II. NUMERICAL METHODS FOR INTEGRATION

II.1 Quadrature formulas

II.1.1 Introductory notes

Numerička integracija funkcija sastoji se u približnom izračunavanju odredjenih integrala na osnovu niza vrednosti podintegralne funkcije po odredjenoj formuli.

Formule za numeričko izračunavanje jednostrukih integrala nazivaju se **kvadraturne formule**. Slično, formule za dvostrukeintegrale nazivaju se kubaturne formule.

U našem izlaganju zadržaćemo se uglavnom na kvadraturnim formulama.

Potreba za numeričkom integracijom javlja se u velikom broju slučajeva. Naime, Newton-Leibnitzova formula (1.1.1)

$$\int_a^b f(x)dx = f(b) - f(a), \tag{1.1.1}$$

gde je F primitivna funkcija za funkiju f, ne može se uvek uspešno primeniti. Navešćemo neke od tih slučajeva:

1. Funkcija F se ne može predstaviti pomoću konačnog broja elementarnih funkcija (na primer, kada je $f(x) = e^{-x^2}$).
2. Primena formule (1.1.1) često dovodi do vrlo složenog izraza, čak i kod izračunavanja integrala jednostavnijih funkcija; na primer
$$\int_a^b \frac{dx}{1+x^3} = \log \sqrt[3]{|a+1|} - \frac{1}{6}\log a^2 - a + 1 + \frac{1}{\sqrt{3}} arctg \frac{a\sqrt{3}}{2-a}.$$
3. Kod integracije funkcija čije su vrednosti poznate na diskretnom skupu tačaka (dobijene, na primer, eksperimentalno), nije mogućno primeniti formulu (1.1.1).

Veliki broj kvadraturnih formula ima oblik

$$\int_a^b f(x)dx \cong \sum_{k=0}^{n} A_k f_k, \tag{1.1.2}$$

gde je $f_k = f(x_k)(a \leq x_0 < \ldots < x_n \leq b)$. Ako je $x_0 = a$ i $x_n = b$, za formulu (1.1.2) kažemo da je zatvorenog tipa, dok u ostalim slučajevima kažemo da je otvorenog tipa. Napomenimo da se za integraciju diferencijabilnih funckija koriste i formule u kojima se pored vrednosti funkcije pojavljuju i vrednosti izvoda. Od interesa su i formule za izračunavanje integrala

$$\int_a^b p(x)f(x)dx,$$

gde je $x \to p(x)$ data težinska funkcija.

Jedan prost način za konstrukciju kvadraturnih formula zasniva se na primeni interpolacije. Formule dobijene na ovaj način nazivaju se kvadraturne formule **interpolacionog tipa**.

Neka su vrednosti funkcije f u datim tačkama $x_0, x_1, \ldots, x_n (\in [a, b])$ redom $f_0, f_1, \ldots, f_n$, tj.

$$f_k \equiv f(x_k) \qquad (k = 0, 1, \ldots, n).$$

Na osnovu ovih podataka, možemo konstruisati Lagrangeov interpolacioni polinom

$$P_n(x) = \sum_{k=0}^{n} f_k \frac{\omega(x)}{(x - x_k)\omega'(x_k)},$$

gde je $\omega(x) = (x - x_0)(x - x_1) \cdots (x - x_n)$.

Tada je

$$\int_a^b p(x)f(x)dx = \int_a^b p(x)P_n(x)d\,x + R_{n+11}(f),$$

tj.

$$\int_a^b p(x)f(x)dx = \sum_{k=0}^{n} A_k f_k + R_{n+1}(f), \tag{1.1.3}$$

gde smo stavili

$$A_k = \int_a^b \frac{p(x)\omega(x)}{(x-x_k)\omega'(x_k)}dx \qquad (k=0,1,\ldots,n).$$

U formuli (1.1.3), veličina $R_{n+1}(f)$ naziva se ostatak kvadraturne formule i predstavlja grešku koja se čini zamenom integrala konačnom sumom. Indeks $n+1$ u ostatku označava da se integral približno izračunava na osnovu vrednosti podintegralne funkcije u $n+1$ tačaka.

Sa Π_n označimo skup svih polinoma ne višeg stepena od n.

Kako je za $f(x) = x^k \quad (k = 0,1,\ldots,n),\ f(x) \equiv P_n(x)$, imamo $R_{n+1}(x^k) \equiv 0 \quad (k = 0,1,\ldots,n)$, odakle zaklju ujemo da je formula (1.1.3) tačna za svako $f \in \Pi_n$, bez obzira na izbor interpolacionih čvorova x_k $(k = 0,1,\ldots,n)$ i u ovom slučaju kažemo da formula (1.1.3) ima algebarski stepen tačnosti n.

II.1.2 Newton-Cotesove formule

U ovom odeljku izvešćemo kvadraturne formule zatvorenog tipa u kojima su interpolacioni čvorovi $x_k = x_0 + kh$ $(k = 0,1,\ldots,n)$ uzeti ekvidistantno sa korakom $h = \frac{b-a}{n}$.

Ako uvedemo smenu $x - x_0 = ph$, imamo

$$\omega(x) = (x-x_0)(x-x_1)\ldots(x-x_n) = h^{n+1}p(p-1)\ldots(p-n) \tag{1.2.1}$$

i

$$\begin{aligned}\omega'(x_k) &= (x_k-x_0)(x_k-x_1)\ldots(x_k-x_{k-1})(x_k-x_{k+1})\ldots(x-x_n)\\ &= h^n(-1)^{n-k}k!1(n-k)!\end{aligned} \tag{1.2.2}$$

Uvodjenjem oznake za uopšteni stepen $x^{(s)} = x(x-1)\dots(x-s+1)$, na osnovu (1.2.1), (1.2.2) i rezultata iz prethodnog odeljka dobijamo

$$A_k = \int_0^n \frac{(-1)^{n-k} p^{(n+1)} h}{(p-k)k!(n-k)!} dp \quad (k = 0, 1, \dots, n),$$

tj.

$$A_k = (b-a)H_k \quad (k = 0, 1, \dots, n),$$

gde smo stavili

$$(1.2.3)\; H_k \equiv H_k(n) = \frac{(-1)^{n-k}}{n!n} \binom{n}{k} \int_0^n \frac{p^{(n+1)}}{p-k} dp \quad (k = 0, 1, \dots, n).$$

Koeficijenti H_k u literaturi su poznati kao Newton-Cotesovi koeficijenti, a odgovarajuće formule

$$(1.2.4) \qquad \int_{x_0=a}^{x_n=b} f(x)dx = (b-a) \sum_{k=0}^{n} H_k f(a + k\frac{b-a}{n}) \quad (k \in N)$$

kao Newton-Cotesove formule.

U daljem izlaganju daćemo pregled Newton-Cotesovih formula za $n \leq 8$. Pri ovome koristimo oznake $h = \frac{b-a}{n}, f_k = f(x_k)$ $(k = 0, 1, \dots, n)$.

1. $n = 1$ (trapezno pravilo)

$$\int_{x_0}^{x_1} f(x)dx = \frac{h}{2}(f_0 + f_1) - \frac{h^3}{12} f''(\xi_1);$$

2. $n = 2$ (Simpsonovo pravilo)

$$\int_{x_0}^{x_2} f(x)dx = \frac{h}{3}(f_0 + 4f_1 + f_2) - \frac{h^5}{90} f^{IV}(\xi_2);$$

3. $n = 3$ (Simpsonovo pravilo $\frac{3}{8}$)

$$\int_{x_0}^{x_3} f(x)dx = \frac{3h}{8}(f_0 + 3f_1 + 3f_2 + f_3) - \frac{3h^5}{80} f^{IV}(\xi_3);$$

4. $n = 4$ (Booleovo pravilo)

$$\int_{x_0}^{x_4} f(x)dx = \frac{2h}{45}(7f_0 + 32f_1 + 12f_2 + 32f_3 + 7f_4) - \frac{8h^7}{945} f^{VI}(\xi_4);$$

5. $n = 5$

$$\int_{x_0}^{x_5} f(x)dx = \frac{5h}{288}(19f_0 + 75f_1 + 50f_2 + 50f_3 + 75f_4 + 19f_5) - \frac{275h^7}{12096} f^{VI}(\xi_5);$$

6. $n = 6$

$$\int_{x_0}^{x_6} f(x)dx = \frac{h}{140}(41f_0 + 216f_1 + 27f_2 + 272f_3 + 27f_4 + 216f_5 + 41f_6) - \frac{9h^9}{1400} f^8(\xi_6);$$

7. $n = 7$

$$\int_{x_0}^{x_7} f(x)dx = \frac{7h}{17280}(751f_0 + 3577f_1 + 1323f_2 + 2989f_3 + 2989f_4 + 1323f_5 + 3577f_6 + 751f_7) - \frac{8183h^9}{518400} f^8(\xi_7);$$

8. $n = 8$

$$\int_{x_0}^{x_8} f(x)dx = \frac{4h}{14175}(989f_0 + 5888f_1 - 928f_2 + 10496f_3 - 4540f_4 + 10496f_5 - 928f_6 + 5888f_7 + 989f_8) - \frac{2368h^{11}}{467775} f^{10}(\xi_8);$$

gde $\xi_k \in (x_0, x_k)$ $(k = 1, \ldots, 8)$.
U opštem slučaju ostatak $R_{n+1}(f)$ ima oblik

$$R_{n+1}(f) = C_n h^m f^{(m-1)}(\xi_n) \quad (x_0 < \xi_n < x_n),$$

gde je $m = 2[\frac{n}{2}] + 3$. Data jednakost ima smisla ukoliko funkcija $f \in C^{m-1}[a, b]$.

II.1.3 Uopštene kvadraturne formule

Da bismo tačnije izračunali vrednost integrala potrebno je podeliti segment $[a, b]$ na niz podsegmenata, a zatim na svakom od njih primeniti neku od kvadraturnih formula. Na taj način dobijamo uopštene ili kompozitne formule. U ovom odeljku razmotrićemo uopštene formule dobijene na bazi trapezne i Simpsonove formule.

Podelimo segment $[a, b]$ na niz podsegmenata $[x_{i-1}, x_i]$ tako da je $x_i = a + ih$ i $h = (b - a)/n$.

sl.1.3.1

Primenom trapezne formule na svaki od podsegmenata dobijamo

$$\int_a^b f(x)dx = \sum_{i=1}^{n} \int_{x_{i-1}}^{x_i} f(x)dx = \sum_{i=1}^{n} (\frac{h}{2}(f_{i-1} + f_i) - \frac{h^3}{12} f''(\xi_i)),$$

tj.

$$\int_a^b f(x)dx = T_n - \frac{h^3}{12}\sum_{i=1}^{n} f''(\xi_i),$$

gde su

$$T_n \equiv T_n(f;h) = h(\frac{1}{2}f_0 + f_1 + \cdots + f_{n-1} + \frac{1}{2}f_n)$$

i

$$x_{i-1} < \xi_i < x_i \quad (i = 1, 2, \ldots, n).$$

Teorema 1.3.1. *Ako $f \in C^2[a,b]$ važi jednakost*

$$\int_a^b f(x)dx - T_n = -\frac{(b-a)^2}{12n^2} f''(\xi) \quad (a < \xi < b).$$

Kvadraturna formula

$$\int_a^b f(x)dx \cong T_n(f;h) \quad (h = \frac{b-a}{n})$$

naziva se uopštena trapezna formula.

Pretpostavimo sada da je $h = \frac{b-a}{2n}$, tj. $x_i = a + ih \quad i = 0, 1, \ldots, 2n)$ (videti sl. 1.3.2), a zatim na podsegmente

$$[x_0, x_2], \ldots, [x_{2n-2}, x_{2n}]$$

primenimo Simpsonovu formulu. Na ovaj način dobijamo uopštenu Simpsonovu formulu

$$\int_a^b f(x)dx \cong S_n(f;h) \quad (h = \frac{b-a}{2n}),$$

gde je

$$S_n \equiv S_n(f,h) = \frac{h}{3}\{f_0 + 4(f_1 + \ldots + f_{2n-1}) + 2(f_2 + \ldots + f_{2n-2}) + f_{2n}\}.$$

sl.1.3.2

Teorema 1.3.2. *Ako* $f \in C^4[a,b]$ *važi jednakost*

$$\int_a^b f(x)dx - S_n = -\frac{(b-a)^2}{2880n^4} f^{(IV)}(\xi) \quad (a < \xi < b).$$

II.1.4 Rombergova integracija

Za izračunavanje odredjenih integrala, u praksi se najčešće koristi uopštena trapezna formula u jednom specijalnom obliku, koji je poznat kao **Rombergova integracija.**

Sa $T_k^{(0)}$ označimo trapeznu aproksimaciju $T_n(f;h)(n = 2^k$, tj. $h = \frac{(b-a)}{2^k})$. Rombergova integracija se sastoji u konstrukiji dvodimenzionalnog niza $T_k^{(m)}$ $(m = 0, 1, \ldots, k; k = 0, 1, \ldots)$ pomoću

$$T_k^{(m)} = \frac{4^m T_{k+1}^{(m-1)} - T_k^{(m-1)}}{4^m - 1}. \tag{1.4.1}$$

Na osnovu (1.4.1) može se konstruisati tzv. T tabela

$$\begin{array}{ccccccc}
T_0^{(0)} & \longrightarrow & T_0^{(1)} & \longrightarrow & T_0^{(2)} & \longrightarrow & T_0^{(3)} \\
 & \nearrow & & \nearrow & & \nearrow & \vdots \\
T_1^{(0)} & \longrightarrow & T_1^{(1)} & \longrightarrow & T_1^{(2)} & \longrightarrow & \\
 & \nearrow & & \nearrow & \vdots & & \\
T_2^{(0)} & \longrightarrow & T_2^{(1)} & \longrightarrow & & & \\
 & \nearrow & \vdots & & & & \\
T_3^{(0)} & \longrightarrow & & & & &
\end{array}$$

uzimajući $k = 0, 1, \ldots$ i $m = 1, 2, \ldots$. U prvoj koloni ove tabele nalaze se redom približne vrednosti integrala I dobijene primenom trapezne formule sa $h_k = (b-a)/2^k$ $(k = 0, 1, \ldots)$. Druga kolona ove tabele dobija se na osnovu prve, korišćenjem formule (1.4.1), treća na osnovu druge, itd.

Iterativni proces, definisan sa (1.4.1) predstavlja standardni Rombergov metod za numeričku integraciju. Može se dokazati da nizovi $\{T_k^{(m)}\}_{k\in N_0}$ i $\{T_k^{(m)}\}_{m\in N_0}$ (po kolonama i vrstama u T-tabeli) konvergiraju ka I. Kod praktične primene Rombergove integracije, iterativni proces (1.4.1) se najčešće prekida kada je

$|T_0^{(m)} - T_0^{(m-1)}| \leq \varepsilon$, gde je ε unapred data dozvoljena greška, i tada se uzima $I \cong T_0^{(m)}$.

II.1.5 Programska realizacija

U ovom odeljku dajemo programsku realizaciju Simpsonove i Rombergove integracije.

Program 1.5.1.

Za integraciju po uopštenoj Simpsonovoj formuli realizovan je potprogram INTEG. Parametri u listi imaju značenje koje je objašnjeno `C`-komentarima u potprogramu. Podintegralna funkcija se zadaje u potprogramu `FUN`, i može zavisiti od jednog parametra `Z`. Celobrojnim parametrom `J` je obezbedjeno istovremeno zadavanje više podintegralnih funkcija.

Potprogram `INTEG` je organizovan tako da se početni broj podsegmenata može povećati (redukcijom koraka h na $h/2$) do `MAX=1000`. U slučaju kada je relativna razlika u vrednosti integrala, dobijenog korakom h i korakom $h/2$, manja od 10^{-5}, računski proces se prekida i vrednost integrala izračunata sa dotad najmanjim korakom se uzima kao definitivna vrednost integrala. Ukoliko se ovaj kriterijum ne može ispuniti sa manje od `MAX` podsegmenata daje se poruka `KBR=1` (u protivnom je `KBR=0`).

Za testiranje ovog potprograma uzeli smo izračunavanje

sledećih integrala:

$$\int_0^1 \frac{e^{2x}}{x^2+z^2}dx \qquad (z = 1.0(0.1)1.5),$$

$$\int_0^{1/2} \pi \sin(\pi z x)dx \qquad (z = 1.0(0.2)1.4)$$

$$\int_1^2 \frac{\log(x+z)}{z^2+e^x}\frac{\sin x}{x}dx \qquad (z = 0.0(0.1)0.5)$$

.

Potprogrami, glavni program i izlazna lista imaju oblik:

```
C=======================================================
C IZRACUNAVANJE ODREDENOG INTEGRALA FUNKCIJE F(X,Z,J)
C SIMPSONOVOM  FORMULOM
C=======================================================
      SUBROUTINE INTEG(A, B, S, F, J, KBR, Z)
C  A - DONJA GRANICA INTEGRALA
C  B - GORNJA GRANICA INTEGRALA
C  S - VREDNOST INTEGRALA SA TACNOSCU EPS=1.E-5
C  KBR - KONTROLNI BROJ
C  KBR=0 INTEGRAL KOREKTNO IZRACUNAT
C  KBR=1 INTEGRAL NIJE IZRACUNAT SA ZAHTEVANOM TACNOSCU
C Z - PARAMETAR U PODINTEGRALNOJ FUNKCIJI
C POCETNI BROJ PODEOKA JE 2*MP,A MAKSIMALNI MAX=1000
      MP=15
      MAX=1000
      KBR=0
      N=2.*MP
      S0=0.
      SAB=F(A,Z,J)+F(B,Z,J)
      H=(B-A)/FLOAT(N)
      X=A
      S1=0.
      N2=N-2
      DO 5 I=2, N2, 2
      X=X+2.*H
5     S1=S1+F(X,Z,J)
```

```
10      S2=0.
        X=A-H
        N1=N-1
        DO 15 I=1, N1, 2
        X=X+2.*H
15      S2=S2+F(X,Z,J)
        S=H/3.*(SAB+2.*S1+4.*S2)
        REL=(S-S0)/S
        IF (ABS(REL)-1.E-5) 35,35,20
20      IF (N-MAX) 25,25,30
25      N=2*N
        H=0.5*H
C  BROJ PODEOKA SE UVECAVA DVA PUTA I
C IZRACUNAVA SE NOVA VREDNOST ZA S1
        S1=S1+S2
        S0=S
        GO TO 10
30      KBR=1
35      RETURN
        END
        FUNCTION FUN(X,Z,J)
        GO TO (10,20,30),J
10      FUN=EXP(Z*X)/(X*X+Z*Z)
        RETURN
20      PI=3.1415926535
        FUN=PI*SIN(PI*X*Z)
        RETURN
30      FUN=ALOG(X+Z)/(Z*Z+EXP(X))*SIN (X)/X
        RETURN
        END
        EXTERNAL FUN
        OPEN(8,File='Simpson.IN')
        OPEN(6,File='Simpson.out')
        WRITE(6,5)
5       FORMAT (1H1,2X, 'IZRACUNAVANJE VREDNOSTI INTEGRALA',
     1 ' PRIMENOM SIMPSONOVE FORMULE ' //14X,
     2 'TACNOST IZRACUNAVANJA EPS=1.E-5'
     3 ///11X,'J',4X,'DONJA',5X,'GORNJA',3X,'PARAMETAR',
     4 3X,' VREDNOST'/ 16X, 'GRANICA', 3X,'GRANICA',
     5  5X,'Z',7X,'INTEGRALA'//)
        DO 40 J=1,3
        READ (8,15) DG, GG, ZP, DZ, ZK
15      FORMAT(5F5.1)
        Z=ZP-DZ
```

```
18      Z=Z+DZ
        IF (Z.GT.ZK+0.000001) GO TO 40
        CALL INTEG (DG,GG,S,FUN,J,KBR,Z)
        IF(KBR) 20,25,20
20      WRITE (6,30)
30      FORMAT (/11X, 'INTEGRAL NIJE KOREKTNO IZRACUNAT'/)
        GO TO 18
25      WRITE (6,35) J,DG,GG,Z,S
35      FORMAT (11X,I1,F8.1,2F10.1,F15.6/)
        GO TO 18
40      CONTINUE
        STOP
        END


    0.,1.,1.,0.1,1.5
    0.,0.5,1.,0.2,1.4
    1.,2.,0.,0.1,0.5


IZRACUNAVANJE VREDNOSTI INTEGRALA PRIMENOM SIMPSONOVE FORMULE
           TACNOST IZRACUNAVANJA EPS=1.E-5
        J    DONJA     GORNJA   PARAMETAR    VREDNOST
             GRANICA   GRANICA     Z        INTEGRALA
        1      .0        1.0       1.0       1.270724
        1      .0        1.0       1.1       1.153890
        1      .0        1.0       1.2       1.059770
        1      .0        1.0       1.3        .983069
        1      .0        1.0       1.4        .920013
        1      .0        1.0       1.5        .867848
        2      .0         .5       1.0       1.000000
        2      .0         .5       1.2       1.090848
        2      .0         .5       1.4       1.134133
        3     1.0       2.0        .0         .048047
        3     1.0       2.0        .1         .059595
        3     1.0       2.0        .2         .069940
        3     1.0       2.0        .3         .079052
        3     1.0       2.0        .4         .086920
        3     1.0       2.0        .5         .093558
```

Program 1.5.2.

Sada dajemo programsku realizaciju Rombergove integracije (videti odeljak II.1.4) u dvostrukoj tačnosti. Lista u potprogramu `ROMBI` ima sledeće značenje:

`DG` - donja granica integrala:

`GG` - gornja granica integrala;

`FUN` - ime funkcijskog potprograma kojim se definiše podintegralna funkcija;

`EPS` - zahteva tačnost izračunavanja;

`VINT` - vrednost integrala sa tačnošću `EPS`, ukoliko je `KB=0`;

`KB` - kontrolni broj (`KB=0` integral korektno izračunat; `KB=1` tačnost izračunavanja integrala nije postignuta sa `15` predvidjenih koraka, tj. sa brojem podsegmenata 2^{15}). Za testiranje ovog potprograma uzeto je tabelianje funkcije

$$F(x) = \int_0^x e^{-t^2} dt \qquad (x = 0.1(0.1)1.0),$$

sa tačnošću 10^{-5} . Programi i izlazna lista imaju oblik:

```
C=============================================================
C               ROMBERGOVA INTEGRACIJA
C=============================================================
      DOUBLE PRECISION GG, FUN, VINT
      EXTERNAL FUN
      open(6,file='romberg.out')
EPS=1.E-8
      WRITE (6,11)
11    FORMAT(1H0,5X,'X',7X,'INTEGRAL(0.,X)'/)
      DO 10 I=1, 10
      GG=0.1*I
      CALL ROMBI(0.D0,GG,FUN,EPS,VINT,KB)
      IF (KB) 5,15,5
  5   WRITE (6,20) GG
20    FORMAT (5X,F3.1,4X,'TACNOST NE ZADOVOLJAVA'//)
      GO TO 10
15    WRITE(6,25)GG,VINT
25    FORMAT(5X,F3.1,4X,F14.9)
10    CONTINUE
```

```
      STOP
      END
      SUBROUTINE ROMBI (DG,GG,FUN,EPS,VINT,KB)
      DOUBLE PRECISION FUN,VINT,T(15),DG,GG,H,A,POM,B,X
      KB=0
      H=GG-DG
      A=(FUN(DG)+FUN(GG))/2.
      POM=H*A
      DO 50 K=1, 15
      X=DG+H/2.
10    A=A+FUN (X)
      X=X+H
      IF (X.LT.GG) GO TO 10
      T(K)=H/2.*A
      B=1.
      IF (K.EQ.1) GO TO 20
      K1=K-1
      DO 15 M=1, K1
      I=K-M
      B=4.*B
15    T(I)=(B*T(I+1)-T(I))/(B-1.)
20    B=4.*B
      VINT=(B*T(1)-POM)/(B-1.)
      IF(DABS(VINT-POM).LE.EPS) RETURN
      POM=VINT
50    H=H/2.
      KB=1
      RETURN
      END
      FUNCTION FUN(X)
      DOUBLE PRECISION FUN,X
      FUN=DEXP(-X*X)
      RETURN
      END
```

```
0     X       INTEGRAL(0.,X)
      .1        .099667666
      .2        .197365034
      .3        .291237887
      .4        .379652845
      .5        .461281012
      .6        .535153533
      .7        .600685674
      .8        .657669863
```

```
 .9          .706241521
1.0          .746824138
```

II.1.6 O numeričkom izračunavanju jedne klase dvostrukih integrala

U ovom odeljku ukazaćemo na jedan način za približno izračunavanje dvostrukih integrala oblika

$$\iint_G f(x,y)\ dxdy, \tag{1.6.1}$$

gde je oblast integracije jedinični krug, tj. $G = \{(x,y)|x^2+y^2 \leq 1\}$. Naime, za numeričko izračunavanje integrala (1.6.1) u literaturi je poznata formula

$$\iint_G f(x,y)\ dxdy \cong \frac{\pi}{8}(2f(0) + \sum_{i=1}^{n} f(M_i)), \tag{1.6.2}$$

gde O predstavlja koordinatni početak, tj. $0 = (0,0)$, dok tačke M_i imaju polarne koordinate

$$r_i = \sqrt{\frac{2}{3}}, \quad \Phi = \frac{\pi}{3}(i-1) \quad (i = 1,2,\ldots,6).$$

.

Prema formuli (1.6.2) realizovaćemo program za izračunavanje dvostrukih integrala, gde je oblast integracije jedinični krug. Organizacija programa biće takva da se funkcijskim potprogramom `EF` mogu definisati više različitih podintegralnih funkcija f. Parametri u listi imaće sledeće značenje:

`X` - vrednost argumenta x;

`Y` - vrednost argumenta y;

`K` - celobrojni parametar kojim se definišu različite podintegralne funkcije.

Formula (1.6.2) realizovana je kroz potprogram DVINT, čiji parametri u listi imaju sledeće značenje:

EF - ime funkcijskog potprograma;

K - celobrojni parametar, kao u potprogramu EF;

VRINT - izračunata vrednost integrala, prema kubaturnoj formuli (1.6.2).

```
      SUBROUTINE DVINT(EF, K,VRINT)
      PI=3.1415926535
      RO=SQRT(2./3)
      PI3=PI/3
      FI=-PI3
      VRINT=2.*EF(0.,0.,K)
      DO 10 I=1,6
      FI=FI+PI3
      X=RO*COS(FI)
      Y=RO*SIN(FI)
10    VRINT=VRINT+EF(X,Y,K)
      VRINT=PI/8.*VRINT
      RETURN
      END
```

Glavni program ima oblik:

```
C=================================================
C      IZRACUNAVANJE DVOSTRUKOG INTEGRALA
C=================================================
      EXTERNAL EF
      OPEN(6,FILE='DVINT.OUT')
      WRITE (6,5)
5     FORMAT (1H1//10X,'IZRACUNAVANJE DVOSTRUKOG',
     1' INTEGRALA'//)
      DO 10 K=1,3
      CALL DVINT(EF, K, VRINT)
10    WRITE (6,15)K,VRINT
15    FORMAT (15X,I1,' PRIMER'// 10X,
     1 'VREDNOST INTEGRALA =',F12.6//)
      STOP
      END
```

Primenom ovog programa približno smo izračunali vrednosti sledećih integrala:

$$1^0 \qquad \iint\limits_G \frac{16x^2y^2}{1+x^2+y^2}\,dxdy;$$

$$2^0 \qquad \iint\limits_G \sqrt{1+(1+x)^2+y^2}\,dxdy;$$

$$3^0 \qquad \iint\limits_G \frac{24x^2}{\sqrt{2-x^2-y^2}}\,dxdy.$$

Funkcijski potprogram `EF` i izlazna lista imaju oblik:

```
      FUNCTION EF(X,Y,K)
GO TO (10,20,30),K
10    EF=(16.*X*X*Y*Y)/(1.+X*X+Y*Y)
      RETURN
20    EF=SQRT(1.+Y*Y+(1.+X)**2)
      RETURN
30    EF=(24.*X*X)/SQRT(2.-X*X-Y*Y)
      RETURN
      END
```

```
IZRACUNAVANJE DVOSTRUKOG INTEGRALA
     1 PRIMER
VREDNOST INTEGRALA =    1.256637
     2 PRIMER
VREDNOST INTEGRALA =    4.858376
     3 PRIMER
VREDNOST INTEGRALA =   16.324200
```

IV. NUMERICAL SOFTWARE

Computer Aided Solution

Axiom - Axiom User Guide, NAG, Downers Grove, IL

Derive - DERIVE, Soft Warehouse, Honolulu, HI

Macsyma - VAX UNIX MACSYMA, Reference Manual, Symbolics Inc., Cambridge, MA

MAPLE - MAPLE V Library Reference Manual, Springer, NY, 1991

Mathematica - A System of Doing Mathematica by Computer, Addison-Wesley, Reading, MA

Matlab -

REDUCE - Software for Algebraic Computation, Springer, NY, 1987.

Standard test examples (Indefinite integrals):

$1^0 \quad \int \sin x \, dx;$

$2^0 \quad \int \sqrt{\tan x} \, dx;$

$3^0 \quad \int \frac{x}{x^3 - 1} \, dx;$

$4^0 \quad \int \frac{x}{\sin^2 x} \, dx;$

$5^0 \quad \int \frac{\log x}{\sqrt{x+1}} \, dx;$

6^0 $\displaystyle\int \frac{x}{\sqrt{1+x}+\sqrt{1-x}}\,dx;$

7^0 $\displaystyle\int e^{-ax^2}\,dx;$

8^0 $\displaystyle\int \frac{x}{\log^3 x}\,dx;$

9^0 $\displaystyle\int \frac{\sin x}{x^2}\,dx;$

10^0 $\displaystyle\int \frac{1}{2+\cos x}\,dx;$

Standard test examples (Definite integrals):

1^0 $\displaystyle\int_0^{4\pi} \frac{1}{2+\cos x}\,dx;$

2^0 $\displaystyle\int_{-\infty}^{\infty} \frac{\sin x}{x}\,dx;$

3^0 $\displaystyle\int_0^{\infty} \frac{e^{-x}}{\sqrt{x}}\,dx;$

4^0 $\displaystyle\int_0^{\infty} \frac{x^2 e^{-x}}{1-e^{-2x}}\,dx;$

5^0 $\displaystyle\int_0^{\infty} e^{-x^2}\log^2 x\,dx;$

6^0 $\displaystyle\int_1^{\infty} e^{-x} x^3 \log^2 x\,dx;$

7^0 $\displaystyle\int_0^{\infty} \frac{x^2}{1+x^3}\,dx;$

8^0 $\displaystyle\int_{-1}^{1} \frac{1}{x^2}\,dx;$

9^0 $\displaystyle\int_1^{\infty} e^{-x} x^{11/3}\,dx;$

Software Libraries

CMLIB- Collection of code from many sources: CDRIV and SDRIV (Kahaner, *Numerical Methods and Software*, Prentice-Hall, Englewood Cliffs, NJ, 1989., DEPACK, FISHPACK, VHS3;

ACM - Collected algorithms;

IMSL- Houston, TX;

NAG- Numerical Algorithms Group, Downers Grove, IL;

NMS - NIST name,*Numerical Methods and Software*, Prentice-Hall, Englewood Cliffs, NJ, 1989.

PORT - *The PORT Mathematical Subroutine Library Manual*, Bell Laboratories, Murray Hill, NJ, 1989;

Sci. Desk - Scientific Desk, NIST, Gaithersburg, MD;

SCRUNCH - Old, unsupported code in BASIC, G. Forsythe, et all., *Computer Methods for Mathematical Computations*, Prentice Hall, Englewood Cliffs, NJ, 1989.

QUADPACK - R. Piessens, et all., *QUADPACK, A Subroutine Package for Automatic Integration*, Springer, Berlin, 1983;

CUBTRI - Cubature Formulae Over Triangle;

SSP - IBM Numerical Software.

II.2 INTEGRALNE JEDNAČINE

II.2.1 Introduction

Jednačina

$$y(x) = f(x) = \lambda \int_a^b K(x,t)y(t)\,dt, \tag{2.1.1}$$

gde su f i K poznate funkcije, y nepoznata funkcija i λ numerički parametar, naziva se Fredholmova integralna jednačina druge vrste.

Funkcija K se naziva jezgro integralne jednačine (2.1.1). U našim razmatranjima pretpostavljaćemo uvek da je jezgro definisano i neprekidno na $D = \{(x,t)|a \leq x \leq b,\ a \leq t \leq b\}$.

Ako je $f(x) \not\equiv 0$, jednačina (2.1.1) se naziva nehomogenom, a u slučaju kada je $f(x) \equiv 0$, jedna ina je homogena.

Integralna jednačina oblika

$$f(x) + \lambda \int_a^b K(x,t)y(t)\,dt = 0$$

naziva se Fredholmova integralna jednačina prve vrste.

Volterraove integralne jednačine prve i druge vrste imaju oblike

$$f(x) + \lambda \int_a^x K(x,t)y(t)\,dt = 0$$

i

$$y(x) = f(x) + \lambda \int_a^x K(x,t)y(t)\,dt\,,$$

respektivno.

Za rešavanje Fredholmove jednačine (2.1.1) često se koristi metod sukcesivnih aproksimacija koji se zasniva na jednakosti

$$y_n(x) = f(x) + \lambda \int_a^b K(x,t) y_{n-1}(t)\, dt \quad (n = 1, 2, \ldots), \tag{2.1.2}$$

pri čemu se uzima $y_0 = f(x)$. Naime, ako definišemo niz funkcija $\{\overline{y}_k\}$ pomoću

$$\overline{y}_0(x) = y_0(x) = f(x),\ \overline{y}_k(x) = \int_a^b K(x,t)\overline{y}_{k-1}(t)\, dt \quad (k = 1, 2 \ldots).$$

tada se (2.1.2) može predstaviti u obliku

$$y_n(x) = \sum_{k=0}^{n} \lambda^k \overline{y}_k(x) \quad (n = 1, 2, \ldots). \tag{2.1.3}$$

Može se pokazati da niz y_n konvergira ka tačnom rešenju jednačine (2.1.1) ako je ispunjen uslov $|\lambda| < \dfrac{1}{M(b-a)}$ gde je

$$M = \max_{x,t \in [a,b]} |K(x,t)|$$

.

II.2.2 Primena kvadraturnih formula na rešavanje Fredholmove integralne jednačine druge vrste

U cilju rešavanja jednačine (2.2.1) uzmimo kvadraturnu formulu

$$\int_a^b F(x)\, dx = \sum_{j=1}^{n} A_j F(x_j) + R_n(F), \tag{2.2.1}$$

gde su apscise $x-1,\ldots,x_n$ iz $[a,b]$, A_j težinski koeficijenti koji ne zavise od F i $R_n(F)$ odgovarajući ostatak.

Ako u (2.1.1) stavimo redom $x=x_i$ $(i=1,\ldots,n)$, imamo

$$y(x_i)=f(x_i)+\lambda\int_a^b K(x,t)y(t)\,dt \quad (1=1,\ldots,n),$$

odakle primenom kvadraturne formule (2.2.1) sleduje

$$y(x_i)=f(x_i)+\lambda\sum_{k=1}^{n}A_jK(x_i,x_j)\,y(x_j)+R_n(F) \qquad i=1,\ldots,n), \tag{2.2.2}$$

gde je $F_i(t)K(x_i,t)\,y(t)$ $(i=1,\ldots,n)$. Odbacivanjem članova $R_n(F_i)$ $(i=1,\ldots,n)$, na osnovu (2.2.2) dobijamo sistem linearnih jednačina

$$y_i-\lambda\sum_{i=1}^{n}A_jK_{ij}y_j=f_i \quad (i=1,\ldots,n) \tag{2.2.3}$$

gde smo stavili $y_i=y(x_i),\ f_i=f(x_i),\ K_{ij}=K(x_i,x_j)$. Sistem (2.2.3) se može predstaviti i u matričnom obliku

$$\begin{bmatrix} 1-\lambda A_1K_{11} & -\lambda A_2K_{12} & \ldots & -\lambda A_nK_{1n} \\ -\lambda A_1K_{21} & 1-\lambda A_2K_{22} & \ldots & -\lambda A_nK_{2n} \\ \vdots & & & \\ -\lambda A_1K_{n1} & -\lambda A_2K_{n2} & \ldots & -\lambda 1-A_nK_{nn} \end{bmatrix}\cdot\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}=\begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_n \end{bmatrix}.$$

Rešavanjem dobijenog sistema linearnih jednačina po y_1,..., y_n, približno rešenje jednačine (2.1.1) se može predstaviti u obliku

$$\tilde{y}(x)=f(x)+\lambda\sum_{j=1}^{n}A_jK(x,x_j)\,y_j. \tag{2.2.4}$$

II.2.3 Programska realizacija

Metod izložen u prethodnom odeljku realizovaćemo korišćenjem uopštene Simpsonove kvadraturne formule, kod koje je

$$h = \frac{b-a}{2m},\ n = 2m+1,\ x_i = a + (i-1)h \quad (i = 1, \ldots, n),$$

$$A_1 = A_{2m+1} = \frac{h}{3},\ A_2 = A_4 = \ldots = A_{2m} = \frac{4h}{3},$$

$$A_3 = A_5 = \ldots = A_{2m-1} = \frac{2h}{3}.$$

Za rešavanje sistema linearnih jednačina (2.2.3) koristićemo potprograme `LRFAK` i `RSTS` iz I.4.6 (poglavlje I.4), sledećeg izgleda:

```
      SUBROUTINE LRFAK(A,N,IP,DET,KB)
      DIMENSION A(1),IP(1)
      KB=0
      N1=N-1
      INV=0
      DO 45 K=1,N1
      IGE=(K-1)*N+K
C
C  FINDING THE PIVOTAL ELEMENT IN  K-TH
C  ELIMINATION STEP
C
      GE=A(IGE)
      I1=IGE+1
      I2=K*N
      IMAX=IGE
      DO 20 I=I1,I2
      IF(ABS(A(I))-ABS(GE)) 20,20,10
 10   GE=A(I)
      IMAX=I
 20   CONTINUE
      IF(GE)25,15,25
 15   KB=1
C
C  MATRIX OF SYSTEM IS SINGULAR
C
```

```
      RETURN
 25   IP(K)=IMAX-N*(K-1)
      IF(IP(K)-K) 30,40,30
 30   I=K
      IK=IP(K)
C
C  ROW PERMUTATION
C
      DO 35 J=1,N
      S=A(I)
      A(I)=A(IK)
      A(IK)=S
      I=I+N
 35   IK=IK+N
      INV=INV+1
C
C  K-TH ELIMINATION STEP
C
 40   DO 45 I=I1,I2
      A(I)=A(I)/GE
      IA=I
      IC=IGE
      K1=K+1
      DO 45 J=K1,N
      IA=IA+N
      IC=IC+N
 45   A(IA)=A(IA)-A(I)*A(IC)
C
C  CALCULATION OF DETERMINANT
C
      DET=1.
      DO 50 I=1,N
      IND=I+(I-1)*N
 50   DET=DET*A(IND)
      IF(INV-INV/2*2) 55,55,60
 60   DET=-DET
 55   RETURN
      END
C
C
      SUBROUTINE RSTS(A,N,IP,B)
      DIMENSION A(1),IP(1),B(1)
C
C  SUCCESSIVE SOLUTION OF TRIANGULAR SYSTEMS
```

```
C
      N1=N-1
C  VECTOR B PERMUTATION
      DO 10 I=1,N1
      I1=IP(I)
      IF(I1-I) 5,10,5
 5    S=B(I)
      B(I)=B(I1)
      B(I1)=S
 10   CONTINUE
C  SOLUTION OF LOWER TRIANGULAR SYSTEM
      DO 15 K=2,N
      IA=-N+K
      K1=K-1
      DO 15 I=1,K1
      IA=IA+N
 15   B(K)=B(K)-A(IA)*B(I)
C  SOLUTION OF UPPER TRIANGULAR SYSTEM
      NN=N*N
      B(N)=B(N)/A(NN)
      DO 25 KK=1,N1
      K=N-KK
      IA=NN-KK
      I=N+1
      DO 20 J=1,KK
      I=I-1
      B(K)=B(K)-A(IA)*B(I)
 20   IA=IA-N
 25   B(K)=B(K)/A(IA)
      RETURN
      END
```

Parameters in subprogram list of `LRFAK` are of following meaning:

`A` - Ulazna matrica reda `N` memorisana kao niz kolona po kolona. Posle `N-1` eliminacionih koraka matrica `A` se transformiše u matricu koja sadrži trougaone matrice `L` i `R` (videti odeljak I.2.4);

`N` - red matrice A;

IP - vektor dužine N-1, koji se formira u procesu eliminacije i predstavlja niz indeksa glavnih elemenata (videti odeljak I.2.4);

DET - izlazna veličina koja daje vrednost determinante matrice sistema A, kao proizvod elemenata na glavnoj dijagonali u matrici R, sa tačnošću do na znak. Ova vrednost se koriguje znakom, na kraju potprograma, imajući u vidu broj permutacija vrsta u matrici u toku eliminacionog procesa;

KB - kontrolni broj sa vrednostima KB=0 ako je faktorizacija korektno izvedena i KB=1 ako je matrica sistema singularna. U poslednjem slučaju LR faktorizacija ne egzistira;

Potprogram RSTS sukcesivno rešava sisteme jednačina (2.3.4). Parametri potprogramskoj listi imaju sledeće značenje:

A - matrica dobijena u potprogramu LRFAK;

N - red matrice A;

IP - vektor dobijen u potprogramu LRFAK;

B - vektor slobodnih članova u sistemu jednačina koji se rešava. Ovaj vektor se transformiše u vektor rešenja datog sistema.

Glavni program je organizovan tako da se, najpre, data matrica A faktorizuje, pomoću potprograma LRFAK, a zatim je mogućno rešiti sistem jednačina $\mathbf{A}\vec{x} = \vec{b}$ za proizvoljan broj različitih vektora $\vec{b}$, pozivanjem potprograma **RSTS**.

U potprogramu FRED formira se sistem jednačina (2.2.3). Parametri u listi ovog potprograma imaju sledeće značenje:

X - vektor apscisa kvadraturne formule;

A - vektor težinskih koeficijenata kvadraturne formule;

FK - ime funkcijskog potprograma kojim se zadaje funkcija f i jezgro K;

PL - vrednost parametra λ;

C - matrica sistema (2.2.3), memorisana kao vektor (kolona po kolona);

F - vektor slobodnih članova u sistemu jednačina (2.2.3).

```
      SUBROUTINE FRED(X,A,N,FK,PL,C,F)
      DIMENSION X(1), A(1),C(1),F(1)
      IND=-N
      DO 15 J=1,N
      IND=IND+N
      DO 10 I=1,N
      IJ=IND+I
      C(IJ)=-PL*A(J)*FK(X(I),X(J),2)
      IF(I-J)10,5,10
5     C(IJ)=1+C(IJ)
10    CONTINUE
15    F(J)=FK(X(J),X(I),1)
      RETURN
      END
```

Funkcijski potprogram FK ima sledeće parametre u listi:

X i T - vrednosti argumenata x i t respektivno;

M - celobrojni parametar, koji definiše izračunavanje vrednosti funkcije f (M=1) i izračunavanje vrednosti jezgra K (M=2) pri zadatim vrednostima argumenata.

```
      FUNCTION FK(X,T,M)
      GO TO (10,20), M
10    FK=EXP(X)
      RETURN
 20   FK=X*EXP(X*T)
      RETURN
      END
```

Uzimajući kao primer jednačinu

$$y(x) = e^x - \int_0^1 x e^{xt}\, y(t)\, dt$$

i M=1,2 (N=3,5) dobijeni su rezultati koje navodimo iza odgovarajućeg programa. Primetimo da je tačno rešenje date jednačine $y(x) = 1$.

```
      EXTERNAL FK
      DIMENSION X(10), A(10), C(100),B(10),IP(9)
      OPEN(8,FILE='FRED.IN')
      OPEN(5,FILE='FRED.OUT')
      READ(8,5)PL,DG,GG
 5    FORMAT(3F5.0)
 10   READ(8,15,END=60) M
15    FORMAT(I2)
      N=2*M+1
      H=(GG-DG)/(2.*FLOAT(M))
      X(1)=DG
      DO 20 I=2,N
 20   X(I)=X(I-1)+H
      Q=H/3.
      A(1)=Q
      A(N)=Q
      DO 25 I=1,M
25    A(2*I)=4.*Q
      DO 30 I=2,M
30    A(2*I-1)=2.*Q
      CALL FRED(X,A,N,FK,PL,C,B)
      CALL LRFAK(C,N,IP,DET,KB)
      IF(KB) 35,40,35
 35   WRITE(5,45)
 45   FORMAT(1H0,'MATRICA SISTEMA SINGULARNA'//)
      GO TO 60
40    CALL RSTS(C,N,IP,B)
      WRITE(5,50)(B(I),I=1,N)
 50   FORMAT(/5X,'RESENJE'//(10F10.5))
      GO TO 10
60    CLOSE(5)
      CLOSE(8)
      STOP
      END
```

```
  RESENJE
1.00000   1.70929   2.88593
  RESENJE
1.00000   1.30995   1.70919   2.22328   2.88509
```

II.3 OBIČNE DIFERENCIJALNE JEDNAČINE

II.3.1 Introduction

Ovo poglavlje je posvećeno, uglavnom, rešavanju Cauchyevog problema kod običnih diferencijalnih jednaina, tj. problema sa početnim uslovima. Metodi su razvrstani u dve opšte klase i to:

1. Klasa linearnih višekoračnih metoda,
2. Klasa metoda Runge-Kutta.

Takodje, ukazaćemo i na rešavanje konturnih problema kod običnih diferencijalnih jednačina.

II.3.2 Eulerov metod

Eulerov metod je najprostiji numerički metod za rešavanje Caucyhyevog problema

$$y' = f(x, y), \quad y(x_o) = y_0 \tag{3.2.1}$$

i bazira se na približnoj jednakosti

$$y(x) = y(x_0) + (x - x_0)y'(x_0),$$

tj.

$$y(x) = y(x_0) + (x - x_0)f(x_0, y_0), \tag{3.2.2}$$

s obzirom na (3.2.1). Ako sa y_1 označimo približnu vrednost za $y(x_1)$, na osnovu (3.2.2) imamo

$$y_1 = y_0 + (x_1 - x_0)f(x_0, y_0).$$

U opštem slučaju, za proizvoljan skup tačaka $x_0 < x_1 < x_2 < \ldots$, približne vrednosti za $y(x_n)$, u oznaci y_n, možemo odrediti pomoću

$$y_{n+1} = y_n + (x_{n+1} - x_n) f(x_n, y_n) \quad (n = 0, 1, \cdots). \tag{3.2.3}$$

Poslednja formula definiše Eulerov metod, čija je geometrijska interpretacija data na sl. 3.2.1.

Sl. 3.2.1

Poligonalna linija $(x_0, y_0) - (x_1, y_1) - (x_2, y_2) - \ldots$ poznata je kao Eulerov poligon.

Najčešće se tačke biraju ekvidistantno, tj. $x_{n+1} - x_n = h = const.(> 0)$ $(n = 0, 1, \ldots)$ i u tom sluča ju (3.2.3) se svodi na

$$y_{n+1} = y_n + h f(x_n, y_n) \qquad (n = 0, 1, \ldots).$$

II.3.3 Opšti linearni višekoračni metod

U ovom i narednim odeljcima ovog poglavlja razmatraćemo jednu opštu klasu metoda za rešavanje Cauchyevog problema

$$y' = f(x, y), \quad y(x_0) = y_0 \quad (x_0 \leq x \leq b). \tag{3.3.1}$$

Ako segment $[x_0, b]$ podelimo na N podsegmenata dužine $h = \frac{b - x_0}{N}$, dobijamo niz tačaka x_n odredjen sa

$$x_n = x_0 + nh \quad (n = 0, 1, \ldots, N).$$

Neka y_n označava niz približnih vrednosti rešenja problema (3.3.1) u tačkama x_n i neka je $f_n \equiv f(x_n, y_n)$. Pred nas se postavlja problem odredj ivanja niza y_n. Za rešavanje ovog problema razradjen je veliki broj metoda. Jedan od ovih metoda je i Eulerov metod koji je razmatran u prethodnom odeljku. Kod Eulerovog metoda niz y_n se izrač unava rekurzivno pomoću

$$y_{n+1} - y_n = hf_n \quad (n = 0, 1, \ldots, N), \tag{3.3.2}$$

pri čemu postoji linearna veza izmedju y_n, y_{n+1} i f_n. U opštem slučaju za izračunavanje niza mogu se koristiti složenije rekurentne relacije, nego što je (3.3.2). Medju metodima koji proističu iz ovih relacija, važnu ulogu igraju metodi kod kojih postoji linearna veza izmedju $y_{n+i}, f_{n+i} \quad (i = 0, 1, \ldots k)$ i oni čine klasu tzv. linearnih višekoračnih metoda (multi-step methods).

Opšti linearni višekoračni metod može se predstaviti u obliku

$$\sum_{i=0}^{k} \alpha_i y_{n+1} = h \sum_{i=0}^{k} \beta f_{n+i} \quad (n = 0, 1, \ldots), \tag{3.3.3}$$

gde su α i β konstantni koeficijenti odredjeni sa tačnošću do na multiplikativnu konstantu. Da bismo obezbedili njihovu jednoznačnost uzećemo $\alpha_k = 1$.

Ako je $\beta_k = 0$, kažemo da je metod (3.3.3) otvorenog tipa ili da je eksplicitan; u protivnom metod je zatvorenog tipa ili implicitan.

U opštem slučaju (3.3.3) predstavlja nelinearnu diferencnu jednačinu, s obzirim da je $f_{n+i} \equiv f(x_{n+i}, y_{n+i})$.

Za odre—ivanje niza y_n primenom metoda (3.3.3) potrebno je poznavanje startnih vrednosti y_i $(i = 0, 1, \ldots, k-1)$. Kako nam je unapred poznata jedino vrednost y_0, poseban problem u primeni višekoračnih metoda (3.3.3) predstavlja odredjivanje ostalih startnih vrednosti. Ovom problemu biće posvećen poseban odeljak.

Pod pretpostavkom da su poznate startne vrednosti y_i $(i = 0, 1, \ldots, k-1)$, kod eksplicitnih metoda direktno se izračunavaju $y_k, y_{k+1}, \ldots, y_N$ pomoću

$$y_{n+k} = h\sum_{i=0}^{k-1} \beta_i f_{n+i} - \sum_{i=0}^{k-1} \alpha_i y_{n+i} \quad (n = 0, 1, \ldots, N-k).$$

Medjutim, kod implicitnih metoda za odredjivanje vrednosti y_{n+k} treba rešiti jednačinu

$$y_{n+k} = h\beta f(x_{n+k}, y_{n+k}) + \Phi, \tag{3.3.4}$$

gde je

$$\Phi = h\sum_{i=0}^{k-1} \beta_i f_{n+i} - \sum_{i=0}^{k-1} \alpha_i y_{n+i}.$$

Kada je $(x, y) \to f(x, y)$ nelinearna funkcija koja zadovoljava Lipschitzov uslov po y sa konstantom L, jednačina (3.3.4) se može rešiti iterativnim procesom

$$y_{n+k}^{[s+1]} = h\beta_k f(x_{n+k}, y_{n+k}^{[s]}) + \Phi, \tag{3.3.5}$$

polazeći i od proizvoljne vrednosti $y_{n+k}^{[0]}$ ako je

$$h|\beta_k|L < 1.$$

Uslov dat ovom nejednakošću obezbedjuje konvergenciju iterativnog procesa (3.3.5). Za metod (3.3.3) definišimo diferencni operator $L_h : C^1[x_0, b] \to C[x_0, b]$ pomoću

$$L_h[y] = \sum_{i=0}^{k} [\alpha_i y(x + ih) - h\beta_i y'(x + ih)]. \tag{3.3.6}$$

Neka funkcija $g \in C^1[x_0, b]$. Tada se $L_h[g]$ može predstaviti u obliku

$$L_h[g] = C_0 g(x) + C_1 h g'(x) + c_2 h^2 g''(x) + \cdots, \tag{3.3.7}$$

gde su $C_j (j = 0, 1, \ldots)$ konstante, koje ne zavise od h i g.

Definicija 3.3.1. *Linearni višekoračni metod* (3.3.3) *ima red* p *ako je u razvoju* (3.3.7)

$$C_0 = C_1 = \ldots = C_p = 0 \text{ i } C_{p+1} \neq 0.$$

Neka je $x \to y(x)$ tačno rešenje problema (3.3.1) i y_n niz približnih vrednosti ovog rešenja u tačkama $x_n = x_0 + nh$ $n = 0, 1, \ldots, N)$ dobijen primenom metoda (3.3.3), sa startnim vrednostima $y_i = s_i(h)$ $(i = 0, 1, \ldots, k-1)$.

Definicija 3.3.2. *Za linearni višekoračni metod* (3.3.3) *se kaže da je konvergentan ako je za svako* $x \in [x_0, b]$

$$\lim_{\substack{x \to 0 \\ x - x_0 = nh}} y_n = y(x)$$

i ako za startne vrednosti važi

$$\lim_{h \to 0} s_i(h) = y_0 \quad (i = 0, 1, \ldots, k-1).$$

Linearni višekoračni metod (3.3.3) se može okarakterisati prvim i drugim karakteističnim polinomom koji su dati respektivno pomoću

$$\rho(\xi) = \sum_{i=0}^{k} \alpha_i \xi^i \text{ i } \sigma(\xi) = \sum_{i=0}^{k} \beta_i \xi^i.$$

Dve važne klase konvergentnih višekoračnih metoda koje se sreću u primenama su:

1. Metod kod kojih je $\rho(\xi) = \xi^k - \xi^{k-1}$;

2. Metod kod kojih je $\rho(\xi) = \xi^k - \xi^{k-2}$.

Eksplicitni metodi prve klase nazivaju se Adams-Bashfortovim, a implicitni Adams-Moultonovi. Slično, ekplicitni metodi druge klase nose naziv Nystromovi metodi, dok se odgovarajući implicitni metodi nazivaju generalisani Milne-Simpsonovi.

Naravno, postoje metodi koji ne pripadaju ovim klasama.

II.3.4 Izbor startnih vrednosti

Kao što je ranije napomenuto, kod primene linearnih višekora nih metoda na rešavanje problema (3.3.1), potrebno je poznavanje startnih vrednosti $y_i = s_i(h)$, takvih da je

$$\lim_{h\to 0} s_i(h) = y_0 \quad (i = 1, \ldots, k-1).$$

Naravno, ovaj problem se postavlja kada je $k > 1$.

Ako je metod (3.3.3) reda p, tada očigledno startne vrednosti $s_i(h)$ treba birati tako da je

$$s_i(h) - y(x_i) = O(h^{p+1}) \quad (i = 1, \ldots, k-1),$$

gde je $x \to y(x)$ tačno rešenje problema (3.3.1).

U ovom odeljku navešćemo jednu klasu metoda za odredjivanje potrebnih startnih vrednosti.

Pretpostavimo da je funkcija f u diferencijalnoj jednačini (3.3.1) dovoljan broj puta diferencijabilna. Tada na osnovu Taylorovog metoda imamo

$$y(x_0 + h) = y(x_0) + hy'(x_0) + \frac{h^2}{2!}y''(x_0) + \cdots + \frac{h^p}{p!}y^{(p)}(x_0) + O(h^{p+1}).$$

Poslednja jednakost ukazuje na to da se može uzeti

$$s_i(h) = y(x_0) + hy'(x_0) + \frac{h^2}{2!}y''(x_0) + \cdots + \frac{h^p}{p!}y^{(p)}(x_0),$$

s obzirom da je tada $s_i(h) - y(x_1) = O(h^{p+1})$ $(x_1 = x_0 + h)$. Isti postupak se može primeniti i na odredjivanje ostalih startnih vrednosti. Naime, u opštem slučaju, imamo

$$s_i(h) = y(x_{i-1} + hy'(x_{i-1}) + \frac{h^2}{2!}y''(x_{i-1}) + \cdots + \frac{h^p}{p!}y^{(p)}(x_{i-1}) \qquad (i = 1, \ldots, k-1),$$

pri čemu za $y(x_{i-1})$ uzimamo $s_{i-1}(h)$.

II.3.5 Prediktor-korektor metodi

Kao što je navedeno u odeljku II.3.3 primena implicitnih metoda je skopčana sa rešavanjem jednačine (3.3.4) na svakom koraku integracije, pri čemu se za ovo rešenje koristi iterativni proces (3.3.5). Bez obzira na ovu teškoću implicitni metodi se dosta koriste za rešavanje Cauchyevog problema, s obzirom da imaju niz prednosti nad eksplicitnim metodima (viši red, bolja numerička stabilnost). Početna vrednost $y_{n+k}^{[0]}$, u primenama se odre—uje korišćenjem nekog eksplicitnog metoda, koji tada nazivamo prediktor. Implicitni metod (3.3.4) nazivamo korektor. Metod dobijen ovakvom kombinacijom nazivamo prdiktor-korektor metod.

Za odredjivanje y_{n+k} , iterativni proces (3.3.5) treba primenjivati sve dok ne bude ispunjen uslov

$$|y_{n+k}^{[s+1]} - y_{n+k}^{[s+]}| < \varepsilon,$$

gde je ε dozvoljena greška, obično reda lokalne greške zaokrugljivanja. Tada se za y_{n+k} može uzeti $y_{n+k}^{[s]}$.

Medjutim, ovakav način se najčešće ne primenjuje u praksi, s obzirom da zahteva veliki broj izračunavanja vrednosti funkcije f po jednom koraku i uz to je ovaj broj promenljiv od koraka do koraka. Da bi se smanjio ovaj broj izračunavanja,

broj iteracija u (3.3.5) se fiksira. Dakle, uzima se samo $s = 0, 1, \ldots, m-1$.

II.3.6 Programska realizacija višekoačnih metoda

U ovom odeljku daćemo programsku realizaciju kako eksplicitnih tako i implicitnih metoda. Dobijene programe testiraćemo na primeru (sa $h = 0.1$).

$$y' = x^2 + y, \quad y(1) = 1 \quad (1 \leq x \leq 2).$$

Tačno rešenje ovog problema je $y(x) = 6e^{x-1} - x^2 - 2x - 2$.

3.6.1. Eulerov metod dat izrazom

$$y_{n+1} - y_n = hf_n \quad (n = 0, 1, \ldots),$$

čiji je red $p = 1$, i Adams-Bashfortov metod trećeg reda

$$y_{n+3} - y_{n+2} = \frac{h}{12}(23f_{n+2} - 16f_{n+1} + 5f_n) \quad (n = 0, 1, \ldots),$$

realizovani su pomoću potprograma `EULER` i `ADAMS` respektivno.

```
      SUBROUTINE EULER (XP,XK,H,Y,FUN)
      DIMENSION Y(1)
      N=(XK-XP+0.00001)/H
      X=XP
      DO 11 I=1,N
      Y(I+1)=Y(I)+H*FUN(X,Y(I))
   11 X=X+H
      RETURN
      END
C
      FUNCTION FUN (X,Y)
      FUN=X*X+Y
      RETURN
      END
C
```

```
      SUBROUTINE ADAMS (XP, XK, H, Y, FUN)
      DIMENSION Y(1)
      N=(XK-XP+0.00001)/H
      X=XP
      F0=FUN (X,Y(1))
      F1=FUN (X+H,Y(2))
      N2=N-2
      DO 11 I=1,N2
      F2=FUN(X+2.*H,Y(I+2))
      Y(I+3)=Y(I+2)+H*(23.*F2-16.*F1+5.*F0)/12.
      F0=F1
      F1=F2
   11 X=X+H
      RETURN
      END
```

Parametri u potprogramskoj listi imaju sledeće značenje:

`XP` i `XK` - početna i krajnja tačka intervala integracije;

`H` - korak integracije;

`Y` - vektor približnih vrednosti rešenja dobijen višekoračnim metodom, pri čemu kod Eulerovog metoda `Y(1)` predstavlja datu početnu vrednost, a kod Adamsovog metoda startne vrednosti su date kroz `Y(1)`, `Y(2)` i `Y(3)`;

`FUN` - ime funkcijskog potprograma kojim se definiše desna strana diferencijalne jednačine $f(x, y)$. Startne vrednosti za Adamsov metod odredjujemo primenom Taylorovog metod za $p = 3$ (videti odeljak 5.3.4). Naime, kako je

$$y(1) = 1, \quad y'(1) = 2, \quad y''(1) = 4, \quad y'''(1) = 6, \quad h = 0.1,$$

dobijamo `Y(1)=1.`, `Y(2)=1.12321`, `Y(3)=1.48836`.

Glavni program i izlazna lista imaju oblik:

```
C
C=================================================
C      RESAVANJE DIFERENCIJALNIH JEDNACINA
C            EKSPLICITNIM METODIMA
C=================================================
```

```
      EXTERNAL FUN
      DIMENSION Y(100),Z(100)
      F(X)=6.*EXP(X-1.)-X*X-2.*X-2.
      OPEN(5,FILE='EULER.OUT')
      WRITE (5,10)
   10 FORMAT(8X,'RESAVANJE DIFERENCIJAL.JED.EKSPLICITNIM',
     1' METODIMA'//8X,'XN',8X,'YN(I)',5X,'GRESKA(%)',3X,
     2'YN(II)',4X,'GRESKA (%)'/)
      XP=1.
      XK=2.
      H=0.1
      Y(1)=1.
      CALL EULER (XP,XK,H,Y,FUN)
      Z(1)=Y(1)
      Z(2)=1.221
      Z(3)=1.48836
      CALL ADAMS (XP,XK,H,Z,FUN)
      N=(XK-XP+0.00001)/H
      NN=N+1
      X=XP
      DO 22 I=1,NN
      G1=ABS((Y(I)-F(X))/F(X))*100.
      G2=ABS((Z(I)-F(X))/F(X))*100.
      WRITE (5,20)X,Y(I),G1,Z(I),G2
   22 X=X+H
   20 FORMAT (8X,F3.1,2(4X,F9.5,4X,F5.2))
      CLOSE(5)
      STOP
      END
```

```
  RESAVANJE DIFERENCIJAL.JED.EKSPLICITNIM METODIMA
  XN         YN(I)     GRESKA(%)   YN(II)    GRESKA
(%)
  1.0      1.00000       .00      1.00000       .00
  1.1      1.20000      1.72      1.22100       .00
  1.2      1.44100      3.19      1.48836       .00
  1.3      1.72910      4.42      1.80883       .02
  1.4      2.07101      5.47      2.19028       .03
  1.5      2.47411      6.37      2.64126       .04
  1.6      2.94652      7.13      3.17116       .05
  1.7      3.49717      7.79      3.79040       .06
  1.8      4.13589      8.36      4.51045       .06
  1.9      4.87348      8.87      5.34403       .07
  2.0      5.72183      9.32      6.30518       .07
```

3.6.2. Uzimajući Eulerov metod kao prediktor i trapezno pravilo ($p = 2$)

$$y_{n+1} - y_n = \frac{h}{2}(f_n + f_{n+1}) \quad (n = 0, 1, \ldots),$$

kao korektor (sa brojem iteracija $m = 2$) obrazivan je potprogram PREKOR. Glavni program, potprogram i izlazni rezultati imaju oblik:

```
C=============================================================
C     RESAVANJE DIF.JED. METODOM PREDIKTOR-KOREKTOR
C=============================================================
      EXTERNAL FUN
      DIMENSION Y(100)
      F(X)=6.*EXP(X-1.)-X*X-2.*X-2.
      OPEN(5,FILE='PREKOR.OUT')
      OPEN(8,FILE='PREKOR.TXT')
      WRITE(5,10)
   10 FORMAT(8X,'RESAVANJE DIF. JED. METODOM PREDIKTOR-
KOREKTOR'
     1,//15X,'XN',13X,'YN',10X,'GRESKA(%)'/)
      READ(8,5)XP,XK,YP,H
    5 FORMAT(4F6.1)
      CALL PREKOR(XP,XK,YP,H,Y,FUN)
      N=(XK-XP+0.00001)/H
      NN=N+1
      X=XP
      DO 11 I=1,NN
      G=ABS((Y(I)-F(X))/F(X))*100.
      WRITE(5,15)X,Y(I),G
   15 FORMAT(15X,F3.1,8X,F9.5,8X,F5.2)
   11 X=X+H
      STOP
      END
C
C
      SUBROUTINE PREKOR(XP,XK,YP,H,Y,FUN)
      DIMENSION Y(100)
      N=(XK-XP+0.00001)/H
      X=XP
      Y(1)=YP
```

```
      DO 10 I=1,N
C  PROGNOZIRANJE VREDNOSTI
      FXY=FUN(X,Y(I))
      YP=Y(I)+H*FXY
C  KOREKCIJA VREDNOSTI
      DO 20 M=1,2
   20 YP=Y(I)+H/2.*(FXY+FUN(X+H,YP))
      Y(I+1)=YP
   10 X=X+H
      RETURN
      END
C
C
      FUNCTION FUN(X,Y)
      FUN=X*X+Y
      RETURN
      END
```

```
RESAVANJE DIF. JED. METODOM PREDIKTOR-KOREKTOR
       XN              YN          GRESKA(%)
       1.0          1.00000           .00
       1.1          1.22152           .04
       1.2          1.48952           .07
       1.3          1.81097           .10
       1.4          2.19363           .12
       1.5          2.64602           .14
       1.6          3.17760           .15
       1.7          3.79881           .17
       1.8          4.52118           .18
       1.9          5.35747           .18
       2.0          6.32177           .19
```

II.3.7 Metodi Runge-Kutta

U prethodnim odeljcima razmatrani su linearni višekoračni metodi za rešavanje Cauchyevog problema (3.3.1). Red ovih metoda se može povećati povećanjem broja koraka. Medjutim, ukoliko se žrtvuje linearnost koju poseduju ovi metodi, moguće je konstruisati jednokoračne metode sa proizvoljnim redom.

Za rešavanje Cauchyevog problema oblika (3.3.1) sa dovoljno puta diferencijabilnom funkcijom f, moguće je, takodje konstruisati jednokoračne metode višeg reda (na primer, Taylorov metod).

Posmatrajmo opšti eksplicitni jednokoračni metod

$$y_{n+1} - y_n = h\Phi(x_n, y_n, h) \tag{3.7.1}$$

Definicija 3.7.1. *Metod* (3.7.1) *je reda* p *ako je* p *najveći ceo broj za koji važi*

$$y(x+h) - y(x) - h\Phi(x, y(x), h) = O(h^{p+1}),$$

gde je $x \to y(x)$ *tačno rešenje problema* (3.3.1).

Definicija 3.7.2. *Metod* (3.7.1) *je konzistentan ako je* $\Phi(x, y, 0) \equiv f(x, y)$.

Primetimo da je Taylorov metod specijalan slučaj metoda (3.7.1). Naime, kod Taylorovog metoda reda p imamo

$$\Phi(x, y, h) = \Phi_T(x, y, h) = \sum_{i=0}^{p-1} \frac{h^i}{(i-1)!} (\frac{\delta}{\delta x} + f\frac{\delta}{\delta y})^i f(x, y). \tag{3.7.2}$$

U specijalnom slučaju, kod Eulerovog metoda je $\Phi(x, y, h) = f(x, y)$.

U ovom odeljku razmatraćemo jednu specijalnu klasu metoda, oblika (3.7.1), koju je 1895. godine predložio C. Runge. Kasnije, ovu klasu metoda, oblika (3.7.1) razvili su W. Kutta i K. Heun.

Kao što ćemo kasnije videti, svi ovi metodi sadrže slobodne parametre. S obzirom na vreme u kome su se pojavili ovi metodi, slobodni parametri su birani tako da se dobiju što jednostavnije formule za praktično računanje. Medjutim, ovakve vrednosti parametra ne obezbedjuju optimalne karakteristike

posmatranih metoda. U daljem tekstu ove metode zvaćemo klasičnim. Opšti eksplicitni metod Runge-Kutta ima oblik

$$y_{n+1} - y_n = h\Phi(x_n, y_n, h) \tag{3.7.3}$$

gde su

$$\Phi(x, y, h) = \sum_{i=1}^{m} c_i k_i,$$

$$k_1 = f(x, y),$$

$$k_i = f(x + a_i, y + b_i h) \quad (i = 2, \ldots, m).$$

$$a_i = \sum_{j=1}^{i-1} \alpha_{ij}, \quad b_i = \sum_{j=1}^{i-1} \alpha_{ij} k_j \tag{3.7.4}$$

Primetimo da iz uslova konzistencije metoda (3.7.3) sleduje

$$\sum_{i=1}^{m} c_i = 1.$$

Nepoznate koeficijente koji figurišu u ovom metodu, odredjujemo iz uslova da metod ima maksimalni red. Pri ovome, koristimo sledeću činjenicu: Ako se $\Phi(x, y, h)$, razvijeno po stepenima od h, može predstaviti u obliku

$$\Phi(x, y, h) = \Phi_T(x, y, h) = O(h^p),$$

gde je Φ_T definisano pomoću (3.7.2), tada je metod (3.7.3) reda p.

Prethodno nadjimo razvoj $\Phi_T(x, y, h)$ po stepenim od h. Korišćenjem Mongeovih oznaka za parcijalne izvode imamo

$$(\frac{\delta}{\delta x} + \frac{\delta}{\delta y}) = f_x + f f_y = F$$

i

$$(\frac{\delta}{\delta x} + f\frac{\delta}{\delta y})^2 = (\frac{\delta}{\delta x} + f\frac{\delta}{\delta y})F = G + f_y F,$$

gde smo stavili $G = f_{xx} + 2ff_{xy} + f^2 f_{yy}$. Tada iz (3.7.2) sleduje

$$\Phi_T(x, y, h) = f + \frac{1}{2}hF + \frac{1}{6}h^2(G + f_y F) + O(h^3). \tag{3.7.5}$$

Razmotrićemo sada samo metode Runge-Kutta, čiji je red $p \leq 3$. Pokazuje se da je za dobijanje metoda trećeg reda dovoljno uzeti $m = 3$. U tom slučaju, formule (3.7.3) se svode na

$$\begin{aligned} \Phi(x, y, h) &= c_1 k_1 + c_2 k_2 + c_3 k_3 \\ k_1 &= f(x, y) \\ k_2 &= f(x + a_2 h, y + b_2 h), \\ k_3 &= f(x + a_3 h, y + b_3 h) \end{aligned}$$

i

$$\begin{aligned} a_2 &= \alpha_{21}, \quad b_2 = \alpha_{21} k_1, \\ a_3 &= \alpha 31 + \alpha_{32}, \quad b_3 = \alpha_{32} k_1 + \alpha_{32} k_2. \end{aligned}$$

Razvijanjem funkcije u Taylorov red, u okolini tačke (x, y), dobijamo

$$k_2 = f + a_2 F h + \frac{1}{2} + a_2^2 G h^2 + O(h^3).$$

Kako je

$$b_3 = \alpha_{31} k_1 + \alpha_{32} k_2 = \alpha_{31} f + \alpha_{32}(f + a_2 F h + \frac{1}{2} a_2^2 g h^2) + O(h^3)$$

imamo

$$b_3 = a_3 f + a_2 \alpha_{32} F h + O(h^2)$$

i

$$b_3^2 = a_3^2 f^2 + O(h).$$

Razvijenjem funkcije k_3 u okolini tačke (x, y) i korišćenjem poslednjih jednakosti imamo

$$k_3 = f + a_3 F h + \frac{1}{2}(2 a_3 \alpha_{32} F f_y + a_3^2 G) h^2 + O(h^3).$$

Najzad, zamenom dobijenih izraza za k_1, k_2, k_3 u izrazu za $\Phi(x, y, h)$ dobijamo

$$\begin{aligned}\Phi(x, y, h) =&(c_1 + c_2 + c_3)f + (c_2a_2 + c_3a_3)Fh \\ &+ (c_2a_2^2G + 2c_3a_2\alpha_{32}Ff_y + c_3a_3^2G)\frac{h^2}{2} + O(h^3).\end{aligned}$$

Poslednja jednakost dozvoljava konstrukciju metoda za $m = 1, 2, 3$.

Slučaj m=1. Kako je $c_2 = c_3 = 0$, imamo

$$\Phi(x, y, h) = c_1 f + O(h^3).$$

Uporedjivanjem sa (3.7.5) dobijamo

$$\Phi_T(x, y, h) - \Phi(x, y, h) = (1 - c_1)f + \frac{1}{2}h^2(G + f_yF) + O(h^3),$$

odakle zaključujemo da se za $c_1 = 1$ dobija metod

$$y_{n+1} - y_n = hf_n,$$

čiji je red $p = 1$. S obzirom da je ovo Eulerov metod mi vidimo da on pripada i klasi metoda Runge-Kutta.

Slučaj m=2. Ovde je $c_3 = 0$ i

$$\Phi_T(x, y, h) = (c_1 + c_2)f + c_2a_2Fh + \frac{1}{2}c_2a_2^2Gh^2 + O(h^3).$$

Kako je

$$\begin{aligned}\Phi_T(x, y, h) - \Phi(x, y, h) = &(c_1 + c_2 - 1)f + (c_2a_2 - \frac{1}{2})Fh \\ &+ \frac{1}{6}[(3c_2a_2^2 - 1)G - f_yF]h^2 + O(h^3),\end{aligned}$$

zaključujemo da se pod uslovima

$$c_1 + c_2 = 1 \qquad \text{i} \qquad c_2 a_2 = \frac{1}{2}, \tag{3.7.6}$$

dobija metod drugog reda sa jednim slobodnim parametrom. Naime, iz sistema jednakosti (3.7.6) sleduje

$$c_2 = \frac{1}{2a_2} \qquad \text{i } c_1 = \frac{2a_2 - 1}{2a_2},$$

gde je $a_2 (\neq 0)$ slobodan parametar. Dakle, sa $m = 2$ imamo jednoparametarsku familiju metoda

$$\begin{aligned} y_{n+1} - y_n &= \frac{h}{2a_2}((2a_2 - 1)k_1 + k_2), \\ k_1 &= f(x_n, y_n), \\ k_2 &= f(x_n + a_2 h, y_n + a_2 k_1 h). \end{aligned}$$

U specijalnom slučaju, za $a_2 = \frac{1}{2}$, dobijamo Euler-Cauchyev metod

$$y_{n+1} - y_n = hf(x_n + \frac{1}{2}h, y_n + \frac{1}{2}hf(x_n, y_n)).$$

Slično, za $a_2 = 1$, dobijamo tzv. poboljšan Euler-Cauchyev metod

$$y_{n+1} - y_n = \frac{h}{2}[f(x_n, y_n) + f(x_n + h, y_n + hf(x_n, y_n)].$$

O geometrijskoj interpretaciji dobijenih metoda videti, na primer, u [M. Bertolino, *Numerička analiza*, Beograd, 1977.]

Slučaj m=3. Kako je

$$\begin{aligned} \Phi(x, y, h) - \Phi_T(x, y, h) &= (c_1 + c_2 + c_3 - 1)f + (c_2 a_2 + c_3 a_3 - \frac{1}{2})Fh \\ &+ [(c_2 a_2^2 + c_3 a_3^2 - \frac{1}{3}G + (2c_3 a_2 \alpha_{32} - \frac{1}{3})Ff_y]\frac{h^2}{2} + O(\frac{h}{3}), \end{aligned}$$

zaključujemo da su za dobijanje metoda trećeg reda dovoljni uslovi

$$\begin{aligned} c_1 + c_2 + c_3 &= 1, \\ c_2 a_2 + c_3 a_3 &= \frac{1}{2}, \\ c_2 a_2^2 + c_3 a_3^2 &= \frac{1}{3}, \\ c_3 a_2 \alpha_{32} &= \frac{1}{6}. \end{aligned} \tag{3.7.7}$$

S obzirom da imamo četiri jednačine sa šest nepoznatih, izlazi da, u slučaju $m = 3$, imamo dvoparametarsku familiju metoda Runge-Kutta. Može se pokazati da medju metodima ove familije ne postoji ni jedan metod čiji je red veći od tri.

U specijalnom slučaju kada je $a_2 = \frac{1}{3}$ i $a_3 = \frac{2}{3}$, iz (3.7.7) sleduje $c_1 = \frac{1}{4}, c_2 = 0, c_3 = \frac{3}{4}, \alpha_{32} = \frac{2}{3}$ Dakle, dobili smo metod

$$\begin{aligned} y_{n+1} - y_n &= \frac{h}{4}(k_1 + 3k_3), \\ k_1 &= f(x_n, y_n), \\ k_2 &= f(x_n + \frac{h}{3}, y_n + \frac{h}{3}k_1), \\ k_3 &= f(x_n + \frac{2h}{3}, y_n + \frac{h}{3}k_2 1), \end{aligned}$$

koji se u literaturi sreće kao Heunov metod.

Za $a_2 = \frac{1}{2}, a_3 = 1 (\Rightarrow c_1 = c_3 = \frac{1}{6}, c_2 = \frac{2}{3}, \alpha_{32} = 2)$ dobijamo metod

$$\begin{aligned} y_{n+1} - y_n &= \frac{h}{6}(k_1 + 4k_2 + k_3), \\ k_1 &= f(x_n, y_n), \\ k_2 &= f(x_n + \frac{h}{2}, y_n + \frac{h}{2}k_1), \\ k_3 &= f(x_n + h, y_n - hk_1 + 2hk_2), \end{aligned}$$

koji je najpopularniji medju metodima trećeg reda sa stanovišta ručnog izračunavanja.

U slučaju kada je $m = 4$, dobijamo dvoparametarsku familiju metode četvrtog reda. Naime, ovde se, analogno sistemu (3.7.7), javlja sistem od 11 jednačina sa 13 nepoznatih.

Sada navodimo, bez dokaza, metod Runge-Kutta četvrtog reda

$$
\begin{aligned}
y_{n+1} - y_n &= \frac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4), \\
k_1 &= f(x_n, y_n), \\
k_2 &= f(x_n + \frac{h}{2}, y_n + \frac{h}{2}k_1), \\
k_3 &= f(x_n + \frac{h}{2}, y_n \frac{h}{2}k_2), \\
k_4 &= f(x_n + h, y_n + hk_3,
\end{aligned}
\tag{3.7.8}
$$

koji se u primenama tradicionalno najviše koristi.

Od metoda četvrtog reda često se koristi i tzv. Gillova vari-

janta, koja se može iskazati sledećim rekurzivnim postupkom:

$$
\begin{aligned}
& n := 0, \quad Q_0 := 0 \\
(*)\ & Y_0 := y_n, \\
& k_1 := hf(x_n, Y_0), \quad Y_1 := Y_0 + \frac{1}{2}(k_1 - 2Q_0), \\
& Q_1 := Q_0 + \frac{3}{2}(k_1 - 2Q_0) - \frac{1}{2}k_1, \\
& k_2 := hf(x_n + \frac{h}{2}, Y_1), \quad Y_2 := Y_1 + (1 - \sqrt{1/2})(k_2 - Q_1), \\
& Q_2 := Q_1 + 3(1 - \sqrt{1/2})(k_2 - Q_1) - (1 - \sqrt{1/2})k_2, \\
& k_3 := hf(x_n + \frac{h}{2}, Y_2), \quad Y_3 := Y_2 + (1 + \sqrt{1/2})(k_3 - Q_2), \\
& Q_3 := Q_2 + 3(1 + \sqrt{1/2})(k_3 - Q_2) - (1 + \sqrt{1/2})k_3, \\
& k_4 := hf(x_n + h, Y_3), \quad Y_4 := Y_3 + \frac{1}{6}(k_4 - 2Q_3), \\
& Q_0 := Q_3 + \frac{1}{2}(k_4 - 2Q_3) - \frac{1}{2}k_4, \\
& y_{n+1} := Y_4, \\
& n := n + 1 \\
& \text{preći na } (*).
\end{aligned}
$$

Za razliku od linearnih višekoračnih metoda, metodi Runge-Kutta ne zahtevaju poznavanje startnih vrednosti (sem $y(x_0) = y_0$, koja, inače, definiše Cauchyev problem), ali su za praktičnu primenu znatno komplikovaniji, s obzirom da zahtevaju m izračunavanja vrednosti funkcije f u svakom koraku.

II.3.8. Programska realizacija metoda Runge-Kutta

U ovom odeljku dajemo programsku realizaciju Euler-Cauchyevog, poboljšanog Euel-Cauchyevog metoda, kao i metoda četvrtog reda (3.7.8) i Gillove varijante metoda Runge-Kutta. Dobijene programe testiraćemo na primeru iz odeljka 5.3.6

Program 3.8.1.

Potprogramom EULCAU realizovani su Euler-Cauchyev i poboljšan Euler-Cauchyev metod. Parametri u listi imaju sledeće značenje:

XP - početna tačka intervala integracije;

H - korak integracije;

N - ceo broj, takav da je N+1 dužina vektora Y;

M - ceo broj koji definiše način konstrukcije vektora Y. Naime, u vektoru Y se redom memoriše svaka M-ta vrednost rešenja dobijena u procesu integracije;

Y - vektor rešenja dužine N+1, pri čemu Y(1) predstavlja zadati početni uslov , Y(2) je vrednost rešenja dobijena integracijom u tački XP + M*H, itd.

FUN - ime funkcijskog potprograma, kojim se definiše desna strana diferencijalne jednačine $f(x, y)$;

K - ceo broj, sa vrednostima K=1 i K=2, kojom se zadaje integracija po Euler-Cauchyevom i poboljšanom Euler-Cauchyevom metodu respektivno.

Potprogram EULCAU ima oblik:

```
      SUBROUTINE EULCAU(XP,H,N,M,Y,FUN,K)
      DIMENSION Y(1)
      X=XP
      Y1=Y(1)
      NN=N+1
      DO 10 I=2,NN
      DO 20 J=1,M
      Y0=Y1
      Y1=FUN(X,Y0)
      GO TO (1,2),K
 1    Y1=Y0+H*FUN(X+0.5*H,Y0+0.5*H*Y1)
      GO TO 20
 2    Y1=Y0+H*(Y1+FUN(X+H,Y0+H*Y1))/2.
 20   X=X+H
10    Y(I)=Y1
      RETURN
```

```
      END
C
      FUNCTION FUN(X,Y)
      FUN=X*X+Y
      RETURN
      END
```

Glavni program i izlazna lista su dati u daljem tekstu. Kao ulazne parametre za integraciju smo uzeli `H=0.1, N=10, M=1`, a u drugom slučaju `H=0.05, N=10, M=2`. Kolone `Y1N` i `Y2N`, u izlaznoj listi daju vrednosti za rešenje datog Cauchyevog problema, po običnom i poboljšanom Euler-Cauchyevom metodu respektivno. Pored ovih kolona, u izlaznoj listi su date i kolone sa odgovarajućim greškama (izražene u %) u odnosu na tačno rešenje.

```
C=============================================================
C RESAVANJE DIF. JED. EULER-CAUCHYEVIM I POBOLJSANIM METODOM
C=============================================================
      EXTERNAL FUN
      DIMENSION Y(100), Z(100)
      F(X)=6.*EXP(X-1.)-X*X-2.*X-2.
      OPEN(5,FILE='EULCAU.OUT')
      OPEN(8,FILE='EULCAU.IN')
      WRITE(5,10)
10    FORMAT(10X,'RESAVANJE DIF. JED. EULER-CAUCHYEVIM'
     1' I POBOLJSANIM METODOM')
20    READ(8,25,END=99)XP,Y(1),H,N,M
25    FORMAT(3F6.1,2I3)
      CALL EULCAU(XP,H,N,M,Y,FUN,1)
      Z(1)=Y(1)
      CALL EULCAU(XP,H,N,M,Z,FUN,2)
      WRITE(5,30)H
30    FORMAT(1H0,30X,'(H=',F6.4,')'//15X,'XN',8X,'Y1N',4X,
     1'GRESKA(%)',5X,'Y2N',4X,'GRESKA(%)'/)
      NN=N+1
      X=XP
      DO 11 I=1,NN
      G1=ABS((Y(I)-F(X))/F(X))*100.
      G2=ABS((Z(I)-F(X))/F(X))*100.
      WRITE(5,15)X,Y(I),G1,Z(I),G2
15    FORMAT(15X,F3.1,3X,F9.6,2X,F7.5,3X,F9.6,2X,F7.5)
```

```
11    X=X+H*M
      GO TO 20
99    CLOSE(5)
      CLOSE(8)
      STOP
      END
```

```
RESAVANJE DIF. JED. EULER-CAUCHYEVIM I POBOLJSANIM METODOM
                    (H= .1000)
 XN        Y1N     GRESKA(%)      Y2N     GRESKA(%)
 1.0    1.000000    .00000    1.000000    .00000
 1.1    1.220250    .06352    1.220500    .04304
 1.2    1.486676    .11693    1.487203    .08157
 1.3    1.806227    .16173    1.807059    .11576
 1.4    2.186581    .19934    2.187750    .14599
 1.5    2.636222    .23109    2.637764    .17274
 1.6    3.164526    .25808    3.166479    .19650
 1.7    3.781851    .28125    3.784260    .21773
 1.8    4.499645    .30138    4.502557    .23685
 1.9    5.330558    .31907    5.334026    .25422
 2.0    6.288567    .33483    6.292649    .27013
                    (H= .0500)
 XN        Y1N     GRESKA(%)      Y2N     GRESKA(%)
 1.0    1.000000    .00000    1.000000    .00000
 1.1    1.220824    .01655    1.220888    .01130
 1.2    1.487963    .03046    1.488098    .02140
 1.3    1.808391    .04213    1.808604    .03034
 1.4    2.189811    .05192    2.190111    .03824
 1.5    2.640738    .06019    2.641133    .04523
 1.6    3.170581    .06721    3.171082    .05143
 1.7    3.789740    .07324    3.790357    .05696
 1.8    4.509705    .07848    4.510451    .06195
 1.9    5.343177    .08309    5.344066    .06647
 2.0    6.304192    .08719    6.305238    .07061
```

Program 3.8.2.

Prema formulama (3.7.8) za standardni metod Runge-Kutta četvrtog reda obrazovan je potprogram RK4:

```
      SUBROUTINE RK4(X0,Y0,H,M,N,YVEK,F)
```

```
C=============================================================
C     METOD RUNGE-KUTTA CETVRTOG REDA
C=============================================================
      DIMENSION YVEK(1)
      T=H/2.
      X=X0
      Y=Y0
      DO 20 I=1,N
      DO 10 J=1,M
      A=F(X,Y)
      B=F(X+T,Y+T*A)
      C=F(X+T,Y+T*B)
      D=F(X+H,Y+H*C)
      X=X+H
 10   Y=Y+H/6.*(A+2.*B+2.*C+D)
 20   YVEK(I)=Y
      RETURN
      END
```

Parametri u listi imaju sledeće značenje:

`X0,X0` - definišu zadati početni uslov (`Y0`=y(`X0`);

`H` - korak integracije;

`M, N` - celi brojevi sa značenjem sličnim kao u potprogramu `EULCAU`;

`YVEK` - vektor dužine n koji se dobija kao rezultat numeričke integracije, pri čemu je `Y(1)` vrednost dobijena u tački `X0+M*H`, `Y(2)` vrednost u tački `X0+2M*H`, itd.;

`F` - ime funkcijskog potprograma kojim se definiše desna strana diferencijalne jednačine $f(x, y)$.

Glavni program ima oblik:

```
C=============================================================
C    RESAVANJE DIF.JED. METODOM RUGE-KUTTA
C=============================================================
      EXTERNAL FUN
      DIMENSION Y (100)
      F(X)=6.*EXP(X-1.)-X*X-2.*X-2.
      OPEN(5,FILE='RK4.OUT')
      OPEN(8,FILE='RK4.IN')
```

```
      WRITE(5,10)
 10   FORMAT (14X,'RESAVANJE DIF.JED. METODOM RUNG-KUTTA')
 20   READ (8,5,END=99)X0,Y0,H,N,M
 5    FORMAT (3F6.1,2I3)
      CALL RK4(X0,Y0,H,M,N,Y,FUN)
      G=0.
      WRITE (5,25) H,X0,Y0,G
25    FORMAT( 28X,'(H=',F6.4,')'//15X,'XN',13X,'YN',10X,
     1'GRESKA(%)'//15X,F3.1,8X,F9.6,7X,F7.5)
      X=X0
      DO 11 I=1,N
      X=X+H*M
      G=ABS((Y(I)-F(X))/F(X))*100.
11    WRITE (5,15)X,Y(I),G
15    FORMAT (15X,F3.1,8X,F9.6,7X,F7.5)
      GO TO 20
99    CLOSE(5)
      CLOSE(8)
      STOP
      END
C
      FUNCTION FUN(X,Y)
      FUN=X*X+Y
      RETURN
      END
```

Uzimajući H=0.1, N=10, M=1 dobijeni su sledeći rezultati:

```
RESAVANJE DIF.JED. METODOM RUNG-KUTTA
               (H= .1000)
 XN              YN          GRESKA(%)
 1.0         1.000000         .00000
 1.1         1.221025         .00002
 1.2         1.488416         .00005
 1.3         1.809152         .00007
 1.4         2.190946         .00009
 1.5         2.642325         .00011
 1.6         3.172709         .00012
 1.7         3.792512         .00014
 1.8         4.513240         .00015
 1.9         5.347611         .00017
 2.0         6.309682         .00018
```

Program 3.8.3.

Gillovu varijantu metoda Runge-Kutta realizovaćemo u dvostrukoj tačnosti. Parametri u listi potprograma `GILL, X0, H, N, M, Y, FUN` imaju isto značenje respektivno kao parametri `HP, H, N, M, Y, FUN` u potprogramu `EULCAU`. Primetimo da je ovaj potprogram realizovan tako da je izvršena optimizacija u pogledu broja promenljivih.

Ulazne parametre za integraciju smo uzeli kao u programu **3.8.1.**

```
C=============================================================
C   RESAVANJE DIF.JED. METODOM RUNGE-KUTTA (GILLOVA VAR-
IJANTA)
C=============================================================
      EXTERNAL FUN
      REAL*8 Y(100),F,FUN,X0,X,H,G
      F(X)=6.*DEXP(X-1.)-X*X-2.*X-2.
      OPEN(8,FILE='GILL.IN')
      OPEN(5,FILE='GILL.OUT')
      WRITE(5,10)
   10 FORMAT(8X,'RESAVANJE DIF.JED.METODOM RUNGE-KUTTA'
     1' (GILLOVA VARIJANTA)' )
   20 READ(8,25,END=99)X,Y(1),H,N,M
   25 FORMAT(3F6.1,2I3)
      X0=X
      CALL GILL(X0,H,N,M,Y,FUN)
      WRITE(5,30)H
   30 FORMAT(/28X,'(H=',F6.4,')'//15X,'XN',13X,'YN',10X,
     1'GRESKA(%)'/)
      NN=N+1
      DO 11 I=1,NN
      G=DABS((Y(I)-F(X))/F(X))*100.
      WRITE(5,15)X,Y(I),G
   15 FORMAT(15X,F3.1,8X,F9.6,6X,D10.3)
   11 X=X+H*M
      GO TO 20
   99 CLOSE(5)
      CLOSE(8)
      STOP
      END
```

```
C
C
      SUBROUTINE GILL(X0,H,N,M,Y,FUN)
      REAL*8 Y(1),H,FUN,X0,Y0,Q,K,A,B
      B=DSQRT(0.5D0)
      Q=0.D0
      Y0=Y(1)
      NN=N+1
      DO 10 I=2,NN
      DO 20 J=1,M
      K=H*FUN(X0,Y0)
      A=0.5*(K-2.*Q)
      Y0=Y0+A
      Q=Q+3.*A-0.5*K
      K=H*FUN(X0+H/2.,Y0)
      A=(1.-B)*(K-Q)
      Y0=Y0+A
      Q=Q+3.*A-(1.-B)*K
      K=H*FUN(X0+H/2,Y0)
      A=(1.+B)*(K-Q)
      Y0=Y0+A
      Q=Q+3.*A-(1.+B)*K
      K=H*FUN(X0+H,Y0)
      A=(K-2.*Q)/6.
      Y0=Y0+A
      Q=Q+3.*A-K/2.
   20 X0=X0+H
   10 Y(I)=Y0
      RETURN
      END
C
      FUNCTION FUN(X,Y)
      REAL*8 FUN,X,Y
      FUN=X*X+Y
      RETURN
      END
```

```
RESAVANJE DIF.JED.METODOM RUNGE-KUTTA (GILLOVA VARIJANTA)
                    (H= .1000)
       XN              YN          GRESKA(%)
       1.0          1.000000         .000D+00
       1.1          1.221025         .246D-04
       1.2          1.488416         .460D-04
       1.3          1.809152         .647D-04
```

```
1.4          2.190946          .808D-04
1.5          2.642325          .949D-04
1.6          3.172709          .107D-03
1.7          3.792512          .118D-03
1.8          4.513240          .128D-03
1.9          5.347611          .136D-03
2.0          6.309682          .144D-03
              (H= .0500)
XN              YN           GRESKA(%)
1.0          1.000000          .000D+00
1.1          1.221025          .162D-05
1.2          1.488417          .303D-05
1.3          1.809153          .425D-05
1.4          2.190948          .531D-05
1.5          2.642327          .623D-05
1.6          3.172713          .704D-05
1.7          3.792516          .775D-05
1.8          4.513245          .838D-05
1.9          5.347618          .894D-05
2.0          6.309690          .946D-05
```

II.3.9. Rešavanje sistema jednačina i jednačina višeg reda

Metodi koji su bili razmatrani u prethodnim odeljcima mogu se uopštiti u tom smislu da budu primenljivi za rešavanje Cauchyevog problema za sistem od p jednačina prvog reda

$$y_i' = f_i(x; y_1, \ldots, y_p), \quad y_i(x_0) = y_{i0} \quad (i = 1, \ldots, p). \tag{3.9.1}$$

U ovom slučaju, sistem jedna ina (3.9.1) treba predstaviti u vektorskom obliku

$$\vec{y}\,' = \vec{f}(x, \vec{y}), \quad \vec{y}(x_0) = \vec{y}_0, \tag{3.9.2}$$

gde su

$$\vec{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}, \quad \vec{y}_0 = \begin{bmatrix} y_{10} \\ y_{20} \\ \vdots \\ y_{p0} \end{bmatrix}, \quad \vec{f}(x, \vec{y}) = \begin{bmatrix} f_1(x & ; y_1, \ldots, y_p) \\ \vdots & \\ f_p(x & ; y_1, \ldots, y_p) \end{bmatrix}.$$

Od interesa je i rešavanje Cauchyevog problema za diferencijalne jednačine višeg reda. Primetimo, medjutim, da se ovaj problem može svesti na prethodni. Naime, neka je data diferencijalna jednačine reda p

$$y^{(p)} = f(x, y, y', \ldots, y_{(p-1)}) \tag{3.9.3}$$

sa početnim uslovima

$$y^{(i)}(x_0) = y_{i0} \quad (i = 0, \ldots, p-1). \tag{3.9.4}$$

Tada se, supstitucijama

$$z_1 = y, \quad z_2 = y', \ldots, z_p = y_{(p-1)},$$

jedna ina (3.9.3) sa uslovima (3.9.4), svodi na sistem

$$\begin{aligned} z_1' &= z_2 \\ z_1' &= z_2 \\ &\vdots \\ z_{p-1}' &= z_m \\ z_p' &= f(x, z_1, z_2, \ldots, z_p), \end{aligned}$$

sa uslovima $z_i(x_0) = z_{i0} = y_{i0} \quad (i = 1, \ldots, p)$.

Linearni višekoračni metodi, koje smo do sada razmatrali, mogu se formalno generalisati na vektorski oblik

$$\sum_{i=0}^{k} \alpha_i \vec{y}_{n+i} = h \sum_{i=0}^{k} \beta_i \vec{f}_{n+i},$$

gde je $\vec{f}_{n+i} = \vec{f}(x_{n+i}, \vec{y}_{n+i})$, a zatim se kao takvi mogu primeniti na reavanje Cauchyevog problema (3.9.2).

Takodje, metodi Runge-Kutta za rešavanje Cauchyevog problema (3.9.2) imaju oblik

$$\vec{y}_{n+1} - \vec{y}_n = h\vec{\Phi}(x_n, \vec{y}_n, h),$$

gde su

$$\begin{aligned}
&\vec{\psi}(x, \vec{y}, h) = \sum_{i=1}^{m} c_i \vec{k}_i, \\
&\vec{k}_1 = \vec{f}(x, \vec{y}), \\
&\vec{k}_i = \vec{f}(x + a_i h, \vec{y} + \vec{b}_i h) \\
&a_i = \sum_{j=1}^{i-1} \alpha_{ij}, \quad \vec{b}_i = \sum_{j=1}^{i-1} \alpha_{ij} \vec{k}_j \quad (i = 2, \ldots, m).
\end{aligned}$$

Sva analiza, koja je data u prethodnim poglavljima formalno se može preneti na navedene vektorske metode.

Kao primer realizujmo standardni metod Runge-Kutta četvrtog reda (3.7.8) za rešavanje sistema od dve diferencijalne jednačine

$$y' = f_1(x, y, z), \quad z' = f_2(x; y, z),$$

pri uslovima $y(x_0) = y_0$ i $z(x_0) = z_0$.

Odgovarajući potprogram ima oblik:

```
SUBROUTINE RKS(XP,XKRAJ,YP,ZP,H,N,YY,ZZ)
REAL KY1,KY2,KY3,KY4,KZ1,KZ2,KZ3,KZ4
DIMENSION YY(1),ZZ(1)
K=(XKRAJ-XP)/(H*FLOAT(N))
N1=N+1
X=XP
Y=YP
Z=ZP
T=H/2.
YY(1)=Y
ZZ(1)=Z
DO 6 I=2,N1
DO 7 J=1,K
```

```
      KY1=FUN(1,X,Y,Z)
      KZ1=FUN(2,X,Y,Z)
      KY2=FUN(1,X+T,Y+T*KY1,Z+T*KZ1)
      KZ2=FUN(2,X+T,Y+T*KY1,Z+T*KZ1)
      KY3=FUN(1,X+T,Y+T*KY2,Z+T*KZ2)
      KZ3=FUN(2,X+T,Y+T*KY2,Z+T*KZ2)
      KY4=FUN(1,X+H,Y+H*KY3,Z+H*KZ3)
      KZ4=FUN(2,X+H,Y+H*KY3,Z+H*KZ3)
      Y=Y+H*(KY1+2.*(KY2+KY3)+KY4)/6.
      Z=Z+H*(KZ1+2.*(KZ2+KZ3)+KZ4)/6.
    7 X=X+H
      YY(I)=Y
    6 ZZ(I)=Z
      RETURN
      END
```

Koristeći ovaj potprogram rešili smo sistem jednačina

$$y' = xyz, \quad z' = xy/z,$$

pri uslovima $y(1) = 1/3$ i $z(1) = 1$ na segmentu $[1, 2.5]$ uzimajući korak integracije $h = 0.01$, dok na izlazu štampamo x sa korakom 0.1 i odgovarajuće vrednosti za y, y_T, z, z_T, gde su y_T i z_T tačna rešenja ovog sistema i data su sa

$$y_T = \frac{72}{(7 - x^2)^3} \quad \text{i} \quad z_T = \frac{6}{7 - x^2}.$$

Odgovarajući program i izlazna lista imaju sledeći oblik:

```
C=====================================================
C   RESAVANJE SISTEMA DIF. JED. METODOM RUNGE-KUTTA
C=====================================================
      DIMENSION YT(16),ZT(16),YY(16),ZZ(16),X(16)
      YEG(P)=72./(7.-P*P)**3
      ZEG(P)=6./(7.-P*P)
      OPEN(8,FILE='RKS.IN')
      OPEN(5,FILE='RKS.OUT')
      READ(8,15)N,XP,YP,ZP,XKRAJ
15    FORMAT(I2,4F3.1)
```

```
      YP=YP/3.
      H=0.1
      N1=N+1
      DO 5 I=1,N1
      X(I)=XP+H*FLOAT(I-1)
      YT(I)=YEG(X(I))
    5 ZT(I)=ZEG(X(I))
      WRITE(5,22)
      H=0.01
      CALL RKS(XP,XKRAJ,YP,ZP,H,N,YY,ZZ)
      WRITE(5,18)H,(X(I),YY(I),YT(I),ZZ(I),ZT(I),I=1,N1)
   18  FORMAT(//7X,'KORAK INTEGRACIJE H=',F6.3//7X,'X',
     111X,'Y',10X,'TACNO',11X,'Z',10X,'ZTACNO'//
     2(F10.2,4F14.7))
   22  FORMAT(1H1,9X,'RESAVANJE SISTEMA SIMULTANIH',
     1'DIFERENCIJALNIH JEDNACINA'//33X,'Y''=XYZ'//33X,
     2'Z''=XY/Z')
      CLOSE(5)
      CLOSE(8)
      STOP
      END
C
      FUNCTION FUN(J,X,Y,Z)
      GO TO (50,60),J
   50 FUN=X*Y*Z
      RETURN
   60 FUN=X*Y/Z
      RETURN
      END
```

```
    RESAVANJE SISTEMA SIMULTANIHDIFERENCIJALNIH JEDNACINA
                          Y'=XYZ
                          Z'=XY/Z
 KORAK INTEGRACIJE H=  .010
 X           Y           TACNO           Z           ZTACNO
1.00      .3333333      .3333333     1.0000000     1.0000000
1.10      .3709342      .3709342     1.0362690     1.0362690
1.20      .4188979      .4188979     1.0791370     1.0791370
1.30      .4808936      .4808935     1.1299430     1.1299430
1.40      .5623943      .5623943     1.1904760     1.1904760
1.50      .6718181      .6718181     1.2631580     1.2631580
1.60      .8225902      .8225904     1.3513510     1.3513510
1.70     1.0370670     1.0370680     1.4598540     1.4598540
1.80     1.3544680     1.3544680     1.5957440     1.5957450
```

1.90	1.8481330	1.8481340	1.7699110	1.7699110
2.00	2.6666650	2.6666670	2.0000000	2.0000000
2.10	4.1441250	4.1441260	2.3166020	2.3166020
2.20	7.1444800	7.1444920	2.7777760	2.7777780
2.30	14.3993600	14.3993900	3.5087690	3.5087710
2.40	37.7628900	37.7631300	4.8387000	4.8387110
2.50	170.6632000	170.6667000	7.9999230	8.0000000

II.3.10. Konturni problemi

U ovom odeljku ukazaćemo na diferencni metod za rešavanje konturnog problema

$$y'' + p(x)y' + q(x)y = f(x); \quad y(a) = A,\ y(b) = B, \tag{3.10.1}$$

gde su funkcije p, q, f neprekidne na $[a, b]$.

Segment $[a, b]$ podelimo na $N+1$ podsegmenata dužine $h = \dfrac{b-a}{N+1}$, tako da je $x_n = a+nh$ $(n = 0, 1, ..., N+1)$. U tačkama x_n $(n = 1, \ldots, N)$ diferencijalnu jednačinu iz (3.10.1) aproksimirajmo sa

$$\frac{y_{n+1} - 2y_n + Y_{n-1}}{h^2} + p_n \frac{y_{n+1} - y_{n-1}}{2h} + q_n y_n = f_n \qquad (n = 1, \ldots, N), \tag{3.10.2}$$

gde su $p_n \equiv p(x_n),\ q_n \equiv q(x_n),\ f_n \equiv f(x_n)$.

Ako uvedemo smene

$$a_n = 1 - \frac{h}{2}p_n,\ b_n = h^2 q_n - 2,\ c_n = 1 + \frac{h}{2}p_n,$$

(3.10.2) se može predstaviti u obliku

$$a_n y_{n-1} + b_b y_n + c_n y_{n+1} = h^2 f_n \quad (n = 1, \ldots, N). \tag{3.10.3}$$

S obzirom da su konturni uslovi $y_0 = A$ i $Y_{N+1} = B$, pred nas se postavlja problem rešavanja sistema linearnih jednačina

$\mathbf{T}\vec{y} = \vec{d}$, gde su

$$\vec{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}, \quad \vec{d} = \begin{bmatrix} h^2 f_1 - A a_1 \\ h^2 f_2 \\ \vdots \\ h^2 f_N - B c_N \end{bmatrix}, \quad \mathbf{T} = \begin{bmatrix} b_1 & c_1 & 0 & \dots & 0 \\ a_2 & b_2 & c_2 & 0 & \\ \vdots & & & & \\ 0 & 0 & 0 & \dots & b_N \end{bmatrix}.$$

Matrica sistema je trodijagonalna. Za rešavanje ovog sistema pogodno je izvršiti dekompoziciju matrice **T** u obliku **T=LR** (videti odeljak *I*.1.1), čime se problem svodi na sukcesivno rešavanje dva trougaona sistema linearnih jednač ina. Ovakav postupak za rešavanje konturnog problema (3.10.1), u literaturi je poznat kao matrična faktorizacija.

Sledeći program je realizovan na osnovu izloženog postupka.

```
      DIMENSION A(100),B(100),C(100),D(100)
C================================================
C  MATRICNA FAKTORIZACIJA ZA RESAVANJE
C  KONTURNIH PROBLEMA KOD  LINEARNIH
C  DIFERENCIJALNIH JEDNACINA II REDA
C  Y''+ P(X)Y'+ Q(X)Y = F(X)
C  Y(DG) = YA, Y(GG) = YB
C ================================================
      OPEN(8,FILE='KONTUR.IN')
      OPEN(7,FILE='KONTUR.OUT')
      READ(8,5) DG,YA,GG,YB
 5    FORMAT(4F10.5)
C UCITAVANJE BROJA MEDJUTACAKA
 10   WRITE(*,14)
 14   FORMAT(1X,'UNETI BROJ MEDJUTACAKA'
     1' U FORMATU I2'/ 5X,'(ZA N=0 => KRAJ)')
      READ(5,15) N
 15   FORMAT(I2)
      N1=N+1
      IF(N.EQ.0) GO TO 60
      H=(GG-DG)/FLOAT(N1)
      HH=H*H
      X=DG
```

```
      DO 20 I=1,N
      X=X+H
      Y=H/2.*PQF(X,1)
      A(I)=1.-Y
      C(I)=1.+Y
      B(I)=HH*PQF(X,2)-2.
20    D(I)=HH*PQF(X,3)
      D(1)=D(1)-YA*A(1)
      D(N)=D(N)-YB*C(N)
      C(1)=C(1)/B(1)
      DO 25 I=2,N
      B(I)=B(I)-A(I)*C(I-1)
25    C(I)=C(I)/B(I)
      D(1)=D(1)/B(1)
      DO 30 I=2,N
30    D(I)=(D(I)-A(I)*D(I-1))/B(I)
      NM=N-1
      DO 35 I=1,NM
      J=NM-I+1
35    D(J)=D(J)-C(J)*D(J+1)
      WRITE(7,40)N,(I,I=1,N1)
40    FORMAT(///5X,'BROJ MEDJUTACAKA N='
     1 ,I3///5X,'I',6X,'O',9I10)
      DO 45 I=1,N
      C(I)=DG+H*FLOAT(I)
45    B(I)=PQF(C(I),4)
      WRITE(7,50)DG,(C(I),I=1,N),GG
      WRITE(7,55)YA,(D(I),I=1,N),YB
      WRITE(7,65)YA,(B(I),I=1,N),YB
50    FORMAT(/5X,'X(I)',10(F6.2,4X))
55    FORMAT(/5X,'Y(I)',10F10.6)
65    FORMAT(/5X,'YEGZ',10F10.6)
      GO TO 10
60    CLOSE(7)
      CLOSE(8)
      STOP
      END
```

Primetimo da je program tako realizovan da se broj medjutačaka N učitava na ulazu. U slučaju kada je $N = 0$ program se završava. Takodje, u programu je predvidjeno i tabeliranje tačnog rešenja u posmatranim tačkama, radi kontrole. Jasno je, medjutim, da ovo poslednje ima smisla samo u školskim

primerima gde nam je rešenje poznato. Tako je, na primer, za konturni problem

$$y'' - 2xy' - 2y = -4x; \quad y(0) = 1,\ y(1) = 1 + e \cong 3.7182818,$$

ta no rešenje $y = x + exp(x^2)$.

Za ovaj konturni problem funkcijski potprogram za definisanje funkcija p, q, f, kao i tačnog rešenja, nazvali smo `PQF`. Za slučaj `N=4`, dobili smo rezultate date u nastavku.

```
      FUNCTION PQF(X,M)
      GO TO (10,20,30,40),M
10    PQF=-2.*X
      RETURN
20    PQF=-2.
      RETURN
30    PQF=-4.*X
      RETURN
40    PQF=X+EXP(X*X)
      RETURN
      END
```

```
      BROJ MEDJUTACAKA N=  4
      I       0          1          2          3          4
      X(I)    .00        .20        .40        .60        .80
1.00
      Y(I)  1.000000  1.243014  1.576530  2.035572  2.695769
3.711828
      YEGZ  1.000000  1.240811  1.573511  2.033329  2.696481
3.711828
```

II.4 PARCIJALNE DIFERENCIJALNE JEDNAČINE

II.4.1 Metod mreža

U ovom proglavlju ukazaćemo samo na jedan način za numeričko rešavanje parcijalnih jednačina. Naime, razmotrićemo samo kako se metodom mreža rešava Laplaceova jednačina (eliptičkog tipa) i talasna jednačina (hiperboličkog tipa). Slično se rešavaju i ostale jednačine; na primer jednačina provodjenja toplote (paraboličkog tipa).

Metod mreža ili diferencni metod, kako se često naziva, predstavlja osnovni metod za rešavanje jednačina matematičke fizike (parcijalne jednačine koje se javljaju u fizici i tehnici).

Neka je data linearna parcijalna diferencijalna jedna ina

$$\mathbf{L}u = f \tag{4.1.1}$$

i neka se u oblasti $\mathbf{D}$, koja je ograničena krivom $\Gamma(\mathbf{D} = int\ \Gamma)$, traži ono njeno rešenje koje na krivoj Γ zadovoljava dati konturni uslov

$$\mathbf{K}u = \Psi \quad ((x, y) \in \Gamma). \tag{4.1.2}$$

U primeni metoda mreža, najpre treba izabrati diskretni skup tačaka D_h, koji pripada oblasti $\overline{D}(= D \cup \Gamma)$ i koji se naziva mrežom. Najčešće se u primenama za mrežu uzima familija paralelnih pravih $x_i = x_0 + ih,\ y_j = y_0 + jl\ (i, j = 0, \pm 1, \pm 2, \ldots)$. Tačke preseka ovih pravih se nazivaju čvorovima mreža, a veličine h i l koracima mreže. Dva čvora mreže se nazivaju susednim ako su udaljena po x i y osi samo za jedan korak. Ako sva četiri susedna čvora nekog čvora pripadaju oblasti $\overline{D}$, onda se taj čvor naziva unutrašnjim; u protivnom čvor mreže D_h se naziva graničnim čvorom. U primenama pored pravougaone mreže koriste se i drugi oblici mreža.

Metod mreža se sastoji u aproksimaciji jednačina (4.1.1) i (4.1.2) pomoću odgovarajućih doferencnih jednačina. Naime, operator **L** možemo aproksimirati diferencnim operatorom, veoma jednostavno, zamenom izvoda odgovarajućim diferencama, u unutrašnjim čvorovima mreže. Pri ovome se koriste sledeće formule

$$\frac{\delta u(x_i, y_j)}{\delta x} \cong \frac{u_{i+1,j} - u_{i,j}}{h}$$

$$\frac{\delta u(x_i, y_j)}{\delta y} \cong \frac{u_{i+1,j} - u_{i,j}}{2h}$$

$$\frac{\delta^2 u(x_i, y_j)}{\delta x^2} \cong \frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{h^2}, \quad \text{itd.}$$

Potpuno simetrične su formule za parcijalne izvode po promenljivoj y. Aproksimacija konturnih uslova, u nekim slučajevima, može biti vrlo složen problem, što zavisi od oblika operatora K i konture Γ . Kod tzv. konturnih uslova prve vreste, kod kojih je $Ku = u$, jedan praktičan način za aproksimaciju predložio je L. Collatz i on se sastoji u sledeem:

Neka je graničnom čvoru A najbliža tačka sa konture Γ , tačka B i neka je njihovo rastojanje δ (videti sl. 4.1.1).

sl. 4.1.1

Na osnovu vrednosti funkcije u tačkama B i C, linearnom interpolacijom dobijamo

$$u(A) \cong \frac{h\Psi(B) + \delta u(C)}{h + \delta}.$$

Aproksimacija konturnog uslova (4.1.2) o ovom slučaju se sastoji u drfinisanju jednačina gornjeg oblika za svaki granični čvor.

Jednačine dobijene aproksimacijom jednačine (4.1.1) i konturnog uslova (4.1.2) predstavljaju sistem linearnih jednačina, čijim se rešavanjem dobijaju tražena numerička rešenja postavljenog problema.

U daljem izlaganju ukazaćemo na dva osnovna primera.

II.4.2 Laplaceova jednačina

Neka je potrebno naći rešenje Laplaceove jednačine

$$\Delta u = \frac{\delta^2 u}{\delta x^2} + \frac{\delta^2 u}{\delta y^2} = 0 \quad ((x,y) \in D),$$

koje na konturi kvadrata $D = \{(x,y)|0 < x < 1,\ 0 < y < 1\}$ ispunjava odredjeni uslov $u(x,y) = \Psi(x,y)$ $((x,y) \in Gamma)$. Izaberimo mrežu D_h kod koje je $l = h = \frac{1}{N-1}$, , tako da su čvorovi mreže tačke $(x_i, y_i) = ((i-1)h,\ (j-1)l)$ $(i,j = 1,\ldots,N)$. Standardna diferencna aproksimacija (šema) za rešavanje Laplaceove jednačine ima oblik

$$\frac{1}{h^2} u_{i+1,j} + u_{i-1,j} + u_{i,j-1} - 4u_{i,j} = 0,$$

ili

$$u_{i,j} = \frac{1}{4} u_{i,j+1} + u_{i,j-1} + u_{i-1,j} + u_{i+1,j}).$$

Uzimajući $i,j = 2,\ldots,N-1$ u poslednjoj jednakosti dobijamo sistem od $(N-2)^2$ linearnih jednačina. Za rešavanje ovog sistema obično se koristi metod proste iteracije ili, što je još prostije, Gauss-Seidelov metod (videti odeljak 4.3.3).

Odgovarajući program za rešavanje posmatranog problema ima oblik

```
C==============================================
```

```
C  RESAVANJE LAPLACE-OVE JEDNACINA
C=============================================
      DIMENSION U(25,25)
      OPEN(8,FILE='LAPLACE.IN')
      OPEN(5,FILE='LAPLACE.OUT')
      READ(8,4)N
 4    FORMAT(I2)
      M=N-1
      READ(8,1)(U(1,J),J=1,N),(U(N,J),J=1,N),
     1(U(I,1),I=2,M),(U(I,N),I=2,M)
 1    FORMAT(8F10.0)
      DO 10 I=2,M
      DO 10 J=2,M
 10   U(I,J)=0.
      IMAX=0
 20   WRITE(*,5)
  5   FORMAT(5X,'UNETI MAKSIMALNI BROJ ITERACIJA'/
     110X, '(ZA MAX=0 => KRAJ)')
      READ(*,4)MAX
      IF(MAX.EQ.0) GOTO 100
      DO 30 ITER=1,MAX
      DO 30 I=2,M
      DO 30 J=2,M
 30   U(I,J)=(U(I,J+1)+U(I,J-1)+U(I-1,J)+U(I+1,J))/4.
      IMAX=IMAX+MAX
      WRITE(5,65) IMAX,(J,J=1,N)
 65   FORMAT(//26X,'BROJ ITERACIJA JE',I3//17X,
     14(5X,'J=',I2))
      DO 60 I=1,N
 60   WRITE(5,66) I,(U(I,J),J=1,N)
 66   FORMAT(13X,'I =',I2,6F10.4)
      GO TO 20
100   CLOSE(8)
      CLOSE(5)
      STOP
      END
```

Za rešavanje sistema linearnih jednačina koristili smo Gauss-Seidelov metod sa početnim uslovima $u_{i,j} = 0$ $(i, j = 2, \ldots, N-1)$, pri čemu se na broj iteracija može uticati na ulazu. Za `N=4`

i konturne uslove

$$u_{11} = 0,\ u_{1,2} = 30,\ u_{13} = 60,\ u_{1,4} = 90,$$
$$u_{41} = 180,\ u_{4,2} = 120,\ u_{43} = 60,\ u_{4,4} = 0,$$
$$u_{21} = 60,\ u_{3,1} = 120,\ u_{24} = 60,\ u_{3,4} = 30,$$

dobijeni su sledeći rezultati:

```
              BROJ ITERACIJA JE  2
           J= 1      J= 2      J= 3      J= 4
I = 1       .0000    30.0000    60.0000    90.0000
I = 2     60.0000    47.8125    53.9063    60.0000
I = 3    120.0000    83.9063    56.9531    30.0000
I = 4    180.0000   120.0000    60.0000      .0000
              BROJ ITERACIJA JE  7
           J= 1      J= 2      J= 3      J= 4
I = 1       .0000    30.0000    60.0000    90.0000
I = 2     60.0000    59.9881    59.9940    60.0000
I = 3    120.0000    89.9940    59.9970    30.0000
I = 4    180.0000   120.0000    60.0000      .0000
              BROJ ITERACIJA JE  9
           J= 1      J= 2      J= 3      J= 4
I = 1       .0000    30.0000    60.0000    90.0000
I = 2     60.0000    59.9993    59.9996    60.0000
I = 3    120.0000    89.9996    59.9998    30.0000
I = 4    180.0000   120.0000    60.0000      .0000
              BROJ ITERACIJA JE 10
           J= 1      J= 2      J= 3      J= 4
I = 1       .0000    30.0000    60.0000    90.0000
I = 2     60.0000    59.9998    59.9999    60.0000
I = 3    120.0000    89.9999    60.0000    30.0000
I = 4    180.0000   120.0000    60.0000      .0000
              BROJ ITERACIJA JE 21
           J= 1      J= 2      J= 3      J= 4
I = 1       .0000    30.0000    60.0000    90.0000
I = 2     60.0000    60.0000    60.0000    60.0000
I = 3    120.0000    90.0000    60.0000    30.0000
I = 4    180.0000   120.0000    60.0000      .0000
```

II.4.3 Talasna jednačina

Posmatrajmo talasnu jednačinu

$$\frac{\delta^2 u}{\delta x^2} = \frac{1}{a^2}.\frac{\delta^2 u}{\delta x^2} \tag{4.3.1}$$

sa početnim uslovima

$$u(x,0) = f(x),\ u_1(x,0) = g(x) \quad (0 < x < h) \tag{4.3.2}$$

i konturnim uslovima

$$u(0,t) = \Phi(t),\ u(b,t) = \Xi(t) \quad (t \geq 0). \tag{4.3.3}$$

Korišćenjem konačnih razlika, jednačina (4.3.1) se može aproksimirati pomoću

$$u_{i+1,j} - 2u_{i,j} + u_{i-1,j} = \frac{1}{r^2}(u_{i,j+1} - 2u_{i,j} + ui, j-1), \tag{4.3.4}$$

gde je $r = a\frac{1}{h}$ (h i l su koraci po x i t osi respektivno) i $u_{i,j} \cong u(x_i, t_j)$. Na osnovu prve jednakosti u (4.3.2) imamo

$$u_{i,0} = f(x_i) = f_i. \tag{4.3.5}$$

Uvodjenjem fiktivnotog sloja $j = -1$, drugi početni uslov u (4.3.2) može se jednostavno aproksimirati pomoću

$$u_i(x_i, 0) = g(x_i) = g_i \cong \frac{u_{i,1} - u_{i,-1}}{2l}. \tag{4.3.6}$$

Ako u (4.3.4) stavimo $j = 0$ dobijamo

$$f_{i+1} - 2f_i + f_{i-1} - \frac{1}{r^2}(u_{i,1} - 2f_i + u_{i,-1}) = 0,$$

odakle, s obzirom na (4.3.6) sleduje

$$u_{i,1} = lg_i + f_i + \frac{1}{2}r^2(f_{i+1} - 2f_i + f_{i-1}),$$

tj.

$$u_{i,1} = lg_i + (1 - r^2)f_i + \frac{1}{2}r^2(f_{i+1} + f_{i-1}). \tag{4.3.7}$$

S druge strane iz (4.3.4) sleduje

$$u_{i,j+1} = \frac{1}{2}(u_{i+1,j} + u_{i-1,j}) - u_{i,j-1} + 2(\frac{1}{r^2} - 1)u_{i,j}. \tag{4.3.8}$$

Ako stavimo $h = b/N$ i $x_i = (i-1)h$ $(i = 1, 2, \ldots, N+1)$, na osnovu konturnih uslova (4.3.3) imamo

$$u_{1,j} = \Phi_j,\ u_{N+1,j} = \Psi(t_j) = \Psi_j, \tag{4.3.9}$$

gde je $j = 0, 1, \ldots$ Za odredjivanje rešenja u pravougaoniku $P = \{(x,t)|0 < x < b, 0 < t < T_{max}\}$, maksimalna vrednost indeksa j je celobrojni deo od T_{max}/l tj. $j_{max} = M = [T_{max}/l]$.

Na osnovu jednakosti (4.3.5), (4.3.7), (4.3.8), (4.3.9) jednostavno se nalaze približna rešenja datog problema u čvorovima mreže pravougaonika P, što je realizovano sledećim programom.

```
C================================================
C RESAVANJE PARCIJALNE DIF. JED. HIPERBOLICNOG TIPA
C================================================
      DIMENSION U(3,9)
      OPEN(8,FILE='TALAS.IN')
      OPEN(5,FILE='TALAS.OUT')
      READ (8,5)N,A,B,R,TMAX
 5    FORMAT(I2,4F5.2)
      N1=N+1
      WRITE (5,10) (I,I=1,N1)
 10   FORMAT(10X,1HJ,<N+1>(4X,'U(',I1,',J)')/)
      H=B/FLOAT(N)
      EL=R*H/A
      M=TMAX/EL
      T=0.
      DO 15 K=1,2
      U(K,1)=FF(T,B,3)
```

```
      U(K,N1)=FF(T,B,4)
15    T=T+EL
      X=0.
      R2=R*R
      DO 20 I=2,N
      X=X+H
      U(1,I)=FF(X,B,1)
20    U(2,I)=EL*FF(X,B,2)+(1.-R)*U(1,I)
      DO 25 I=2,N
25    U(2,I)=U(2,I)+R2/2.*(U(1,I+1)+U(1,I-1))
      J=0
30    WRITE(5,35)J,(U(1,I),I=1,N1)
35    FORMAT(7X,I5,<N1>F10.4)
      IF(J.EQ.M)GO TO 50
      J=J+1
      U(3,1)=FF(T,B,3)
      U(3,N1)=FF(T,B,4)
      DO 40 I=2,N
40    U(3,I)=(U(2,I+1)+U(2,I-1))/R2-U(1,I)-2.
     1*(1./R2-1.)*U(2,I)
      T=T+EL
      DO 45 I=1,N1
      U(1,I)=U(2,I)
45    U(2,I)=U(3,I)
      GO TO 30
50    CLOSE(5)
      CLOSE(5)
      STOP
      END
```

Primetimo da se vrednosti rešenja u tri uzastopna sloja $j-1, j, j+1$, pamte u prvoj, drugoj i trečoj vrsti matrice U, respektivno.

Funkcije f, g, Φ, Psi definišu se funkcijskim potprogramom `FF` za `I=1,2,3,4`, respektivno.

U konkretnom slučaju za $a = 2$, $b = 4$, $T_{max} = 6$, $f(x) = x(4-x)$, $g(x) = 0$, $\Phi(t) = 0$, $\Psi(t) = 0$, $N = 4$, i $r = 1$, potprogram `FF` i odgovarajući rezultati imaju oblik:

```
      FUNCTION FF(X,B,I)
      GO TO(10,20,30,40),I
```

```
 10   FF=X*(B-X)
      RETURN
 20   FF=0.
      RETURN
 30   FF=0.
      RETURN
 40   FF=0.
      RETURN
      END
```

J	U(1,J)	U(2,J)	U(3,J)	U(4,J)	U(5,J)
0	.0000	3.0000	4.0000	3.0000	.0000
1	.0000	2.0000	3.0000	2.0000	.0000
2	.0000	.0000	.0000	.0000	.0000
3	.0000	-2.0000	-3.0000	-2.0000	.0000
4	.0000	-3.0000	-4.0000	-3.0000	.0000
5	.0000	-2.0000	-3.0000	-2.0000	.0000
6	.0000	.0000	.0000	.0000	.0000
7	.0000	2.0000	3.0000	2.0000	.0000
8	.0000	3.0000	4.0000	3.0000	.0000
9	.0000	2.0000	3.0000	2.0000	.0000
10	.0000	.0000	.0000	.0000	.0000
11	.0000	-2.0000	-3.0000	-2.0000	.0000
12	.0000	-3.0000	-4.0000	-3.0000	.0000

LITERATURA

1. N. Parezanović: *Računske mašine i programiranje - Programski jezik Fortran IV*. Beograd, 1973.
2. G. V. Milovanović:*Numerička analiza - I deo*. Niš, 1979.
3. M. Bertolino: *Numerička analiza*, Beograd, 1977.
4. S.Gill: *A process for the step-by-step integration of differential equations in an automatic computing machine*. Proc. Cambridge Phil. Soc. 47 (1951), 96-108.
5. G.V.Milovanović: *Numerička matematika - Diferencijalne i integralne jednačine*, Niš, 1981.
6. D.D.McCracken and W.S.Dorn:*Numerical Methods and Fortran Programming*, New York,1964.

7. B.Carnahan, H.A. Luther, J.O.Wilkes:*Applied Numerical Methods*. New York,1969.