**Министерство образования и науки Российской Федерации**

**Новосибирский Государственный Технический Университет**

## Кафедра экономической информатики

## РАСЧЕТНО-ГРАФИЧЕСКАЯ РАБОТА

**по предмету «ЧИСЛЕННЫЕ МЕТОДЫ» на тему:**

Исследование метода дифференцирования по параметру для решения нелинейных САУ

Новосибирск

2004

# **СОДЕРЖАНИЕ**

ВВЕДЕНИЕ

1. ПОСТАНОВКА ЗАДАЧИ (МАТЕМАТИЧЕСКОЕ ОПИСАНИЕ МЕТОДОВ)

* 1. Обобщенный алгоритм решения нелинейных САУ
  2. Метод дифференцирования по параметру
  3. Явные методы Рунге-Кутта
  4. Метод Ньютона
  5. Дискретный метод Ньютона

2. ОПИСАНИЕ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

2.1 Общие сведения

2.2 Функциональное назначение

2.3 Описание логической структуры

2.4 Используемые технические средства

2.5 Вызов и загрузка

2.6 Входные данные

2.7 Выходные данные

3. ОПИСАНИЕ ТЕСТОВЫХ ЗАДАЧ

4. АНАЛИЗ РЕЗУЛЬТАТОВ. ВЫВОДЫ

ЗАКЛЮЧЕНИЕ

##### СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

# **ВВЕДЕНИЕ**

Целью данной работы является исследование метода дифференцирования по параметру для решения нелинейных САУ. Для реализации данного исследования используется система MatLab Version 5.1. Поставленными в начале работы задачами являются разработка программного обеспечения для решения нелинейных САУ методом дифференцирования по параметру, а также исследование влияние метода интегрирования на точность получаемого решения. Также в работе должны быть представлены графики переходных процессов для трех методов с различными начальными значениями вектора X0.

# **1. ПОСТАНОВКА ЗАДАЧИ (МАТЕМАТИЧЕСКОЕ ОПИСАНИЕ МЕТОДОВ)**

Экономический объект при определенных условиях описывается системой нелинейных алгебраических уравнений вида

*0=F(X\*,U\*).*

Если при этом входной сигнал *U\** известен, то для определения соответствующего значения *Х\** необходимо решить систему нелинейных АУ вида

*F(X)=0*.

Точно решить эту систему удается редко, поэтому решение находим в два этапа:

* определение приближенного значения;
* уточнение приближенного значения с помощью некоторого итерационного метода до некоторой заданной степени точности.

Часто значение *Х0* бывает известно из каких-либо практических соображений, связанных со знанием ЭО. Для малых *n* значения вектора *Х0* можно определить графически. Если метод решения обладает глобальной сходимостью, то *Х0* может быть любым. Сосредоточим свое внимание на втором этапе (уточнение приближенного значения).

Численный метод, в котором производится последовательное, шаг за шагом, уточнение первоначального грубого приближения *Х0* называется итерационным. Каждый шаг в этом методе называется итерацией. Если с каждым шагом получаемое значение *Х* все ближе к точному *Х\*,* то метод сходится.

**1.1 Обобщенный алгоритм решения нелинейных САУ**

дифференцирование алгебраическое нелинейное уравнение

Большинство известных итерационных методов решения системы F(X)=0 можно записать одной общей формулой

*Хm+1=G(Хm, Хm-1,…,Хm-p+1),*

где *G* – вектор-функция размерности *n* , которая определяется способом построения итерационного процесса; *р* – количество предыдущих значений *Х*, используемых в данном итерационном процессе.

Если в итерационном процессе используется только одна предыдущая точка *(р=1),* то

*Хm+1=G(Хm)*

Метод дифференцирования по параметру относится к этому случаю.

Основные характеристики итерационных методов:

1. Сходимость итераций. Итерации сходятся, если

*lim Хm =Х\* при m→∞*

Вектор-функция *G(Х)* называется изображением итерационного процесса. Спектральным радиусом квадратной матрицы *А ρ(А)* называется максимальный из модулей ее собственных значений. Предположим, что функция *G(Х)* определена и непрерывна вместе со своей первой производной

*G/Х = δG/δХ*

Теорема сходимости.

Если спектральный радиус матрицы *G/Х ρ(G/Х)<1* и если векторы *Хm+1=G(Хm)* не выходят за области определения вектор - функций *F* и *G*, то процесс итераций *Хm+1=G(Хm)* сходится. При этом предельный вектор *Х\* = limХm*при *m→∞* является единственной точкой притяжения итераций.

Эта теорема справедлива для любого начального приближения *Х0* и поэтому относится к теоремам о глобальной сходимости.

Трудность применения теоремы о глобальной сходимости состоит в том, что надо определять величины *ri, i=1,n* на каждом m-шаге итерационного процесса. Это практически невозможно.

Поэтому нашли применение теоремы локальной сходимости. При этом предполагают, что точка *Х0* лежит близко к *Х\**. Спектральный радиус матрицы *G/Х* вычисляется только в точке *Х0: ρ(G/(Х0))<1.*

1. Выбор величины начального приближения.

Выбор величины *Х0* зависит от вида сходимости метода. Если метод имеет локальную сходимость, то *Х0* должно быть близко к *Х\**, если глобальную, то *Х0* - любой. Часто *Х0 = 0*.

1. Скорость сходимости итераций.

Скорость сходимости итераций оценивается по скорости уменьшения величины ошибки

*Em = |Хm-Х\*|*

Если условия сходимости выполняются, то часто скорость можно оценить формулой

*||Em+1|| = с ||Em ||k* , где *k*- целое число*; с*- константа.

Если *k=1*, то итерационный метод имеет линейную сходимость. При этом, если *с≈1*, то сходимость медленная (метод простой итерации).

Если *k=2*, то метод обладает квадратичной скоростью сходимости. Так как *||Em ||<1*, то *||Em||2* будет величиной второго порядка малости и поэтому скорость велика (метод Ньютона).

1. Критерий окончания итераций.

Расчеты по формуле *Хm+1=G(Хm)* не могут длиться бесконечно долго. Очевидно, что критерием окончания итерационного процесса могла бы служить величина *Em*, но нам неизвестно значение *Х\**. В связи с этим величину *Em* можно оценить косвенно.

Способ 1. Остановить процесс вычислений, когда *||F(Хm)|| ≤ εдоп* заданной допустимой погрешности. Заметим, что *lim ||F(Хm)||=0* при *m→∞.*

Способ 2. Остановить процесс вычислений, когда *||∆Хm || < εдоп*. *∆Хm* *= Хm+1 – Xm.* Чем ближе к *Х\*,* тем меньше величина *||∆Хm ||.*

Выбор способа зависит от характера поведения функций *fi(Х)* вблизи решения.

*Xi*

Eim

*fi fi*

Eim

εдоп

Хi\*Xim Xi\* Xim Xi

εдоп

Рис. 1.1 Рис. 1.2

Из рис. 1.1 видно, что если заканчивать итерационный процесс по величине *||F||,* то при этом можно оказаться довольно далеко от *Хi\** по *Хim* . На рис. 1.2 – наоборот, итерационный процесс заканчивается при малых значениях *||∆Хm ||,* что приводит к большим ошибкам по *||Fm ||.*

Способ 3. Чтобы избежать недостатков первых двух способов, контролируют обе нормы, а итерационный процесс заканчивают при том значении *m*, при котором

*max{||∆Хm ||, ||F(Хm)|| }< εдоп*

Следует заметить, что при плохой обусловленности матрицы *G/Х* вблизи *Х\** возможны колебания значений норм. Тогда нужно применять специальные методы уменьшения этих колебаний.

## 1.2 Метод дифференцирования по параметру

Здесь алгебраическая задача сводится к задаче интегрирования системы ОДУ, которая формируется следующим образом. Рассмотрим функцию *Н(Х, t)* как функцию параметра *tЄ[0,1],* т.е. обозначим *Ф(t)=Н(Х, t).* Пусть *Ф(t)* непрерывно дифференцируема по *t* на интервале *[0,1],* тогда

*Ф/(t)=( δH/δХ)·X/(t)+ δH/δt.*

Функция *H(Х, t)* удовлетворяет тем же требованиям, что и в методе продолжения решения по параметру. Следовательно, функция *Х(t)* удовлетворяет уравнению *H(Х, t)=0*, откуда получаем *Ф/(t)=0*. Значит, из последнего соотношения имеем систему ОДУ вида

*X/(t)= - ( δH/δХ)-1 ·δH/δt.*

Система ОДУ решается при начальных условиях *t=0, X(0)=X0*. Время меняется от 0 до 1. При *t=1* получим решение системы *F(X)=0* - вектор *Х\** с точностью, зависящей от точности метода интегрирования системы. Если *Н(Х,t)= F(X) + (t - 1) · F(X0)* , то получим систему ОДУ

*X(t)=-J-1(Х(t))•F(X0),*

которая является нелинейной по *Х*.

Данная система решается явными методами Рунге – Кутта, а затем полученное приближенное значение уточняется дискретным методом Ньютона за несколько итераций.

## 1.3 Явные методы Рунге-Кутта

Свойства методов Рунге-Кутта:

1. Методы являются одношаговыми; чтобы найти *Xm+1 ,* нужна информация только о предыдущей точке *Xm, tт*.

2*.* Они согласуются с рядом Тейлора вплоть до членов порядка *hs*, где степень S различна для различных методов и называется порядком метода.

3. Методы не требуют вычисления производных функций *fi (X,t), i=1,n,* а только самой функции в нескольких точкахна шаге *hm .*

Методом Рунге-Кутта 1-го порядка является явный метод Эйлера:

Х/ =F(Х, t) ;

*Xm+1 = Xm + hm ·F (Xm, tm)*

Ошибка аппроксимации *εα ~ h2т.* Область абсолютной устойчивости – круг радиусом, равным 1 и центром в точке (0, -1) – см. рис. 1.3, кривая 1; область относительной устойчивости – вся правая полуплоскость.

Рассмотрим методы Рунге-Кутта 2-го и 4-го порядка, которые также используются довольно часто.

Алгоритм метода Рунге-Кутта 2-го порядка состоит в следующем:

*Xm+1 = Xm + (hm/2)·(K1 + K2),* где

*K1= F(Xm, tm), K2= F(Xm + hm·K1, tm + hm).*

Ошибка аппроксимации *εα=kh3т.* Область абсолютной устойчивости показана на рис. 1.3 (кривая 2). Область относительной устойчивости - вся правая полуплоскость.

### Алгоритм метода Рунге-Кутта 4-го порядка:

*Xm+1 = Xm + (hm/6) · (K1 + 2K2 + 2K3 + K4),* где

*K1= F(Xm, tm), K2= F(Xm + (hm/2)·K1, tm + hm/2);*

*K3= F(Xm + (hm/2) ·K2, tm + hm/2);*

*K4= F(Xm + h·K3, tm + hm);*

Ошибка аппроксимации εα=*kh5т.* Область абсолютной устойчивости показана на рис. 1.3 (кривая 3). Область относительной устойчивости - вся правая полуплоскость.

Re(h)

Область

относительной устойчивости

Im(h)

Область

абсолютной устойчивости

1

2

3

Рис. 1.3

## 1.4 Метод Ньютона

Итерационная формула дискретного метода Ньютона имеет вид

*Xm+1=Xm – J-1(Xm) ·F(Xm)* ,

где *J (Хm) = F/X / X=Xm*– матрица Якоби.

Характеристики метода.

1. Сходимость.

Условие сходимости метода

*ρ(G’Х(Х))= ρ(I – (J-1(Х) ·F(Х))/Х)<1.*

Имеем *ρ(I – J-1(Х\*) · F/Х(Х\*))=0*; это означает, что метод обладает локальной сходимостью, т.е. сходится только вблизи точного решения. Поэтому при реализации метода дифференцирования по параметру вначале нужно получить приближенное значение. Чем ближе к *Х\**, тем быстрее сходится метод.

1. Выбор начального приближения *Х0*.

Поскольку метод сходится только вблизи точного решения, значит, начальное приближение также нужно выбирать вблизи *Х\*.* Насколько близко, зависит от скорости изменения функции *F(Х)* вблизи *Х\**: чем выше скорость, тем меньше область, где соблюдается условие сходимости.

1. Скорость сходимости.

Она определяется соотношением

*||Em+1||=Cm ||Em||1+ Р ,*

*0<р<1*. При *Х→Х\** величина *р→1*. Это значит, что вблизи точного решения скорость сходимости близка к квадратичной.

1. Критерий окончания итераций.

Таким критерием может быть любое соотношение из описанных выше в обобщенном алгоритме решения нелинейных САУ.

**1.5 Дискретный метод Ньютона**

Трудности практического применения метода Ньютона состоят в следующем:

1. Необходимость определения матрицы *J = F/X.*

При этом существует два подхода:

* аналитический способ. Здесь метод Ньютона особенно эффективен. Однако точные формулы могут быть слишком громоздкими, что повышает вероятность ошибки. Кроме того функции *F(X)* могут быть заданы таблично;
* конечно-разностная аппроксимация. При этом используется формула:

*δfi/δxj = (fi(x1, …, xj + ∆xj, …, xn) - fi(x1, …, xj - ∆xj, …, xn)) / 2δxj .*

В этом случае имеем дискретный метод Ньютона, который уже не обладает квадратичной сходимостью. Скорость сходимости можно увеличить, уменьшая *∆xj* по мере приближения к *X\*.*

1. Вычисление матрицы *J-1* на каждом шаге требует значительных вычислительных затрат. Поэтому часто вместо этого решают систему линейных АУ, которая формируется следующим образом. Очевидно, что *∆Xm = Xm+1 - Xm*. Тогда после алгебраических преобразований алгоритм *Xm+1=Xm – J-1(Xm) ·F(Xm)* примет вид:

*J(Xm) · ∆Xm = -F(Xm).*

На каждом *m*-м шаге матрицы *J(Xm)* и *F(Xm)* известны. Необходимо найти *∆Xm*, как решение системы линейных АУ *J(Xm) · ∆Xm = -F(Xm)*. Тогда

*Xm+1 = Xm + ∆Xm.*

Решение системы *J(Xm) · ∆Xm = -F(Xm)* – наиболее трудоемкий этап, который определяет вычислительную эффективность каждой итерации.

**2. ОПИСАНИЕ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ**

## 2.1 Общие сведения

Наименования файлов, из которых состоит пакет: main.m, rk1.m, rk2, rk4.m, funf.m, dmn.m, dif.m .

Головная программа – файл main.m, остальные – внешние функции данной программы.

Программное обеспечение: ОС класса Windows.

Язык, на котором написана программа: MatLab (Version 5.1).

## 2.2 Функциональное назначение

Программа разработана для решения нелинейных систем алгебраических уравнений методом дифференцирования по параметру. Решение проводится с использованием трех различных методов Рунге – Кутта первого, второго и четвертого порядка. Решение уточняется с помощью дискретного метода Ньютона.

## 2.3 Описание логической структуры

Файл описания системы - funf.m. С помощью файла dif.m формируется система ОДУ, затем система интегрируется с помощью явных методов Рунге-Кутта (rk1.m, rk2, rk4.m), и полученное решение уточняется дискретным методом Ньютона (dmn.m). В файле main.m можно изменять шаг интегрирования, начальное решение системы, допустимую ошибку при уточнении, начальный и конечный моменты времени…

Головная программа

В головной программе задаются начальные значения необходимых параметров, а также производится вызов основных функций, необходимых для реализации метода дифференцирования по параметру. В течение одного цикла работы программы вызываются последовательно функции rk1.m, rk2.m, rk4.m, вычисляющие приближенные значения по методам Рунге – Кутта 1-го, 2-го и 4-го порядков. После каждого из них производится вызов функции dmn.m, вычисляющей уточненное значение по дискретному методу Ньютона, строятся графики и выводятся значения полученных решений системы и ошибок интегрирования; значения уточненных решений системы и ошибок интегрирования, полученных в процессе итераций. В конце работы каждого цикла выводится число шагов, за которое было получено приближенное решение и число итераций, за которое было получено уточненное решение.

## Текст программы (файл Main.m):

## t0=0;

## tfinal=1;

## y0=[2 0];

## h=0.1;

## trace=1;

## disp('Метод Рунге - Кутта 1го порядка');pause;

## [tout,yout,x]=rk1('dif',t0,tfinal,y0,h,trace);

subplot(2,1,1);

## plot(tout,yout(:,1));

grid;

title(sprintf('Метод Рунге - Кутта 1го порядка'));

ylabel('y(1)'); xlabel('t');

subplot(2,1,2);

plot(tout,yout(:,2));

grid;

ylabel('y(2)'); xlabel('t');

pause;

## x0=x';

## dx=[0.01;0.01];

## Ed=0.000001;

## [x,dx,m] = dmn('funf',x0,dx,Ed);

subplot(111);

## plot(m,x(:,1),m,x(:,2));

## grid;

title(sprintf(‘Уточнение решения системы’));

ylabel('x(m)'); xlabel('m');

pause;

plot(m,dx(:,1),m,dx(:,2));

## grid;

ylabel('dx(m)'); xlabel('m');

pause;

out=[m,x,dx]

disp('Метод Рунге - Кутта 2го порядка');pause;

[tout,yout,x]=rk2('dif',t0,tfinal,y0,h,trace);

subplot(2,1,1);

## plot(tout,yout(:,1));

grid;

title(sprintf('Метод Рунге-Кутта 2го порядка'));

ylabel('y(1)'); xlabel('t');

subplot(2,1,2);

plot(tout,yout(:,2));

grid;

ylabel('y(2)'); xlabel('t');

pause;

x0=x';

dx=[0.01;0.01];

[x,dx,m] = dmn('funf',x0,dx,Ed);

subplot(111);

## plot(m,x(:,1),m,x(:,2));

## grid;

title(sprintf(‘Уточнение решения системы’));

ylabel('x(m)'); xlabel('m');

pause;

plot(m,dx(:,1),m,dx(:,2));

## grid;

ylabel('dx(m)'); xlabel('m');

pause;

out=[m,x,dx]

disp('Метод Рунге - Кутта 4го порядка');pause;

[tout,yout,x]=rk4('dif',t0,tfinal,y0,h,trace);

subplot(2,1,1);

## plot(tout,yout(:,1));

grid;

title(sprintf('Метод Рунге-Кутта 4го порядка'));

ylabel('y(1)'); xlabel('t');

subplot(2,1,2);

plot(tout,yout(:,2));

grid;

ylabel('y(2)'); xlabel('t');

pause;

x0=x';

dx=[0.01;0.01];

[x,dx,m] = dmn('funf',x0,dx,Ed);

subplot(111);

## plot(m,x(:,1),m,x(:,2));

## grid;

title(sprintf(‘Уточнение решения системы’));

ylabel('x(m)'); xlabel('m');

pause;

plot(m,dx(:,1),m,dx(:,2));

## grid;

ylabel('dx(m)'); xlabel('m');

pause;

out=[m,x,dx]

## Функции rk1, rk2, rk4

Данные функции являются программной реализацией методов Рунге-Кутта 1-го, 2-го и 4-го порядка. Здесь производится расчет приближенных значений при интегрировании системы с t = [0; 1]. Входные параметры функции – правые части системы, начальный и конечный моменты времени, начальное значение вектора Y, шаг и признак трассировки. Выходные параметры – вектор моментов времени tout, матрица yout={n x 2}, где n – число шагов метода( по строкам матрицы располагаются вектора Ym, m = 0,n), вектор х, содержащий значения, полученные на последнем шаге работы функции и предназначенный для того, чтобы затем можно было передать значения данного вектора в подпрограмму, реализующую дискретный метод Ньютона.

Тексты программ (файл rk1.m, rk2.m, rk3.m):

function [tout,yout,x] = rk1(dif, t0, tfinal, y0, h, trace)

t=t0;y=y0;

tout=t;

yout=y;

b=0;

if trace

clc, t, h, y

end

while (t<tfinal)

if t+h>tfinal

h=tfinal-t;

end

if trace

clc, t, h, y

end

k1=feval(dif, t, y);

q=h\*k1;

y=y+q';

t=t+h;

b=b+1;

tout=[tout; t];

yout=[yout; y];

if trace

home,t, h, y

end;

end; x=y;

disp('Количество шагов = '); disp(b);

function [tout,yout,x] = rk2(dif, t0, tfinal, y0, h, trace)

t=t0;y=y0;

tout=t;

yout=y;

b=0;

if trace

clc, t, h, y

end

while (t<tfinal)

if t+h>tfinal

h=tfinal-t;

end

if trace

clc, t, h, y

end

k1=feval(dif, t, y);

z=h\*k1;

k2 = feval(dif, t+h, y+z');

q=h/2\*(k1+k2);

y=y+q';

t=t+h;

b=b+1;

tout=[tout; t];

yout=[yout; y];

if trace

home,t, h, y

end;

end; x=y;

disp('Kоличество шагов = '); disp(b);

function [tout,yout,x] = rk4(dif, t0, tfinal, y0, h, trace)

t=t0;y=y0;

tout=t;

yout=y;

b=0;

if trace

clc, t, h, y

end

while (t<tfinal)

if t+h>tfinal

h=tfinal-t;

end

if trace

clc, t, h, y

end

k1=feval(dif, t, y);

z=h\*k1;

k2 = feval(dif, t+h, y+z');

z=h\*k2/2;

k3 = feval(dif, t+h/2, y+z');

z=h\*k3;

k4 = feval(dif, t+h, y+z');

q=h\*(k1+2\*k2+2\*k3+k4)/6;

y=y+q';

t=t+h;

b=b+1;

tout=[tout; t];

yout=[yout; y];

if trace

home,t, h, y

end;

end; x=y;

disp('Kоличество шагов = '); disp(b);

Файл funf.m

#### Файл funf.m содержит исходную систему нелинейных САУ (описание ее правых частей). Пользователь может ввести в файл любую нелинейную систему ОДУ.

Текст файла funf.m:

function [F] = funf(x)

F=[x(1)\*x(1)+x(2)\*x(2)-4; x(1)\*x(2)-1];

## Файл dif.m

Файл формируется для того, чтобы создать систему дифференциальных уравнений, интегрируемых впоследствии по методам Рунге – Кутта. Он содержит матрицы начальных значений системы и значения матрицы Якоби. Функция вычисляет произведение обратной матрицы Якоби и начального решения системы.

Текст программы(файл dif.m):

function yp= dif(t,y)

J=[2\*y(1) 2\*y(2); y(2) y(1)];

J=-inv(J);

a=[0; -1];

b=J\*a;

yp=b;

Файл Dmn.m

Файл содержит подпрограмму, вычисляющую уточненное решение системы по дискретному методу. Он выводит количество пройденных итераций на экран. Выходными данными является вектор mout и матрицы xout, dxout.

Текст программы(файл dif.m):

function[xout,dxout,mout]=dmn(funf,x,dx,edop);

xout=x';

dxout=dx';

x1=x;

m=0; it=0;

mout=m;

nv=[1;1];

n=size(x);

while(max(nv)>edop)

f=feval(funf,x);

nf=norm(f);

for j=1:n,

x1(j)=x(j)+dx(j);

f1=feval(funf,x1);

x1(j)=x(j)-dx(j);

f2=feval(funf,x1);

J(:,j)=(f1-f2)/(2\*dx(j));

x1(j)=x(j);

end;

dx=-J\f;

ndx=norm(dx);

nv=[nf;ndx];

x=x+dx ;

m=m+1; it=it+1;

xout=[xout;x1'];

dxout=[dxout;dx'];

mout=[mout;m];

end;

disp('Количество итераций равно'); disp(it);

**2.4 Используемые технические средства**

ЭВМ, совместимые с IBM PC. Процессор не ниже Pentium1. ОП не меньше 32 Мб, мышь, стандартная клавиатура, видеокарта.

**2.5 Вызов и загрузка**

Для запуска программы из среды MatLab необходимо вызвать головную программу, набрав в командной строке ее имя main. При этом необходимо перейти в тот каталог, в котором находится данный пакет программ. Затем производятся все предусмотренные программой операции. После вывода на экран каждого графика или сообщения система ожидает нажатия любой клавиши.

Программа со всеми сопутствующими файлами, занимает 4,3 Кб.

**2.6 Входные данные**

Правые части системы в файле funf.m, матрица Якоби в файле dif.m, система дифференциальных уравнений, составленная по исходной системе нелинейных САУ (в файле dif.m), начальное приближение, значение шага, интервал времени для интегрирования, допустимая ошибка.

**2.7 Выходные данные**

На выходе программы на экран выводятся приближенные решения, полученные по методам Рунге-Кутта, значения шага и времени на каждом шаге интегрирования, количество шагов интегрирования, графики приближенных решений по методам Рунге-Кутта, количество итераций, необходимых для нахождения уточненного решения дискретным методом Ньютона, уточненное решение при каждой итерации, графики уточненных значений по методу Ньютона и ошибки для каждой составляющей решения.

# **3. ОПИСАНИЕ ТЕСТОВЫХ ЗАДАЧ**

Решение системы нелинейных САУ.

Для интегрирования возьмем систему:

x2+y2-4=0

xy – 1=0

Тогда при запуске программы на экране появляются следующие сообщения:

Метод Рунге - Кутта 1го порядка

t =

0

h =

0.1000

y =

2 0

…

t =

1

h =

1.1102e-016

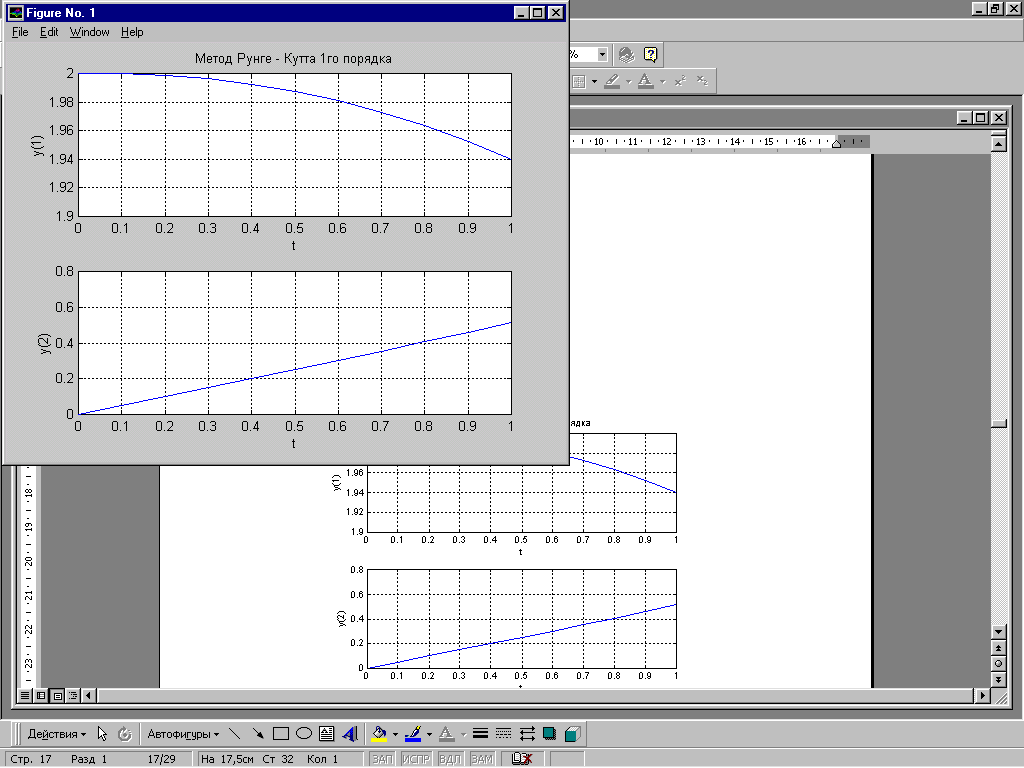
y =

1.9398 0.5139

Количество шагов =

11

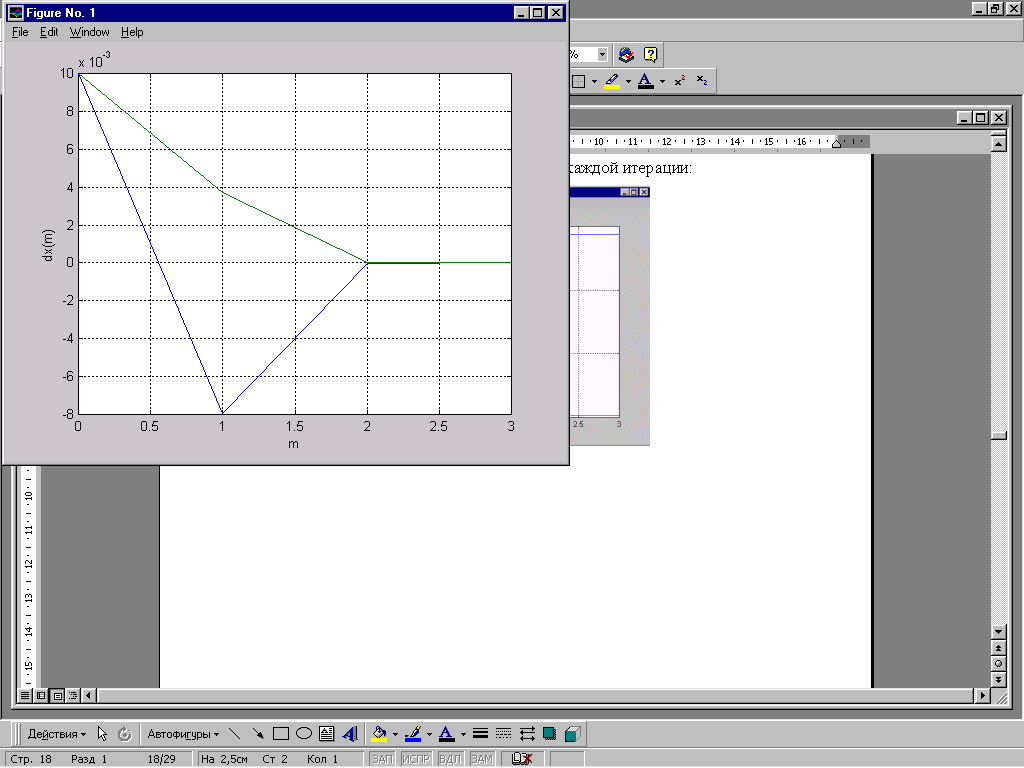
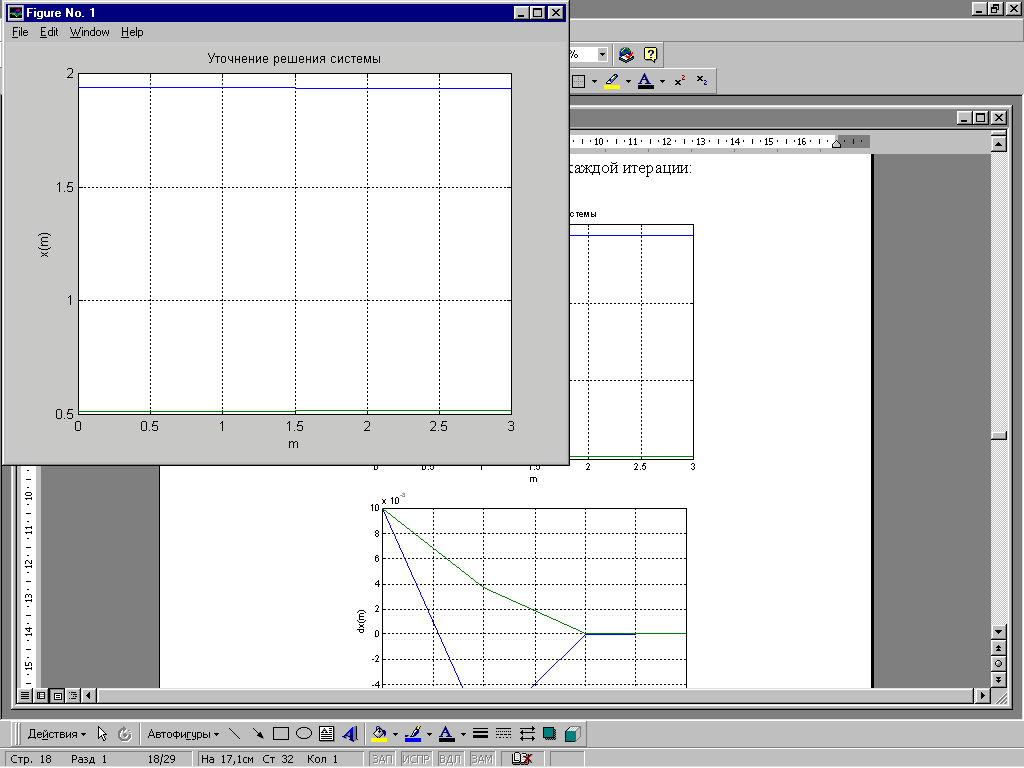
##### График решения системы ДУ:



Количество итераций равно

3

Графики значений системы и ошибок при каждой итерации:



out =

0 1.9398 0.5139 0.0100 0.0100

1.0000 1.9398 0.5139 -0.0079 0.0037

2.0000 1.9319 0.5176 0.0000 0.0000

3.0000 1.9319 0.5176 0.0000 0.0000

Метод Рунге - Кутта 2го порядка

t =

0

h =

0.1000

y =

2 0

…

t =

1

h =

1.1102e-016

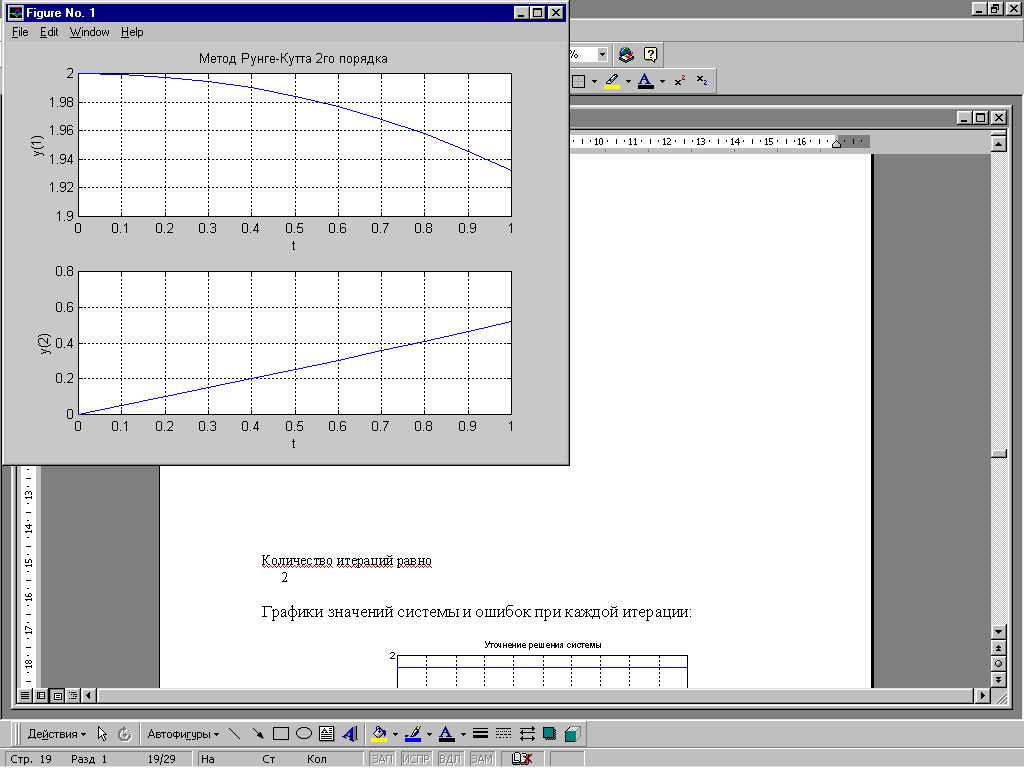
y =

1.9319 0.5176

Kоличество шагов =

11

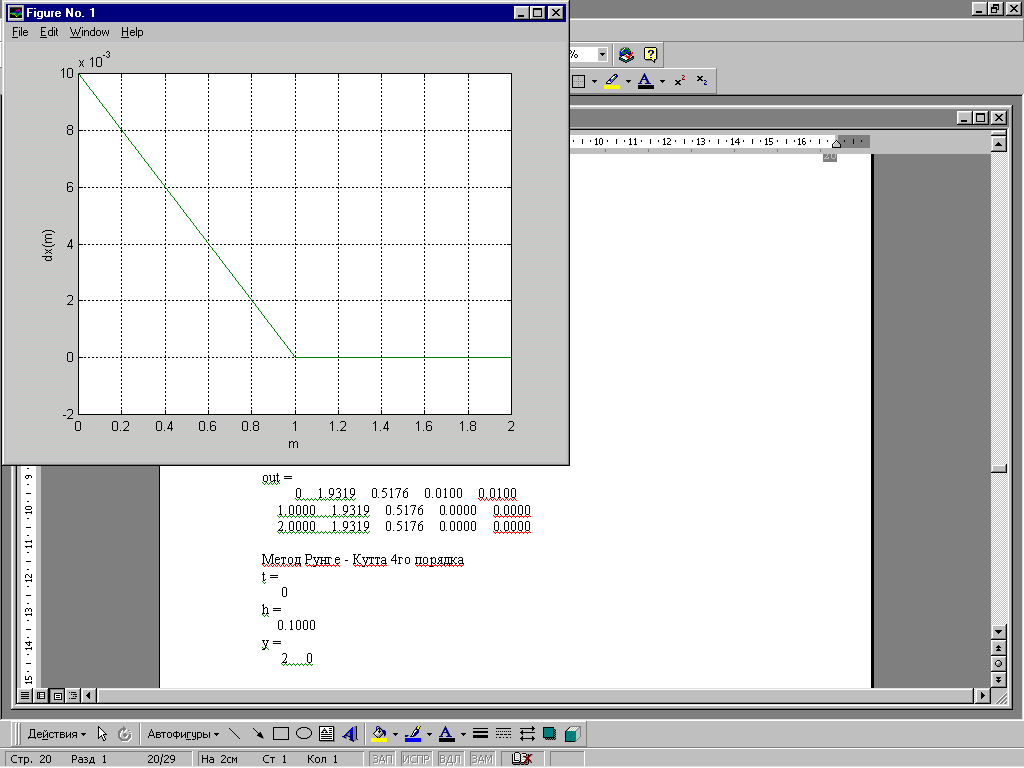
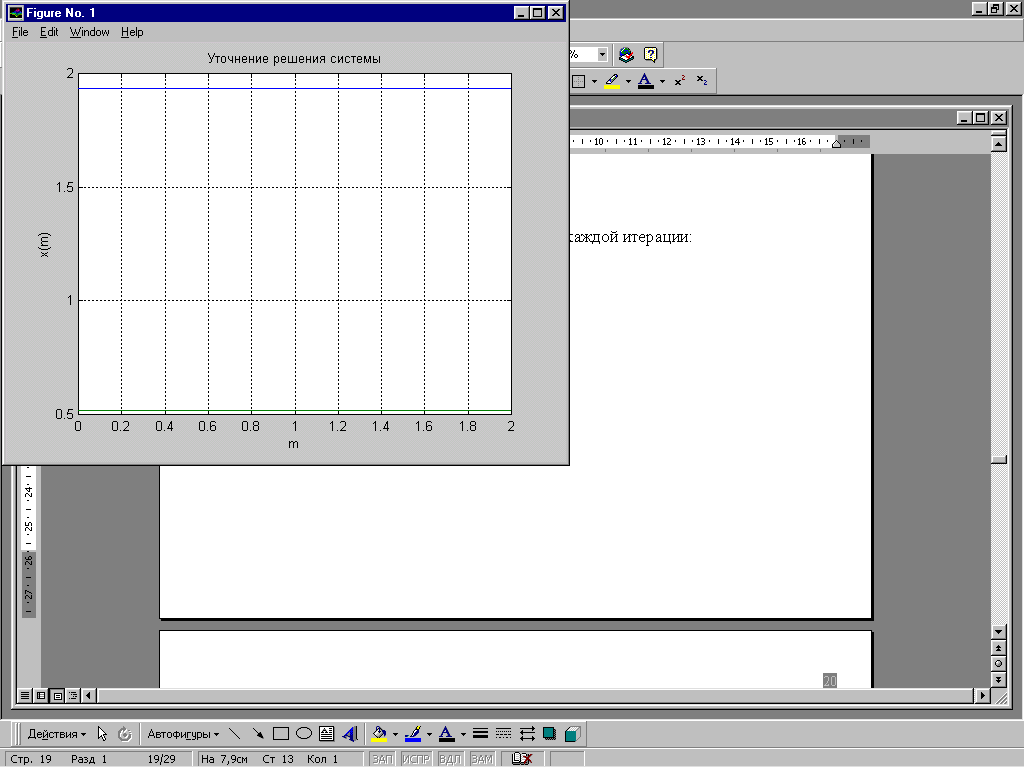
График решения системы ДУ:



Количество итераций равно

2

Графики значений системы и ошибок при каждой итерации:



out =

0 1.9319 0.5176 0.0100 0.0100

1.0000 1.9319 0.5176 0.0000 0.0000

2.0000 1.9319 0.5176 0.0000 0.0000

Метод Рунге - Кутта 4го порядка

t =

0

h =

0.1000

y =

2 0

…

t =

1

h =

1.1102e-016

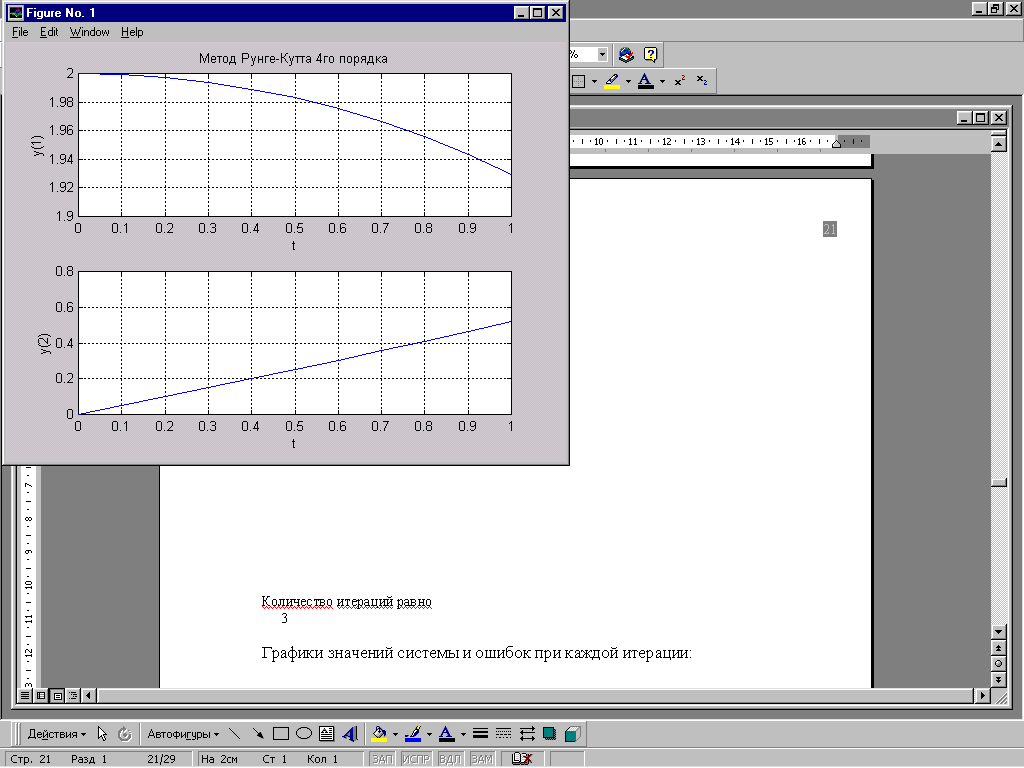
y =

1.9291 0.5190

Kоличество шагов =

11

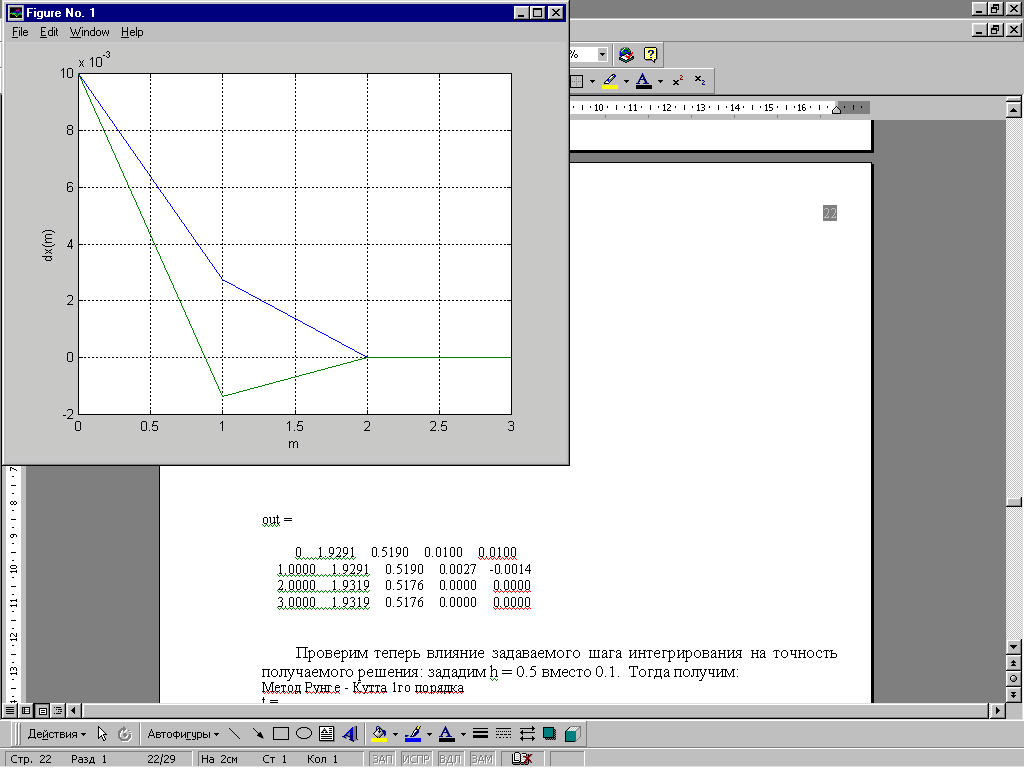
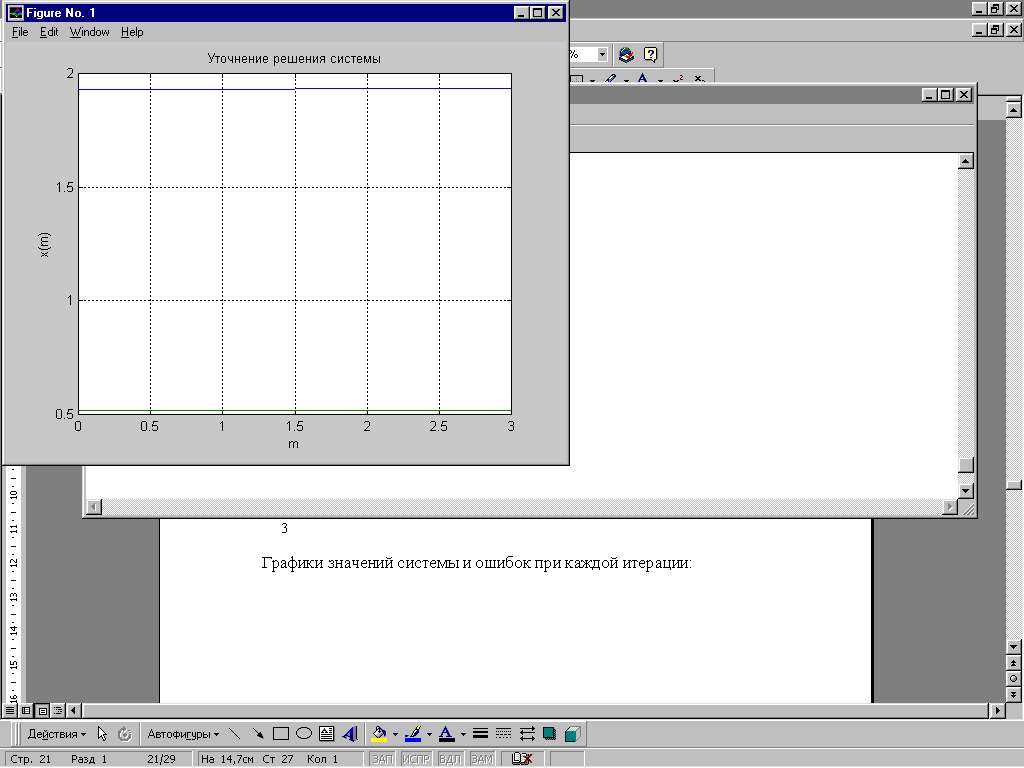
График решения системы ДУ:



Количество итераций равно

3

Графики значений системы и ошибок при каждой итерации:



out =

0 1.9291 0.5190 0.0100 0.0100

1.0000 1.9291 0.5190 0.0027 -0.0014

2.0000 1.9319 0.5176 0.0000 0.0000

3.0000 1.9319 0.5176 0.0000 0.0000

Проверим теперь влияние задаваемого шага интегрирования на точность получаемого решения: зададим h = 0.5 вместо 0.1. Тогда получим:

Метод Рунге - Кутта 1го порядка

t =

0

h =

0.5000

y =

2 0

…

t =

1

h =

0.5000

y =

1.9683 0.5040

Количество шагов =

2

Количество итераций равно

4

out =

0 1.9683 0.5040 0.0100 0.0100

1.0000 1.9683 0.5040 -0.0359 0.0133

2.0000 1.9323 0.5173 -0.0005 0.0004

3.0000 1.9319 0.5176 0.0000 0.0000

4.0000 1.9319 0.5176 0.0000 0.0000

Метод Рунге - Кутта 2го порядка

t =

0

h =

0.5000

y =

2 0

t =

1

h =

0.5000

y =

1.9321 0.5175

Kоличество шагов =

2

Количество итераций равно

2

out =

0 1.9321 0.5175 0.0100 0.0100

1.0000 1.9321 0.5175 -0.0002 0.0002

2.0000 1.9319 0.5176 0.0000 0.0000

Метод Рунге - Кутта 4го порядка

t =

0

h =

0.5000

y =

2 0

t =

1

h =

0.5000

y =

1.9183 0.5243

Kоличество шагов =

2

Количество итераций равно

3

out =

0 1.9183 0.5243 0.0100 0.0100

1.0000 1.9183 0.5243 0.0137 -0.0068

2.0000 1.9319 0.5176 -0.0001 0.0001

3.0000 1.9319 0.5176 0.0000 0.0000

Видим, что при увеличении *h* снизилась точность получаемого приближенного решения, уменьшилось количество шагов по методу Рунге – Кутта (их стало не 11, а 3), и, вследствие этого, увеличилось количество итераций по дискретному методу Ньютона.

Проверим влияние задаваемой допустимой ошибки для дискретного метода Ньютона: зададим edop = 0.001 вместо edop = 0.00001. Получаем:

Метод Рунге - Кутта 1го порядка

t =

0

h =

0.1000

y =

2 0

…

t =

1

h =

1.1102e-016

y =

1.9398 0.5139

Количество шагов =

11

Количество итераций равно

2

out =

0 1.9398 0.5139 0.0100 0.0100

1.0000 1.9398 0.5139 -0.0079 0.0037

2.0000 1.9319 0.5176 0.0000 0.0000

Метод Рунге - Кутта 2го порядка

t =

0

h =

0.1000

y =

2 0

…

t =

1

h =

1.1102e-016

y =

1.9319 0.5176

Kоличество шагов =

11

Количество итераций равно

1

out =

0 1.9319 0.5176 0.0100 0.0100

1.0000 1.9319 0.5176 0.0000 0.0000

Метод Рунге - Кутта 4го порядка

t =

0

h =

0.1000

y =

2 0

…

t =

1

h =

1.1102e-016

y =

1.9291 0.5190

Kоличество шагов =

11

Количество итераций равно

2

out =

0 1.9291 0.5190 0.0100 0.0100

1.0000 1.9291 0.5190 0.0027 -0.0014

2.0000 1.9319 0.5176 0.0000 0.0000

Видим, что при увеличении допустимой ошибки для дискретного метода Ньютона уменьшается число итераций, так как уже при второй, и даже первой, итерации достигается заданная точность решения.

Решим эту же систему при другом начальном приближении Х0 = (3 0).

Метод Рунге - Кутта 1го порядка

t =

0

h =

0.1000

y =

3 0

…

t =

1

h =

1.1102e-016

y =

3.7401 0.2728

Количество шагов =

11

Количество итераций равно

6

out =

0 3.7401 0.2728 0.0100 0.0100

1.0000 3.7401 0.2728 -1.3520 0.0932

2.0000 2.3880 0.3660 -0.3996 0.0984

3.0000 1.9884 0.4644 -0.0539 0.0484

4.0000 1.9345 0.5128 -0.0026 0.0047

5.0000 1.9319 0.5176 0.0000 0.0001

6.0000 1.9319 0.5176 0.0000 0.0000

Метод Рунге - Кутта 2го порядка

t =

0

h =

0.1000

y =

3 0

…

t =

1

h =

1.1102e-016

y =

3.7321 0.2680

Kоличество шагов =

11

Количество итераций равно

6

out =

0 3.7321 0.2680 0.0100 0.0100

1.0000 3.7321 0.2680 -1.3467 0.0967

2.0000 2.3854 0.3646 -0.3973 0.0992

3.0000 1.9881 0.4638 -0.0536 0.0490

4.0000 1.9345 0.5128 -0.0026 0.0047

5.0000 1.9319 0.5176 0.0000 0.0001

6.0000 1.9319 0.5176 0.0000 0.0000

Метод Рунге - Кутта 4го порядка

t =

0

h =

0.1000

y =

3 0

…

t =

1

h =

1.1102e-016

y =

3.7294 0.2664

Kоличество шагов =

11

Количество итераций равно

6

out =

0 3.7294 0.2664 0.0100 0.0100

1.0000 3.7294 0.2664 -1.3449 0.0978

2.0000 2.3845 0.3642 -0.3965 0.0995

3.0000 1.9880 0.4637 -0.0535 0.0492

4.0000 1.9345 0.5128 -0.0026 0.0047

5.0000 1.9319 0.5176 0.0000 0.0001

6.0000 1.9319 0.5176 0.0000 0.0000

###### Видим, что количество итераций для дискретного метода Ньютона увеличивается, так как начальное решение (3 0) немного дальше от точного, чем (2 0), и для уточнения приходится совершать больше итераций.

# **4.АНАЛИЗ РЕЗУЛЬТАТОВ. ВЫВОДЫ**

Видно, что графики приближенных решений по методам Рунге-Кутта ненамного отличаются. Лишь при выводе на экран численных значений решения, можно увидеть отличия. При этом более точное приближенное решение получилось у метода Рунге-Кутта второго порядка (при использовании метода Рунге-Кутта первого порядка получилось приближенное решение, где первая составляющая чуть больше уточненного, а вторая – чуть меньше; при использовании же метода Рунге-Кутта четвертого порядка наоборот). Но на данном этапе нет необходимости получать более точное решение, поэтому с точки зрения вычислительных затрат целесообразнее использовать метод Рунге-Кутта первого порядка

При использовании дискретного метода Ньютона для уточнения решения метод сходится за 2-3 итерации. При чем точность можно регулировать с помощью допустимой ошибки: чем меньше мы зададим допустимую ошибку, тем больше точность.

Можно прийти к выводу, что целесообразнее при решении нелинейных САУ методом дифференцирования по параметру использовать для вычисления приближенного решения метод Рунге-Кутта первого порядка. Так как необходимую точность можно получить потом при уточнении решения. А шаг интегрирования можно даже выбрать 0.5, то есть достаточно большим. Метод сойдется, а вычислительных затрат будет меньше. При уточнении же дискретным методом Ньютона все равно получится достаточно точное решение, а количество итераций станет ненамного больше.

Метод дифференцирования по параметру обладает глобальной сходимостью, поэтому он сойдется даже при достаточно неточном первоначальном приближении (это проверено при Х0 = (3 0)).

Итак, при решении систем нелинейных уравнений методом дифференцирования по параметру получаются достаточно точные значения. Можно сделать вывод, что данный метод эффективен.

# **ЗАКЛЮЧЕНИЕ**

В заключении можно сказать, что проведенное исследование оказалось успешным, задачи, поставленные вначале проекта, выполнены. В работе исследовано влияние метода интегрирования на точность получаемого решения. Получены сведения о зависимости точности интегрирования от величины шага; о зависимости получаемого уточненного решения от величины допустимой ошибки и от начального приближенного решения; а также от выбора порядка метода Рунге – Кутта для получения приближенного решения.

# **СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ**

1. Бахвалов Н.С. Численные методы. - Часть1.- М: Наука, 1975. – 632с.
2. Кузьмик П.К., Маничев В.Б. Автоматизация функционального проектирования. Кн.5. Системы автоматизированного проектирования/ Под ред. И.П. Норенкова. – М: Высшая школа, 1986. – 144 с.
3. Потабенко Н.А. Численные методы. – М.: Изд-во МАИ, 1997. – 88с.: ил.
4. Сарычева О.М. Численные методы в экономике. Конспект лекций. – Новосибирск, 1995. – 65с.