# Введение

В ходе исследования различных явлений мира окружающего нас, относящихся как к гуманитарным, так и к точным областям наук, ученые периодически сталкиваются с тем, что отношения и корелляции между величинами в ходе решения уравнений с производными от искомых функций. Простейшими и самыми распространенными среди них являются функции, содержащие лишь производные первого порядка и которые могут быть записаны в виде

= f(x, y) ,

где у – функция, которую мы ищем, х - независимая переменная, f(x,y) - непрерывная функция от х и у. Но аналитически вычислить решение данного уравнения для произвольной функции f не всегда удается, и лишь для редких частных ситуаций.

С быстрым развитием компьютеров и микро-процессоров (электронной вычислительной техники) в последние годы есть опция воспользоваться приближенными математическими методами в ходе решения таких и подобных задач. В частности, одна из групп таких методов называется группой методов Рунге-Кутты, также существует группа методов Эйлера.

Цель курсового проекта: изучить методы Эйлера и Рунге – Кутта 4-го порядка и использовать их для решения обыкновенных дифференциальных уравнений.

Описание задачи: нужно создать визуальное приложение на языке программирования C# в среде разработки Microsoft Visual Studio, позволяющее решать ОДУ методами Эйлера и Рунге – Кутта 4-го порядка.

# Теоретическое описание

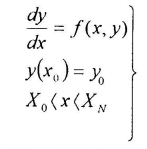
# 1.1 Задача Коши и ОДУ первого порядка

Для упрощения мы будем рассматривать двумерное пространство переменных x и y и принадлежащее ему открытое множество G.

Пусть f (x, y) - непрерывно дифференцируемая функция над этим открытым множеством и уравнение

= f(x, y) (1)

Согласно теореме существования и единственности для каждой точки (x0, y0) ∈G существует решение y = y(x), которое определено в интервале (x0 -δ, x0 + δ) и соответствует условию y(х­0) = y0 такое, где точки y'x ≡ f(x,y(x)) и (x,y(x))∈G и это решение является единственным. Для уравнения (1) с y(x0) = y0 (задача Коши) задача состоит в том, чтобы найти y(x), приводящее уравнение (1) и начальное условие к тождеству. Предположим, что значения, которые принимает независимая переменная x, принадлежат интервалу (X0, XN), и запишем задачу Коши:

(2)

Разделите отрезок [Х0, XN ] на N частей так, чтобы xn+1 – хn = hn ,

n = 0, … ,N-1. Далее без ограничения общности мы будем рассматривать случай, когда разбиение является равномерным, т.е. все hn = h = = const,

n = 0 ,… ,N-1.

# Сущность метода Эйлера

Рассмотрим ОДУ

*y’ = f(x,y)*  (1)

с начальными значениями

*y(x0) = y0*

Подставляя *x0* в уравнение (1), *x0,y0* мы получаем производную в точке :

*y’|x=x0 = f(x0,y0)*

Для наименьших значений *Δx* получается следующее:

*y(x0 + Δx) = y(x1) = y0 + Δy = y0 + y’|x = x0 \* Δx = y0 + f(x0,y0) \* Δx*

Переписываем крайнее равенство в виде: *f(x0,y0)=f0*

*y1 = y0 + f0 \* Δx*       (2)

Если теперь мы возьмем *(x1,y1)* за новую отправную точку *y1*, мы получим то же самое:

*y 2= y1 + f1 \* Δx*

В итоге мы получим:

*yi+1 = yi + fi \*Δx*    (3)

В этом и состоит метод Эйлера. В данном случае *Δx* является шагом интегрирования. Благодаря этому методу мы можем получить приближенные значения y,т.к. производная, в реальности, не является константой во всем диапазоне. Итак, мы получим ошибку в ходе определения значения функции *y’*, тем большую, чем больше *Δx*. Метод Эйлера - является одним из простейших подходов в численном интегрировани дифференциальных уравнений и систем. Его недостатками являются - низкая точность и систематическое накопление ошибок.

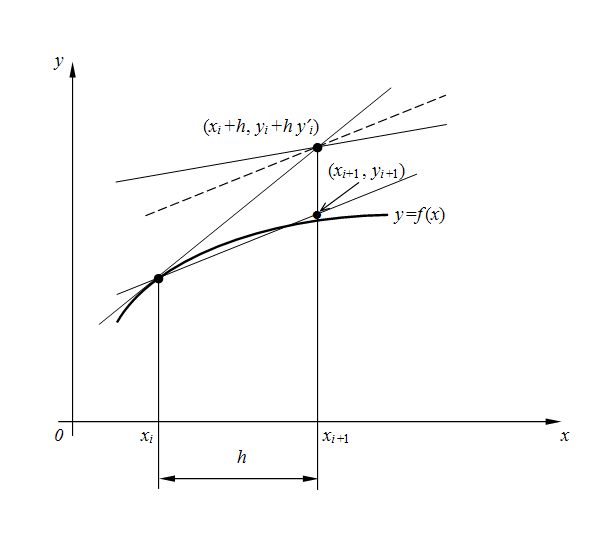
Более точным методом является Модифицированный метод Эйлера с преобразованием. В данном методе мы сначала находим так называемое «грубое приближение» (прогноз) по формуле (3).

http://www.simumath.net/library/materials/Dif_Ur_method_Euler/images/Eqn017.png

далее, в ходе пересчетаhttp://www.simumath.net/library/materials/Dif_Ur_method_Euler/images/Eqn018.png получают тоже приближенное, но заметно более точное значение (коррекция):

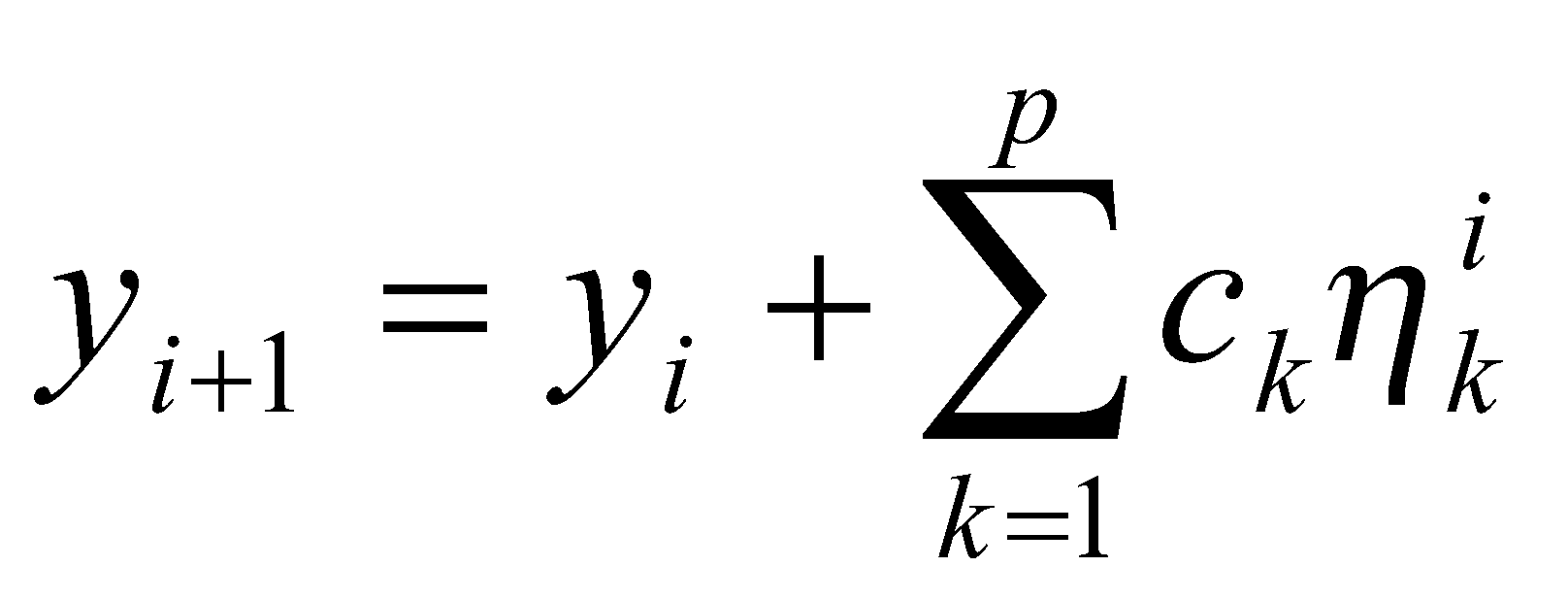
http://www.simumath.net/library/materials/Dif_Ur_method_Euler/images/Eqn019.png          (4)

Фактически, данный метод дает возможность учесть, пусть и приближенно, изменение значений производной *y’* для каждог шага интегрирования *Δx*, так как берутся в рассчет ее значения *fi* в начале и *fi+1* в концеинтервала интегрирования (рис. 1), а затем берется их серединное значение. Метод Эйлера с пересчетом (4) является по своей сути методом Рунге-Кутта 2-го порядка [2], что станет очевидным в дальнейшем.

  
Рис. 1. метод Эйлера с пересчетом

# Суть метода Рунге-Кутта

Для разрешения ДУ широко используются методы Рунге-Кутта. Самый распространенный способ - четвертого порядка.

 (3)

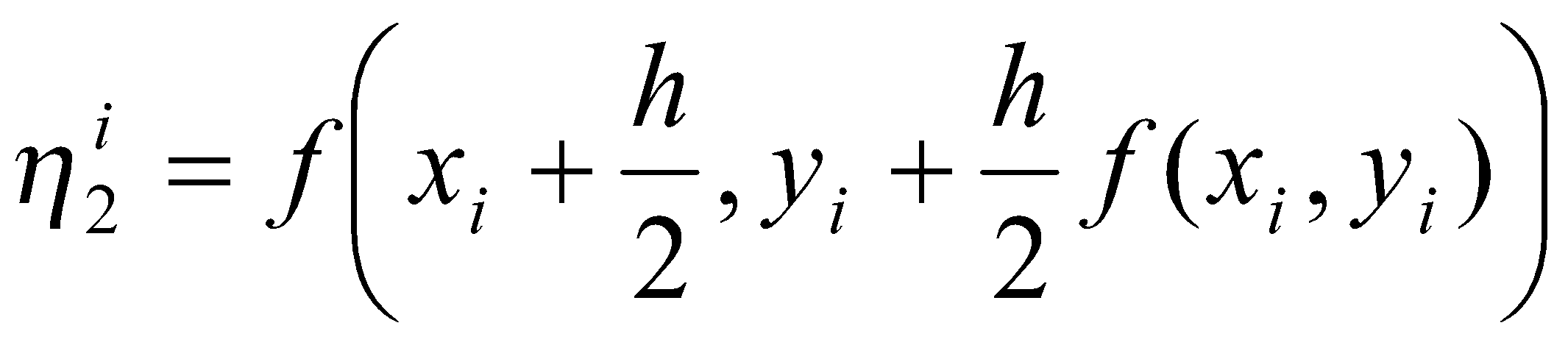
(4)

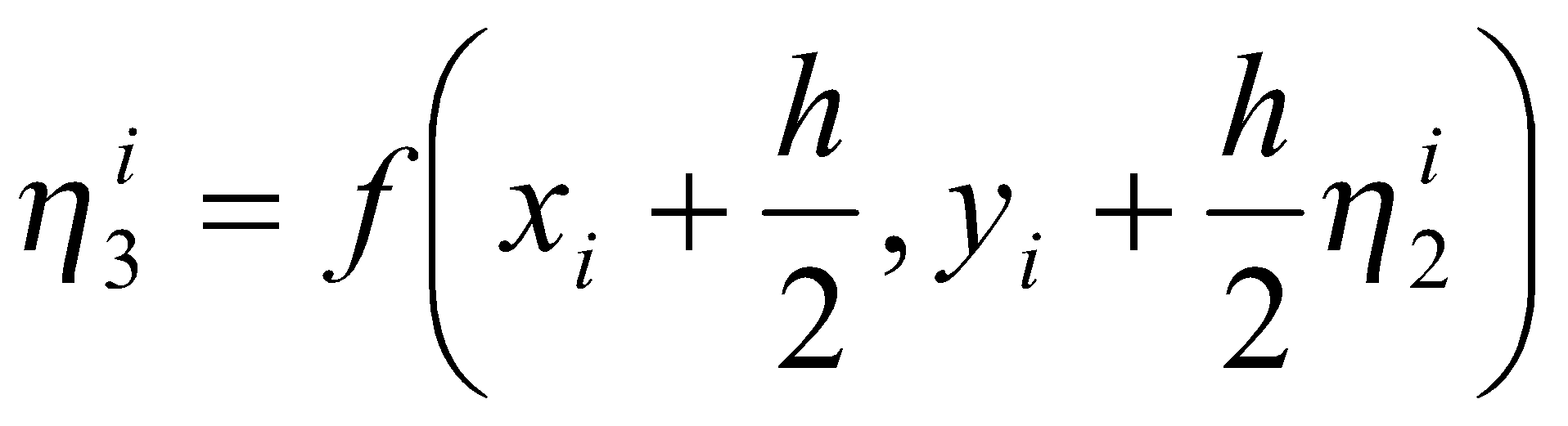
(5)

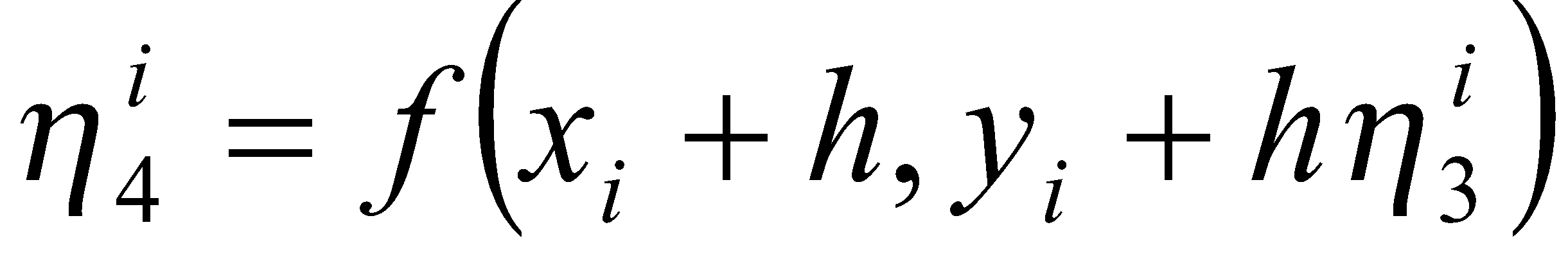
- параметр, определяющий значение функции около точки области определения.

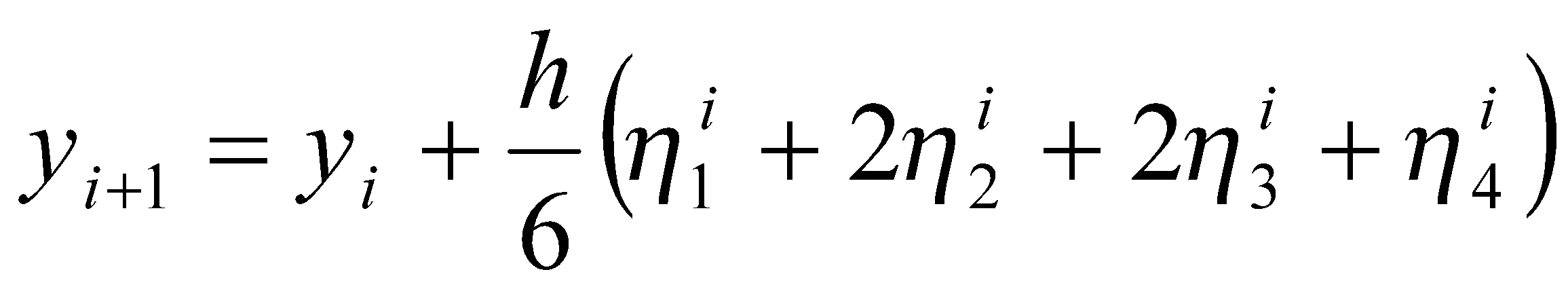
Общепринятый метод четвертого порядка:

(6)

 (7)

 (8)

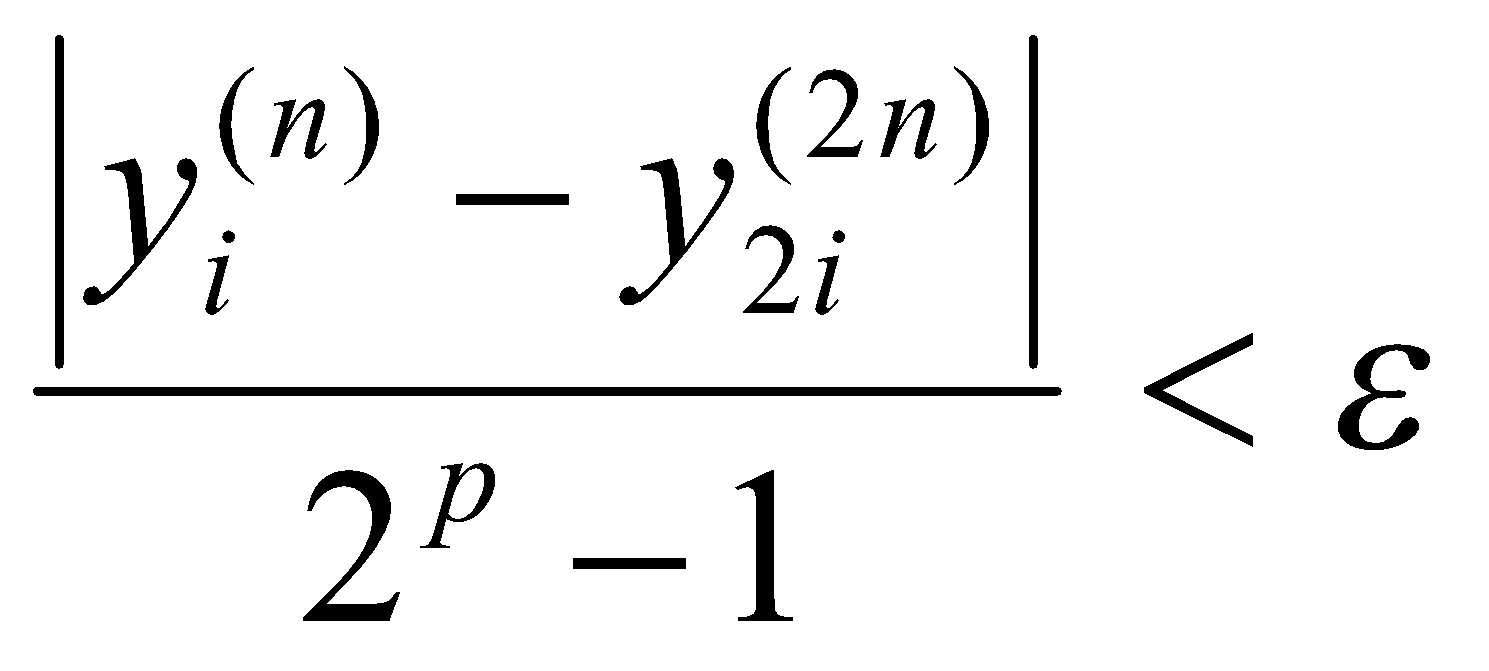
 (9)

 (10)

Ошибка формулы (10) пропорциональна h5.

Этот метод намного точнее методов Эйлера, но также требует большего количества вычислений: положение точки (xi+1, yi+1) определяется четырехкратным вычислением значения функции f (x,y). С появлением компьютера этот недостаток перестал быть существенным, и метод Рунге-Кутты четвертого порядка получил чрезвычайно широкое распространение на практике.

Количество микро сегментов [xi; xi+1], на которые разбивается начальный сегмент [x0;xn], зависит от требуемой точности вычислений. Для достижения необходимой точности задача решается несколько раз последовательным удвоением количества микро сегментов n. Точность считается достигнутой, когда значения (в точках сходимости x) начального и двойного числа n не отличаются более чем на определенное значение:

, i =0, ..,n,(11)

где p – порядок точности метода.

Метод Рунге-Кутта обладает следующими свойствами:

1. Одношаговый метод (для нахождения уm+1 нужна предыдущая точка хm, уm)
2. Он не требует производной функции f (x, y), но необходимо вычислить саму функцию.
3. Имеет маленькую погрешность

# 1.3 Выбор среды разработки

Microsoft Visual Studio - это линейка продуктов от Microsoft, которая включает IDE и множество других инструментов. С помощью этого продукта вы можете создавать консоли, приложения с графическим интерфейсом, а также сайты, приложения, службы в собственном и управляемом коде для всех платформ.

Для своей задачи я выбрал программное обеспечение C#. Потому что он проще в использовании и отвечает всем требованиям, необходимым для создания приложения с графическим интерфейсом.

# Практическая часть

Разработка программы начинается с описания функций.

Листинг 1 «описание функций»

Далее следует объявление и инициализация переменных, которые участвуют в вычислительном процессе: коэффициенты для уравнений, границы заданного отрезка, шаг, начальные условия (они будут получены в ходе использования программы посредством ручного ввода).

Когда все необходимые данные получены, мы переходим непосредственно к решению ОДУ методом Рунге - Кутты 4-го порядка и методом Эйлера.

Листинг 2 «Реализация вычисления ОДУ методом Рунге-Кутта 4-го порядка»

Листинг 3 «Реализация вычисления ОДУ методом Эйлера»

Заключение

В ходе курсовой работы была реализована задача, а именно создана программа с графическим интерфейсом на языке C#, с помощью которой можно решать обыкновенные дифференциальные уравнения по методам Эйлера и Рунге-Кутты 4-го порядка.

В ходе тестирования программы мы получили результаты, показывающие, что решение этими методами в достаточной степени совпадают с аналитическими.

# Список использованных источников

1. Березин И.С., Жидков Н.П., Методы вычислений: Т.2 – М.: ГИФМЛ, 1960. – 620 с.
2. Бахвалов Н. С., Жидков Н. П., Кобельков Г. М. Численные методы. – М.: Бином, 2001 - с. 363-375.
3. Копченова Н.В., Марон И.А., Вычислительная математика в примерах и задачах – М.: Наука, 1972. – 368 с.
4. <https://ru.wikipedia.org/wiki/Microsoft_Visual_Studio>
5. <https://ru.wikipedia.org/wiki/C%2B%2B_Builder>