**Міністерство освіти і науки України**

# **Вінницький державний технічний університет**

**Інститут ІНАЕКСУ**

###### **Факультет АКСУ**

###### **Кафедра АІВТ**

## Курсова робота з дисципліни :

## «Обчислювальні методи та застосування ЕОМ»

Керівник професор, д.т.н. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ??????????

Студент гр. ??????? \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ??????????

2003

Зміст

####  Завдання ………………………………………………………………………..

1.Загальні відомості ……………………………………………………………..

2.Вибір методу інструментальних засобів вирішення задач………………

3.Функціональне призначення програми …………………………………….

##### 4.Розробка та опис логічної частини програми ………………………………

##### 5.Керівництво оператору …………………………………………………………

##### 6.Результати обчислень ………………………………………………………….

##### Висновки …………………………………………………………………………….

##### Література …………………………………………………………………………..

 Додаток А

 Лістинг програми……………………………………………………………..

 Додаток B

 Блок-схема алгоритму ………………………………………………………

### **Анотація**

В даній курсовій роботі проведено дослідження двох чисельних методів вирішення диференційних рівнянь: Ейлера та Рунне-Кутта. Дослідження проводиться на прикладі заданого диференційного рівняння. Дається опис кожного з методів та задачі в цілому.

### **1. Загальні відомості**

Найбільш простим однокроковим методом, який потребує мінімальних затрат обчислювальних ресурсів, але дає змогу обчислювати результат із порівняно низькою точністю, є метод Ейлера.

В цьому методі для оцінки наступної точки на кривій  використовується лише один лінійний член в формулі Тейлора,

,

де  визначається з початкового рівняння.

 Цей процес можна розповсюдити на наступні кроки:

.

Метод Ейлера є методом першого порядку 

,

де , , ,  - визначається як



для всіх  і .

Метод Ейлера, крім значної похибки зрізання часто буває нестійким (малі локальні похибки призводять до значного збільшення глобальної).

Цей метод можна вдосконалити різними способами.

Найбільш відомі два з них: виправлений метод Ейлера і модифікований метод Ейлера (в літературі зустрічаються інші назви цих методів, наприклад, модифікований метод Ейлера й удосконалений метод ламаних).

Ітераційні формули для цих методів мають вигляд, відповідно:



і



де

.

Це методи другого порядку, їх похибка має третій ступінь, що досягається покращенням апроксимації похідної. Ідея полягає у спробі зберегти або оцінити член другого порядку у формулі Тейлора. Однак збільшення точності вимагає додаткових витрат машинного часу на обчислення . Ще більш висока точність може бути досягнута при обчисленні вищих похідних і збереженні більшої кількості членів ряду Тейлора. Такими методами є методи Рунге-Кутта.

Принцип на якому побудований модифікований метод Ейлера, можна пояснити, користуючись рядом Тейлора і зберігаючи в ньому член з . Апроксимація другої похідної  здійснюється кінцевою різницею

.

Аналогічно обчисленню другої похідної в кінцево-різницевому вигляді можна обчислити більш високі похідні: значення n-ї за значеннями попередньої (n-1)-ї.

Метод Рунге-Кутта дає набір формул для обчислення координат внутрішніх точок, які потрібні для реалізації цієї ідеї. Оскільки існує ряд способів знаходження цих точок, то метод Рунге-Кутта обєднує цілий клас методів для розвязання диференціальних рівнянь першого порядку.

Найбільш розповсюджений класичний метод четвертого порядку точності:

,

де

,

.

Метод Ейлера і його модифікації ще називають методами Рунге-Кутта першого і другого порядку. Метод Рунге-Кутта має значно більш високу точність, що дозволяє збільшити крок розвязання. Його максималу величину визначає допустима похибка. Такий вибір часто здійснюється автоматично і включається як складова частина, вбудована в алгоритм, побудований за методом Рунге-Кутта.

Раніше було відзначено, що помилка зрізання при використанні методу Рунге-Кутта n-го порядку . Обчислення верхніх границь для коефіцієнта с являє собою складну задачу, повязану з необхідністю оцінки ряду додаткових параметрів. Існує декілька способів для оперативного обчислення с. Найбільшого поширення набув екстраполяційний метод Річардсона (ще його називають методом Рунге), коли послідовно знаходять  з кроком h і з кроком , а після цього прирівнюють отримані величини та визначають с з рівняння:

,

що відповідає точному значенню .

Отримаємо оціночне співвідношення:

.

**2.Вибір методу інструментальних засобів вирішення задач.**

Розв”язок даної задачі реалізовано на ЕОМ, причому було складено алгоритм та програму в середовищі Borland Delphi 7. Програма є досить простою та зрозумілою для користувача середнього рівня. Готову програму можна використовувати навіть на мінімальних системних параметрах процесора типу Intel P-100, 8 Мb ОЗУ та операційній системі MS-Windows 95.

**3. Функціональне призначення**

Розроблена програма дозволяє розв’язати вказане диференційне рівняння методами Ейлера (прямим та зворотним) та Рунге-Кутта, порівняти їх результати та визначити похибки:

,

де а, b, проміжок х та крок можна задавати в самій програмі.

Результати виводяться графічно (графік) та у текстовій формі.

**4. Розробка та опис логічної частини програми**

В даній курсовій роботі було розроблено програмне забезпечення для розв’язання та дослідження заданого диференційного рівняння. Розвязок ведеться за трьома алгоритмами – прямим та зворотнім Ейлера та Руннге-Кутта, що легко дозволяє проаналізувати різницю між ними, і особливо в похибках обчислень.

Кодування на мові Паскаль проводилося з застосуванням інтуїтивно-зрозумілих назв змінних та процедур. Також відступи та табуляція дозволяє досить легко збагнути структуру програми.

В інтерфейсі також не допущено зайвих елементів.

**5. Керівництво оператору**

Для завантаження програми необхідно запустити програмний файл Kurs.exe. При цьому зявиться вікно (рис. 1), де при виборі відповідного пункту меню вирішується задача по обраному методу.

Графік результуючої функції знаходиться у лівій частині вікна, а супутня інформація – у правій.

Рисунок 1. Інтерфейс програми.

**6. Результати обчислень**

**Висновки**

 При виконані даної курсової роботи я навчився розв”язувати диференційні рівняння.Завдання цієї роботи вимагало детального вивчення методів розвязку диференційних рівнянь. Причому я встановив, що поміж досліджених методів – Рунне-Кутта є найбільш точним.

Література

1. Самарський А.А. **Вступ в чисельні методи**. - М.: Наука,

1987. – 286 с.

2.Квєтний Р.Н., Маліков В.Т. **Обчислювльні методи та використання ЕОМ**. Вища школа, 1989 – 55 с., 104 с.

**Додаток В - Лістинг програми**

// Kurs.cpp : Defines the class behaviors for the application.

//

#include "stdafx.h"

#include "Kurs.h"

#include "MainFrm.h"

#include "ChildFrm.h"

#include ".\kurs.h"

#include "StepDlg.h"

#ifdef \_DEBUG

#define new DEBUG\_NEW

#undef THIS\_FILE

static char THIS\_FILE[] = \_\_FILE\_\_;

#endif

/////////////////////////////////////////////////////////////////////////////

// CKursApp

BEGIN\_MESSAGE\_MAP(CKursApp, CWinApp)

 //{{AFX\_MSG\_MAP(CKursApp)

 ON\_COMMAND(ID\_APP\_ABOUT, OnAppAbout)

 ON\_COMMAND(ID\_FILE\_NEW, OnFileNew)

 ON\_COMMAND(ID\_METD1, OnMetd1)

 ON\_COMMAND(ID\_METD2, OnMetd2)

 ON\_COMMAND(ID\_METD3, OnMetd3)

 //}}AFX\_MSG\_MAP

 ON\_COMMAND(ID\_32777, On32777)

END\_MESSAGE\_MAP()

/////////////////////////////////////////////////////////////////////////////

// CKursApp construction

CKursApp::CKursApp()

: k1(0)

, k2(0)

, k3(0)

{

 // TODO: add construction code here,

 // Place all significant initialization in InitInstance

}

/////////////////////////////////////////////////////////////////////////////

// The one and only CKursApp object

CKursApp theApp;

/////////////////////////////////////////////////////////////////////////////

// CKursApp initialization

BOOL CKursApp::InitInstance()

{

 // Standard initialization

 // If you are not using these features and wish to reduce the size

 // of your final executable, you should remove from the following

 // the specific initialization routines you do not need.

// Enable3dControls(); // Call this when using MFC in a shared DLL

 // Change the registry key under which our settings are stored.

 // TODO: You should modify this string to be something appropriate

 // such as the name of your company or organization.

 SetRegistryKey(\_T("Local AppWizard-Generated Applications"));

 // To create the main window, this code creates a new frame window

 // object and then sets it as the application's main window object.

 CZast dlg;

 dlg.Create(IDD\_DIALOG2);

 dlg.ShowWindow(SW\_SHOW);

 dlg.RedrawWindow();

 Sleep(1000);

 dlg.ShowWindow(SW\_HIDE);

 CMDIFrameWnd\* pFrame = new CMainFrame;

 m\_pMainWnd = pFrame;

 // create main MDI frame window

 if (!pFrame->LoadFrame(IDR\_MAINFRAME))

 return FALSE;

 // try to load shared MDI menus and accelerator table

 //TODO: add additional member variables and load calls for

 // additional menu types your application may need.

 HINSTANCE hInst = AfxGetResourceHandle();

 m\_hMDIMenu = ::LoadMenu(hInst, MAKEINTRESOURCE(IDR\_MAINFRAME));

 m\_hMDIAccel = ::LoadAccelerators(hInst, MAKEINTRESOURCE(IDR\_MAINFRAME));

 // The main window has been initialized, so show and update it.

 pFrame->ShowWindow(m\_nCmdShow);

 pFrame->UpdateWindow();

 k1 = 15;

 k2 = 16;

 k3 = 10;

 return TRUE;

}

/////////////////////////////////////////////////////////////////////////////

// CKursApp message handlers

int CKursApp::ExitInstance()

{

 //TODO: handle additional resources you may have added

 if (m\_hMDIMenu != NULL)

 FreeResource(m\_hMDIMenu);

 if (m\_hMDIAccel != NULL)

 FreeResource(m\_hMDIAccel);

 return CWinApp::ExitInstance();

}

void CKursApp::OnFileNew()

{

 CMainFrame\* pFrame = STATIC\_DOWNCAST(CMainFrame, m\_pMainWnd);

 // create a new MDI child window

 pFrame->CreateNewChild(

 RUNTIME\_CLASS(CChildFrame), IDR\_MAINFRAME, m\_hMDIMenu, m\_hMDIAccel);

}

/////////////////////////////////////////////////////////////////////////////

// CAboutDlg dialog used for App About

class CAboutDlg : public CDialog

{

public:

 CAboutDlg();

// Dialog Data

 //{{AFX\_DATA(CAboutDlg)

 enum { IDD = IDD\_ABOUTBOX };

 //}}AFX\_DATA

 // ClassWizard generated virtual function overrides

 //{{AFX\_VIRTUAL(CAboutDlg)

 protected:

 virtual void DoDataExchange(CDataExchange\* pDX); // DDX/DDV support

 //}}AFX\_VIRTUAL

// Implementation

protected:

 //{{AFX\_MSG(CAboutDlg)

 // No message handlers

 //}}AFX\_MSG

 DECLARE\_MESSAGE\_MAP()

};

CAboutDlg::CAboutDlg() : CDialog(CAboutDlg::IDD)

{

 //{{AFX\_DATA\_INIT(CAboutDlg)

 //}}AFX\_DATA\_INIT

}

void CAboutDlg::DoDataExchange(CDataExchange\* pDX)

{

 CDialog::DoDataExchange(pDX);

 //{{AFX\_DATA\_MAP(CAboutDlg)

 //}}AFX\_DATA\_MAP

}

BEGIN\_MESSAGE\_MAP(CAboutDlg, CDialog)

 //{{AFX\_MSG\_MAP(CAboutDlg)

 // No message handlers

 //}}AFX\_MSG\_MAP

END\_MESSAGE\_MAP()

// App command to run the dialog

void CKursApp::OnAppAbout()

{

 CAboutDlg aboutDlg;

 aboutDlg.DoModal();

}

/////////////////////////////////////////////////////////////////////////////

// CKursApp message handlers

void CKursApp::OnMetd1()

{

 SP\_Eylr();

 CMainFrame\* pFrame = STATIC\_DOWNCAST(CMainFrame, m\_pMainWnd);

 // create a new MDI child window

 CMDIChildWnd\* ch=pFrame->CreateNewChild(

 RUNTIME\_CLASS(CChildFrame), IDR\_MAINFRAME, m\_hMDIMenu, m\_hMDIAccel);

 ch->SetTitle("╠хЄюф ┼щыхЁр");

}

void CKursApp::OnMetd2()

{

 SP\_ZEylr();

 CMainFrame\* pFrame = STATIC\_DOWNCAST(CMainFrame, m\_pMainWnd);

 // create a new MDI child window

 CMDIChildWnd\* ch=pFrame->CreateNewChild(

 RUNTIME\_CLASS(CChildFrame), IDR\_MAINFRAME, m\_hMDIMenu, m\_hMDIAccel);

 ch->SetTitle("╟тюЁюЄэшщ ьхЄюф ┼щыхЁр");

}

void CKursApp::OnMetd3()

{

 SP\_Rung();

 CMainFrame\* pFrame = STATIC\_DOWNCAST(CMainFrame, m\_pMainWnd);

 // create a new MDI child window

 CMDIChildWnd\* ch=pFrame->CreateNewChild(

 RUNTIME\_CLASS(CChildFrame), IDR\_MAINFRAME, m\_hMDIMenu, m\_hMDIAccel);

 ch->SetTitle("╠хЄюф ╨єэух-╩єЄЄр");

}

void CKursApp::SP\_Eylr()

{

 double fromx,tox;

 double M0,M1,M2;

 int i;

 CStepDlg dlg;//(IDD\_DIALOG3);

 dlg.DoModal();

 if (h==0) h = 0.05;

 fromx= 0;

 tox= 1;

 n = (tox-fromx)/h + 1;

 m\_ay[0]=1;

 M0 = f(m\_ax[0],m\_ay[0]);

 M1 = f\_x(m\_ax[0],m\_ay[0]);

 M2 = f\_y(m\_ax[0],m\_ay[0]);

 for (i=0;i<n;i++) m\_ax[i] = fromx + i\*h;

 for (i=1;i<n;i++){

 m\_ay[i] = m\_ay[i-1]+h\*f(m\_ax[i-1],m\_ay[i-1]);

 M0 = max(M0,f(m\_ax[i],m\_ay[i]));

 M1 = max(M1,f\_x(m\_ax[i],m\_ay[i]));

 M2 = max(M2,f\_y(m\_ax[i],m\_ay[i]));

 }

 err=(M1+M0\*M2)/2\*h\*h;

}

void CKursApp::SP\_Rung()

{

 double fromx,tox;

 double M0,M1,M2;

 int i;

 double k0,k1,k2,k3;

 CStepDlg dlg;//(IDD\_DIALOG3);

 dlg.DoModal();

 if (h==0) h = 0.05;

 fromx= 0;

 tox= 1;

 n = (tox-fromx)/h + 1;

 m\_ay[0]=1;

 M0 = f(m\_ax[0],m\_ay[0]);

 M1 = f\_x(m\_ax[0],m\_ay[0]);

 M2 = f\_y(m\_ax[0],m\_ay[0]);

 for (i=0;i<n;i++) m\_ax[i] = fromx + i\*h;

 for (i=1;i<n;i++){

 k0=h\*f(m\_ax[i-1],m\_ay[i-1]);

 k1=h\*f(m\_ax[i-1]+h/2,m\_ay[i-1]+k0/2);

 k2=h\*f(m\_ax[i-1]+h/2,m\_ay[i-1]+k1/2);

 k3=h\*f(m\_ax[i-1]+h,m\_ay[i-1]+k2);

 m\_ay[i] = m\_ay[i-1]+(k0+2\*k1+2\*k2+k3)/6;

 M0 = max(M0,f(m\_ax[i],m\_ay[i]));

 M1 = max(M1,f\_x(m\_ax[i],m\_ay[i]));

 M2 = max(M2,f\_y(m\_ax[i],m\_ay[i]));

 }

 err=(M1+M0\*M2)/2\*h\*h\*h;

}

void CKursApp::SP\_ZEylr()

{

 double fromx,tox;

 double M0,M1,M2;

 int i;

 CStepDlg dlg;//(IDD\_DIALOG3);

 dlg.DoModal();

 if (h==0) h = 0.05;

 fromx= 0;

 tox= 1;

 n = (tox-fromx)/h + 1;

 m\_ay[0]=1;

 M0 = f(m\_ax[0],m\_ay[0]);

 M1 = f\_x(m\_ax[0],m\_ay[0]);

 M2 = f\_y(m\_ax[0],m\_ay[0]);

 for (i=0;i<n;i++) m\_ax[i] = fromx + i\*h;

 for (i=1;i<n;i++){

 m\_ay[i] = m\_ay[i-1]+h\*f(m\_ax[i-1],fy(m\_ax[i],f(m\_ax[i-1],m\_ay[i-1])));

 M0 = max(M0,f(m\_ax[i],m\_ay[i]));

 M1 = max(M1,f\_x(m\_ax[i],m\_ay[i]));

 M2 = max(M2,f\_y(m\_ax[i],m\_ay[i]));

 }

 err=(M1+M0\*M2)/2\*h\*h;

}

void CKursApp::On32777()

{

 CHelp dlg;

 dlg.DoModal();

}

 **Додаток A – Алгоритм роботи програми**

### Початок



Задання початкових умов

для **х** від **хпоч** до **хкін** крок **h**

для **х** від **хпоч** до **хкін** крок **h**



для **х** від **хпоч** до **хкін** крок **h**

.

### Кінець