Кузнецов М.В.

Решение систем дифференциальных уравнений методом Рунге - Кутты 4 порядка

Источник: диск российской компании “AlexSoft”

Обыкновенные дифференциальные уравнения (ОДУ) широко используются для математического моделирования процессов и явлений в различных областях науки и техники. Переходные процессы в радиотехнике, кинетика химических реакций, динамика биологических популяций, движение космических объектов, модели экономического развития исследуются с помощью ОДУ.

В дифференциальное уравнение n-го порядка в качестве неизвестных величин входят функция y(x) и ее первые n производных по аргументу x

$$\varphi(x, y, y^1, \ldots\ y^{(n)})=0. \qquad 1.1$$

Из теории ОДУ известно, что уравнение (1.1) эквивалентно системе n уравнений первого порядка

$$\varphi_k(x, y_1, y_1', y_2, y_2', \ldots, y_n, y_n')=0. \qquad 1.2$$

где k=1, ... , n.

Уравнение (1.1) и эквивалентная ему система (1.2) имеют бесконечное множество решений. Единственные решения выделяют с помощью дополнительных условий, которым должны удовлетворять искомые решения. В зависимости от вида таких условий рассматривают три типа задач, для которых доказано существование и единственность решений.

Первый тип – это задачи Коши, или задачи с начальными условиями. Для таких задач кроме исходного уравнения (1.1) в некоторой точке xo должны быть заданы начальные условия, т.е. значения функции y(x) и ее производных

$$y(x_0)=y_0', \quad y'(x_0)=y_{10}, \ldots, y^{(n-1)}(x_0)=y_{n-1,0}.$$

Для системы ОДУ типа (1.2) начальные условия задаются в виде

$$y_1(x_0)=y_{10}, \quad y_2(x_0)=y_{20}, \ldots, y_n(x_0)=y_{n0}. \qquad 1.3$$

Ко второму типу задач относятся так называемые граничные, или краевые задачи, в которых дополнительные условия задаются в виде функциональных соотношений между искомыми решениями. Количество условий должно совпадать с порядком n уравнения или системы. Если решение задачи определяется в интервале x є $[x_0, x_k]$, то такие условия могут быть заданы как на границах, так и внутри интервала. Минимальный порядок ОДУ, для которых может быть сформулирована граничная задача, равен двум.

Третий тип задач для ОДУ – это задачи на собственные значения. Такие задачи отличаются тем, что кроме искомых функций y(x) и их производных в уравнения входят дополнительно m неизвестных параметров λ1,λ2,…, xm, которые называются собственными значениями. Для единственности решения на интервале [x0,xk] необходимо задать m+n граничных условий. В качестве примера можно назвать задачи определения собственных частот, коэффициентов диссипации, структуры электромагнитных полей и механических напряжений в колебательных системах, задачи нахождения фазовых коэффициентов, коэффициентов затухания, распределения напряженностей полей волновых процессов и т.д.

К численному решению ОДУ приходится обращаться, когда не удается построить аналитическое решение задачи через известные функции. Хотя для некоторых задач численные методы оказываются более эффективными даже при наличии аналитических решений.

Большинство методов решения ОДУ основано на задаче Коши, алгоритмы и программы для которой рассматриваются в дальнейшем.

1. Постановка задачи

Многие процессы химической технологии описываются СДУ - начиная от кинетических исследований и заканчивая химическими технологическими процессами. В основу математических способов описания процессов положены СДУ и СЛАУ. Эти уравнения описывают материальные и тепловые балансы объектов химической технологии, а так же структуры потоков технических веществ в этих аппаратах.

Для получения, распределения технологических параметров во времени и в пространстве (в пределах объекта), необходимо произвести СДУ методом, которых дал бы высокую точность решения при минимальных затратах времени на решение, потому что ЭВМ должна работать в режиме реального времени и успевать за ходом технологического процесса. Если время на решение задачи большое, то управляющее воздействие, выработанное на ЭВМ может привести к отрицательным воздействиям. Методов решения существует очень много. В данной работе будет рассмотрен метод решения СДУ методом Рунге-Кутта 4 порядка.

Для удобства работы на ЭВМ, необходимо данную кинетическую схему преобразовать в удобный для работы на компьютере вид. Для этого необходимо

кинетическую схему процесса представить в виде уравнений. При рассмотрении кинетической схемы процесса необходимо учитывать коэффициенты скоростей реакций. Но, так как процесс протекает при изотермических условиях, коэффициенты скоростей реакций можно считать за константы скоростей химической реакции. Из приведенной ниже схемы мы можем составить ряд дифференциальных уравнений, учитывающих изотермичность процесса.
Так как коэффициенты K1,K2,K3,K4 являются константами, то можно уравнение записать в следующем виде.
Для преобразования данных дифференциальных уравнений для использования их в расчетах тепловых и кинетических схем методами Рунге-Кутты необходимо подставлять вместо производных значений концентраций, значения концентраций данных в начале процесса. Это обусловлено тем, что метод Рунге-Кутты четвертого порядка, который будет использован для расчета кинетической схемы процесса. Так как этот метод требует сведений только об одной точке и значений функции.

2. Суть метода

Разбор и рассмотрение методов, применяемых на практике для решения дифференциальных уравнений, мы начнем с их широкой категории, известной под общим названием методов Рунге-Кутта.

Методы Рунге-Кутта обладают следующими свойствами:

1. *Эти методы являются одноступенчатыми: чтобы найти y_{m+1}, нужна информация о предыдущей точке xm,ym.*

2. *Они согласуются с рядом Тейлора вплоть до членов порядка h^p, где степень p различна для различных методов и называется порядковым номером или порядком метода.*

3. *Они не требуют вычисления производных от f (x,y), а требуют вычисления самой функции.*

Рассмотрим сначала геометрическое построение и выведем некоторые формулы на основе геометрических аналогий. После этого мы подтвердим полученные результаты аналитически.

Предположим, нам известна точка xm,ym на искомой кривой. Тогда мы можем провести прямую линию с тангенсом угла наклона $y'_m=f(x_m,y_m)$, которая пройдет через точку x_m,y_m. Это построение показано на рис.1, где кривая представляет собой точное, но конечно неизвестное решение уравнения, а прямая линия L1 построена так, как это только что описано.
Тогда следующей точкой решения можно считать ту, где прямая L_1 пересечет ординату, проведенную через точку $x=x_{m+1}=x_m+h$.

Уравнение прямой L_1 выглядит так: $y=y_m+y'_m(x-x_m)$ так как $y'=f(x_m,y_m)$ и кроме того, $x_{m+1}=x_m+h$ тогда уравнение примет вид

$$y_{m+1}=y_m+h*f(x_m,y_m) \qquad 1.1$$

Ошибка при $x=x_{m+1}$ показана в виде отрезка e. Очевидно, найденное таким образом приближенное значение согласуется с разложением в ряд Тейлора вплоть до членов порядка h, так что ошибка ограничения равна $e_t=Kh^2$

Заметим, что хотя точка на графике 1 была показана на кривой, в действительности y_m является приближенным значением и не лежит точно на кривой.

Формула 1.1 описывает метод Эйлера, один из самых старых и широко известных методов численного интегрирования дифференциальных уравнений. Отметим, что метод Эйлера является одним из методов Рунге-Кутта первого порядка.

Рассмотрим исправленный метод Эйлера и модификационный метод Эйлера. В исправленном методе Эйлера мы находим средний тангенс угла наклона касательной для двух точек: x_m,y_m и $x_m+h,y_m+hy'_m$. Последняя точка есть та самая, которая в методе Эйлера обозначалась x_{m+1},y_{m+1}. Геометрический процесс нахождения точки x_{m+1},y_{m+1} можно проследить по рис.2. С помощью метода Эйлера находится точка $x_m+h,y_m+hy'_m$, лежащая на прямой L_1. В этой точке снова вычисляется тангенс, дает прямую Ĺ. Наконец, через точку x_m,y_m мы проводим прямую L, параллельную Ĺ. Точка, в которой прямая L пересечется с ординатой, восстановленной из $x=x_{m+1}=x_m+h$, и будет искомой точкой x_{m+1},y_{m+1}.

Тангенс угла наклона прямой Ĺ и прямой L равен

$$\Phi(x_m,y_m,h)=½[f(x_m,y_m)+f(x_m+h,y_m+y'_mh)] \qquad 1.2$$

где $y'_m=f(x_m,y_m)$ 1.3

Уравнение линии L при этом записывается в виде

$$y=y_m+(x-x_m)\Phi(x_m,y_m,h),$$

так что

$$y_{m+1}=y_m+h\Phi(x_m,y_m,h). \qquad 1.4$$

Соотношения 1.2, 1.3, 1.4 описывают исправленный метод Эйлера.

Чтобы выяснить, насколько хорошо этот метод согласуется с разложением в ряд Тейлора, вспомним, что разложение в ряд функции f(x,y) можно записать следующим образом:

$$f(x,y)=f(x_m,y_m)+(x-x_m)\partial f/\partial x+(y-y_m)\partial f/\partial x+\ldots \qquad 1.5$$

где частные производные вычисляются при $x=x_m$ и $y=y_m$.

Подставляя в формулу 1.5 $x=x_m+h$ и $y=y_m+hy'_m$ и используя выражение 1.3 для y'_m, получаем

$$f(x_m+h,y_m+hy'_m)=f+hf_x+hff_y+O(h^2),$$

где снова функция f и ее производные вычисляются в точке x_m,y_m. Подставляя результат в 1.2 и производя необходимые преобразования, получаем

$$\Phi(x_m,y_m,h)=f+h/2(f_x+ff_y)+O(h^2).$$

Подставим полученное выражение в 1.4 и сравним с рядом Тейлора

$$y_{m+1}=y_m+hf+h^2/2(f_x+ff_y)+O(h^3).$$

Как видим, исправленный метод Эйлера согласуется с разложением в ряд Тейлора вплоть до членов степени h^2, являясь, таким образом, методом Рунге-Кутты второго порядка.

Рассмотрим модификационный метод Эйлера. Рассмотрим первоначальное построение сделано так же, как и на рис.2. Но на этот раз мы берем точку, лежащую на пересечении этой прямой и ординатой $x=x+h/2$. На рисунке эта точка образована через P, а ее ордината равна $y=y_m+(h/2)y'_m$. Вычислим тангенс угла наклона касательной в этой точке

$$\Phi(x_m,y_m,h)=f+(x_m+h/2,y_m+h/2*y'_m), \qquad 1.6$$

где $y'_m=f(x_m,y_m)$ 1.7

Прямая с таким наклоном, проходящая через P, обозначена через L^*. Вслед за тем, мы проводим через точку xm,ym прямую параллельную L^*, и обозначаем ее через L_0. Пересечение этой прямой с ординатой $x=x_m+h$ и даст искомую точку x_{m+1},y_{m+1}. Уравнение прямой можно записать в виде $y=y_m+(x-x_m)\Phi(x_m,y_m,h)$,
где Ф задается формулой 1.6. Поэтому

$$y_{m+1}=y_m+h\Phi(x_m,y_m,h) \qquad 1.8$$

Соотношения 1.6, 1.7, 1.8 описывают так называемый модификационный метод Эйлера и является еще одним методом Рунге-Кутта второго порядка. Обобщим оба метода. Заметим, что оба метода описываются формулами вида

$$y_{m+1}=y_m+h\Phi(x_m,y_m,h) \qquad 1.9$$

и в обоих случаях Ф имеет вид

$$\Phi(x_m,y_m,h)=a_1f(x_m,y_m)+a_2f(x_m+b_1h,y_m+b_2hy'_m), \quad 1.10$$

где $y'_m=f(x_m,y_m)$ 1.11

В частности, для исправленного метода Эйлера

$a_1=a_2=1/2;$
$b_1=b_2=1.$

В то время как для модификационного метода Эйлера

$a_1=0,\ a_2=1,$
$b_1=b_2=1/2.$

Формулы 1.9, 1.10, 1.11 описывают некоторый метод типа Рунге-Кутты. Посмотрим, какого порядка метод можно рассчитывать получить в лучшем случае и каковы допустимые значения параметров a_1, a_2, b_1 и b_2 .

Чтобы получить соответствие ряду Тейлора вплоть до членов степени h, в общем случае достаточно одного параметра. Чтобы получить согласование вплоть до членов степени h^2, потребуется еще два параметра, так как необходимо учитывать члены h^2f_x и h^2ff_y. Так как у нас имеется всего четыре параметра, три из которых потребуются для создания согласования с рядом Тейлора вплоть до членов порядка h^2, то самое лучшее, на что здесь можно рассчитывать - это метод второго порядка.

В разложении f(x,y) в ряд 1.5 в окрестности точки x_m,y_m положим $x=x_m+b_1h$,
$y=y_m+b_2hf$.

Тогда $f(x_m+b_1h,y_m+b_2hf)=f+b_1hf_x+b_2hff_y+O(h^2)$, где функция и производные в правой части равенства вычислены в точке x_m,y_m.

Тогда 1.9 можно переписать в виде $y_{m+1}=y_m+h[a_1f+a_2f+h(a_2b_1f_x+a_2b_2ff_y)]+O(h^3)$.

Сравнив эту формулу с разложением в ряд Тейлора, можно переписать в виде

$$y_{m+1}=y_m+h[a_1f+a_2f+h(a_2b_1f_x+a_2b_2ff_y)]+O(h^3).$$

Если потребовать совпадения членов hf, то $a_1+a_2=1$.
Сравнивая члены, содержащие h^2f_x, получаем $a_2b_1=1/2$.
Сравнивая члены, содержащие h^2ff_y, получаем $a_2b_2=1/2$.

Так как мы пришли к трем уравнениям для определения четырех неизвестных, то одно из этих неизвестных можно задать произвольно, исключая, может быть, нуль, в зависимости от того, какой параметр взять в качестве произвольного.

Положим, например, $a_2=\omega\neq0$. тогда $a_1=1-\omega$, $b_1=b_2=1/2\omega$ и соотношения 1.9, 1.10, 1.11 сведутся к

$$y_{m+1}=y_m+h[(1-\omega)f(x_m,y_m)+\omega f(x_m+h/2\omega,y_m+h/2\omega f(x_m,y_m))]+O(h^3) \quad 1.12$$

Это наиболее общая форма записи метода Рунге-Кутта второго порядка. При ω=1/2 мы получаем исправленный метод Эйлера, при ω=1 получаем модификационный метод Эйлера. Для всех ω, отличных от нуля, ошибка ограничения равна

$$e_t=kh^3 \qquad 1.13$$

Методы Рунге-Кутта третьего и четвертого порядков можно вывести совершенно аналогично тому, как это делалось при выводе методов первого и второго порядков. Мы не будем воспроизводить выкладки, а ограничимся тем, что приведем формулы, описывающие метод четвертого порядка, один из самых употребляемых методов интегрирования дифференциальных уравнений. Этот классический метод Рунге-Кутта описывается системой следующих пяти соотношений

$$y_{m+1}=y_m+h/6(R_1+2R_2+2R_3+R_4) \qquad 1.14$$

где $R_1=f(x_m,y_m)$, 1.15

$R_2=f(x_m+h/2,y_m+hR_1/2)$, 1.16

$R_3=f(x_m+h/2,y_m+hR_2/2)$, 1.17

$R_4=f(x_m+h/2,y_m+hR_3/2)$. 1.18

Ошибка ограничения для этого метода равна $e_t=kh^5$
так что формулы 1.14-1.18 описывают метод четвертого порядка. Заметим, что при использовании этого метода функцию необходимо вычислять четыре раза.

3. Выбор метода реализации программы

Исходя из вышеизложенного, для решения систем дифференциальных уравнений мы выбираем наиболее точный метод решения – метод Рунге-Кутта 4 порядка, один из самых употребляемых методов интегрирования дифференциальных уравнений.

- - *этот метод является одноступенчатым и одношаговым*
- - *требует информацию только об одной точке*
- - *имеет небольшую погрешность*
- - *значение функции рассчитывается при каждом шаге*

4. Блок-схема программмы

НАЧАЛО

INIT

RUN

КОНЕЦ

Основная программа

Процедура **INIT**

5. Программа

```
PROGRAM smith_04;
USES crt;
  VAR
  i,n:integer;
  sum,k1,k2,k3,k4,p,dp,eps,Xn,Xk,X,dX:real;
  rSR,C,dC,r1,r2,r3,r4,cPR:array[1..3] of real;
  f1,f2:text;

  PROCEDURE Difur;
 BEGIN
        dC[1]:=C[3]*k2+C[2]*k4-C[1]*k1-C[1]*k3; {dcA}
        dC[2]:=C[1]*k3-C[2]*k4;                 {dcB}
        dC[3]:=C[1]*k1-C[3]*k2;                 {dcC}
 END;

  PROCEDURE RK_4;
 BEGIN
      Difur;
    FOR i:=1 TO n DO BEGIN
                    r1[i]:=dC[i];
                    C[i]:=cPR[i]+r1[i]*(dX/2);
                     END;
      Difur;
    FOR i:=1 TO n DO BEGIN
                    r2[i]:=dC[i];
                    C[i]:=cPr[i]+r2[i]*(dX/2);
                     END;
      Difur;
    FOR i:=1 TO n DO BEGIN
                    r3[i]:=dC[i];
                    C[i]:=cPR[I]+r3[i]*dX;
                     END;
      Difur;
    FOR i:=1 TO n DO r4[i]:=dC[i];
```

```
    FOR i:=1 TO n DO rSR[i]:=((r1[i]+r2[i])*(r2[i]+r3[i])*(r3[i]+r4[i]))/6;
 END;

  PROCEDURE STROKA;
 BEGIN
WRITE(f2,'|',x:4:1,'|',c[1]:7:3,'|',c[2]:7:3,'|',c[3]:7:3,'|');
WRITE(f2,sum:3:0,'|',dc[1]:7:3,'|',dc[2]:7:3,'|',dc[3]:7:3,'|');
WRITELN(f2);
 END;

  PROCEDURE RUN;
 BEGIN
  WRITE('Step 3: Calculating data and writting results to file : out.rez');
  X:=Xn;
  dX:=0.05;
  REPEAT
       IF (ABS(x-p)<eps)  THEN BEGIN
            Difur;
            sum:=C[1]+C[2]+C[3];
            STROKA;
              p:=p+dp;         END;
       FOR i:=1 TO n DO Cpr[i]:=C[i];
         RK_4;
       X:=X+dX;
  UNTIL(X>Xk);
  WRITELN('       - done.');
 END;

  PROCEDURE INIT;
 BEGIN
  ClrScr;
  WRITELN('Smith-04: v1.0 (c) 1998 by Mike Smith smith01@home.bar.ru ');
  WRITELN;
  WRITELN;
  WRITE('Step 1: Read data from file                              : in.dat');
  ASSIGN(f1,'in.dat');
  RESET(f1);
  READLN(f1,C[1],C[2],C[3]);
  READLN(f1,k1,k2,k3,k4);
  READLN(f1,Xn,Xk,dp,n,eps,p);
  WRITELN('        - done.');
  ASSIGN(f2,'out.rez');
  REWRITE(f2);
  WRITE('Step 2: Write header to file                             : out.rez');
  WRITELN(f2,'============================================================');
  WRITELN(f2,'| t,c| Ca,% | Cb,%| Cc,%  |  SUM |  dCa  |  dCb  |  dCc  |');
  WRITELN(f2,'============================================================');
  WRITELN('       - done.');
 END;

  PROCEDURE DONE;
 BEGIN
  WRITELN('Step 4: Close all files and exiting...');
  CLOSE(f1);
  WRITELN(f2,'============================================================');
  CLOSE(f2);
  WRITELN;
 END;

BEGIN

      INIT;
      RUN;
      DONE;
```

```
END.
```

6. Обсуждение результатов расчета.

В результате расчета кинетической схемы процесса на языке Паскаль методом Рунге-Кутты, были получены результаты зависимости изменения концентрации реагирующих веществ во времени. Исходя из полученных результатов, можно сделать вывод, что расчет произведен **верно,** так как, исходя из полученных значений скоростей реакций можно сделать вывод, что соблюдается баланс скоростей химической реакции.

Рассмотрим процесс подробнее. Вещество А на протяжении всего процесса расходуется на образование веществ В и С. Концентрации вещества А в начальный момент времени расходуется быстрее, чем концентрации его же в конце процесса. Это обусловлено тем, что скорость химической реакции зависит от концентрации реагирующего вещества. Производная имеет знак «минус». Это говорит о том, что вещество расходуется. Следовательно, чем выше концентрация вещества, вступающего в процесс, тем выше скорость его реагирования с другими веществами. Вещества В и С образуются пропорционально, так как, исходя из кинетической схемы процесса и значений констант скоростей химической реакции, видно, что образование этих веществ и расходование этих веществ, ***одинаково***. Производная имеет знак ***«плюс»***. Это говорит о том, что вещество ***образуется.***

Это видно также и по результатам расчета, на протяжении всего времени исследования процесса концентрации и скорости веществ В и С одинаковы. В этом можно убедиться по виду графической зависимости концентрации веществ В и С от времени.

Можно сказать, что *процесс протекает в сторону увеличения концентрации веществ* В и С и *уменьшения концентрации вещества* А. Процесс будет протекать до момента установления равновесия, но в данном случае равновесие не установлено, так как вещества продолжают расходоваться и образовываться. На протяжении всего процесса ни одно из образующихся веществ не поменяло знак производной. Это говорит о том, что *процесс протекает в одну сторону.*

7. Инструкция к программме

Итак, программа состоит из 3 основных процедур:

1) 1) **Init** - процедура инициалиации, включающую в себя ввод данных;
2) 2) **Run** - процедура вычисления и обработки результатов, включает в себя вызов двух вспомогательных процедур **Difur**, **RK-4**, **Stroka**, первая из которых отвечает за вычисление, а последняя - за вывод результатов в файл в табличном виде;
3) 3) **Done -** процедура подготовки к выходу из программы;

и трех вспомогательных:

a) a) **Difur** - процедура вычисления производных (изменение концентрации веществ за единицу времени)

b) b) **RK-4** - используя значения производных, вычисленных процедурой Difur, вычисляет последущие концентрации веществ методом Рунге-Кутта

c) c) **Stroka** - процедура вывода результата в файл в табличном виде

Рассмотрим все эти процедуры поподробнее:

Процедура INIT:

В данной процедуре задействованы операторы ввода/вывода **Wite/Read**, оператор модуля Crt - **CrlScr** - очистка экрана, файлового ввода/вывода - **Reset/Rewrite** – открытие файла для чтения и создание нового файла, соответственно. Данная процедура выполняет функцию инициализации программных данных, считывание данных из файла **in.dat**, создание, открытие на запись файла **out.rez** и запись в него шапки таблицы результатов.

Процедура RUN:

```
 PROCEDURE RUN;
 BEGIN
X:=Xn;
 dX:=0.05;
 REPEAT
      IF (ABS(x-p)<eps)  THEN
BEGIN
           Difur;
           sum:=C[1]+C[2]+C[3];
           STROKA;
           p:=p+dp;       END;
      FOR i:=1 TO n DO Cpr[i]:=C[i];
      RK_4;
      X:=X+dX;
 UNTIL (X>Xk);
```

В данной процедуре задействованы операторы цикла **Repeat/Until**, и **For/Do** с операторами условного перехода **IF/Then**. В зависимости от условий вызываются процедуры Difur и Strok. В теле цикла постоянно вызывается процедура RK-4 вызывающая 4 раза функцию **Difur**.

Процедура DONE:

```
                DONE;
 BEGIN
CLOSE(f1);

WRITELN(f2,'_____________________
  CLOSE(f2);
```

В данной процедуре задействованы оператор работы с файлами Close, который закрывает файлы с исходными данными и файл с полученными в резуультате вычислений результатами.

Процедура DIFUR:

```
PROCEDURE Difur;
 BEGIN
      dC[1]:=C[3]*k2+C[2]*k4-C[1]*k1-
C[1]*k3;
      dC[2]:=C[1]*k3-C[2]*k4;
      dC[3]:=C[1]*k1-C[3]*k2;
```

Данная процедура вычисляет производную изменения концентрации везества за единицу времени.

Процедура STROKA:

```
PROCEDURE STROKA;
 BEGIN
WRITE(f2,'|',x:4:1,'|',c[1]:7:3,'|',c[2]:7:3,'|',c[3
WRITE(f2,sum:3:0,'|',dc[1]:7:3,'|',dc[2]:7:3,'|',
WRITELN(f2);
 END;
```

Данная процедура с помощью оператора вывода **WRITE** записывает результаты в файл, соответствующий файловой переменной F2, назначенной коммандой **ASSIGN** в процедуре **INIT**

Процедура RK-4:

```
 PROCEDURE RK_4;
BEGIN
    Difur;
  FOR i:=1 TO n DO BEGIN
                  r1[i]:=dC[i];
                  C[i]:=cPR[i]+r1[i]*(dX/2);
               END;
    Difur;
  FOR i:=1 TO n DO BEGIN
                  r2[i]:=dC[i];
                  C[i]:=cPr[i]+r2[i]*(dX/2);
               END;
    Difur;
  FOR i:=1 TO n DO BEGIN
                  r3[i]:=dC[i];
```

Данная процедура, используя вызовы процедур **Difur**, а также циклы операторы цикла **FOR**, вычисляет последуущие концентрации веществ по предидущим точкам.

Программа представляет собой 2 файла – файл с исходным текстом на языке Паскаль **smith.pas** и исполняемый модуль **smith.exe** скомпилированный компилятором TNT Pascal 3.25 фирмы Layer`s Ins.

Исполняемый модуль программы предназначен для запуска в операционных системах: MS Dos, Windows95, Windows NT, OS/2, а также в X-windows под Linux (при наличии эмулятора)

Для нормальной работы программе необходимо 640 кb «нижней» памяти и 20 kb дискового пространства. Согласитесь – требования минимальные, учитывая то, что сама программа абсолютно не требовательна к процессору.

В процессе работы программа считывает данные из файла **in.dat** и записывает результаты работы в файл **out.rez** в табличном виде. Исходный файл программма открывает стандартными средствами ОС, не проверяя его наличие перед работой, поэтому, если данный файл не будет доступен в каталоге, в котором расположена программа, компилятор выдаст сообщение об ошибке. Если Вы после запуска программы увидели что-то типа «Runtime error 202 at 0000:0A86» - это всего лишь значит, что программа не смогла найти файл с исходными данными в текущем каталоге. Если Вы забыли поместить его туда, ***скопируйте*** этот файл в каталог с программой и ***запустите*** исполняемый модуль еще раз. Если данный файл у Вас ***отсутствует***, Вам прийдется сделать его самому.
Для этого в любом текстовом редакторе наберите 3 выделенных строчки и сохраните созданный файл с именем **in.dat**

100 0 0
0.2 0.1 0.2 0.1
0 10 0.5 3 0.05 0

Создав файл и скопировав его к исполняемому модулю программы, запустите исполняемый модуль еще раз.

В процессе работы программа будет выдавать сообщения об успешном окончании каждого блока. Если все прошло нормально, то на экране своего компьютера Вы увидите следуще сообщения:

```
Step 1: Read data from file                          : in.dat     - done.
Step 2: Write header to file                         : out.rez    - done.
Step 3: Calculating data and writting results to file : out.rez    - done.
Step 4: Close all files and exiting...
```

Первый шаг (step1) сообщает, что данные из файла **in.dat** были успешно прочитаны

Второй – о том что программа успешно создала выходной файл **out.rez** и записала в него шапку таблицы с данными

В третьем сообщении сказано, что данные успешно посчитаны и записаны в выходной файл **out.rez**

Четвертое сообщение сообщает об окончании вычислений и завершении программы.

После того, как программа отработает, Вы сможете познакомится с результатами, которые были вычислены и помещены в файл результатов **out.rez.** Просмотрев его любой программой просмотра текстовых файлов или вывев его на печать, вы получите таблицу с результатами.

8. Заключение.

В результате выполнения расчета получена зависимость изменения концентрации вещества во времени. Из расчета следует, что на протяжении всего процесса вещество А расходовалось на образование В и С. Процесс не достиг конечного состояния (не достиг равновесия) Максимум концентрации вещества наблюдался при следующих значениях времени:

при *начальном значении* времени max соответствовал веществу А;

при *значении времени, равном 10 часам*, max соответствовал веществам В и С, однако, это не является максимумом концентрации веществ в процессе вообще, так как вещества В и С продолжают образовываться;

В ходе выполнения работы был произведен расчет системы дифференциальных уравнений методом Рунге-Кутты четвертого порядка, произведен расчет кинетической схемы процесса при изотермических условиях при данных значениях концентраций и констант скоростей. Расчет произведен с малой величиной погрешности.

Литература.

1. Мудров А.Е.Численные методы для ПЭВМ на языках Паскаль, Фортран и Бейсик. МП “Раско”, Томск, 1991 г.