. Этот прием позволяет сократить число операций необходимых для вычисления значения элементарной функции за счет уменьшения количества итераций или использования многочленных и рациональных приближении, содержащих сравнительно небольшое число членов.

Способ уменьшения интервала изменения аргумента зависит от свойств функции. Если функция периодическая, то имеет смысл вычислять ее только на одном периоде изменения аргумента. Если функция симметричная, то это свойство также можно использовать для уменьшения интервала. Существует распространенный прием уменьшения диапазона изменения аргумента, который основывается на использовании теорем сложения и умножения элементарных функций. Одним из наиболее простых и универсальных приемов является разбиение всего диапазона изменения на ряд интервалов (сегментная аппроксимация). Обычно аргумент приводят к интервалу [- 1, 1 ] или [ 0, 1]. Такой выбор объясняется наилучшей изученностью поведения функции на этих интервалах, возможностью работы в режиме с фиксированной точкой, наличием точки нуль, которая для многих функций является осью симметрии, и тем, что на данных отрезках существуют ортогональные многочлены.

Этапы приведения аргумента к интервалу аппроксимации и постаброботки специфичны для каждой элементарной функции и мало зависит от выбранного метода вычислений. Наибольшую часть выполнения которого зависит прежде всего от выбранного метода вычисления элементарной функции. Метод должен обладать высокой скоростью сходимости и требовать для своей реализации минимального количества арифметических операций, обеспечивать заданную точность вычисления значений элементарной функции, быть универсальным.

Метод вычисления элементарных функций можно разделить на две группы: алгоритмические и таблично алгоритмические. Методы первой группы основаны на чисто алгоритмических приемах вычислений, которые начинаются “с нуля” и вследствие этого требуют значительных затрат машинного времени. К ним относятся и терационные, полиномиальные методы, в том числе, степенные ряды и др.

Отличительной особенностью таблично алгоритмических методов является использование в той или иной мере предварительно вычисленных табличных значений. Эти методы находят широкое применение для программной и аппаратурной реализации.

4. Теоретические основы таблично алгоритмического метода

В основе методов, основанных на применении таблиц, лежит разбиение интервала аппроксимации на промежутки h (шаг таблицы), длина которых выбирается пропорционально основанию используемой системы счисления. Значения функций, подлежащей реализации, предварительно вычисляются для концов промежутков и заносятся в таблицу, роль которой выполняет ПЗУ. Однако данный подход кроме значительной емкости ПЗУ трудно реализовать вследствие значительности объема предварительных вычислений.

В связи с этим широкое применение в настоящее время нашли таблично алгоритмические методы вычислений, сочетающие поиск по таблице с грубым значением аргумента и введением поправок на точное значение. При определении поправки, характерной для данных методов, может использоваться следующая формула:

 F(x)=F(xs+Δ xn-s)= F(xs)+Ф(Δ xn-s, xs),



где xs – s-разрядный код старшей части аргумента Х,

0≤Δ xn-s<2-s – (n-s)-разрядный код младшей части аргумента Х,

F(xs) – табличное значение функции F(x) для точки xs,

Ф(Δ xn-s, xs) – поправка.

В данном случае величина s, являясь разрядностью аргумента xs , задает значение шага таблицы h=2-s, а также количество табличных значений функции F(xs), равное 2s. Поправки Ф(Δ xn-s, xs) определяются в процессе работы микроЭВМ. Их находят с помощью различных методов, к которым, прежде всего, относятся: разложение в степенные ряды, цепные дроби, дробно рациональные приближения, интерполирование, численное интегрирование, итерационные процессы.

Принципы использования частных методов вычисления поправок, основанных на применении известных тождеств для каждой конкретной элементарной функции. Этот подход характеризуется точностью базовой формулы, зависимостью поправки для большинства элементарных функций только от младшей части аргумента. Чаще всего данный метод является нерациональным из-за значительного числа дополнительных операций.

Формула для вычисления значения F(x) на интервале (xs, xs+h) с использованием ряда Тейлора имеет следующий вид:

 F(x)= F(xs) + Δ xn-s/k!)\*F(k)( xs),

где m – количество членов ряда , Δ xn-s=x - xs,

F(k)( xs) – значение k-ой производной функции F(x) в точке xs.

 Методическая погрешность приведенного алгоритма можно оценить по формуле:

 εм≤((Δ xn-s)m+1/(m+1)!)\*F(m+1)max

где F(m+1)max – максимальное значение (m+1)-ой производной функции F(x) на интервале (xs, xs+h).

 Формула для вычисления значения F(x) на интервале (xs, xs+h) с помощью линейной интерполяции имеет следующий вид:

 F(x)= F(xs) + (Δ xn-s/h)\*(F(xs, xs+h) - F(xs))

 Методическая погрешность приведенного алгоритма можно оценить по формуле:

 εм≤(Δ xn-s(Δ xn-s – h)/2)\*F(2)max

Формулы для вычисления погрешностей:

1. полной ε = Fэт(х) - Fm(xm)/Mf
2. методической εм ≤ ((Δ xn-s)m-1/(m+1)!)\*F(m+1)max
3. вычислительной εв = Fэт\* - Fm(xm)/Mf

**5. Расчет параметров алгоритма**

Одной из задач проектирования алгоритмов таблично алгоритмических методов является удовлетворение требований по времени вычисления элементарной функции в микроЭВМ и точности вычислений, которая обусловлена заданной разрядностью представления аргумента.

Требование по времени tдоп можно выполнить, ограничивая такие параметры численных методов как число членов ряда, полинома, цепной дроби, шагов итераций и т. п. При этом учитывается время выполнения в микро-ЭВМ арифметических, логических и других используемых операций.

Необходимая точность вычислений обеспечивается путем выполнения баланса методической и вычислительной погрешностей алгоритма в виде условия εм = εв где в качестве εв выступает ее минимальное значение 2-(n+1).

В случае разложения в ряд Тейлора баланс погрешностей имеет вид:

 ((Δ xn-s)m+1/(m+1)!)\*|F(m+1)max| = 2-(n+1)

Учитывая что Δ xn-s = 2-s имеем:

 S ≥ (n + 1 - log2((m+1)!)/|F(m+1)max|)/(m+1)

Для большинства элементарных функций зависимость s(m) хорошо аппроксимируется соотношением:

 S ≥ (n + 1 – 2m\*log2e)/(m+1)

Следовательно, по известной разрядности аргумента n и количеству членов ряда m можно оценить минимальную разрядность s старшей части аргумента X, обеспечивающую требуемый баланс методической и вычислительной погрешностей.

В случае использования кусочно-линейной интерполяции баланс погрешностей имеет вид:

 (Δ xn-s|Δ xn-s – h|/2)\*|F(2)max| = 2-(n+1)

Поскольку h=2-s, то левая часть этого неравенства достигает максимального абсолютного значения при Δ xn-s = 2-(s+1).Тогда

2-2s \*|F(2)max|/8 = 2-(n+1)

Отсюда окончательно получаем:

 S ≥ (n + log2|F(2)max| - 2)/2

Величина s является тем параметром, который обеспечивает баланс методической и вычислительной погрешностей, а также определяет не только разрядность старшей части аргумента Х, но и значение шага таблицы h = 2-s и количество табличных значений функции F(xs) равное 2s.

**5.1. Расчёт величины S и h**

 y=sh(x) - исходная функция

Найдем первую и вторую производную y

y'=ch(x) у''=sh(x)

|f'max| = 1.128 |f''max| = 0.521

Зная |f''max| найдем s и h

S ≥ (n + 1 - log2((m+1)!)/|f(m+1)max|)/(m+1) , где n = 8 и m =1

Вычисляя получим s = 4

Тогда h = 2-s = 2-4 = 0.0625

**5.2.Выбор масштабных коэффициентов**

# Мf(x) = = 26 MΔx = = = 26

Mf(xs) = = 26  Mf '(x) = = 25 Mx==27

**6.Масштабирование алгоритма**

f(x) = f(xs) + Δ xn-s⋅f'(xs),

F(X)/Mf(x) = F(Xs)/Mf(xs) + (Δ Xn-s/ MΔx)⋅F'(Xs)/ Mf '(x)

Умножим все выражение на Mf(x)

F(X) = F(Xs)⋅Mf(x)/Mf(xs) + Δ Xn-s⋅F'(Xs) Mf(x)/(Mf '(x)⋅MΔx)

Подставляя значения полученные выше получим

F(X) = F(Xs) + 2-5 ⋅Δ Xn-s⋅F'(Xs)

где F(X)=f(x)⋅Mf(x), F(Xs)=f(xs)⋅Mf(xs), Δ Xn-s= Δ xn-s⋅MΔx,

F'(Xs)= f'(xs)⋅Mf '(x), X = x\*Mx - масштабированные значения

**7. Граф – схема программы**

Вычисление табличных значений функций f(xs) и f '(xs) с шагом h = 2-s на интервале аппроксимации в виде вещественных чисел с помощью библиотек функций TURBO PASCAL

Масштабирование табличных значений F(xs) = f(xs)⋅Mf , F'(xs) = f '(xs)⋅Mf ' , округление масштабированных значений до ближайшего целого и преставление их в виде целых чисел

Получение масштабированного значения аргумента Х=х⋅Мх в виде целого числа для исходного не масштабированного значения х из заданного интервала аппроксимации

Вычисление приближенных, целочисленных (масштабированных) значений функции F(x) по разработанному алгоритму с использованием подпрограммы функции

Вычисление эталонного значения функции f(x) в виде вещественного числа с использованием библиотеки TURBO PASCAL, (εм ≈ 0, εв ≈ 0 – для эталона)

Вычисление полной погрешности ε = f(x) – F(x)/Mf

Вычисление функции f(\*)(x) по разработанному не масштабированному алгоритму в виде вещественного числа (εв≈0)

# Нахождение вычислительной погрешности εв=f\*(x) – F(x)/Mf

# Нахождение методической погрешности εм=ε-εв или f(x)–f\*(x)

Вычисления закончены?

1. **Листинг и описание подпрограммы функции**

**8.1.Листинг подпрограммы функции**

function Fx(masFx,masFhx:m;Xm:word;var ad:word):longint;

 var

 dx,fx1:word;

 begin

 ad:=Xm shr 3;

 dx:=Xm and $6;

 fx1:=dx\*masFhx[ad];

 fx1:=fx1 shr 5;

 Fx:=masFx[ad]+fx1;

 end;

**8.2. граф – схема алгоритма**

Сдвиг Xm на 3 разряда вправо

dx:=Xm&0110

fx1:=dx⋅masfx[ad]

Сдвиг fx1 на 5 разрядов вправо

#### Fx:=masFx[ad]+fx1

**9. Листинг программы**

program kir;

uses crt,dos,graph;

type

 m=array[0..8] of word;

const

 h=0.0625;

 Mh=64;

 Mf=64;

 Mfh=32;

 Mx=128;

function Fx(masFx,masFhx:m;Xm:word;var ad:word):longint;

 var

 dx,fx1:word;

 begin

 ad:=Xm shr 3;

 dx:=Xm and $6;

 fx1:=dx\*masFhx[ad];

 fx1:=fx1 shr 5;

 Fx:=masFx[ad]+fx1;

 end;

var

 masFx,masFhx:m;

 masfxs,masfhxs,x:array[0..8] of real;

 i,Xm,ad:word;

 work,F,fxn,e,ev,em,fz:real;

 gd,gm:integer;

 xnext,xprev,y1next,y1prev:integer;

 y2next,y2prev,y3next,y3prev:integer;

 y4next,y4prev:integer;

begin

 clrscr;

 for i:=0 to 8 do begin

 x[i]:=h\*i;

 masfxs[i]:= (exp(x[i])-exp(-x[i]))/2;

 masfhxs[i]:= (exp(x[i])+exp(-x[i]))/2;

 masFx[i]:=trunc(masfxs[i]\*Mf);

 masFhx[i]:=trunc(masfhxs[i]\*Mfh);

 end;

 work:=0;

 repeat

 begin

 Xm:=trunc(work\*Mx);

 F:=Fx(masFx,masFhx,Xm,ad);

 fxn:=(exp(work)-exp(-work))/2;

 e:=fxn-F/Mf;

 fz:=masfxs[ad]+(work-x[ad])\*masfhxs[ad];

 ev:=fz-F/Mf;

 em:=e-ev;

 write(' ',work:5:4,' ',Xm:2,' ',fxn:4:4,' ',F:4:0,' ');

 writeln(e:4:4,' ',ev:4:4,' ',em:4:4);

 work:=work+h/4;

 end;

 until work>0.5;

 writeln(' x X f(x) F(x) E Ev Em');

 readln;

 gd:=detect;

 initgraph(gd,gm,'c:\language\bp\bgi');

 cleardevice;

 floodfill (0,0,white);

 setcolor(black);

 line(0,0,0,480);

 line(0,240,640,240);

 work:=0;

 xnext:=1;

 xprev:=0;

 y2next:=240;

 y1prev:=240-trunc((exp(work)-exp(-work))\*120);y2prev:=240;

 y3prev:=240;y4prev:=240;

 repeat

 begin

 work:=work+h/4;

 fxn:= (exp(work)-exp(-work))/2;

 y1next:=240-trunc(fxn\*240);

 setcolor(green);

 line(xprev,y1prev,xnext,y1next);

 y1prev:=y1next;

 Xm:=trunc(work\*Mx);

 F:=Fx(masFx,masFhx,Xm,ad);

 e:=fxn-F/Mf;

 y2next:=240-trunc(e\*240);

 setcolor(cyan);

 line(xprev,y2prev,xnext,y2next);

 y2prev:=y2next;

 fz:=masfxs[ad]+(work-x[ad])\*masfhxs[ad];

 ev:=fz-F/Mf;

 y3next:=240-trunc(ev\*240);

 setcolor(magenta);

 line(xprev,y3prev,xnext,y3next);

 y3prev:=y3next;

 em:=e-ev;

 y4next:=240-trunc(em\*240);

 setcolor(lightgray);

 line(xprev,y4prev,xnext,y4next);

 y4prev:=y4next;

 xprev:=xnext;

 xnext:=xnext+trunc(64\*work);

 end;

 until work=0.5;

 setcolor(green);

 outtextxy(0,470,'f(x)');

 setcolor(cyan);

 outtextxy(40,470,'E');

 setcolor(magenta);

 outtextxy(60,470,'Ev');

 setcolor(lightgray);

 outtextxy(90,470,'Em');

 readln;

end.

1. **Результаты работы программы**

 0.1563 20 0.1569 12 -0.0306 -0.0307 0.0001

 0.1719 22 0.1727 14 -0.0460 -0.0462 0.0002

 0.1875 24 0.1886 12 0.0011 0.0011 0.0000

 0.2031 26 0.2045 14 -0.0142 -0.0142 0.0000

 0.2188 28 0.2205 16 -0.0295 -0.0296 0.0001

 0.2344 30 0.2365 18 -0.0447 -0.0449 0.0002

 0.2500 32 0.2526 16 0.0026 0.0026 0.0000

 0.2656 34 0.2688 18 -0.0125 -0.0125 0.0000

 0.2813 36 0.2850 20 -0.0275 -0.0277 0.0001

 0.2969 38 0.3013 22 -0.0425 -0.0428 0.0003

 0.3125 40 0.3176 20 0.0051 0.0051 0.0000

 0.3281 42 0.3340 22 -0.0097 -0.0097 0.0000

 0.3438 44 0.3506 24 -0.0244 -0.0246 0.0002

 0.3594 46 0.3672 26 -0.0391 -0.0395 0.0004

 0.3750 48 0.3839 24 0.0089 0.0089 0.0000

 0.3906 50 0.4006 26 -0.0056 -0.0057 0.0000

 0.4063 52 0.4175 28 -0.0200 -0.0202 0.0002

 0.4219 54 0.4345 30 -0.0342 -0.0347 0.0004

 0.4375 56 0.4516 28 0.0141 0.0141 0.0000

 0.4531 58 0.4688 30 0.0000 -0.0000 0.0001

 0.4688 60 0.4861 32 -0.0139 -0.0141 0.0002

 0.4844 62 0.5035 34 -0.0277 -0.0282 0.0005

 0.5000 64 0.5211 33 0.0055 0.0055 0.0000

 x X f(x) F(x) E Ev Em

**11. Заключение**

 В процессе выполнения курсовой работы, используя литературу и методические указания, мы выполнили проектирование алгоритма вычисления элементарной функции с использованием таблично – алгоритмического метода. Алгоритм ориентируется на целочисленные вычисления в формате байт со знаком в дополнительном коде. Разработали программу для вычисления функции на заданном интервале аппроксимации и экспериментального анализа, полной, методической и вычислительной погрешности. Программа выводит на экран графики функции и всех погрешностей на интервале аппроксимации.

**12. Список литературы**

1. Руководство к лабораторной работе №4 “проектирование алгоритмов вычисления элементарных функций” *по курсу*:

 Основы обработки данных и моделирования

# 2. Конспект лекций