Министерство образования и науки Российской Федерации
Санкт-Петербургский политехнический университет Петра Великого
—
Институт компьютерных наук и технологий
Кафедра «Информационная безопасность компьютерных систем»

ОТЧЕТ
ПО КУРСОВОЙ РАБОТЕ

по дисциплине «Вычислительная математика»

Выполнил
студент гр. 23508/4 Е.Г. Проценко

Проверил
профессор С.М. Устинов

Санкт-Петербург
2016

1 ФОРМУЛИРОВКА ЗАДАНИЯ (ВАРИАНТ 13)

Для решения нелинейной краевой задачи относительно $y(x)$ на интервале $0 \leq x \leq 1$

$\frac{d^2y}{dx^2} = y^2 - 1,$

$y(0) = 0,$

$y(1) = 1$

может быть использован следующий подход.

Исходное уравнение переписываем в виде $\frac{d}{dx}\left(\frac{(y')^2}{2} - \frac{y^3}{3} + y\right) = 0$

Отсюда, $\frac{(y')^2}{2} - \frac{y^3}{3} + y = \alpha$ (1),

где α – некоторая константа. Поскольку $y(0) = 0$, то $y'(0) = \sqrt{2\alpha}$. Если бы мы могли вычислить α, то исходная задача свелась бы к задаче Коши, легко решаемой с помощью подпрограммы *RKF45*.

Интегрирование уравнения (1) дает $x = \int_0^x \frac{dy}{\sqrt{2}\left(\alpha + \frac{y^3}{3} - y\right)^{\frac{1}{2}}}$.

Используя граничное условие $y(1) = 1$, получим уравнения для α: $1 = \int_0^1 \frac{dy}{\sqrt{2}\left(\alpha + \frac{y^3}{3} - y\right)^{\frac{1}{2}}}$, которое может быть решено с помощью подпрограммы QUANC8 и ZEROIN.

Реализовать этот подход к решению задачи. Оценить погрешность результатов и погрешность определяемую неточностью в исходных данных.

2 РЕЗУЛЬТАТЫ РАБОТЫ

2.1 Нахождение параметра α

Единственное, что нам мешает решать сразу дифференциальное уравнение, используя подпрограмму $RKF45$, это то, что параметр α нам неизвестен. Для его нахождения предложено использовать подпрограмму $ZEROIN$. Т.к. в крайнюю названную программу поступает функция содержащая интеграл, то мы будем использовать $QUANC8$ внутри той функции, которая отправляется в $ZEROIN$.

```
a := zeroin(ax,bx,tol,@F1);
----------------------------------
function F1(new_a: float) : float;
begin
      a := new_a;
      abserr := 0;
      relerr := 0.00001;
      quanc8(@F2, 0, 1, abserr, relerr, result, errest, nofun, flag);
      writeln('a:   ', a, ';   result:   ', result);
      F1 := result - 1;
end;
-----------------------------------
function F2(y: float) : float;
begin
      F2 := (1/(sqrt(2) * power((a + power(y, 3) / 3 - y), 1/2)));
end;
```

Результат работы $ZEROIN$:

```
a:    0.0000000000E+00;   result:    1.4695270930E+00
a:    1.0000000000E+00;   result:    9.6974961983E-01
a:    9.3947230158E-01;   result:    1.0370331420E+00
a:    9.7278700895E-01;   result:    9.9814726788E-01
a:    9.7109615836E-01;   result:    1.0000046078E+00
a:    9.7110035309E-01;   result:    9.9999998570E-01
a:    9.7110034012E-01;   result:    1.0000000000E+00
 9.7110034012E-01
```

0.97110034012 – это α, полученное от подпрограммы $ZEROIN$.

2.2 Решение дифференциального уравнения второго порядка

Получив α, мы можем подставить его в соответствующую формулу. Помимо этого нам стал известен $y'(0) = \sqrt{2\alpha}$.

Следующим шагов является решение дифференциального уравнения второго порядка. У нас есть подпрограмма $RKF45$, которая работает с дифференциальными уравнениями, но первого порядка. Поэтому на нужно используя замену разложить дифференциальное уравнение первого порядка на 2 дифференциального уравнения первого порядка:

$$\begin{cases} w = y' \\ u = y \end{cases}$$

$$\begin{cases} w' = u^2 - 1 \\ u' = w \end{cases}$$

По коду: $z[1] = w,\ z[2] = u$.

```
procedure F4(t: float; var z, dz: floatvector);
begin
      dz[1] := power(z[2], 2) - 1;
      dz[2] := z[1];
end;
```

Начальные условия:

$$\begin{cases} w(0) = y'(0) \\ u(0) = y(0) \end{cases}$$

$$\begin{cases} w = \sqrt{2\alpha} \\ u = 0 \end{cases}$$

Теперь можно воспользоваться подпрограммой $RKF45$.

Результат работы $RKF45$:

```
t=   0.00  w  1.393628602  u=  0.000000000   Flag= 2
t=   0.10  w  1.294299581  u=  0.134384138   Flag= 2
t=   0.20  w  1.198328751  u=  0.258974619   Flag= 2
t=   0.30  w  1.108511475  u=  0.374255721   Flag= 2
t=   0.40  w  1.026949052  u=  0.480952649   Flag= 2
t=   0.50  w  0.955233342  u=  0.579974184   Flag= 2
t=   0.60  w  0.894576838  u=  0.672368247   Flag= 2
t=   0.70  w  0.845938987  u=  0.759290330   Flag= 2
t=   0.80  w  0.810145735  u=  0.841984229   Flag= 2
t=   0.90  w  0.788002435  u=  0.921774387   Flag= 2
t=   1.00  w  0.780402759  u=  1.000069294   Flag= 2
```

2.3 Влияние погрешности исходных данных на решение

Касательно погрешности подпрограмм $QUANC8$ и $RKF45$ можно сказать, что вычисления получены добротные, $ESP(QUANC8) = 10^{-7}$, $ESP(RKF45) = 10^{-6}$. Трогать их не будем.

Но что будет, если мы получим параметр α с погрешностью $+1\%$.

```
a := zeroin(ax,bx,tol,@F1);
a := a * 1.01;
```

Новое $\alpha = 0{,}98081134352$.

Результат работы RKF45:

```
t=   0.00  w  1.400579411  v=  0.000000000   Flag= 2
t=   0.10  w  1.301256955  v=  0.135079404   Flag= 2
t=   0.20  w  1.205328906  v=  0.260367315   Flag= 2
t=   0.30  w  1.115624003  v=  0.376353341   Flag= 2
t=   0.40  w  1.034273869  v=  0.483771182   Flag= 2
t=   0.50  w  0.962899218  v=  0.583541060   Flag= 2
t=   0.60  w  0.902741549  v=  0.676725212   Flag= 2
t=   0.70  w  0.854791335  v=  0.764496437   Flag= 2
t=   0.80  w  0.819909701  v=  0.848119123   Flag= 2
t=   0.90  w  0.798943842  v=  0.928942137   Flag= 2
t=   1.00  w  0.792838963  v=  1.008403033   Flag= 2
```

Теперь посмотрим, что будет, если параметр α меньше того, что мы получили ранее на 1%.

```
a := zeroin(ax,bx,tol,@F1);
a := a * 0.99;
```

Новое $\alpha = 0{,}96138933671$.

Результат работы RKF45:

```
t=   0.00  w  1.386642951  v=  0.000000000   Flag= 2
t=   0.10  w  1.287307364  v=  0.133685388   Flag= 2
t=   0.20  w  1.191293767  v=  0.257574956   Flag= 2
t=   0.30  w  1.101364178  v=  0.372147653   Flag= 2
t=   0.40  w  1.019589638  v=  0.478120198   Flag= 2
t=   0.50  w  0.947533256  v=  0.576389948   Flag= 2
t=   0.60  w  0.886378680  v=  0.667990534   Flag= 2
t=   0.70  w  0.837054575  v=  0.754060194   Flag= 2
t=   0.80  w  0.800352057  v=  0.835822207   Flag= 2
t=   0.90  w  0.777035153  v=  0.914576714   Flag= 2
t=   1.00  w  0.767946741  v=  0.991703325   Flag= 2
```

3 ВЫВОД

Зная, что $y(1) = 1$, получаем, что при том α, что мы получили изначально, можно сделать вывод, что погрешность появилась только в 5-ом разряде.

При увеличении и уменьшении α на 1%, погрешность перепрыгнула с пятого порядка на третий. Можно с уверенностью сказать, что такая система не является очень то устойчивой, т.к. при изменении исходных данных всего на 1% мы получили погрешность в 100 раз больше, чем раньше.

4 ПРИЛОЖЕНИЕ

Листинг написанной программы:

```
uses FMM, CRT, MATH;

label rinse;

Var
	a, a_etalon: float;
	ax, bx, tol: float;
	y: float;
	repits:integer;

	z, zp: floatvector;
	t, tout, tfinal, tprint: float;
	iflag: integer;
	work: rvecn;
	iwork: ivec5;

	abserr, relerr: float;
	nofun: longint;
	flag: float;
	result: float;
	errest: float;


{$F+}

function F2(y: float) : float;
begin
	F2 := (1/(sqrt(2) * power((a + power(y, 3) / 3 - y), 1/2)));
end;

function F1(new_a: float) : float;
{Var}
begin
	a := new_a;
	abserr := 0;
	relerr := 0.0000001;
	quanc8(@F2, 0, 1, abserr, relerr, result, errest, nofun, flag);
	writeln('a:   ', a, ';   result:   ', result);
	F1 := result - 1;
end;

procedure F4(t: float; var z, dz: floatvector);
begin
	dz[1] := power(z[2], 2) - 1;
	dz[2] := z[1];
end;

{$F-}

begin
	clrscr;

	{ZEROIN + QUANC8}
	ax := 0;
	bx := 1;

	tol := 1e-10;
	a := zeroin(ax,bx,tol,@F1);
	writeln(a);

	readln;
```

```
repit:
       {RFK45}
       t := 0;
       tfinal := 1;
       tout := t;
       tprint := 0.1;
       relerr := 0.000001;
       abserr := 0;
       iflag := 1;
       z[1] := sqrt(2 * a);
       z[2] := 0;

rinse:
       rkf45(@F4,2,z,t,tout,relerr,abserr,iflag,work,iwork);
       writeln('  t= ',t:6:2, '  w',z[1]:13:9,'  v=',
                     z[2]:13:9,'   Flag=',iflag:2);
       case iflag of
              1, 8   :      exit;
              2             :      begin
                                          tout := t + tprint;
                                          if t < tfinal then goto rinse
                                   end;
              4             :      goto rinse;
              5             :      begin
                                          abserr := 1E-9;
                                          goto rinse
                                   end;
              6             :      begin
                                          relerr := 10 * relerr;
                                          iflag := 2;
                                          goto rinse
                                   end;
              7             :      begin
                                          iflag := 2;
                                          goto rinse
                                   end;
       end;

       readln;

end.
```